UNIVERSITY OF CALIFORNIA SAN DIEGO

Robust Distributed Multi-Robot Information Based Exploration and Sampling

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering
(Intelligent Systems, Robotics and Control)

by

Carlos Porfirio Nieto Granda

2021

DEDICATION

To the curious kid that lives inside of us who is hungry to learn new things and never gives up. To the kid that is always looking up to find the best point of view of everything. To the kid that never stops dreaming and working hard to make those dreams come true.

EPIGRAPH

"Knowledge and understanding are life's faithful companions who will never prove untrue to you. For knowledge is your crown, and understanding your staff; and when they are with you, you can possess no greater treasures."

— Kahlil Gibran

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

The completion of my dissertation would not have been possible without the support of the *U.S. Army Research Laboratory* (ARL) through the projects: Micro Autonomous Systems and Technology (MAST), Distributed and Collaborative Intelligent Systems and Technology (DCIST) Collaborative Research Alliance and the Robotics Collaborative Technology Alliance (RCTA), who funded me during my Doctor of Philosophy (Ph.D.) studies.

I would like to express my deepest appreciation to my advisor, Professor Henrik Christensen, Qualcomm Chancellor's Chair of Robot Systems, for believing in me and guiding me to reach one of my childhood dreams. I am thankful for meeting him at ICRA during that fantastic DARPA Grand Challenge workshop. He gave me the most exceptional opportunity to work on the project MAST, solving multi-robot problems for the Army. Besides that, he taught me during talks and meetings about how to perform quality research while also guiding me with his experience.

I would like to acknowledge my co-advisor, Professor Nikolay Atanasov, for convincing me that theory and practice live in perfect synergy, and that it is fun making proofs that your algorithms can run on robots and the whiteboard.

I am also grateful to Professor Sonia Martinez for her support and for providing me with great feedback for future directions for my thesis during my post-doctoral research. I am happy to know that I will work with her in the future. ¡Muchas Gracias!

I also wish to thank Professor Tara Javidi for her feedback about my work on information adaptive sampling and how information theory can be applied in my future papers. As well, she made me think more carefully about how to prove optimality and use better metrics in my research.

Thank you also to Professor Mohan Trivedi for encouraging me to perform more experiments at the University of California, San Diego, to make me feel proud of my work as a Triton!

I am extremely grateful to Professor Vijay Kumar, Dean of Penn Engineering. Thank you for allowing me to visit the GRASP Laboratory at the University of Pennsylvania so many times. I am very thankful for shared ideas with him, all of his students, and researchers. Also, because of him, I was lucky to be introduced to the world of UAVs and, after that, the heterogeneous multi-robot UAV/UGV world.

I must also thank all my co-authors who helped me to make our ideas into scientific papers and open more branches to continue solving tough problems. They are acknowledged at the end of each chapter of this dissertation.

I want to thank my friends from Tec de Monterrey and the Centro de Investigación en Matemáticas A.C. (CIMAT), who allowed me to learn and work with them during their Ph.D. studies while I was an undergraduate student. Lulú Muñoz, Moises Alencastre, Erik Millan, Benjamin Hernandéz and Fernando Godinéz showed me the world of graduate school and made my undergraduate a fantastic experience by working with all of them. I am deeply indebted to Professor Rafael Murrieta, who encouraged me to find an idea, work hard on it, and supported me with his advice during my Ph.D. Thank you to Professor Ricardo Swain, who allowed me to join his robotics laboratory and helped me to do research like a graduate student. Besides that, he allowed me to work with Professor Seth Hutchinson and Professor Steve LaValle. I would also like to thank Professor Gildardo Sanchéz, President of Universidad Politécnica de Yucatán, for introducing me to Professor Jean Claude Latombe, who encouraged me to stop working on many problems and focus only on one. Also, I am very thankful to him for helping me choose universities for Graduate School and encouraging me to apply to Georgia Tech to pursue my Ph.D. under the mentorship of Professor Henrik Christensen. Special thanks to Professor Jean-Bernard Hayet who invited me to do a summer internship at CIMAT, and showed me the beauty of monocular visual SLAM and how geometry is your best friend finding the best landmarks.

I am grateful for my friends at Georgia Tech, with whom I spent most of my days during my Ph.D. When I arrived in the lab, Michael Kaess, Kai Ni, Richard Roberts, Grant Schindler, and Sangmin Oh received me with open arms. It was terrific to meet again with my good friend Gian Luca Mariotini, who encouraged me to focus on my research and believe in my ideas without thinking about others would think about it. During my time at the lab, I had the honor to meet Michael and Kate Novitsky, and I do not have enough words to thank them for all of their support and friendship. I will never forget those long nights at the lab sharing ideas and food with George Mathai, Ana Huaman, Neil Dantam, Pushkar Kolhe, Hae Won Park, Tobias Kunz, Carl Erdogan, and Manohar Paluri, Yong-Dian, Krunal Chande, Doru Balcan, Denise Chew, Alex Cunningham, Duy-Nguyen, Vadim Indelman, Tesca Fitzgerald, Crystal Chao, Vivian Chu, Andrew Price, Mustafa Mukadam, Jake Huckaby, Alicia Bargar, Changyun Choi, Niharika Arora, Himanshu Sahni, Maya Chalkmak, Arya Irani, Asif Rana, Shray Bansal, Ahmad Humayun, Eric Huang, Daniel Castro, Steven Hickson, Mario Romero, Kaushik Subramanian, Jacob Panikulam, and Siddharth Choudhary. I was fortunate to work with Heni Ben Amor, Pablo Escaramilla (which the only bad thing is that he is a fan of Real Madrid), and Luca Carlone. I learned a lot from our talks and coding sessions while we were working on research papers. I am thankful to have had the honor to work with and learn from John G. Rogers III, whom I admire. I am happy to have the great opportunity to continue working with him at ARL. Also, I would like to thank Alex Trevor for showing me the beauty of coding and playing World of Warcraft. Both of them made me feel that I was part of the best SLAM team, and I am grateful for their continuing support and advice.

As well, I would like to thank my brother Philip Twu. I could not be luckier to have the best roommate and best friend with whom I spent many nights talking about science and technology. I am looking forward to seeing him being successful at the Jet Propulsion Laboratory (NASA). “To Mars and Beyond”.

I would like to thank my friends from the University of Pennsylvania: Shaojie Shen, Sikang Liu, Mickey Whitzer, Kartik Mohta, Monroe Kennedy III, Yash Mulgaonkar, Luis Guerrero, Dinesh Thakur and Loianno Giuseppe. I spent many nights with them learning about

UAVs, eating food from Wawa, and sharing amazing stories and ideas. I am happy that distance does not break friendships.

Thank you to my friends at the Massachusetts Institute of Technology (MIT): Varun Murali, with whom I spent many nights working on crazy projects that made everybody think that we were crazy, and Winter Guerra whose passion for technology and skills amaze me every day.

While working on the MAST project, I had the greatest opportunity to do internships at the U.S. Army Research Laboratory. I would like to extend my sincere thanks to Stuart Young, DARPA/ARL Program Manager, who invited me to visit and work with his group on different robotics problems, which at the end became my Ph.D. dissertation. I also had the honor to meet and work with Jon Fink, Ethan Stump, Jason Gregory, Jeff Twigg, David Baran, Maggie Wigness, Alec Koppel, Garrett Warnell, Nick Fung and Chris Reardon. I learned so much from all of them, and I am happy to continue working with them.

The last years of my Ph.D. I had the pleasure to meet and work with new friends at the University of California, San Diego. Andrew Saad, Priyam Parashar, Ruffin White, Shengye Wang, Michael Hazoglou, Ahmed Qureshi, Ludwig, Vikas Dhiman, Jacob Johnson, Sachiko Matsumoto, Sanmi Adeleye, Michelle Sit, Ehsan Ziaee, Kendra Struffenegger, Shinxin Li, David Paz, Pojung Lai, Keshav Rungta, Cassie Qiu, Darren Chan, Natalie Golaszewski, Anwesan Pal, Ali Mirzaei, Elham Serahati, Andrea Frank, Maya Azarova, Anthony Simeonov, James Smith, Angelique Taylor, Maryam Pourebadi, Florian Richter, Dimitri Schreiber, Jessica Block, Ciara Lugo, Leah Kent, Teresa Ta, and Stephanie Mathew.

I apologize to anyone I forgot to mention here.

THANK YOU TO EVERYONE!

I truly believe that love is the most important thing that you can provide and receive. I am very thankful to receive the unconditional love from my family. I want to thank my family for all their support since the beginning to pursue my career and dreams without questioning my decisions.

My dad Carlos supported me with my academic education and helped me to go to my internship at Cornell University, which encouraged my decision to study my Ph.D. in the USA. My Mom Marilu made me appreciate science since I was a kid by taking me to museums, observatories, reading the encyclopedia, watching documentaries, and buying me magazines to learn how to build robots. She encouraged me to work hard to be the best without losing humility, and gave me the confidence to pursue my scientific career. She also taught me to help others without expecting to receive something back.

"Thank you mami-you raise me up, to more than I can be".

My sister Lulú who taught me that our dreams become true if you work hard, even if you fail at times. You have to believe in yourself and never throw in the towel.

¡Ustedes han estado conmigo desde el comienzo de lo que ha sido la
más grande aventura de mi vida. No tengo suficientes
palabras para agradecerles todo su apoyo y amor.
Solo puedo decirles que los amo y
siempre están en mi corazón!

Thank you to my parents-in-law, Judy and Julius, for their support during the last years of my Ph.D. and allowing me to marry the love of my life.

Last but not least, I want to thank my wife, Mary, for her love, companionship, support, and understanding during my studies and internships far from home. Your support and belief in me when I have a crazy idea gives me the strength to accomplish my dreams. I am fortunate to have you next to me, and I am excited to see the next chapter of our lives together.

"I Will Grow Old With You, My Love"

Chapter 2, in full, is a reprint of the material as it appears on the following published papers:

- Alexander J. B. Trevor, John G. Rogers III, Carlos Nieto-Granda, and Henrik I. Christensen. Applying domain knowledge to SLAM using virtual measurements. *In IEEE International Conference on Robotics and Automation (ICRA)*, pages 5389–5394. Anchorage, Alaska, USA, May 3-7, 2010.

- Alex Trevor, John Rogers, Carlos Nieto-Granda, and Henrik Iskov Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3D. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop: Semantic Mapping and Autonomous Knowledge Acquisition. Taipei, Taiwan, October 18-22, 2010.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. SLAM with expectation maximization for moveable object tracking. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 2077–2082. Taipei, Taiwan, October 18-22, 2010.

- A. Trevor, J. Rogers, C. Nieto, and H. Christensen. Feature-based mapping with grounded landmark and place labels. *In the Robotics Science & Systems (RSS) Workshop on Grounding Human-Robot Dialog for Spatial Tasks*. Los Angeles, CA, USA. June, 2011.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. Simultaneous localization and mapping with learned object recognition and semantic data association. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 1264–1270. San Francisco, CA. USA, October 9-14, 2011.

- John G. Rogers, Stuart Young, Jason M. Gregory, Carlos Nieto-Granda, and H. I. Christensen. Robot mapping in large-scale mixed indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, Alexander Cunningham, Manohar Paluri, Nathan Michael, Frank Dellaert, Henrik I. Christensen, and Vijay Kumar. Effects of sensory precision on mobile robot localization and mapping. *In the 11th International Symposium on Experimental Robotics (ISER)*, pages 433–446. New Delhi and Agra, India, December18-21, 2010.

- Varun Murali, Carlos Nieto-Granda, Siddharth Choudhary, and Henrik I. Christensen. Active planning based extrinsic calibration of exteroceptive sensors in unknown environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Daejeon, South Korea, October 9-14, 2016.

The dissertation author was the primary investigator and co-author of these papers.

Chapter 3, in full is a reprint of the material as it appears on the following published papers:

- Anwesan Pal, Carlos Nieto-Granda and Henrik Christensen. DEDUCE: Diverse scEne Detection methods in Unseen Challenging Environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Macau, China, November 4-8, 2019.

- C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Object guided autonomous exploration for mobile robots in indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XVI*, volume 9084, Baltimore, MD, April 2014.

- Carlos Nieto-Granda, John Rogers, Alexander J. B. Trevor, and Henrik Iskov Christensen. Semantic map partitioning in indoor environments using regional analysis. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 1451–1456. Taipei, Taiwan, October 18-22, 2010.

- Alex Trevor, John Rogers, Carlos Nieto-Granda,and Henrik Iskov Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3d. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop: Semantic Mapping and Autonomous Knowledge Acquisition. Taipei, Taiwan, October 18-22, 2010.

The dissertation author was the primary investigator and co-author of these papers.

Chapter 4, in full is a reprint of the material as it appears on the following published papers:

- Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Henrik I. Christensen, and Frank Dellaert. Distributed mapping with privacy and communication constraints: Light weight algorithms and object-based models. *In the International Journal of Robotics Research (IJRR)*, 36(12):1286–1311, 2017.

- Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Zhen Liu, Henrik I. Christensen, and Frank Dellaert. Multi-robot object-based SLAM. *In the 14th International Symposium on Experimental Robotics (ISER)*. Tokyo, Japan, October 3-6, 2016.

- Siddharth Choudhary, Luca Carlone, Carlos Nieto-Granda, John Rogers, Henrik I. Christensen, and Frank Dellaert. Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed gauss-seidel approach. *In the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5261–5268. Stockholm, Sweden, May 16-21, 2016.

- C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Object guided autonomous exploration for mobile robots in indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XVI*, volume 9084, Baltimore, MD, April 2014.

- Carlos Nieto-Granda, John G. Rogers, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the International Journal of Robotics Research (IJRR)*, 33(4):519–533, February 2014.

- C. Nieto-Granda, John G. Rogers, and H. I. Christensen. Multi-robot exploration strategies for tactical tasks in urban environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

- Neil Dantam, Carlos Nieto-Granda, Henrik I. Christensen, and Mike Stilman. Linguistic composition of semantic maps and hybrid controllers. *In the 12th International Symposium on Experimental Robotics (ISER)*. Québec City, Canada, October 3-6, 2012.

- John Rogers, Carlos Nieto-Granda, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the 12th International Symposium on Experimental Robotics (ISER)*, pages 231–243. Québec City, Canada, October 3-6, 2012.

The dissertation author was the primary investigator and co-author of these papers.

Chapter 5, in full is a reprint of the material as it appears on the following published papers:

- Nicholas Fung, John G. Rogers, Carlos Nieto-Granda, Henrik Christensen, Stephanie Kemna, and Gaurav sukhatme. Coordinating multi-robot systems through environment partitioning for adaptive informative sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, May 20-24, 2019.

- Carlos Nieto-Granda, John G. Rogers, Nicolas Fung, Stephanie Kenma, Henrik I. Christensen, and Gaurav Sukhatme. On-line coordination tasks for multi-robot systems using adaptive informative sampling. *In 2018 International Symposium on Experimental Robotics (ISER)*. Buenos Aires, Argentina, November 5-8, 2018.

- Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav Sukhatme. Multi-robot task allocation for collaborative adaptive informative sampling in structured environments. workshop on informative path planning and adaptive sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia, May 21-25, 2018.

- Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav S. Sukhatme. Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments. In the IEEE International Conference on Robotics and Automation (ICRA). Singapore, May 29-June 3, 2017.

- Nick Fung, Carlos Nieto-Granda, Jason Gregory, and John Rogers. Autonomous exploration using an information gain metric. *In US Army Research Laboratory (ARL-TR-7638)*, Adelphi, MD. United States, March 2016.

The dissertation author was the primary investigator and co-author of these papers.

Chapter 6, in full is a reprint of the material as it appears on the following published papers:

- Carlos Nieto-Granda, Shengye Wang, Vikas Dhiman, John G. Rogers III and Henrik Christensen. Distributed Heterogeneous Multi-robot source seeking using information based sampling with visual recognition. Accepted to the *2020 International Symposium on Experimental Robotics (ISER)*. Malta 2020.

- Carlos Nieto-Granda, Vikas Dhiman and Henrik Christensen. Multi-robot source seeking in partially observable environment. Accepted to *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Auckland, New Zealand, May 9-13, 2020.

The dissertation author was the primary investigator and co-author of these papers.

VITA

2004	Summer student. Institut National de Recherche en Informatique et en Automatique (INRIA). Unité de Recherche Sophia Antipolis, France.
2005	Research Assistant. Department of Computer Science. Cornell University. Ithaca, New York, USA.
2007	Bachelor of Science in Electronics Systems Engineering , Tecnológico de Monterrey Campus, Estado de México, México.
2009	Visitor Researcher, Mathematics Research Center (CIMAT). Computer Science Department. Guanajuato, Gto, Mexico.
2012	Master in Computer Science with a specialization in Robotics and Computer Vision, Georgia Institute of Technology, Atlanta GA, USA.
2013	Visitor Researcher, General Robotics, Automation, Sensing & Perception Lab (GRASP). University of Pennsylvania, Philadelphia PA, USA.
2013	Research Technician IV. Institute for Robotics and Intelligent Machines. Georgia Institute of Technology, Atlanta GA, USA.
2014-2018	Summer Visitor Researcher. Computational and Information Sciences Directorate. Asset Control and Behavior Branch. U.S. Army Research Laboratory. Adelphi, Maryland, USA.
2021	Doctor of Philosophy in Electrical Engineering (Intelligent Systems, Robotics, and Control), University of California San Diego, CA, USA.
2020	Post-doctoral Fellow. U.S. Army Combat Capabilities Development Command. Army Research Laboratory (ARL). Adelphi, Maryland, USA.

PUBLICATIONS

Carlos Nieto-Granda, Shengye Wang, Vikas Dhiman, John G. Rogers and Henrik Christensen. Distributed Heterogeneous Multi-robot source seeking using information based sampling with visual recognition. Accepted to the *2020 International Symposium on Experimental Robotics (ISER)*. Malta 2020.

Carlos Nieto-Granda, Vikas Dhiman and Henrik Christensen. Multi-robot source seeking in partially observable environment. Submitted to *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Auckland, New Zealand, May 9-13, 2020.

Michelle Sit, Carlos Nieto-Granda and Henrik Christensen. Dynamic target tracking and energy-efficient AUV path planning for trash collection using ocean currents. Accepted to *OCEANS 2020*. Singapore 2020.

Christopher Reardon, Jason Gregory, Carlos Nieto, John Rogers. Enabling Situational Awareness of Autonomous Robot-Based Environmental Change Detection via Mixed Reality. Accepted in the *22nd International Conference on Human-Computer Interaction (HCI International 2020)*. Copenhagen, Denmark. July 19-24, 2020.

Anwesan Pal, Carlos Nieto-Granda and Henrik Christensen. DEDUCE: Diverse scEne Detection methods in Unseen Challenging Environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Macau, China, November 4-8, 2019.

Nicholas Fung, John G. Rogers, Carlos Nieto-Granda, Henrik Christensen, Stephanie Kemna, and Gaurav sukhatme. Coordinating multi-robot systems through environment partitioning for adaptive informative sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, May 20-24, 2019.

Carlos Nieto-Granda, John G. Rogers, Nicolas Fung, Stephanie Kemna, Henrik I. Christensen, and Gaurav Sukhatme. On-line coordination tasks for multi-robot systems using adaptive informative sampling. *In 2018 International Symposium on Experimental Robotics (ISER)*. Buenos Aires, Argentina, November 5-8, 2018.

Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav Sukhatme. Multi-robot task allocation for collaborative adaptive informative sampling in structured environments. workshop on informative path planning and adaptive sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia, May 21-25, 2018.

Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Henrik I Christensen, and Frank Dellaert. Distributed mapping with privacy and communication constraints: Light weight algorithms and object-based models. *In the International Journal of Robotics Research (IJRR)*, Volume 36. Number 12. 2017.

Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav S. Sukhatme. Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments. In the IEEE International Conference on Robotics and Automation (ICRA). Singapore, May 29-June 3, 2017.

Varun Murali, Carlos Nieto-Granda, Siddharth Choudhary, and Henrik I. Christensen. Active planning based extrinsic calibration of exteroceptive sensors in unknown environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Daejeon, South Korea, October 9-14, 2016.

Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Zhen Liu, Henrik I. Christensen, and Frank Dellaert. Multi robot object-based SLAM. *In the 14th International Symposium on Experimental Robotics (ISER)*. Tokyo, Japan, October 3-6, 2016.

Siddharth Choudhary, Luca Carlone, Carlos Nieto-Granda, John Rogers, Henrik I. Christensen, and Frank Dellaert. Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed gauss-seidel approach. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm, Sweden, May 16-21, 2016.

Nick Fung, Carlos Nieto-Granda, Jason Gregory, and John Rogers. Autonomous exploration using an information gain metric. *In US Army Research Laboratory (ARL-TR-7638)*, Adelphi, MD. United States, March 2016.

C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Towards contextual awareness in robot mapping: extracting semantic hierarchy from point cloud data. *In SPIE Defense + Security conference. Unmanned Systems Technology XVII*, volume 9468, Baltimore, MD, April 2015.

C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Object guided autonomous exploration for mobile robots in indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XVI*, volume 9084, Baltimore, MD, April 2014.

Carlos Nieto-Granda, John G. Rogers, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the International Journal of Robotics Research (IJRR)*, Volume 33. Number 4. February 2014.

C. Nieto-Granda, John G. Rogers, and H. I. Christensen. Multi-robot exploration strategies for tactical tasks in urban environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

John G. Rogers, Stuart Young, Jason M. Gregory, Carlos Nieto-Granda, and H. I. Christensen. Robot mapping in large-scale mixed indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

Neil Dantam, Carlos Nieto-Granda, Henrik I. Christensen, and Mike Stilman. Linguistic composition of semantic maps and hybrid controllers. *In the 12th International Symposium on Experimental Robotics (ISER)*. Québec City, Canada, October 3-6, 2012.

Manohar Paluri, Carlos Nieto-Granda, and Henrik I. Christensen. A case study on effects of various parameters in a localization and navigation system. *In the 12th International Symposium on Experimental Robotics (ISER)*. Québec City, Canada, October 3-6, 2012.

John Rogers, Carlos Nieto-Granda, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the 12th International Symposium on Experimental Robotics (ISER)*. Québec City, Canada, October 3-6, 2012.

John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. Simultaneous localization and mapping with learned object recognition and semantic data association. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. San Francisco, CA. USA, October 9-14, 2011.

A. Trevor, J. Rogers, C. Nieto, and H. Christensen. Feature-based mapping with grounded landmark and place labels. *In the Robotics Science & Systems (RSS) Workshop on Grounding Human-Robot Dialog for Spatial Tasks*. Los Angeles, CA, USA. June, 2011.

Alex Trevor, John Rogers, Carlos Nieto, and Henrik Christensen. Virtual measurements for robotic mapping. *In GTRC ID 5160 Invention Disclosure Serial number 61/328, 818.*, Atlanta GA, USA, 2010.

John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, Alexander Cunningham, Manohar Paluri, Nathan Michael, Frank Dellaert, Henrik I. Christensen, and Vijay Kumar. Effects of sensory precision on mobile robot localization and mapping. *In the 11th International Symposium on Experimental Robotics (ISER)*. New Delhi and Agra, India, December18-21, 2010.

Carlos Nieto-Granda, John Rogers, Alexander J. B. Trevor, and Henrik Iskov Christensen. Semantic map partitioning in indoor environments using regional analysis. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Taipei, Taiwan, October 18-22, 2010.

John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. SLAM with expectation maximization for moveable object tracking. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Taipei, Taiwan, October 18-22, 2010.

Alex Trevor, John Rogers,Carlos Nieto-Granda,and Henrik Iskov Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3d. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop: Semantic Mapping and Autonomous Knowledge Acquisition. Taipei, Taiwan, October 18-22, 2010.

Alexander J. B. Trevor, John Rogers, Carlos Nieto-Granda, and Henrik Iskov Christensen. Applying domain knowledge to SLAM using virtual measurements. *In IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage, Alaska, USA, May 3-7, 2010.

A. Trevor, J. Rogers, C. Nieto, and H. Christensen. *Virtual Measurements for Robotic Mapping*. GTRC ID 5160 Invention Disclosure Serial number 61/328, 818. Atlanta GA, USA, 2010

ABSTRACT OF THE DISSERTATION

Robust Distributed Multi-Robot Information Based Exploration and Sampling

by

Carlos Porfirio Nieto Granda

Doctor of Philosophy in Electrical Engineering
(Intelligent Systems, Robotics and Control)

University of California San Diego, 2021

Professor Henrik Iskov Christensen, Chair
Professor Nikolay Atanasov, Co-Chair

Military operations require the capability and capacity to gain a situational understanding of complex megacity environments. This is often formulated in Intelligence, Surveillance, and Reconnaissance (ISR) missions. These mission types occur during varied phases of the battle, including Combat Operations and Stability and Support Operations (SASO). Teams of autonomous mobile robots can be tasked to conduct patrol and reconnaissance missions in known dynamic urban environments, in support of Soldiers.

In this thesis, we aim to develop a probabilistic framework called Risk Maps. The autonomous robot will use Risk Maps to plan its actions, which indicate a tactically relevant location where exposure or the environment might enable an attack to inflict maximum damage (e.g., possible IED or sniper locations).

Risk Maps are based in a decision process, which allocate robot patrols against adaptive adversary events. These techniques will use temporal evolution to prevent an inevitable adversary adaption to these strategies, which can nullify much of their effectiveness.

Using a multi-robot coordination approach for decentralized, informative and adaptive sampling application has not a single point of failure. It allows for anytime prediction, and any robot at any point in time to have a reasonable model of the environment. Furthermore, it keeps the required amount of communication to a minimum. Besides that, proper Geographic Information System (GIS) technology provides military commanders the means to rapidly integrate data sets, assess conditions, plan strategies, and evaluate options.

Chapter 1

Introduction

In the future, humans will collaborate closely with artificially intelligent systems. Intelligent systems will be team members and will serve to extend the reach and capability of individual units to enable unprecedented capabilities.

Intelligent exploration and robust collaborative surveillance by autonomous robots will be essential to urban operations, allowing them to prevent future risk of vulnerability and threats. This thesis addresses how prior knowledge of an environment and the history of actions in similar scenarios can predict and prevent future attacks. In this thesis, we present a probabilistic framework into which a set of domain expert rules can be incorporated with spatial and semantic knowledge to enable an autonomous agent to gather information. Autonomous agents can then use this evolving framework to plan optimal actions with respect to this changing information landscape to best accomplish its mission. Our approach extends the techniques described in [Pit+08; ZST15] for use with the Information Based-Exploration framework presented in this thesis used by the *MAST/ARL*[*] navigation module. Pita et al. created the system architecture: ARMOR. This system provides a monthly calendar that satisfies all the key requirements by LAX officers for checkpoint and canine deployment at LAX.

[*]Micro Autonomous Systems and Technology *MAST* Collaborative Technology and Research Alliances. U.S. Combat Capabilities Development Command Army Research Laboratory (ARL)

A major challenge to provide support to the troops by the multi-robot team is to understand how the environment is dynamically changing to provide security of zones in which a convoy should choose the most obvious or convenient route. To address this challenge, it is of interest in utilizing GIS data and activity logs about specific locations. One approach to achieve it is to use an information-based map (Risk Map) formed by a set of modular components which represent knowledge of enemy strategy when evaluating the prior probability of risk. Also, the risk map has a temporal component that blends gradually back to the prior map state, representing the fog of war.

We consider realistic scenarios in which a team of robots with different capabilities explores an unknown environment, and each robot acquires and computes its own map and exchanges this information with the other members of the team considering constraint communication, i.e., robots can only communicate within a specific distance and the exchange of the amount of information is due to bandwidth constraints. Furthermore, each robot is capable of switching from exploration to a source seeking a task and can provide or request assistance when it is needed.

Thesis Statement

Coordination strategies for multi-robot exploration and navigation utilizing adaptive informative sampling to enable robot platforms to autonomously perform intelligence, surveillance and reconnaissance (ISR) missions in unknown environments allowing them to prevent future risk of vulnerability and threats.

Contributions

This statement is supported through the following contributions:

- A complete distributed multi-robot SLAM solution performing feature-based, object-based as well as pose-based graph SLAM, complete with probabilistic data associations and loop closure.

- A distributed algorithm to estimate the 3D trajectories of multiple cooperative robots or mobile devices from relative pose measurements.
- A system framework which provides high and low threat-level information, which will be used as an action plan for movement formation and patrols.
- Novel coordination strategies for heterogeneous multi-robot adaptive informative sampling which is decentralized and robust.

*All the contributions of this thesis are validated through experimental results using both simulated and real data

Chapter 2

Simultaneous Localization and Mapping - "*SLAM*"

2.1 Introduction

The deployment of distributed systems in the real world poses many technical challenges, ranging from coordination and formation control, to task allocation and distributed sensor fusion. In this work we tackle a specific instance of the sensor fusion problem. We consider the case in which a team of robots or mobile devices explores an unknown environment and each robot or mobile device has to estimate its trajectory from its own sensor data and leveraging information exchange with the teammates. Trajectory estimation is relevant as it constitutes the backbone for many estimation and control tasks e.g., geo-tagging sensor data, 3D map reconstruction, position-aware task allocation. Indeed, in our application, trajectory estimation enables distributed 3D reconstruction and localization. Poses are shown in red, static landmarks are shown in green, and moveable landmarks are shown in blue, connected by a blue dotted line showing their movement. We consider a realistic scenario, in which the robots or mobile devices can only communicate when they are within a given distance. Moreover, also during a rendezvous (i.e., when the robots or mobile devices are close enough to communicate). The robots cannot exchange a large amount of information, due to bandwidth constraints (e.g., there exist an upper bound on the number of bytes that the robots or mobile devices can exchange).

2.2 SLAM as a Probabilistic Framework

In landmark-based SLAM, a robot, while navigating, tries to localize itself and at the same time build a map of the environment (represented using landmarks). Assuming a pose of the robot at the i^{th} time step is x_i with $i \in 0 \dots M$, a landmark is l_j with $j \in 0 \dots N$ and a measurement is z_k, with $k \in 0 \dots K$, the joint probability model is given as:

$$P(X,L,Z) = P(x_o) \prod_{i=1}^{M} P(x_i|x_{i-1},u_i) \prod_{k=1}^{K} P\left(z_k|x_{ik},l_{jk}\right)$$

where $P(x_o)$ is a prior on the initial state, $P(x_i|x_{i-1},u_i)$ is the motion model, parametrized by a control input u_i and $P(z_k|x_{ik},l_{jk})$ is the landmark measurement model, x_{ik} and l_{jk} corresponds to measurement z_k. Assuming the motion and measurement models are Gaussian, $P(x_i|x_{i-1},u_i) \propto \exp -\frac{1}{2}\|f_i(x_{i-1},u_i) - x_i\|^2_{\Lambda_i}$ and $P(z_k|x_{ik},l_{jk}) \propto \exp -\frac{1}{2}\|h_k(x_{ik},l_{jk}) - z_k\|^2_{\Sigma_k}$ where $f()$ is the robot motion equation and $h()$ is a landmark measurement equation with Λ_i and Σ_k as the respective covariances. We use a factor graph to represent the joint probability model $P(X,L,Z)$ where each factor represents either $P(x_o)$ or $P(x_i|x_{i-1},u_i)$ or $P\left(z_k|x_{ik},l_{jk}\right)$. Therefore the joint probability model can be written as $P(X,L,Z) \propto g(\Theta) = \prod_i g_i(\Theta_i)$ where Θ_i is the set of variables θ_j adjacent to the factor g_i. Figure 2.2 shows the corresponding factor graph. Given all the measurements, we obtain the maximum a posteriori (MAP) estimate by maximizing the joint probability $P(X,L,Z)$.

$$\Theta^* = \arg\max_{\Theta} P(X,L|Z) = \arg\min_{\Theta} \left(-\log g(\Theta)\right) \tag{2.1}$$

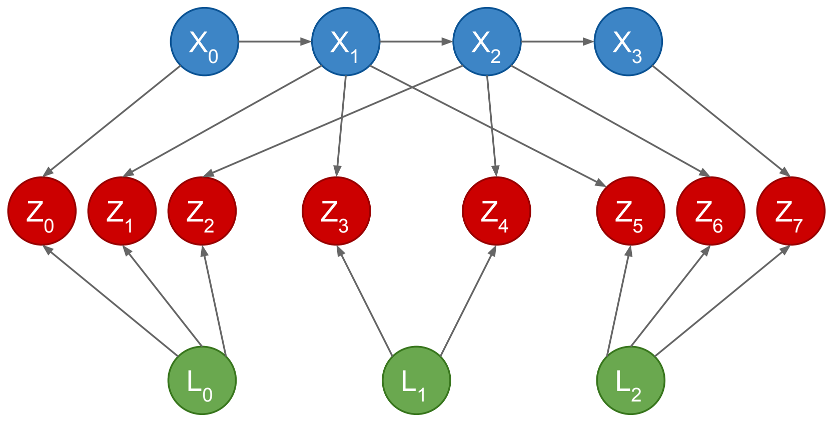


Figure 2.1: Belief Net corresponding to the landmark-based SLAM problem. The pose of the robot at i^{th} time step is x_i with $i \in 0 \ldots M$, a landmark is l_j with $j \in 1 \ldots N$ and a measurement is z_k, with $k \in 1 \ldots K$.

which leads to the following non-linear least squares problem:

$$\Theta^* = \arg\min_{\Theta} \sum_{i=1}^{M} \|f_i(x_{i-1}, u_i) - x_i\|^2_{\Lambda_i} + \sum_{k=1}^{K} \|h_k(x_{ik}, l_{jk}) - z_k\|^2_{\Sigma_k}.$$

The non-linear least squares problem is solved using a non-linear optimization method such as the Levenberg-Marquardt algorithm which solves a succession of linear approximations in order to approach the minimum. We use Square Root SAM (Dellaert and Kaess [DK06b]) to optimize the resulting factor graph.

2.3 SLAM with Expectation Maximization for Moveable Object Tracking

Many SLAM algorithms rely on the assumption that the environment is static, and will perform poorly or fail if mapped objects move. However, to operate in real world dynamic environments, algorithms will need to recognize moveable objects. Consider, for example, a robot that makes a map of a room, and then returns several days later. Some of the features in

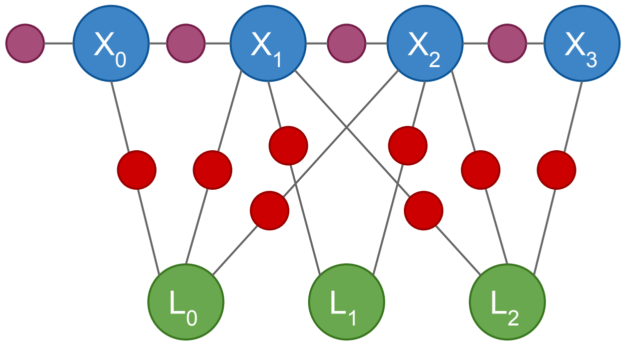


Figure 2.2: Factor graph representation of the SLAM problem. Blue circles denote the poses (X) and green circles denote landmarks (L). Small purple circles represent odometry constraints and red circles represent landmark-pose constraint.

its map might correspond to immobile objects, such as walls, while some might correspond to objects that may have moved, such as furniture. If someone has moved some of the objects in the room that were part of the robot's map, it will be potentially catastrophic because the SLAM system will make an inconsistent map out of incompatible measurements.

In this section, we propose an Expectation Maximization (EM) approach to data association and static vs. moveable object determination for performing SLAM in a pathological office environment where mapped landmarks move. Our algorithm has been validated in an indoor environment.

A common assumption is that the environment being mapped is static. There are two main research directions which attempt to relax this assumption. One approach partitions the model into two maps; one map holds only the static landmarks and the other holds the dynamic landmarks. Hähnel *et al.* use an EM based technique to split the occupancy grid map into static and dynamic maps over multiple iterations in a batch process. This technique is shown in [HSB02] and [Häh+03] to be effective in generating useful maps in environments with moving people. Biswas *et al.* take a finite set of snapshots of the map and employ an EM algorithm to separate the moving components to generate a map of the static environment and a series of

separate maps of the dynamic objects at each snapshot. Wolf and Sukhatme [WS05] [WS04] [WS03] are able to separate static and dynamic maps with an online algorithm. Stachniss and Burgard developed an algorithm in [SB05] which identifies dynamic parts of the environment which engages a finite number of states.
The map is represented with a “patch map” that identifies the alternative appearances of the portions of the map which are dynamic, like doors.

The second direction with respect to relaxing the static world assumption is to track moving objects while mapping the static landmarks. Wang *et al.* in [WT02] and [WTT03] are able to track moving objects and separate the maps in an online fashion by deferring the classification between static and dynamic objects until several laser scans can be analyzed to make this determination. Bibby *et.al* [BR07] use an EM based technique over a finite time-window to perform dynamic vs. static landmark determination; however, their approach differs from ours in several important ways. First, they use a finite sliding window after which no data associations can be changed. Our approach has no such limitation as we are attempting to determine which aspects of the environment are static vs dynamic over the entire mapping run. Bibby’s technique does a good job of detecting *moving* objects but it will not be able to detect *moveable* objects over longer time intervals, that might move when they aren’t being observed by the robot. Additionally, Bibby addresses the static data association problem by maintaining a distribution of data associations across the sliding window; the data association decision is made permanent at the end of the window (6 steps in Bibby’s implementation). An infinite sliding window would be computationally intractable in this implementation due to exponential growth of the interpretation tree; however, our approach offers an alternative solution to the static data association problem which does not suffer from finite history.

Approach

Our algorithm is based upon the Square Root SAM of Dellaert [Del05b]. We have modified this algorithm with a per-landmark weighting term which enables discrimination between stationary and mobile landmarks to allow for more reliable localization. The remaining landmarks which have a low weight are classified as being moveable and are now tracked by the robot without influencing the robot's trajectory.

Square Root SAM

Our implementation of Square Root SAM finds the assignment for the robot trajectory and landmark positions that minimizes the least squares error in the observed measurements. As is common in the SLAM literature, our motion and measurement models assume Gaussian noise. Each adjacent pose in the robot trajectory is modeled by the motion model in equation 2.2.

$$x_i = f_i(x_{i-1}, u_i) + \nu_i \tag{2.2}$$

where $f_i(.)$ is the nonlinear motion model and u_i is the observed odometry from the robot, and ν_i is the process noise. In our case, we use a differential drive robot which has a three-dimensional pose (x_i, y_i, θ_i). The model is shown in equation 2.3.

$$\begin{pmatrix} \Delta x_i \\ \Delta y_i \\ \Delta \theta_i \end{pmatrix} = \begin{pmatrix} u_0 \cos(\tilde{\theta}) - u_1 \sin(\tilde{\theta}) \\ u_0 \sin(\tilde{\theta}) + u_1 \cos(\tilde{\theta}) \\ u_2 \end{pmatrix} \tag{2.3}$$

where u_0 is the forward motion, u_1 is the side motion, u_2 is the angular motion of the robot, and $\tilde{\theta} = \theta_{i-1} + \frac{u_2}{2}$. The measurement model determines the range and bearing to the landmarks. It has the form shown in equation 2.4.

$$h(x_i, l_j) = \begin{pmatrix} \sqrt{(x_i - l_j^x)^2 + (y_i - l_j^y)^2} \\ \tan^{-1}\left(\frac{(l_j^y - y_i)}{(l_j^x - x_i)}\right) - \theta_i \end{pmatrix} \tag{2.4}$$

The linearized least squares problem is formed from the Jacobians of these motion and measurement models as is seen in [Del05b]. By organizing the Jacobians appropriately in matrix A and collecting the innovation of the measurements and odometry in vector b we can iteratively solve for the robot trajectory and landmarks which are stacked in Θ as seen in equation 2.5.

$$\Theta^* = \arg\min_{\Theta} ||A\Theta - b||^2 \tag{2.5}$$

After each iteration, the Jacobians are re-linearized about the current solution Θ. The solution to this minimization problem can be found quickly by direct QR factorization of the matrix A using Householder reflectors followed by back-substitution. The source paper for this technique [Del05b] exploits sparsity to vastly improve performance; however, we are currently using dense matrices. The optimization currently runs in approximately one second per iteration for a SAM problem of around 50 poses and 100 measurements with dense matrices. We anticipate achieving much better performance once we utilize sparse matrices.

Expectation Maximization

To establish an EM algorithm for SAM with moveable objects, we first must express the joint probability model in equation 2.6.

$$P(X, M, Z) = \prod_{poses} P(x_i | x_{i-1}, u_i) * \prod_{landmarks} P(z_k | x_{i_k}, l_{j_k}) \tag{2.6}$$

where $P(x_i|x_{i-1}, u_i)$ is the motion model and $P(z_k|x_{i_k}, l_{j_k})$ is the sensor model. We add a hidden variable ω_k to each landmark measurement, which changes the joint probability model to equation 2.7.

$$P(X,M,Z,\Omega) = \prod_{poses} P(x_i|x_{i-1},u_i) * \prod_{landmarks} P(z_k|x_{i_k},l_{j_k},\omega_k) \tag{2.7}$$

With a Gaussian representation for the sensor model, this new set of parameters ω_k results in the sensor model in equation 2.8.

$$P(z_k|x_{i_k},l_{j_k},\omega_k) \propto \exp -(\omega_k((z_k - h(x_{i_k},l_{j_k})^T \Sigma^{-1}(z_k - h(x_{i_k},l_{j_k}))) \tag{2.8}$$

With the interpretation that ω_k is the likelihood that this measurement comes from a static landmark, if the landmark is not static (i.e. $\omega_k = 0$), then the measurement does not affect the joint likelihood since $P(z_k|x_{i_k},l_{j_k},\omega_k) = 1$ for all assignments to the robot poses and landmark positions. When the landmark is static (i.e. $\omega_k = 1$), then this weighting term makes this measurement behave like normal. Obviously, the hidden variables ω_k cannot be directly observed by the robot and must instead be estimated from multiple observations of each object. The M step selects new assignments for the ω_k to maximize the joint likelihood. Since the likelihood can be trivially maximized by setting all $\omega_k = 0$, we introduce a Lagrange multiplier to penalize setting too many moveable landmarks. The non-constant portions of the log likelihood for the measurements is now seen in equation 2.9.

$$l(Z,X,L,\Omega) = \sum_{measurements} (-\omega_k(\eta_k^T \Sigma_k^{-1} \eta_k)) - \lambda(1-\omega)^T(1-\omega) \tag{2.9}$$

where η_k is the innovation of the k-th measurement (the k-th measurement minus its predicted value). This log likelihood is maximized when

$$\frac{\delta l(Z,X,L,\Omega)}{\delta\Omega} = 0 \tag{2.10}$$

For each ω_k we get the equation 2.11

$$-(\eta_k^T \Sigma_k^{-1} \eta_k) + 2\lambda - 2\lambda\omega_k = 0 \tag{2.11}$$

so

$$\omega_k = 1 - \frac{\eta_k^T \Sigma_k^{-1} \eta_k}{2\lambda} \tag{2.12}$$

We have made the additional modification that the ω_k is not assigned per measurement, but instead since these measurements come from objects we would like to treat the objects as the things that are moveable instead of the measurements being unreliable. This is a simple modification which changes equation 2.9 into equation 2.13.

$$l(Z,X,L,\Omega) = \sum_{measurements} (-\omega_{l_k}(\eta_k^T \Sigma_k^{-1} \eta_k)) - \lambda(1-\omega)^T(1-\omega) \tag{2.13}$$

where ω_{l_k} is the weight of landmark l involved in the kth measurement. For each ω_{l_k} we get the equation

$$\sum_{k \in K_l} -(\eta_k^T \Sigma_k^{-1} \eta_k) + 2\lambda - 2\lambda\omega_l = 0 \tag{2.14}$$

so

$$\omega_l = 1 - \frac{\sum_{k \in K_l} \eta_k^T \Sigma_k^{-1} \eta_k}{2\lambda} \tag{2.15}$$

where K_l is the set of measurements of landmark l. The Lagrange multiplier λ can be assigned to trade off the penalty for having moveable landmarks. This was the M step of EM.

The E step of EM computes the robot trajectory and the landmark positions with the current estimates of the weighting terms ω_l. The least-squares problem solved by Square Root SAM to find the most likely map is simply the weighted least squares problem, which is to add a weighting term $\omega_l \in [0,1]$ to each landmark measurement row i.e.

$$H * x_i + J * l_{j_i} = z_{ij} - h(x_i, l_{j_i}) \tag{2.16}$$

becomes

$$\omega_{l_{j_i}} * (H * x_i + J * l_{j_i}) = \omega_{l_{j_i}} * (z_{ij} - h(x_i, l_{j_i})) \tag{2.17}$$

where $H = \frac{\delta \nu}{\delta x_i}$ and $J = \frac{\delta \nu}{\delta l_{j_i}}$ with $\omega_{l_{j_i}}$ is the weight assigned to the landmark which we are measuring in this row. In the least squares formulation, this will have the effect of scaling the contribution of this measurement to the overall solution.

2.3.1 Moveable Landmark Tracking

After the terminal iteration of the EM algorithm, landmarks which have a weight factor falling below a specific threshold are removed from the SAM optimization and collected in a separate data structure. The final map is optimized once again with the moveable landmarks removed. Measurements on the dynamic landmarks are used with the final trajectory to compute global locations for the dynamic landmarks. These landmarks are now moved into a separate list of moveable landmarks where they could be referred to later to find a list of observed positions. If the moveable landmarks were tagged with semantic information, the robot would then be able to use this data structure as a candidate list of search locations for the object for a retrieval task. The robot can start with the most recently seen position for this object and then try the other places that the object has been seen in the past. Currently, the distribution of positions of the moveable landmarks is being represented as a list of observed locations.

2.3.2 Experiments

To verify the performance of our algorithm, we performed a series of experiments with moveable landmarks in our office environment.

Robot Platform

To test our system, we collected data with a Mobile Robotics Peoplebot. Our robot is equipped with a SICK LMS-291 laser scanner, as well as a Logitech webcam. As measurements, we detect ARToolKit Plus [WS07] markers in the camera images. The ARToolKit was used in these experiments because the emphasis here is on the detection of static and mobile landmarks. Having landmarks with trivial data association helps us focus on the key contribution of this paper; however, there is no loss of generality and natural landmarks will be considered in future work. The resulting measurements give the relative pose of each marker with respect to the camera. Laser data was also logged, but is only used for visualization purposes. Wheel odometry is also logged, and is used as the input for the motion model.

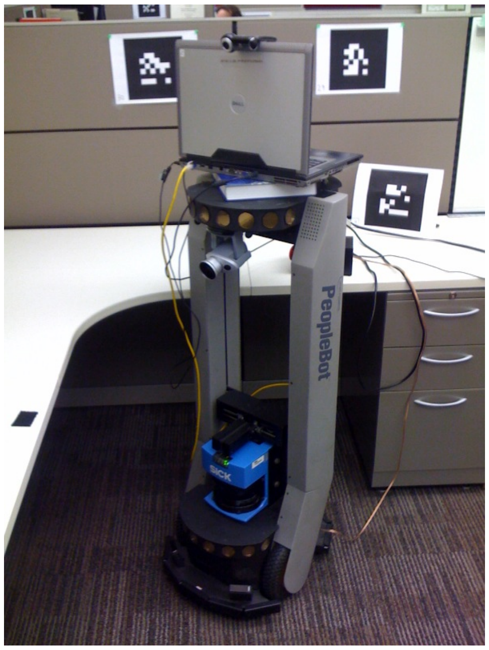


Figure 2.3: The robot platform used for these experiments. The webcam attached to the laptop is the one that is used to collect visual measurements.

Experimental Setup

The environment used for our experiment was a portion of our lab, consisting of two student offices and the corridors connecting them. ARToolKit markers were placed throughout the environment to serve as landmarks. Some of the markers were pinned to the walls, while others were held by moveable frames which facilitated their movement during the experiments.

Procedure

Our first experiments were to move the robot in a circle in one of our student offices measuring 4.5 meters on a side. This office had 12 ARToolKit markers pinned to the walls. In addition to these static landmarks, this office had a total of 10 moveable landmarks which were placed upon the desks and shelves. We performed tests of our implementation of the standard SAM algorithm by moving the robot in this office to collect measurements of landmarks without moving them during the test run. The next experiment was to move the landmarks to a second location within this same cubicle midway through the data collection.

We performed a larger scale experiment in which the robot moved between both of our group's student offices. Each of these offices is 4.5 meters on a side, and they are separated by about 12 meters of corridors. We left the 12 ARToolKit markers in the first office from the small scale experiment, and placed 9 markers in the second office. Additionally, we pinned 8 markers to the walls in the corridor between our offices. Several data sets were collected in this setup with varying numbers of moveable landmarks. In each run, the robot was moved in the first office so that each landmark was observed multiple times and then the robot was driven down the corridor. While the robot was being moved down the corridor, the moveable landmarks were transferred from the starting office to random locations in the second office. The robot was then maneuvered in the second office so that each landmark was observed multiple times and then it was driven back to the first office. The robot was driven in the first office in a few loops and was finally placed as close as possible to its starting location. Each test run of this type featured similar trajectories, but always the moveable landmarks were placed in arbitrary positions in the two offices.

The logs were used as input for our algorithm. While the ARToolKit Plus measurements provide the relative pose of the landmark in Cartesian space with respect to the camera, we instead converted this to a range and bearing measurement. As described in section 2.3, the algorithm first considered all measurements, and iteratively adjusted the weights to determine which landmarks were moveable, and which were static. The algorithm iterated until the state converged and the updates were below a specified threshold. Once a stable configuration had been found, the landmarks which had weights below a certain threshold were removed from the SAM problem and were tracked separately. At this point, a final map was generated.

2.3.3 Results

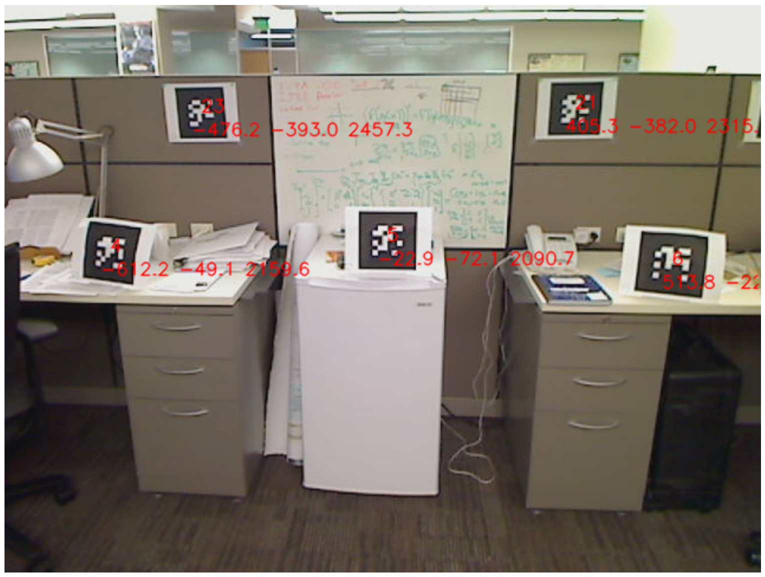
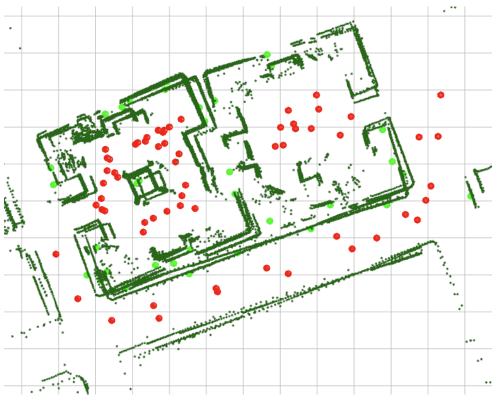
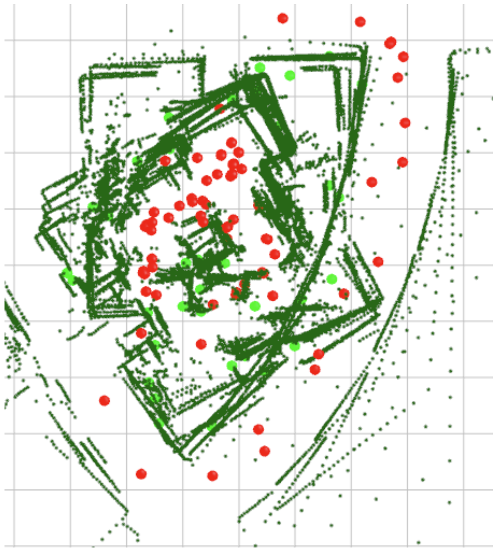


Figure 2.4: One of the images used as part of our experiments, as taken from the robot.

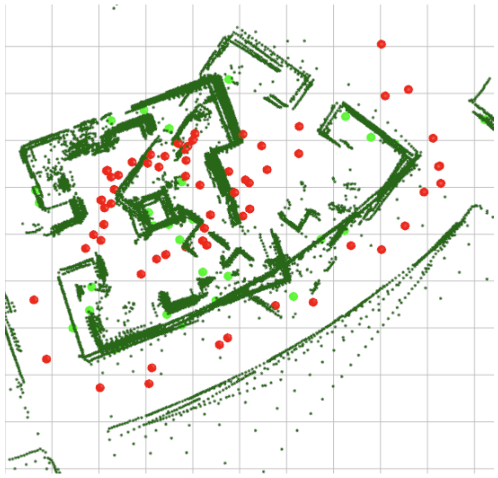
We present the longest test run in detail. The initial state of the problem can be seen in Figure 2.5a. This corresponds to the raw odometry and sensor measurements. The robot starts out in the upper rightmost corner of the left office, facing up in the image. The initial iteration of the EM algorithm will have 1.0 in each ω_k, so each landmark is initially assumed to be static. The SAM optimization is iterated until convergence with these parameters, resulting in a very poor quality map as can be seen in Figure 2.5b. It is apparent in this figure that the moveable landmarks have resulted in incorrect loop closures causing the two offices to intersect almost completely. After two iterations of the EM algorithm the map can be seen in Figure 2.5c. This map is clearly better than the initial state; with two additional iterations of the EM algorithm the low weight landmarks can be thresholded to generate the final map in Figure 2.5d. In this particular run, there were 10 moveable landmarks, 6 of which were detected as moveable. No static landmarks were mistakenly detected as moveable. The remaining 4 moveable landmarks which were mistakenly classified as static were only observed in one of the two offices. Without the observation of the landmark in its second position, the algorithm cannot determine that the



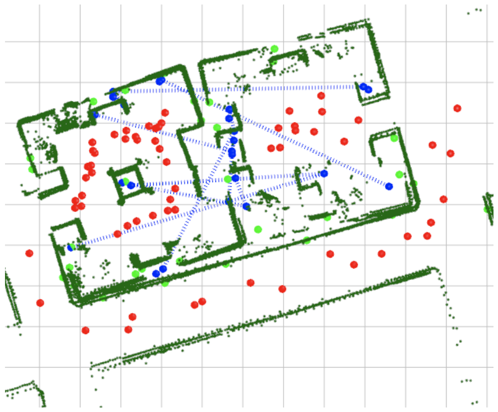
(a) Initial State of the Map.



(b) The resulting map with all measurements (including moveable objects) prior to the first weight assignment.



(c) The resulting map after two iterations of the EM algorithm



(d) The resulting map after all iterations and thresholding to exclude moveable objects.

Figure 2.5: Poses are shown in red, static landmarks are shown in green, and moveable landmarks are shown in blue, connected by a blue dotted line showing their movement. The dark green points are laser scans, and are for visualization purposes only.

landmark had moved. The missing observations can be explained by our use of a webcam and some poor lighting, or the missing landmarks do not appear with a front aspect view which ARToolKit Plus can detect. We have performed two additional test runs of this length and four runs of the single office test, with similar results.

We performed an additional test run where we ignore measurements from the static landmarks. This test was generated by running one of our normal test runs with measurements suppressed from markers that we know to have been static. In this test run, the EM operation was able to correctly identify all of the landmarks as moveable. The final output appears the same as the initial odometry solution. This makes sense because the SAM problem has no measurements between landmarks which affect the trajectory since all of the landmarks had moved.

2.4 OMNIMAPPER

Our mapping system is based upon the *GTsam* library developed at Georgia Tech. This library extends the Square Root SAM technique in [DK06b] with sparse linear algebra in a nonlinear optimization engine. We optimize a graph of measurements between robot poses along a trajectory, and between robot poses and various landmarks in the environment. Measurements come from various software components. Measurements of simple objects like points, lines, and planes are data associated to mapped landmarks with the joint compatibility branch and bound (JCBB) technique from [NT01a]. Measurements of richer landmarks such as objects or signs are data associated based upon interpretation of this semantic information.

OmniMapper is a complete SLAM solution performing feature-based as well as pose graph SLAM, complete with probabilistic data association and loop closure. A complete SLAM solution consists of a *back-end* as well as a *front-end*; these two components must work together to build a map of an environment. There are many options for back-end SLAM libraries; however, they can be described by the two basic categories of *Filtering* approaches like the EKF and

Smoothing approaches like iSAM2.

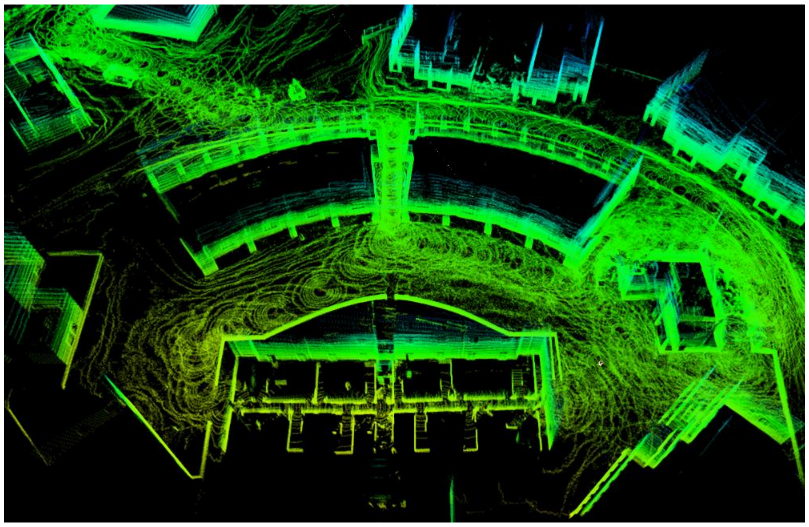


Figure 2.6: Map generated using OmniMapper.

Omnimapper is a plugin architecture using the M-space formulation of Folkesson and Christensen [FC04]. The *M-space* representation makes incorporating new landmark measurement types simple by expressing linearization parameters in terms of simpler components via the chain rule and then multiplying the matrices together to form the linearized measurement model Jacobians.handle multiple landmark types simultaneously such as lines, planes, walls, doors, and objects. Also, we have developed plugins for 2D canonical scan matching (CSM) [Cen08], and full point cloud alignment via generalized iterated closest point (G-ICP) [SHT05].

2.4.1 Object mapping

In [Rog+11a], we developed techniques for mapping visual objects. This study and a novel technique called *semantic data association*. The plugin for handling objects in the *OmniMapper* is simpler than the plane and line plugins described above, because objects are tracked as 3D points.

$$h = R^T * (\vec{p} - \vec{t}) \tag{2.18}$$

The Jacobian with respect to the robot pose is then given by:

$$\frac{\delta h}{\delta X_r} = \begin{bmatrix} 0 & -h_z & h_y & 1 & 0 & 0 \\ h_z & 0 & -h_x & 0 & 1 & 0 \\ -h_y & h_x & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.19}$$

The Jacobian with respect to the landmark is given by:

$$\frac{\delta h}{\delta l} = R^T \tag{2.20}$$

2.4.2 Data association

Data association is the process of assigning sensor measurements to previously mapped or new landmarks. One of the core problems in SLAM is how to perform this data association, and how to recover from erroneous data associations.

There are a number of approaches to performing data association of measurements to mapped landmarks. The various techniques differ in what is needed to make the data association determination. The simplest technique, dubbed *Simple*, is not probabilistically motivated because it does not consider the uncertainty in the location of the robot relative to a measured landmark. A more sophisticated technique, called *Nearest Neighbor* uses the covariance between the robot and the landmark position to compute the probability that a measurement was generated from a mapped landmark. Finally, the most sophisticated technique, called *Joint Compatibility Branch and Bound*, performs probabilistic data association on all measurements simultaneously.

We have implemented these three data association algorithms within the *OmniMapper* via generic template programming. Each of these data association modules takes the current map and sensor measurements. Each module then returns indices referring to mapped landmarks for sensor measurements which correspond to these previously found landmarks. The data association module also inserts new landmarks for measurements which do not correspond to previously mapped landmarks.

The *Simple* data association module takes a measurement function which computes the error corresponding to inserting a new measurement on an existing landmark. This error function does not take into account the *shape* of the covariance between the robot and this mapped landmark, and does not make a probabilistic data association. The error function is illustrated in equation 2.21.

$$e(x,l,m) = \|pred(x,l) - m\| \tag{2.21}$$

Here, a data association is accepted between pose x, landmark l, with measurement m if the predicted measurement between x and l is close to m. This data association module is fast and easy; however, it is not probabilistically motivated and can therefore not resolve fine detail on landmarks which are close. It also cannot handle loop closures well when uncertainty is too great.

The *Nearest Neighbor* data association module does a much better job at resolving fine detail and performing data association in a loop closure. This is because the error function between a measurement and the robot is now the Mahalanobis distance between these entities using the covariance of the predicted measurement, as can be seen in equation 2.22.

$$e(x,l,m) = \|pred(x,l) - m\|_{\Sigma(meas)} \tag{2.22}$$

The covariance of the predicted measurement is computed from the joint distribution of the robot pose in question (usually the new robot pose) and the landmark being tested. This joint covariance is projected into the measurement space using the measurement Jacobians, as can be seen in equation 2.23.

$$\Sigma(meas) = \frac{dh}{dx} * \Sigma(x,l) * \frac{dh}{dx}^T + \Sigma(m) \tag{2.23}$$

Here, $\Sigma(meas)$ is the covariance in measurement space, $\Sigma(x,l)$ is the covariance between the robot and landmark, and $\frac{dh}{dx}$ is the Jacobian for the function transforming landmark and robot positions into measurement space. $\Sigma(m)$ is the sensor error model. A nearest-neighbor data association is accepted if this Mahalanobis distance falls below a probabilistic threshold. It should be noted that a coarse version of the *Simple* data associator is used to filter and avoid testing data associations that are very far apart. This is needed due to the computational complexity of computing the joint distribution between the robot pose and each potential matched landmark.

The *Joint Compatibility Branch and Bound* (JCBB) data associator extends the *Nearest-Neighbor* data associator with an additional joint compatibility test. Once the *Nearest-Neighbor* (or *individual compatibility* test) is passed, the group of measurements is considered together against a χ^2 test with as many degrees of freedom as measurements given. Measurements are added in an interpretation tree search with a branch and bound test. The branch and bound test prevents exploring other interpretations of landmarks and measurements which are worse than one that has already been tried. This module is based upon the paper [NT01a], which should be consulted for the implementation details.

2.5 Robot mapping in large-scale mixed indoor and outdoor environments

Tactical situational awareness in unstructured and mixed indoor/ outdoor scenarios is needed for urban combat as well as rescue operations. Situational awareness gathered by mobile robots can increase the margin of safety for military as well as rescue personnel through increased stand-off as well as the ability to automate the gathering of information and to leverage advanced sensory modalities. To successfully operate in unstructured and unfamiliar environments, mobile robots must be equipped with the ability to build a world model or map of their environment and the ability to use that map to track their position and perform their mission. With the introduction of a wider variety of sensors that can generate dense point clouds it is of interest to provide mapping systems that can utilize these sensors for generation of detailed maps and to utilize the extra information to solve challenging problems such as loop closing. We present a strategy to build dense maps and to do automatic loop closing. The system is designed to have two parallel threads, one building a map and the other detecting potential loop closures. The approach has been integrated into a mapping system named OmniMapper. The system can utilize a variety of different sensors such as the Velodyne 32E, Kinect style 3D cameras, or a Hokuyo sensor mounted on a nodding pan-tilt unit such as the Direct Perception PTU-46, in addition to 2D measurements from standard laser scanners.

2.5.1 Approach

The *OmniMapper* was initially developed for feature-based mapping of landmarks in various types of highly structured environments. Feature-based techniques are useful for mapping man-made buildings, but are less useful in unstructured outdoor environments, or even highly cluttered indoor environments. In the presence of clutter, large sections of planes and walls might be hard to identify; to operate in these situations, mapping plugins based upon iterative closest point (ICP) algorithms were developed. The first of these is designed for indoor environments with a 2.5D planar constraint and lot of clutter; it is based upon canonical scan matching 13(CSM). The second of these is designed for outdoor environments where the robot may have general 6-DOF motion; it is based upon G-ICP. The 2.5D mapping plugin based upon CSM is described in 2.5.2, and the full 3D mapping plugin based on G-ICP is described in 2.5.3.

Both of the featureless mapping plugins described in this section are based upon the ICP algorithm. The ICP algorithm is a two phase algorithm which is repeated until convergence. ICP algorithms take as their input two point clouds (or scans), a reference cloud from the previous scan and a new cloud from the current scan. In the first phase, putative point matches are determined by selecting for each point in the new cloud the closest point in the reference cloud. In the second phase, these putative point matches form a set of constraints which is optimized via a nonlinear estimation algorithm to determine the relative pose between these two clouds. This procedure is now repeated by selecting a new set of putative point matches and solving for the relative pose until a convergence threshold is reached.

2.5.2 OmniMapper CSM

To enable operation in general indoor environments where no level of clutter or structural assumptions can be made, we developed a featureless map plugin called *OmniMapper CSM*. This mapping plugin uses the iterative closest point algorithm based on the point to line metric described in [Cen08] to compute the relative pose between trajectory elements.

Andrea Censi provided a library routine called *canonical scan matching*(CSM) which computes the rigid body transform between two 2D laser scans. This library routine is called from two threads of computation in the OmniMapper CSM plugin. The first thread of computation processes incoming laser scans and registers them to the previously incorporated scan. The robot's odometry is used to provide an initial estimate of the transform between the two scans; the CSM library routine then completes the alignment to refine this transform.

The second thread of computation looks back at previous laser scans and finds loop closures. For each new scan, the centroid of the laser data projected as a point cloud is computed in a map coordinate reference frame. This centroid is compared to the centroids of all previous scans already incorporated in the map. Whichever is the closest to the current scan is then analyzed by the CSM library routine to attempt to find the transform between these scans. If this procedure is successful, then a new pose constraint is added to the trajectory and the map is re-optimized.

The CSM library routine also computes an estimate of the covariance of the laser scan match using the technique described in [Cen07]. This estimate takes into account the statistical noise of the laser scanner as well as the geometry of the environment when computing the covariance estimate. The result is a large covariance along a structure such as a hallway, with a tight covariance across the hallway, in the direction normal to the wall. This covariance estimate is used as the error model for each constraint added to the GTsam nonlinear factor graph.

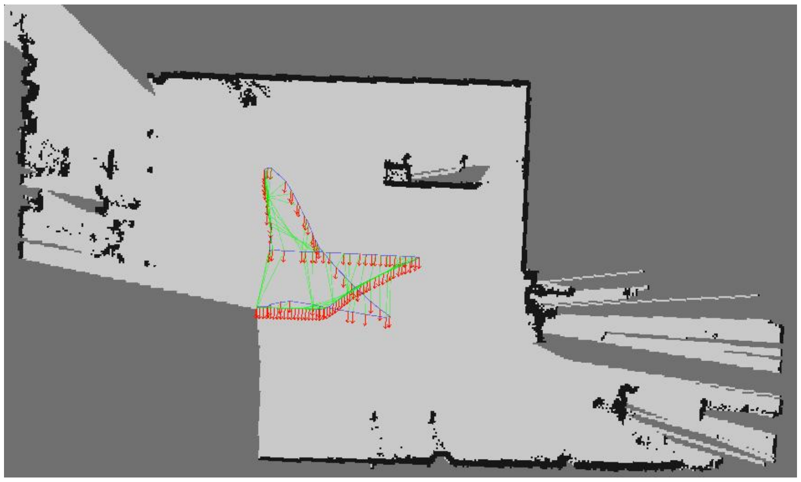


Figure 2.7: OmniMapper CSM builds a map of the Boeing lab at the MIRC building at Georgia Tech on the OmniX platform*. OmniMapper CSM was able to achieve the 1cm accuracy needed to position the platform to perform a drilling demo.

OmniMapper-CSM was used to localize the Boeing Omnix platform in Figure 2.7. In this experiment, the platform was able to determine its position within a few centimeters, which was needed to perform a drilling operation in an industrial setting.

2.5.3 OmniMapper ICP

The OmniMapper has also been extended with a plugin to use 3D iterative closest point (ICP) to build a detailed map in an arbitrary environment. The OmniMapper-CSM plugin described in section 2.5.2 is limited to indoor use because the planar ICP can only function well if the robot remains on flat ground. To support mapping arbitrary geography in outdoor navigation, a full 3D ICP mapping plugin was developed. This plugin uses the *Generalized-ICP* algorithm described in [SHT05] and provided in a *PCL* implementation.

The ICP plugin in *OmniMapper* contains two threads of computation. The first thread of

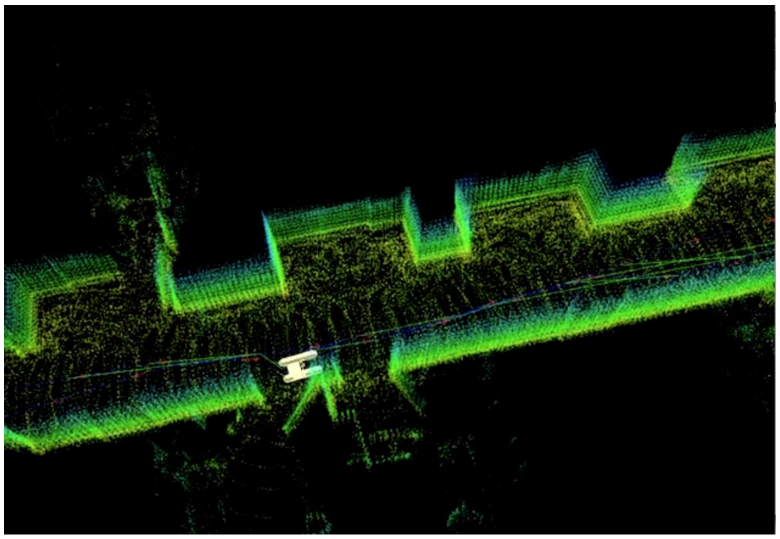
*The OmniX is a large holonomic industrial robot base. It has been augmented with Hokuyo UTM30 laser scanners for localization, mapping, and safety. The robot is also equipped with a drilling rig.

computation processes new point clouds as they are produced from sensor data. These point clouds can be produced from many types of sensors. The sensor modalities which have been used with this plugin include Hokuyo UTM30 laser scanners which are mechanically actuated via Directed Perception pan-tilt units, Kinect type 3D cameras, and Velodyne 32E 3D LIDAR scanners. The second thread of computation looks for loop closures with the most recent point cloud and other point clouds which were already used. These two threads provide pose measurement constraints which are used to compute the robot's trajectory. This trajectory information is used to render all point cloud data into one large point cloud representing the entire map.

The *Generalized ICP* algorithm described in [SHT05], and provided as an implementation in *PCL*, has a number of advantages over the vanilla ICP procedure described above. Due to limited sensor resolution, exact points in the reference cloud are rarely imaged by the new cloud. In the case of a long range sensor with limited (vertical) resolution such as the Velodyne 32E, the reference cloud points could fall in-between the new scan. These points could be up to a meter apart. Due to this effect, the errors are projected onto the surface normals to constrain their correction only in the direction of the normal. This is called the *point-to-plane* error. In Generalized ICP, both the new and the reference cloud normals are used in computing the point-to-plane error; this technique significantly improves performance as is shown in [SHT05] and empirically verified in our own comparisons.

The ICP algorithm is costly in terms of computation time. To maintain online operation, the resolution and density of the point clouds is reduced to a more manageable level. The filter which contributes the most to producing manageable point clouds is a voxel grid filter. This filter limits the density of the point cloud to one point in each 5cm voxel. This reduces the density close to the robot without affecting the long range density, where errors are more apparent and can be used to correct the robot's trajectory.

Even with reducing the density of the incoming point clouds, the ICP routine requires almost 2 seconds per point cloud. Fortunately, since the robot has a good estimate of its relative


(a) A 3D view of a small segment of hallway from OmniMapper ICP

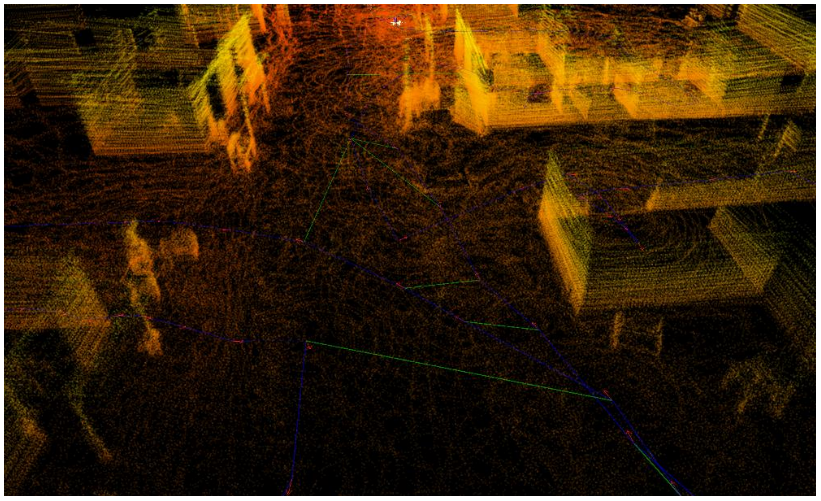
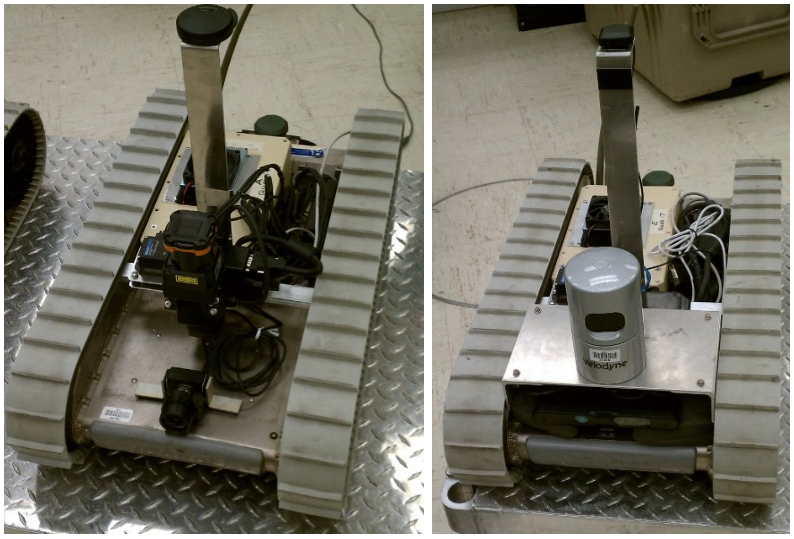

(b) Loop closure illustrated on an outdoor run. Loop closure constraints are drawn as green lines between trajectory elements which are drawn as red arrows. Relative pose constraints solved by ICP are represented with blue lines between adjacent trajectory elements

Figure 2.8: OmniMapper-ICP maps by finding relative poses in adjacent point-clouds and loop closures between distant trajectory elements.

motion from IMU and odometry, the ICP routine can be initialized close to the final solution. This allows for significant platform motion between point clouds; therefore, the robot can still travel quickly and keep track of its motion and build a map. In Figure 2.8a, a robot is seen using OmniMapper-ICP to build a 3D map of a hallway in an office building.

2.5.4 Experiments

Experiments were performed with the iRobot PackBot robot which was augmented with additional computing resources and sensors. Two of the robots used in these experiments is shown in Figure 2.9. These robots are tele-operated in these experiments; however, in other work we have automated exploration while mapping.



(a) A robot with a custom 3D tilt LIDAR.

(b) A robot with a custome Velodyne 32E 3D LIDAR.

Figure 2.9: iRobot PackBots used in these experiments. These robots have been augmented with additional sensors and onboard computing for online mapping

We have tested the featureless mapping components of OmniMapper in a variety of environments representing various operational scenarios. The first environment consists of an underground complex of long and narrow tunnels. The second environment consists of a training facility consisting of a number of buildings made to look like a small village. The third environment consists of the interior of an office building. The final environment is along a roadway between two buildings. The first environment which we mapped with the OmniMapper for these experiments consists of an underground complex of narrow tunnels. The tunnels themselves are smooth and indistinguishable; however,periodically along their length are rooms which contain machinery and salient structures and features. This test environment also contains some outdoor portions where GPS can be used to correct some drift.

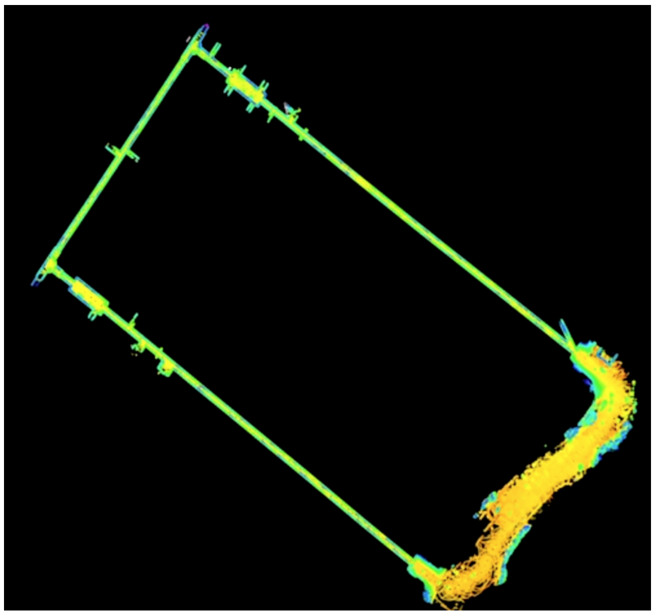


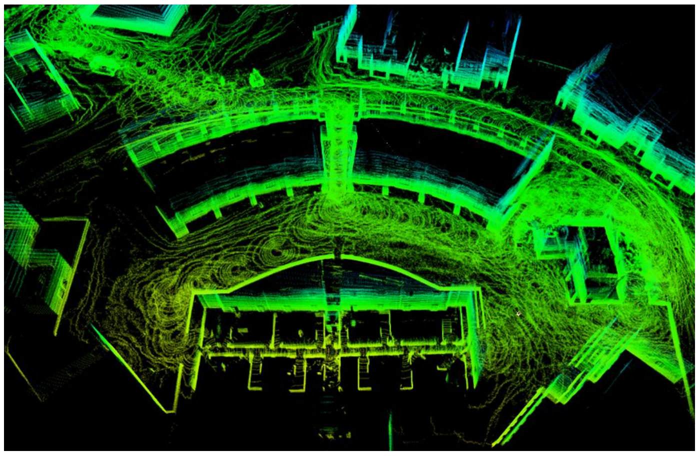
Figure 2.10: A path through an underground tunnel over one kilometer in length. The robot started outside in the fuzzy area in the bottom-right and entered the upper tunnel, proceeding in a counter-clockwise direction. GPS measurements (GPS was sufficient to correct the trajectory enough that ICP was able to close the loop) helped make this loop closure as the robot exited the lower tunnel.

The results from this experiment can be seen in Figure 2.10. In this test run, the robot starts outside in the area shown in the middle-right and proceeds into the tunnel in the top of the image. The robot proceeds around the tunnel complex and exits at the bottom of the image back to the outside. At this point, the robot has accumulated approximately 5 meters of error in its map due to the lack of good constraints in the narrow tunnels. The robot receives a GPS lock which adds a few more constraints to its pose. Once these additional constraints are available, the pose of the robot is corrected to within one meter of error. At this point, the loop closure mechanism is able to identify locations of structure which have been previously observed and corrects the remaining map error. The robot then re-enters the tunnel complex and repeats the mapping procedure down the tunnels. This experiment demonstrated the usefulness of the GPS plugin in the mixed indoor/outdoor mapping application. GPS has some poor performance especially in the transition between outdoor to an indoor environment due to multipath reflections of GPS signals, so its information can only be used with an appropriately weak noise model. Therefore, it takes a series of observations to leverage the information provided by GPS to correct the map. The accuracy of GPS information is also not sufficient to perform mapping alone, but in this experiment it has been shown to provide the constraints needed to guide the ICP process to correct the map.

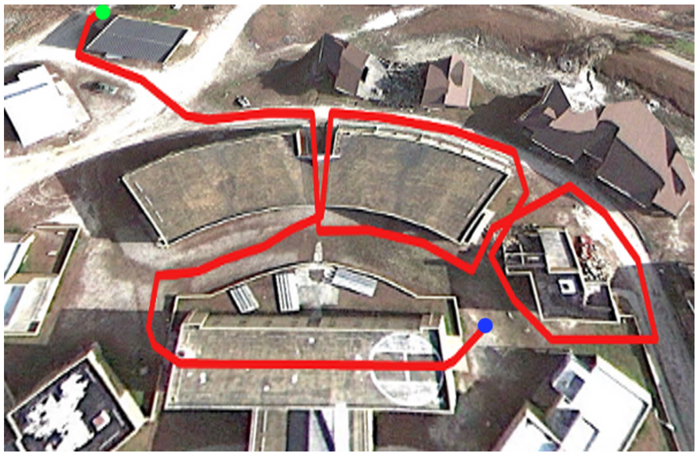
The second experimental environment is a training facility built to resemble a small village. In this experiment, the robot maps the interior and exterior of several buildings and moves across a variety of terrain. The results from one of the runs of this experiment can be seen in Figure 2.8b.

The output from the mapping run can be seen in 2.11a and approximate ground-truth overlaid on an aerial image can be seen in Figure 2.11b. In this test run, the robot starts in the area in the upper-left of the image indicated by the green dot. The robot proceeds down the road and turns to pass under an overpass in the curved building. The robot makes a path around another building and re-visits the overpass again via a second path. The robot then enters the building at the bottom of the image from the left side and exits it from the right side having built a map of its interior. The end of the run is indicated by a blue dot. Aerial imagery is provided for a reference with the robot's approximate trajectory indicated in red in Figure 2.11b. From this view it can be seen that ground robots can provide additional detail not available from aerial surveillance, such as side profile information such as window and door locations. Additionally, ground robots can enter some buildings to yield high quality tactical situational awareness which might be hard to get from the air. These advantages can be seen in Figure 2.11b.

A variety of additional test runs were performed at the same training facility. An additional run is presented in Figure 2.13a. This run is entirely outdoors and covers a larger area of the training facility. The output from the mapping procedure can be seen in Figure 2.13b. In this test run, the robot starts in the area at the bottom-right denoted by a green dot in the approximate



(a) Optimized map generated by 3D ICP on a ground robot featuring mixed indoor and outdoor elements



(b) Approximate ground truth drawn on an overhead view. *Image courtesy of GoogleEarth*

Figure 2.11: An experiment where a robot examines the interior and exterior of a series of buildings at a training facility.

ground-truth aerial image shown in Figure 2.13b. The robot then proceeds to the village in the upper-left and makes a loop along the main road, rejoining its previous trajectory at the blue dot.

The third experimental environment is an outdoor scene along the road between two buildings approximately one kilometer apart. The results from this experiment can be seen in Figure 2.12. OmniMapper-ICP is used together with the GPS plugin for very large outdoor mapping. In this experiment, the robot proceeds from the exterior of one building, along a roadway flanked by trees, to the exterior of another building. This experiment made use of

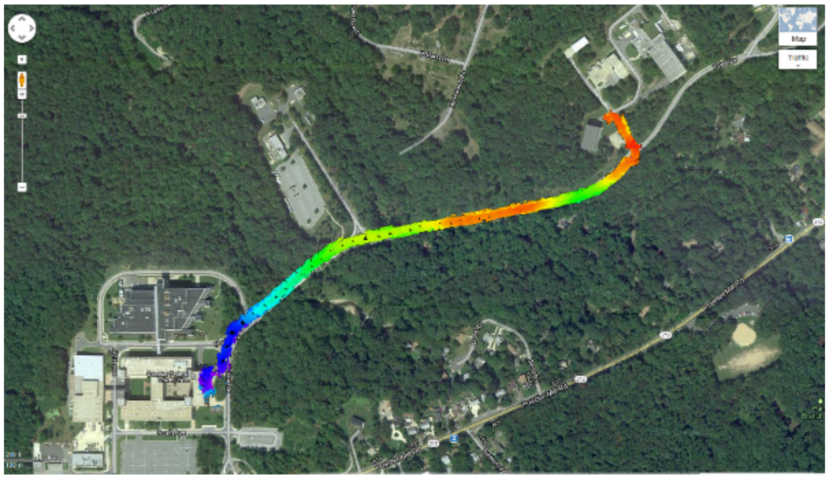
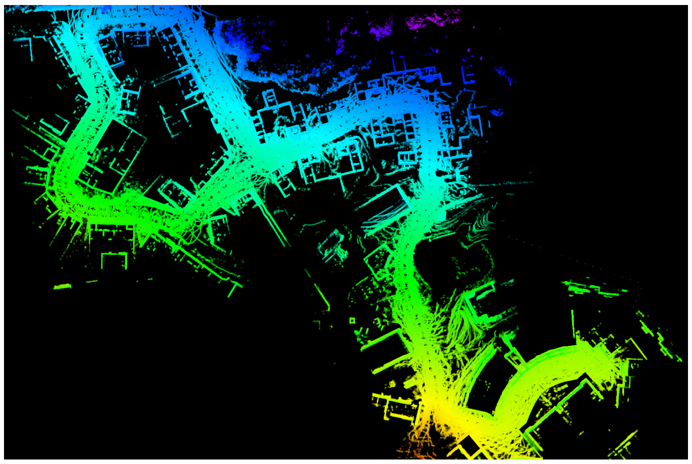


Figure 2.12: A robot uses Omnimapper-ICP together with the GPS plugin to map an outdoor route. Robot point cloud data is shown along optimized trajectory overlaid on Google Maps.

3D ICP measurements of the environment, including more distant trees. In addition to the ICP measurements, this run made use of GPS measurements to provide additional constraints to correct the map. GPS measurements come with significant noise which requires an equally weak noise model to incorporate them into a map. When multiple GPS measurements are incorporated over a long distance into a map via the OmniMapper, they can be helpful in correcting the map and overcoming the noise inherent in GPS measurements. The image shown in Figure 2.12 was generated by manually aligning the resulting mapped point cloud data with an image taken from Google Earth.

The final experiment is to map the interior of a large office building. This is a scenario which would also have been possible with a feature-based mapping technique such as lines or planes. The versatility of featureless mapping techniques allows this structured environment to be mapped successfully in addition to unstructured environments.

The final map built from this environment can be seen in Figure 2.14. This office environment provided significant features and texture to enable robust ICP matching of adjacent robot poses. Due to the presence of constraining measurements in the environment, the trajectories had minimal error before loop closure. The loop closure routine was able to identify multiple


(a) Optimized map generated by 3D ICP on a ground robot featuring mixed indoor and outdoor elements over a longer test run.

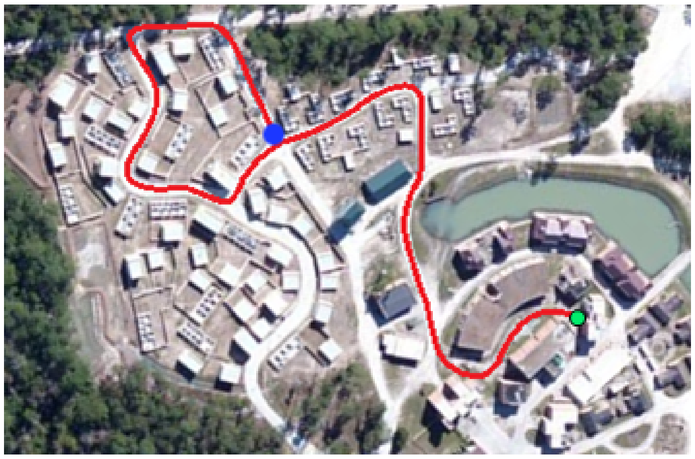

(b) Approximate ground truth drawn on an overhead view. *Image courtesy of GoogleEarth*

Figure 2.13: Another experiment where a robot examines the interior and exterior of a series of buildings at a training facility.

loop closure locations and correct any remaining error.

2.5.5 Discussion

The ability to build a model of the environment is critical to enable mobile robots to provide tactical situational awareness to first responders and military personnel in a variety of hazardous missions. To enable operation in realistic scenarios such as arbitrary cluttered

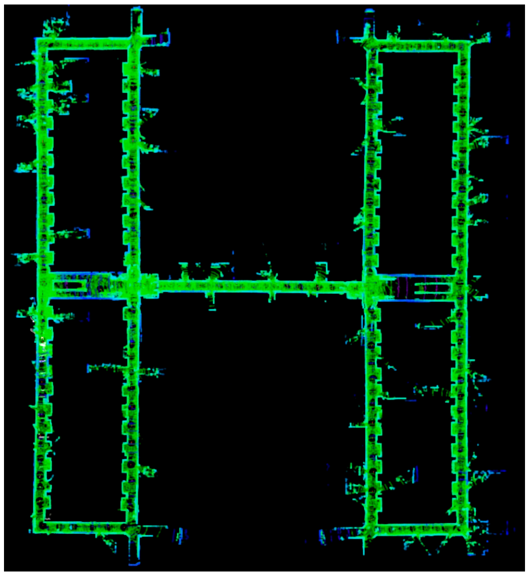


Figure 2.14: OmniMapper-ICP run on an indoor office environment. Measurements are made by a Velodyne 32E 3D laser scanner from a ground robot

environments or even disasters like collapsed buildings after an earthquake, these robots can use current state-of-the-art sensors to build maps of these complex environments. This type of situational awareness can enable planning by rescue and military personnel for operations; it can also enable autonomous mobile robot mission execution. We have demonstrated a system, dubbed OmniMapper, that can be used by a mobile robot to build a map of large-scale mixed indoor and outdoor environments. This system has been tested in a variety of challenging environments combining indoor, outdoor, and mixed elements to examine performance across the spectrum of operations that will be expected of future robot systems. This system works by analyzing point cloud data from powerful 3D sensors, such as the Velodyne 32E, to determine the robot's precise motion in the environment. The sensor data is then rendered at this optimized trajectory, resulting in a high-quality 3D representation. With the prevalence of UAV's to provide broad situational awareness, overview information is currently available to support operations. Due to high altitude flying, this situational awareness does not extend to the interior of buildings or even to low-angle viewing of the side of smaller buildings. This type of information can be provided by ground robots using these types of sensors. Micro aerial platforms could also be used for this purpose.

2.6 Effects of sensory precision on map performance

The current equipment needed to build a domestic service robot is too expensive for the average family to afford. Recent developments in sensor technology such as the Microsoft Kinect 3D camera have been revolutionary in delivering a new sensor modality at a commodity price. Current research sensors such as LIDAR systems are very precise and expensive units. This study will address how feature based mapping techniques using *OmniMapper* can allow for the use of less precise, and therefore cheaper, sensor equipment on mobile robots.

Making robots cheaper also enables new uses in dangerous environments such as disaster sites or urban counter-insurgency operations. Simultaneous Localization and Mapping (SLAM) techniques can be used to build maps which can be shared with emergency workers or soldiers as robots complete exploration missions [BBS10]. Projects such as the Micro Autonomous System Technologies Collaborative Technology Alliance (MAST CTA) [ARL06] are working to develop techniques and components to enable robot mapping with low power and small form factor robots and sensors. MAST's mission is to develop robots which are autonomous and collaborate to provide situational awareness in urban environments for military and rescue operations. MAST platforms are required to have a small form factor and long operational availability, so we must understand where compromises can be made on sensor performance.

There is a need to determine what level of performance is required from sensors to enable SLAM techniques to deliver a map of sufficient quality from autonomous exploration of an unknown environment. Sensors such as laser scanners are appropriate in laboratory applications but other sensory modalities such as radar or vision may be preferred when considering cost levels, power usage, and size. Some of the accuracy, density, annular confinement, and field of view assumptions which are common with laser scanners might no longer be valid with these other sensory modalities. Low cost radar based scanners are currently under development as part of the MAST project; until these are available, we will simulate the performance of radar in software using data from laser scanners. To investigate the effects of various properties of a sensor on the mapping result, we use high quality laser data and degrade properties such as the maximum range, angular resolution, field-of-view, or range accuracy. We call this *de-featuring* the data. The performance requirements for robot mapping must be determined to guide the development of these new miniaturized radar scanners to ensure that they will work well for robot mapping tasks.

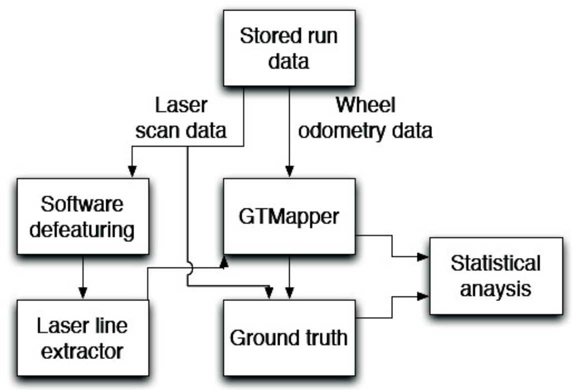
An objective benchmark for comparing SLAM algorithms was reported in [Bur+04]. This benchmark compares the incremental relative error along the robot's trajectory instead of comparing maps. This allows the authors to directly compare the results between different algorithms or sensor types. Ground truth is generated by manually aligning laser scans using ICP as an initial guess. We use the relative pose error metric and ground truth generation method from this benchmarking paper to assess SLAM performance levels under various sensor configurations.

Research has been reported on analysis of localization performance and sensing characteristics. O'Kane and Lavalle proved that it is possible to localize in a known map with a robot equipped with only a contact sensor and a compass in [OL05] and [OL07]. The authors determined that it is impossible to localize a robot in a known map if the compass is replaced with an angular odometer. In the SLAM domain, work has been done to support simpler and less expensive sensor modalities. Bailey developed techniques for delaying initialization of landmarks in bearing-only SLAM in [Bai03]. A paper by Müller *et.al.* relates the performance of their 6D SLAM algorithm to the precision of their 3D laser scanner in [Mül+06].

The feature detectors developed for this mapping application have been designed to be robust to noise and performance degradation. We have chosen to extract line segments from laser range data using the RANSAC technique. We have selected parameters such as minimum line length and maximum gap which correspond to the size of walls and door openings in a typical office environment.

The experiments in this section were run on data collected from two types of mobile robots and were processed offline to compare mapping performance with different sensor de-featuring. Though these techniques were compared in offline operation, it should be noted that the mapper can comfortably run online on a modest CPU.

2.6.1 Experimental Design



(a) The experimental setup

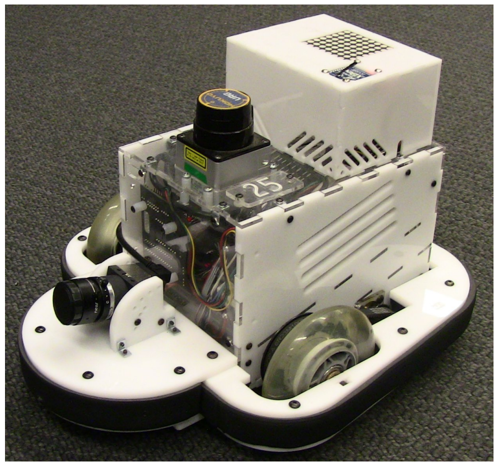

(b) The Scarab mobile robot developed at the University of Pennsylvania

Figure 2.15: Experimental setup and equipment used to gather data for robot experiments

The operation of the experimental framework is shown in Figure 2.15(a). Ground truth was generated by manual alignment of the full unaltered laser scan data in an initial run. This serves as an adequate estimate of ground truth for the purpose of this relative comparison of path trajectory error with different sensor performance levels. It allowed the operator to correct the (x, y, θ) between subsequent poses so that the laser scans came into alignment. This information was then captured in a ground-truth file where it was compared to the posterior relative displacements in the test run trajectories.

Stored robot data was replayed with various settings of laser scan data de-featuring. The laser line extractor extracted line segments from the de-featured laser data and sent them to the mapper. As shown in figure 2.16, once the run was completed, the robot trajectory from the mapper was compared to the ground truth for this run. The average incremental absolute error was then recorded.

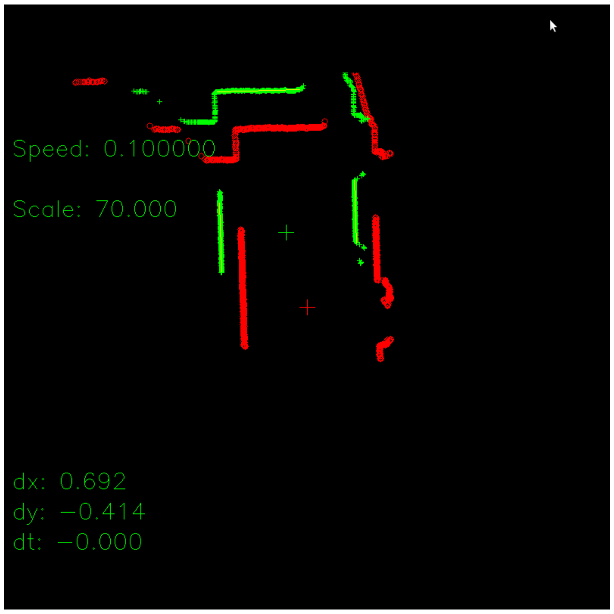


Figure 2.16: The alignment tool used to generate ground truth trajectories. Poses illustrated in this figure are displayed at a poor solution to illustrate operation.

2.6.2 Experiments

The first source of sensor data used in these experiments was a series of logs taken from the *Scarab* robots developed at the University of Pennsylvania. An image of the Scarab robot can be seen in figure 2.15b. The Scarab is a differential drive mobile robot equipped with a Hokuyo URG laser scanner. In this series of experiments, the robot was tele-operated for ten runs in an indoor office environment to collect data which is used for offline processing.

The laser scanner accuracy was de-featured for our analysis to determine what level of performance in the mapping task can be expected with progressively simpler (and therefore less expensive and lower power) sensors. All sensor values were captured during the data collection phase and were de-featured with software in the post-processing phase.

In the first round of experiments, several de-featuring modifications were made to the laser scanner readings. First, the range of the scanner was restricted. Second, the angular precision of the laser scanner was limited by omitting beams from the line extraction process. Thirdly, the angle subtended by the laser scan was reduced. Finally, the range accuracy of the sensor was corrupted by varying degrees of Gaussian noise. This series of experiments will establish which parameters of the scanner can be varied while still yielding sufficient performance in the mapping task.

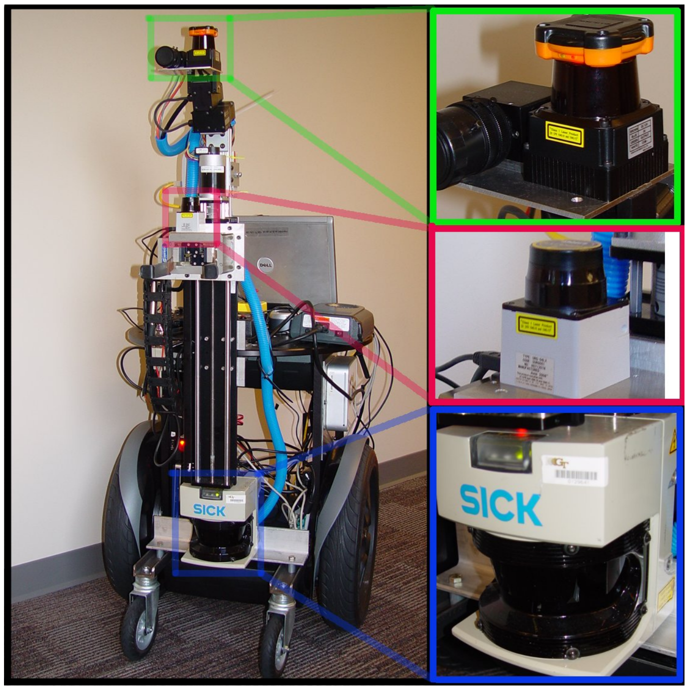
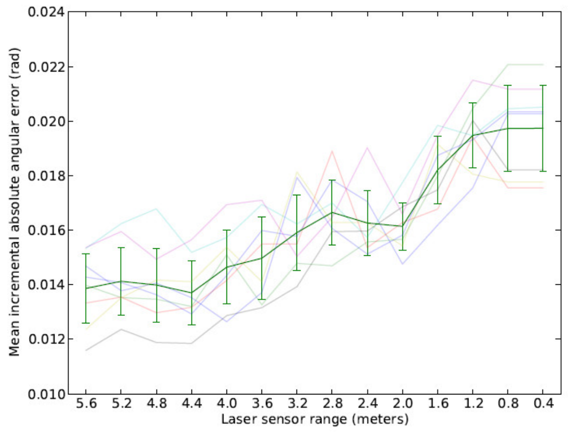


Figure 2.17: The Georgia Tech robot "Jeeves". Right-top(green): Hokuyo UTM-30 laser scanner.Right-middle(magenta): Hokuyo URG laser scanner. Right-bottom(blue): SICK LMS291 laser scanner.

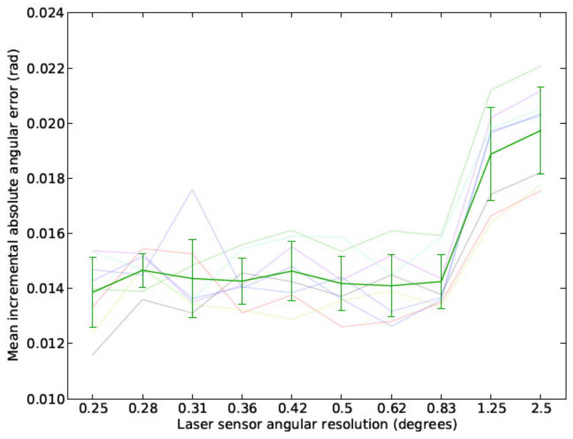
In the second experiment, the Georgia Tech robot "Jeeves" was outfitted with three different laser scanners to collect a run where each laser can be used to compare the mapping results. The robot is a Segway RMP 200 modified to be statically stable, as seen in figure 2.17.

This platform is configured to simultaneously collect data from the SICK LMS291, Hokuyo URG, and the Hokuyo UTM 30 laser scanners, all of which are seen in the right of Figure 2.17. This data is used individually in the mapping process and the resultant relative errors are compared. In this experiment, there is no need to de-feature the performance of the laser scanners in software since each exhibits a unique price and performance level.

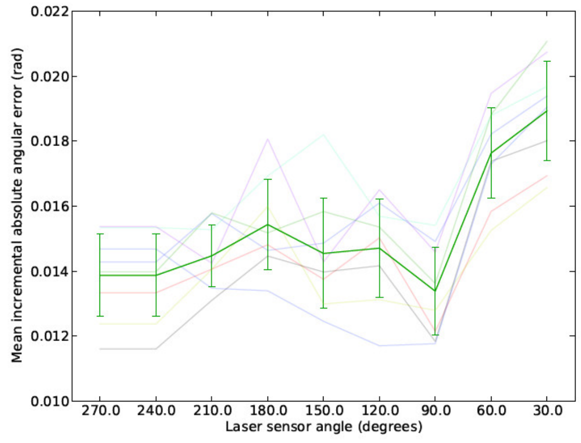
2.6.3 Results



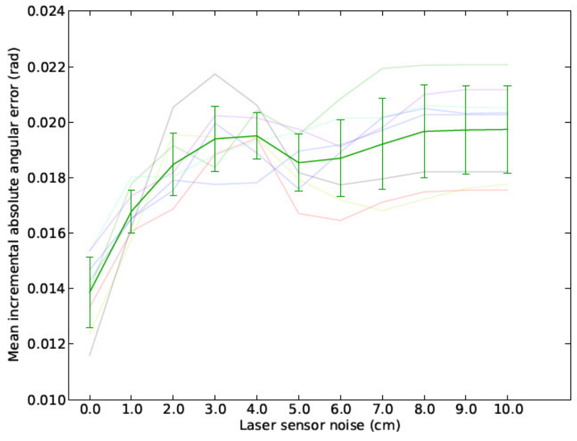
(a) Maximum range restricted between 0.4m to 5.6m



(b) Angular resolution restricted between 0.25° to 2.5°



(c) Angle subtended by laser restricted between 270° to 30°



(d) Gaussian noise added with variance up to 10cm

Figure 2.18: Mean incremental heading error with various types of sensor *defeaturing*

In the first set of experiments, Scarab robots were driven along a similar trajectory in an office building 10 times. Two of the runs were of poor quality due to wheel slip on a thick concrete seam and were eliminated from the analysis. In Figure 2.18(a) the mean trajectory error is displayed from a series of test runs where the maximum range of the laser scanner was varied from 5.6 meters down to 0.4 meters. 5.6 meters is the maximum range of the Hokuyo URG laser scanner. It can be seen from this figure that the performance is poor when the range is restricted to below 2 meters, but it rapidly improves at longer ranges to a nearly constant level of performance. The residual incremental error (due to the inaccuracy in the odometry of the robot) is reached as the range is restricted to 0.8 meters and below. This experiment indicates that the accuracy of the mapper can be maintained until the range of the laser scanner is restricted to about 1.6 meters, roughly the width of the hallways in the test site.

In Figure 2.18(b) the mean trajectory error is displayed from a series of test runs where progressively higher proportions of the laser beams are left out of the line extraction process. This test is meant to simulate a sensor with less angular resolution than the Hokuyo URG. It is apparent from the graph that until the resolution is lowered to around 1 degree, the mapper is able to deliver a similar level of performance to the results from the fully featured laser scan. This corresponds to throwing away 75% of the data coming from the laser scanner. When the resolution is lowered below 1 degree, the accuracy drops to the level of the robot odometry alone.

In Figure 2.18(c) the mean trajectory error is compared with various angles viewed by the laser scanner. The default Hokuyo URG laser scanner can view 270 degrees. This metric is meant to simulate a sensor which views a smaller angle. The viewing angle is always faced towards the front as it is expected that the sensors would need to observe this region in order to avoid hitting obstacles during autonomous operation. This graph suggests that the mapper remains accurate until as little as 90 degrees is viewed from the laser scanner. This is probably due to the fact that in this test the range was kept at its maximum value of 5.6 meters, so the mapper was still able to see wall line segments. If the range was restricted while the angular view was also restricted, then

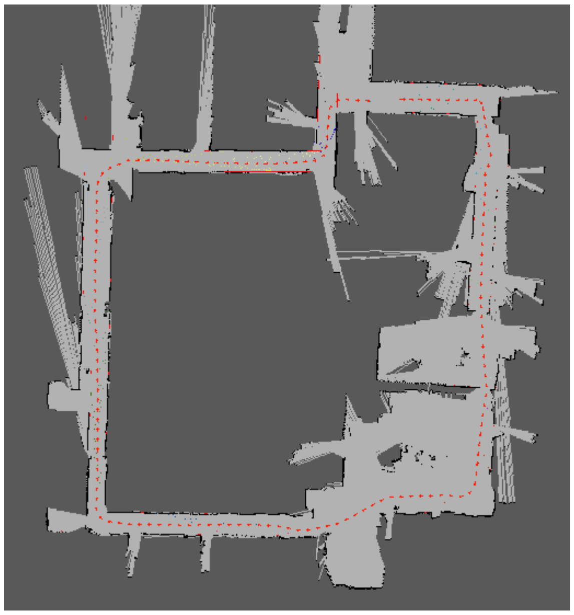


Figure 2.19: The map of the the corridors in our lab. There are two large storage rooms in the lower right. The map is shown as an occupancy grid only for display purposes – all measurements are made based on wall features.

we would expect poor performance.

In Figure 2.18(d) the mean trajectory error is displayed from a series of test runs where Gaussian noise is added to the range measurements. The noise is sampled from a Gaussian distribution with the variance indicated along the x-axis. It can be seen from this figure that with even a moderate 2 centimeter noise is added to the laser scan significant error is introduced into the map. The mapper is particularly susceptible to noise in the laser ranges, though it should be noted that this noise level is significantly higher than in a typical laser scanner.

In the second experiment, mapping results from three different laser scanners which are commonly used by the SLAM research community were compared. An example map from this run is shown in Figure 2.19. The robot successfully closed the loop with each of the laser scanners. The absolute relative incremental position and angular errors can be seen in Table 2.1. It can be seen with these results that the SICK and URG mapping results are similar in displacement error, whereas the UTM30 is much more accurate in recovering the relative displacements. This is to be expected since the UTM30 slightly outperforms the SICK 291 laser scanner in angular resolution, range accuracy, and speed.

Table 2.1: Absolute incremental position and orientation error for test comparing performance of mapping with three typical laboratory laser scanners

Error	URG	SICK	UTM-30
Position error	0.020162	0.020942	0.009785
Angular error	0.009924	0.006612	0.007033
Resolution	$1°$	$1°$	$0.25°$
Range	5.6m	80m	30m
Range Error	1%	$\pm 35mm$	$\pm 30mm$
Angle	$240°$	$180°$	$270°$

2.6.4 Discussion

The longer range of the laser scanner does not appear to improve performance in the office environment. This indicates that the line extraction algorithm is working well when even a small amount of the wall can be seen from the scanner, so the range of the scanner doesn't need to be much more than the width of the hallway in order to achieve good performance. We believe that in general the range of the scanner needs to be no more than such that it can see about 1 meter of the wall from the middle of the hallway.

We were surprised to find that the mapper was so susceptible to range accuracy given that our line extraction algorithm performs a least-squares fit to many laser points and should be able to handle zero-mean noise. It should be noted that the noise levels explored here were relatively large and that the line extraction performs very well for noise levels more typical of laser scanners. We believe that, with additional parameter tuning, the line extractor could be made to perform well with more noise; so this technique should translate well to radar scans.

The angular resolution parameter is effectively reducing the range of the scanner slightly because the missing resolution is causing gaps in the extracted lines beyond a certain range. This prevents long lines from being extracted; however, we have already determined that range much longer than the hallway width is not very important in the range test. We have now also determined that high angular resolution is not needed to perform the mapping task well.

For the second experiment with the robot “Jeeves”, we notice that the incremental angular error is comparable between the SICK and UTM30 laser scanners, with the URG performing slightly worse. We believe this is because the office environment for the second experiment has larger, cluttered rooms for which the limited range would often result in few measurements of the walls with the URG. The incremental displacement error seems to show that the UTM30 is performing better than the SICK laser scanner. In our first experimental analysis we determined that the mapping results are particularly susceptible to range accuracy, but the The UTM30 claims a $\pm 30mm$ range accuracy whereas the SICK claims a $\pm 35mm$ range accuracy. There is only 5 millimeters difference here so this is likely not the main cause. It is more likely that the extended viewing angle combined with the longer range of the UTM30 is allowing the robot to observe both the walls behind it as well as those in front of it for a significant portion of the run. The SICK can only see in front of the robot and is therefore only correcting based on the walls in front – it cannot see the wall behind the robot as it passes through a doorway. We think that this effect was not seen in the first experiment with the Scarab robots because they were operated in an environment with narrow corridors and few larger rooms. The Scarab robots never had the opportunity to pass through doorways and therefore could not benefit from this effect.

Based on these results, we determined that the Hokuyo UTM30 is the ideal laser scanner for use in environments with larger open spaces. It has additional advantages over the LMS291 laser scanner, namely size and power usage. In environments with more confined spaces, the Hokuyo URG performs well enough to make useful maps.

2.7 Simultaneous Calibration, Localization and Mapping

Introduction

Autonomous robots depend on high quality extrinsic calibration of their sensors to reliably perform their tasks. As a motivating example, one of the tasks typically performed by mobile robots is inferring the structure of the environment using sensor measurements. Given a representation of the world, the robot could generate a plan of actions to accomplish a given goal like navigation. Performance of autonomous navigation by mobile robots can be greatly affected by the quality of extrinsic calibration of the utilized sensors. A mobile robot is often assigned the task of performing inference from sensor measurements, in order to build a model of the surrounding environment and to estimate variables of interest. Moreover, it has to generate a plan of actions to accomplish a given goal, such as exploration or navigation.

We study the self calibration problem to autonomously estimate the extrinsic parameters of the sensors. Simultaneous Calibration Localization and Mapping problem attempts to jointly estimate the location of landmarks, trajectory of the robot and the extrinsic parameters of the sensors on the robot. We exploit the structure of this problem to greedily plan a path that would reduce the uncertainty of the extrinsic parameters of the robot. Figure 2.20 shows the output of such a system where the calibration parameters, trajectory and landmarks are jointly estimated.

In this case, the robot's goal is to self-calibrate the extrinsic parameters of the sensors before performing any autonomous task. The variables of interest that are estimated are the extrinsic transform of the sensors to the base of the robot, the poses of the robot and the landmarks in the world in relation to each pose. The motivation is to reduce the amount of manual labor required and eliminates the human error introduced during calibration by offloading the task to the robot itself.

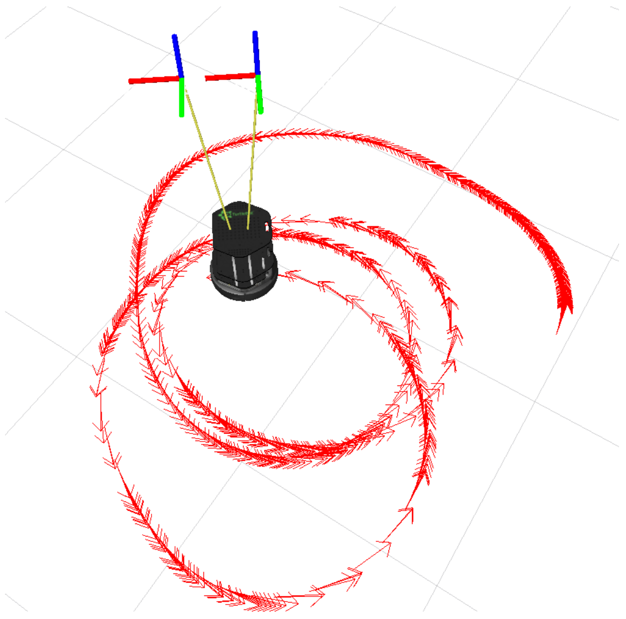


Figure 2.20: The figure above shows the robot's estimated trajectory and the two transforms represent the ground truth transform and the estimated sensor transform with respect to base at that time.

Often a human operator is given a task with easy objectives such as repeatedly aligning the robot's field of view with the calibration grid, a process that could be sped up through automation. This would also allow the robot to be robust to sensor alignment changes between different experiments due to either intentional or accidental reconfigurations.

We propose a novel algorithm that allows the robot to perform a self-calibration routine through autonomous investigative movements around a previously unknown environment that would then enable the robot to autonomously perform other tasks. The main contribution of this paper is to show the possibility of a generalized, multi-modal approach to autonomously calibrate extrinsic parameters of available sensors on the robot by leveraging a greedy planner to quickly estimate the extrinsic sensor transform. This would greatly reduce the amount of time and effort required to setup an autonomous robot. We further demonstrate the utility of planning towards the application of simultaneous localization calibration and mapping.

Approach

As the robot moves in an unknown environment, it continuously maps using the simultaneous localization and mapping (SLAM) pipeline presented by Trevor et al. [TRC14b]. However in contrast to the known extrinsic sensor calibration as assumed in general, we model the extrinsic calibration as an unknown parameter and estimate its uncertainty along with other unknown parameters like landmarks and poses. At each instant the robot plans a trajectory which actively reduces the uncertainty of extrinsic calibration parameters. As a result, the robot autonomously calibrates its sensor by actively moving in the direction of the maximum uncertainty reduction.

Our approach:

- Does not rely on human input
- Requires no prior knowledge of the world
- Need not be provided engineered patterns or environmental infrastructure
- Supports asynchronous sensor measurements
- Intended for calibration and help
- Resulting solutions not only provide relative sensor extrinsics, but also provides an confidence for each
- Performs simultaneous localization calibration and mapping while actively reducing the sensor transform uncertainty

In order to perform simultaneous calibration, localization and mapping we explicitly represent the additional calibration parameters in the joint probability model as

$$P(X,L,Z,\Omega,\Phi) = P(x_o)\prod_{i=1}^{M} P(x_i|x_{i-1},u_i,\Omega)\prod_{k=1}^{K} P\left(z_k|x_{ik},l_{jk},\Phi\right) \tag{2.24}$$

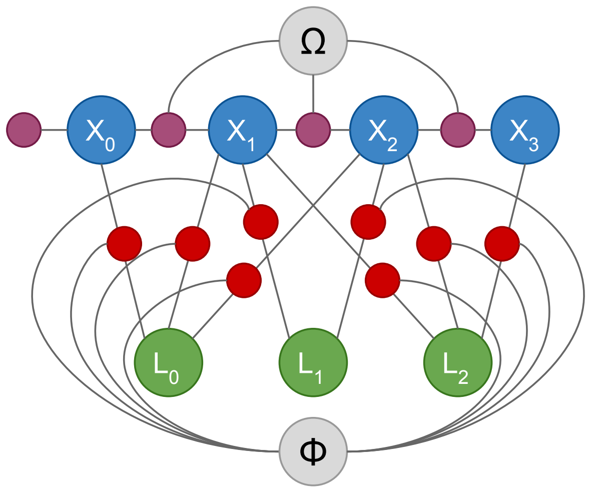


Figure 2.21: Factor graph representing the Simultaneous Calibration, Localization and Mapping problem. Gray circles represent the additional calibration parameters (Ω,Φ) which are connected to other factors resulting in joint optimization over landmarks, poses and calibration parameters

where Ω represents odometry bias parameters and Φ represents the extrinsic transformation between the robot base and the corresponding exteroceptive sensor which has to be calibrated. Figure 2.21 shows the corresponding factor graph. Each sensor will have a different extrinsic transformation Φ but for simplicity we use a single Φ corresponding to a single sensor. Optimization can be done in a similar manner as shown in the previous section with additional variables representing the extrinsic calibration Φ and odometry bias Ω. The resulting non-linear optimization is $\Theta^* = \arg\min_\Theta \sum_{i=1}^{M} \| f_i(x_{i-1}, u_i, \Omega) - x_i \|^2_{\Lambda_i} + \sum_{k=1}^{K} \| h_k(x_{ik}, l_{jk}, \Phi) - z_k \|^2_{\Sigma_k}$

2.7.1 Autonomous Extrinsic Calibration using Active Planning

We use active planning in belief space for autonomous extrinsic calibration of an exteroceptive sensor. Planning is done such that it actively reduces calibration parameter uncertainty at each instant in a previously unknown environment. To achieve that end, we integrate simultaneous calibration, localization and mapping with an active planning framework resulting in *planning for calibration*.

In order to compute the optimal control action over the l look ahead steps, we compute

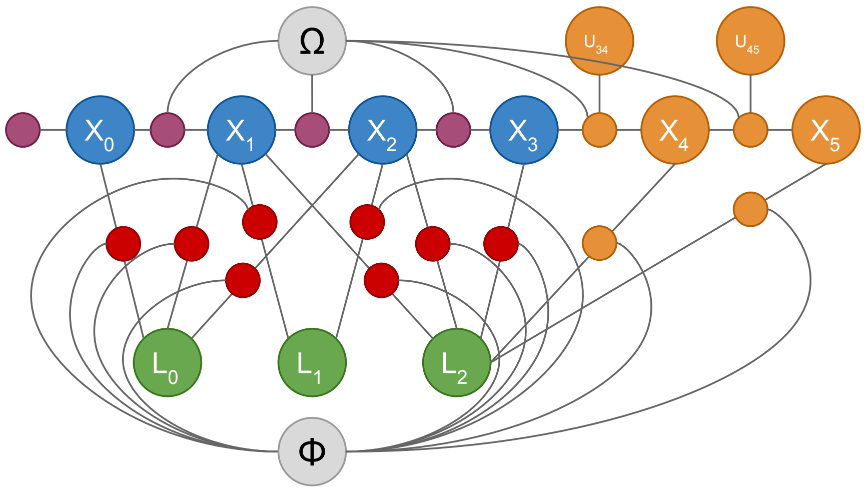


Figure 2.22: Factor graph representing the Planning for Calibration problem. Orange circles to the right represent the future control inputs and predicted poses. Maximum likelihood observations are represented by small orange circles.

the predicted belief over the time horizon. However we don't know the observations $Z_{M+1:M+l}$ ahead of time, given M is the current time step. We assume the maximum likelihood observation as future observation given the current belief over landmarks and poses and future control inputs. Modeling future observations as unknown has insignificant increase in efficiency [VPA12]. We also assume that the number of landmarks do not change when we predict the belief over the time horizon.

Given the estimated extrinsic calibration parameter Φ_M and odometry bias Ω_M, estimated poses and landmarks up to time M, current observations $Z_{1:M}$, control inputs $U_{M+1:M+l}$, and the corresponding predicted observations $Z_{M+1:M+l}$, the predicted calibration belief Φ_{M+l}, Ω_{M+l} at a future time step l is given as:

$$gb(\Phi_{M+l}, \Omega_{M+l}) = P(\Phi_{M+l}, \Omega_{M+l} | X_{1:M+l}, L_{1:M+l}, Z_{1:M+l}, U_{M+1:M+l}) \tag{2.25}$$

Figure 2.22 shows the corresponding factor graph. The control action minimizes the general objective function $J(U_{M+1:M+l})$ over l look ahead steps

$$J(U_{M+1:M+l}) = c_l(gb(\Phi_{M+l}, \Omega_{M+l}))$$

where c_l is defined as the determinant of predicted joint covariance of the calibration parameters Φ_{M+l}, Ω_{M+l}. At each step the general objective function selects the control action that results in the minimum predicted uncertainty of the calibration parameter. In turn, the resulting planning algorithm will ensure that the robot moves in a direction that will reduce the associated calibration uncertainty. Given the control action, the robot is moved in the corresponding direction followed by re-estimation of landmarks and pose beliefs. Using the new added poses and landmarks, we again estimate the predicted belief over the next l look ahead steps.

This algorithm is run until the change in the uncertainty determinant of the calibration parameters is below a certain threshold or the maximum number of steps are taken. Algorithm 1 summarizes the process.

Monte-Carlo Sampling

To select the control action that minimizes the objective function $J(U_{M+1:M+l})$ we perform Monte-Carlo sampling where we randomly select sequence of control actions from among a discrete set of control action sequences and estimate the corresponding value of the objective function. The control action which results in the minimum value of the objective function is taken as the optimal control action. Algorithm 2 summarizes it.

```
1: ${X, L} \leftarrow$ Initialize by taking random step
2: **repeat**
3:     Select control action that minimizes the objective
       function: $\mathbf{U} = \min_{U_{M+1:M+l}} J(U_{M+1:M+l})$
4:     Move the robot in the corresponding direction.
5:     Re-estimate the pose and landmark beliefs
6: **until** stopping criterion is met
```

Algorithm 1: Active Planning based Extrinsic Calibration

We use random sampling in the discrete control space, as opposed to using a gradient descent in continuous space to reduce the probability of being stuck in local minima and to allow for better back-tracking.

```
1:  $V_{min} \leftarrow \infty, U_{min} \leftarrow \emptyset$
2:  **for** $K$ iterations **do**
3:      $U \leftarrow$ Select random sequence of controls $U_{M+1:M+l}$.
4:      Forward simulate the robot poses given the controls.
5:      Add Maximum-Likelihood observations.
6:      Generate predicted graph $g(\Theta)$ and estimates.
7:      $V \leftarrow$ Estimate the value of objective function $J(U)$.
8:      **if** $V \leq V_{min}$ **then**
9:          $V_{min} \leftarrow V$
10:         $U_{min} \leftarrow U$
11:     **end if**
12: **end for**
13: **return** $U_{min}$
```

Algorithm 2: Monte-Carlo Sampling(X,L)

Uncertainty Prediction

Instead of re-optimizing the trajectory in order to evaluate the objective function $J(U)$, we use an approximation to predict the joint covariance of the calibration parameters required for the evaluation. Future pose estimates are predicted given the latest pose and control inputs.

We linearize the current graph around the current estimate and future pose estimates to generate the Jacobian A and the corresponding Hessian $A^T A$. In order to compute the covariance of Ω, Φ, we eliminate rest of the variables (non-calibration parameters: X, L) from the Hessian $A^T A$ using QR factorization and evaluate the determinant of remaining matrix corresponding to probability distribution $P(\Omega, \Phi)$ as shown in Equations 2.26 and 2.27. Algorithm 3 summarizes it.

$$|\Sigma|^{-1} = |A^T A| = \left| \begin{bmatrix} R & T \\ & A_{\Omega,\Phi} \end{bmatrix}^T \begin{bmatrix} R & T \\ & A_{\Omega,\Phi} \end{bmatrix} \right| \tag{2.26}$$

$$P(X,L,\Omega,\Phi) = P(X,L|\Omega,\Phi)P(\Omega,\Phi) \tag{2.27}$$

The algorithm 3 plans a trajectory that minimizes the calibration uncertainty. The resulting plan will ensure that the robot moves in a direction that actively calibrates the sensor transforms.

```
1: $G \leftarrow$ Linearize $g(\Theta)$ around $\hat{X},\hat{L},\hat{\Omega},\hat{\Phi}$
2: $(R, A_{\Omega,\Phi}) \leftarrow$ Eliminate $X, L$ from $G$
3: $logdet \leftarrow \ln|A_{\Omega,\Phi}A^{-1}_{\Omega,\Phi}| = -2 \times \ln|A_{\Omega,\Phi}|$
4: return $\exp(logdet)$
```

Algorithm 3: Uncertainty Prediction($g(\Theta)$,$\hat{X}$,$\hat{L}$,$\hat{\Omega}$,$\hat{\Phi}$)

2.7.2 Experimental Results

In this section, we describe the experimental setup, define the metrics used for evaluation and show the results.

Setup

We evaluate our algorithm in a simulated environment with known data association and ground-truth parameter values and real environment using a robot platform. In rest of the experiments we assume that odometry bias (Ω) is known since the focus of this paper is to perform extrinsic calibration of an exteroceptive sensor. However this algorithm can be easily extended to

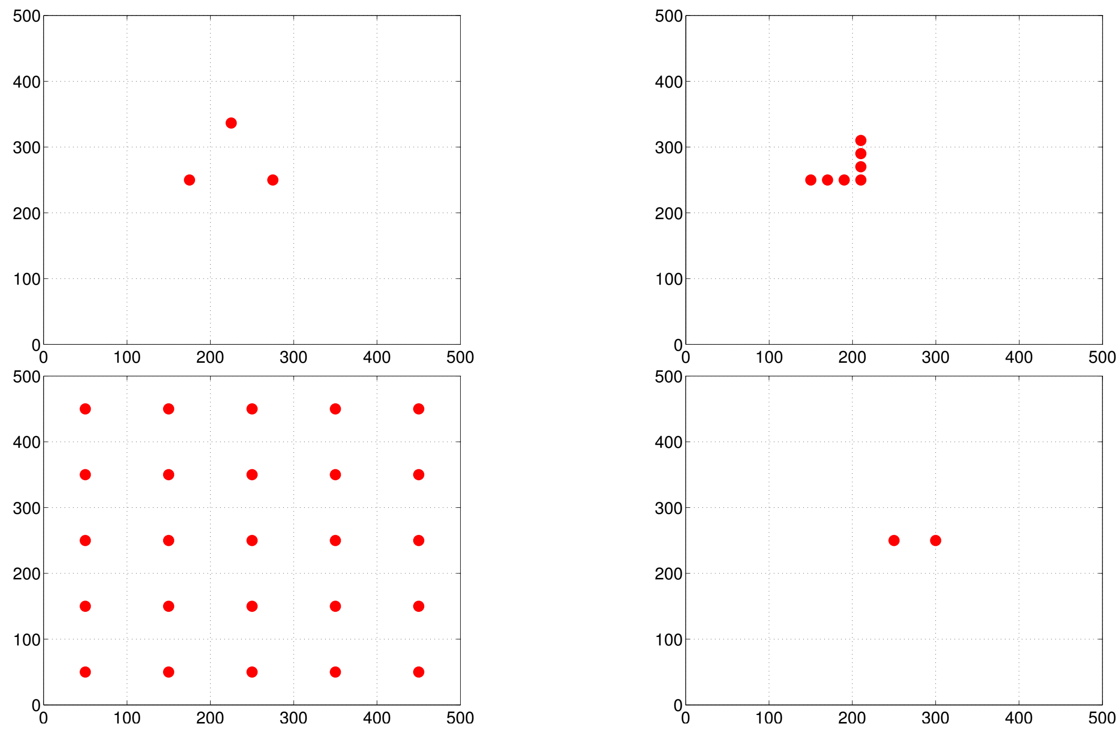


Figure 2.23: Simulated Scenarios used for Experiments. Red dots represent the landmarks. Start location of the robot is near the center of each scene. The size of the world is 500x500.

estimating odometry biases as well at the cost of increased non-linearity in the resulting objective function.

Simulated Environment

In simulation, we directly compare the result of our autonomous calibration algorithm against known extrinsic calibration parameter values for the exteroceptive sensor. It allows us to analyze the performance of the algorithm with different scenarios and parameter initializations, and noise in a controlled setting.

We evaluate the algorithm against random walk algorithm since there is not an existing state of the art for this particular problem. In random walk, at each instant the robot moves in a random direction instead of moving in a direction which results in maximum reduction in uncertainty.

Table 2.2: Default parameter values used in the simulated environment. #steps refer to the number of look ahead steps considered during each iteration. step size is the size of each step taken in any direction. angular resolution divides 360 degrees into set of directions which are considered at each time step. #samples refer to the number of Monte-Carlo samples considered. sensor location is the default ground truth value at which sensor is placed and sensor initialization is the noisy initial sensor location used as input to the algorithm. $\mathcal{N}(\mu,\Sigma)$ is a normally distributed pseudo random number generator having mean μ and variance Σ.

Parameter	**Value**
#steps	1
step size	50
angular resolution	1 deg
#samples	360
sensor location	$(0,50,0)$
sensor initialization	$(0+\mathcal{N}(0,10),50+\mathcal{N}(0,10),0+\mathcal{N}(0,0.1rad))$

For the first set of experiments, we present four different 2D scenarios to compare the proposed approach against the random walk algorithm. These scenarios are shown in Figure 2.23. In each scenario the robot starts at the same location for both random walk algorithm and our proposed approach and we compare the algorithms based on the evaluation metrics proposed in section 2.7.3. We assume that there are no false positive data-associations as the robot moves through the scene. The robot is assumed to be a 2D point robot with a range-bearing sensor attached to the robot with a known extrinsic transformation. For each scenario, we run each algorithm for 10 independent trials given the same start location and measure the covariance determinant of the estimated extrinsic transformation and the median transformation error to the known ground truth calibration. Evaluation metrics are explained in detail below. We log the evolution of these metrics with the number of planning iterations and demonstrate that using active planning results in faster convergence as compared to random walk algorithm. The default parameter values are explained in Table 2.2.

Table 2.3: Default parameter values used in Gazebo and Real environment. #steps refer to the number of look ahead steps considered during each iteration. #samples refer to the number of Monte-Carlo samples considered. Range min refers to the minimum range of the sensor. Range max refers to the maximum range of the sensor. measurement res is the resolution of the sensor.

Parameter	Value
#steps	1
#samples	100
range min	0.10 m
range max	30.0 m
measurement res	0.01 m deg

Gazebo Environment

For the second set of experiments in simulation, we use Gazebo[KH04] simulation environment to run our algorithm on a turtlebot. It is modified to attach a Hokuyo laser scanner at a certain height from the turtlebot base. Instead of maintaining landmarks in the environment and using landmark-pose constraints, we estimate pose-pose constraints through scan matching and add the corresponding pose-pose-transform factor similar to the odometry bias factors. Omnimapper [TRC14b] is used for mapping. We do not have any assumption on data-association as well.

In this experiment we sample the control in the velocity space, and estimate the change in the confidence of the sensor extrinsic parameters to choose the next control. The default parameters for this are shown in Table 2.3. Ground-truth sensor location is at $(0, 0, 1.36m)$ and the sensor initialization used is $(1, -3, 1.36m)$ which is chosen at random. We measured the rate of the convergence of the error of the estimate for 5 independent iterations with random start location.

Real Environment

For experiments in real environment, we run our algorithm on a physical robot. The robot platform consists of a Segway RMP-200 mobile base, which has been modified to be statically

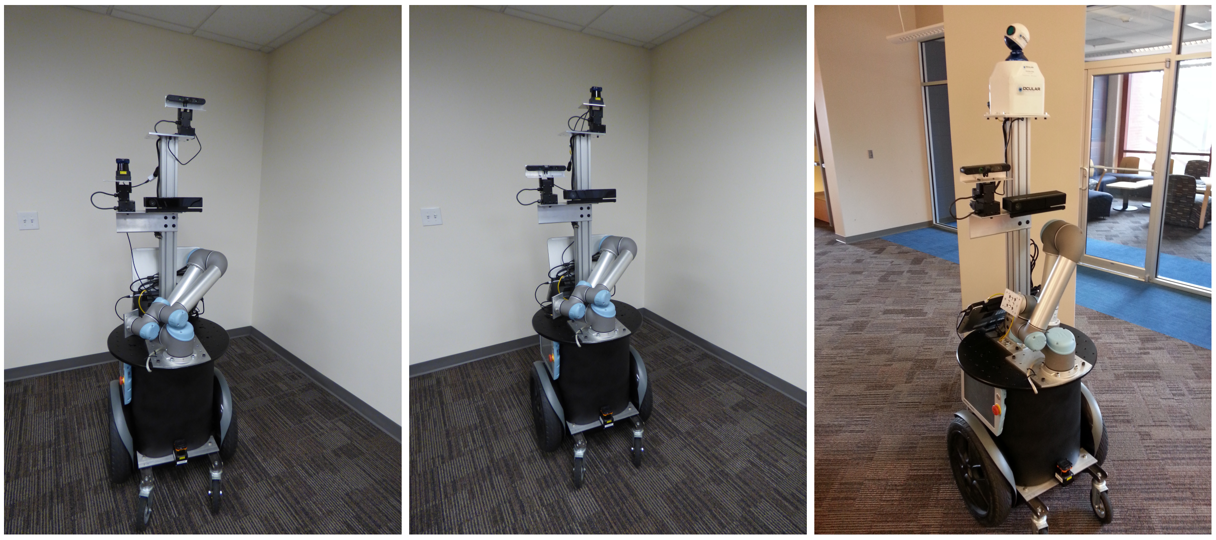

(a) (b) (c)

Figure 2.24: The figures above show an example reconfiguration for the experiments conducted. The figure on the left shows the Hokuyo laser on the top of the robot and the figure on the right shows the Hokuyo laser towards the right of the robot.

stable. For experiments we mounted a Hokuyo UTM30LX-EW laser Range Finder at a certain height. Computation is performed on an on board laptop. To evaluate the system, the robot was allowed to attempt self calibration in different locations in an indoor lab environment. The robot is shown in Figure 2.24. This was an example scenario for the experiment where the system was evaluated to test the variance against the reconfiguration. Similar to the Gazebo environment, we use scan matching to add pose-pose constraints instead of adding landmark-pose constraints.

2.7.3 Evaluation Metrics

Below we explain the evaluation metrics used to compare our algorithm against random walk algorithm.

Median transformation error

It is evaluated by computing the median over the translation errors when comparing the ground truth sensor translation values to the estimated sensor translation values. The translation errors are computed after every planning iteration and the median translation error is computed over 10 independent trials of the algorithm.

Median covariance determinant

Covariance determinant is evaluated by computing determinant of marginal uncertainty of the extrinsic transformation parameter. Similar to the previous metric, we compute covariance determinant after every planning iteration and compute its median over 10 independent trials of the algorithm.

Results

Simulated Environment

We compared our algorithm against random walk algorithm and plotted the evolution of median transformation error and median covariance error as shown in Figure 2.25. We used the default parameters as summarized in Table 2.2 and ran it on scenes shown in Figure 2.23.

In median covariance comparison (Figure 2.25), the median of the covariance of the sensor transform for all of the 10 runs is plotted with the x axis being the planning iteration. The proposed method performs better on all of the scenes. Scene 3 is the only one where the random walk performs comparably.

Similarly in median error comparison (Figure 2.25), the median of the absolute error (translation error between the estimate and the ground truth) for all of the 10 runs is plotted with the x axis being the planning iteration. The proposed method performs significantly better on all the scenes except 3. This is because Scene 3 is a grid of landmarks with a lot of landmarks being visible as the robot moves in the scene. This provides good calibration uncertainty convergence irrespective of the used strategy (random walk or active planning based calibration). Below we analyze the effect of varying parameter values on the evolution of median error and covariance

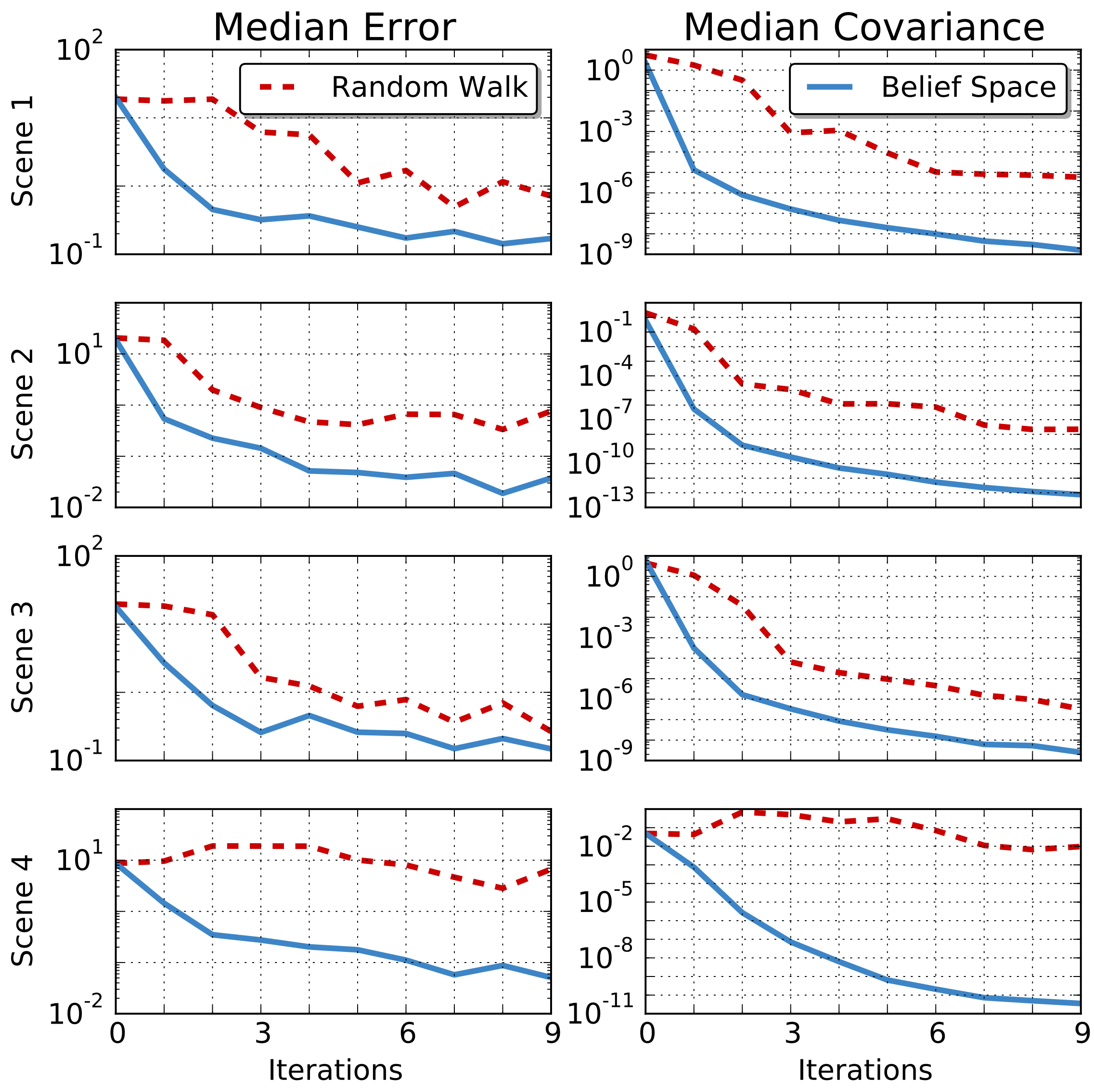


Figure 2.25: Plots showing evolution of median transformation error (left) and median covariance determinant (right) over planning iterations. Blue line shows the results using our approach and Red dotted line shows the result using random walk algorithm.

values.

Rigidity of plan in each iteration

We reduce the rigidity of plan in each iteration by reducing the step size and increasing the number of steps. We experiment with three configurations of rigidity, default: (step size: 50,

#steps: 1), medium rigidity: (step size: 25, #steps: 2), low rigidity: (step size: 10, #steps: 5). Figure 2.26 summarizes the results. Reducing the rigidity of the plan increases the variance in median error but has better median covariance than more rigid plans. The reason behind this is that reducing the rigidity might result in robot getting stuck in local minima even though it leads to faster covariance reduction.

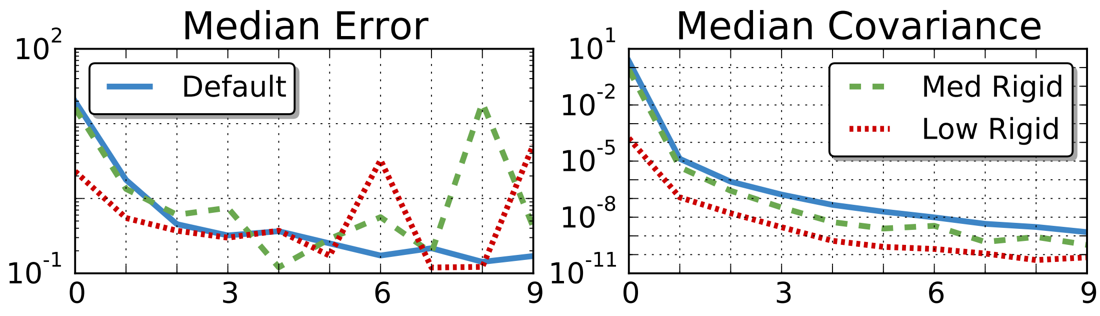


Figure 2.26: Plots showing evolution of error (left) and covariance (right) for different plan rigidities.

Angular Resolution

We experimented with three different values of angular resolution, default: 1 degree, medium: 2 degrees, high: 4 degrees. Figure 2.27 show the results. Varying angular resolution has an insignificant effect on the number of median error and covariance values.

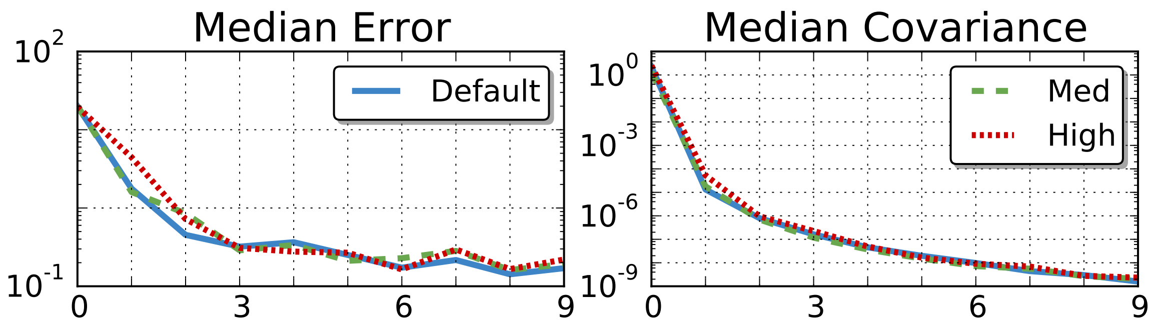


Figure 2.27: Plots showing evolution of error (left) and covariance (right) for different angular resolution. Blue plot uses the default parameter value, green dotted plot uses 2 degrees angular resolution and red dotted plot uses 4 degrees angular resolution.

Number of Monte-Carlo Samples

We experimented with three configurations of the number of monte-carlo samples used in the Algorithm 2. The configurations we experimented with are: default: 360, medium: 180, low: 90. Figure 2.28 show the results. Reducing the number of samples have insignificant effect on the error and covariance values.

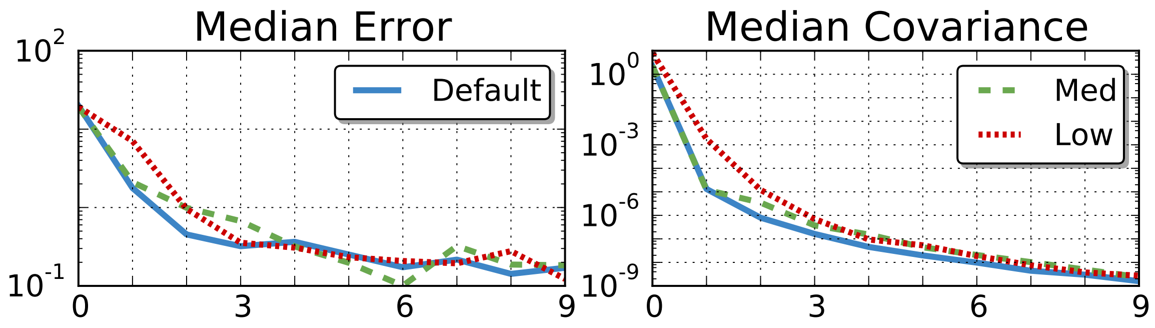


Figure 2.28: Plots showing evolution of error (left) and covariance (right) for different number of samples. Blue plot uses the default (360) number of samples, green dotted plot uses 180 samples and red dotted plot uses 90 samples.

Robustness to bad initialization

To analyze the robustness of our algorithm to bad initializations, we experiment with three configurations of sensor location initializations:

$$\text{default: } (0+\mathcal{N}(0,10), 50+\mathcal{N}(0,10), 0+\mathcal{N}(0,0.1rad)),$$
$$\text{medium noise: } (0+\mathcal{N}(0,20), 50+\mathcal{N}(0,20), 0+\mathcal{N}(0,0.5rad)),$$
$$\text{large noise: } (0+\mathcal{N}(0,40), 50+\mathcal{N}(0,40), 0+\mathcal{N}(0,1rad).$$

$\mathcal{N}(\mu,\Sigma)$ is a normally distributed pseudo random number generator having mean μ and variance Σ. Figure 2.29 summarizes the result. Increasing the initialization noise has an insignificant effect on the convergence. This implies that the objective function has a wide basin of attraction and can converge given bad initializations as well.

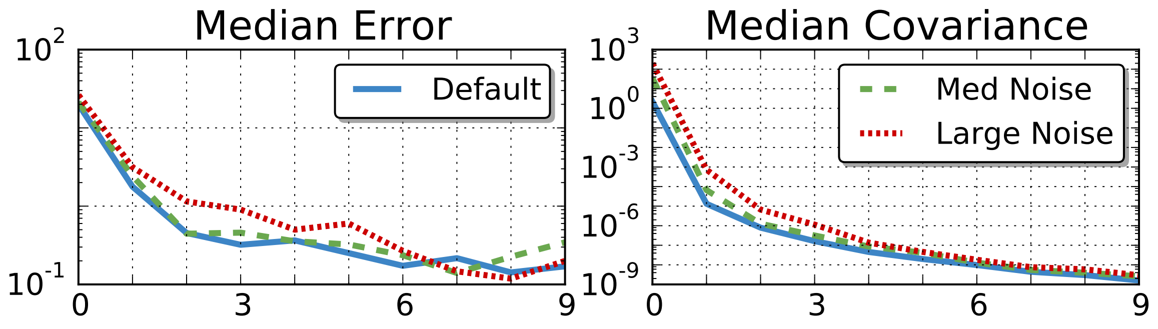


Figure 2.29: Plots showing evolution of error (left) and covariance (right) for different sensor initializations. Blue plot uses the default initialization, green dotted plot uses medium initialization noise and red dotted plot uses high initialization noise.

Gazebo Environment

We experimented in the simulated Gazebo environment with 10 trials with random initial locations and random initial poses of the sensor. We measured the convergence of the absolute error with respect to the known ground truth over time. This is summarized in the Figure 2.30. This shows that the robot is able to estimate the extrinsic of the sensor in a timely fashion and to high degrees of accuracy. This can consistently be seen across the random restarts and this shows that the algorithm is able to converge quickly for each of the test cases. As can be seen in the graph, the robot can autonomously calibrate the sensor with required accuracy within 300 seconds.

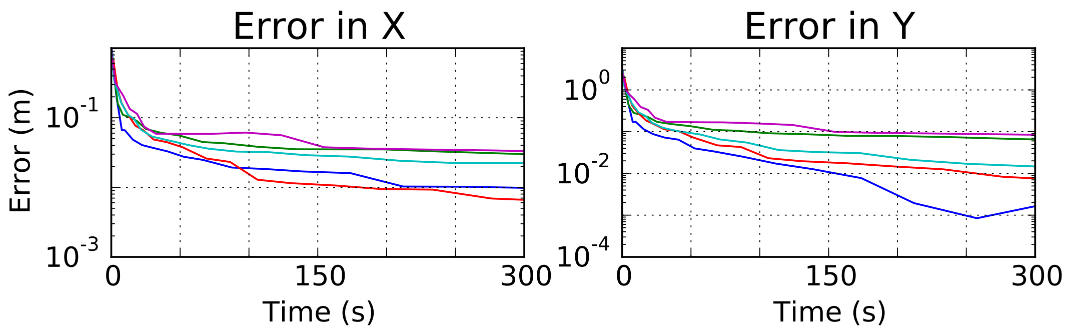


Figure 2.30: Plots showing evolution of error over time for 5 separate gazebo simulations. The y axis is in log scale and the x axis is time in seconds. The lines represent the error over time for each of the random restarts.

Real Environment

For the real experiment, we measured the ground truth extrinsic calibration by hand. We experimented with different configurations to measure the absolute error with respect to time. This is summarized in Figure 2.31. This shows that the approach can be used effectively to estimate the extrinsic parameters of the sensor on the real robot quickly. The estimation of the sensor is also invariant to changing the pose of the sensor with respect to the base. This is consistent with the theoretical analysis and the simulations.

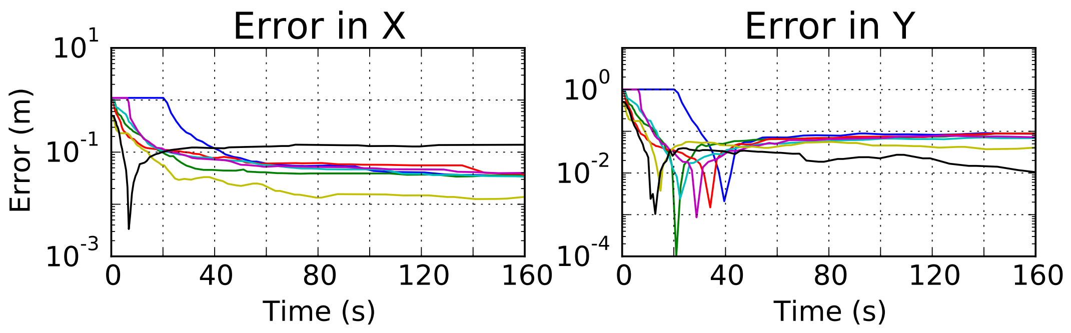


Figure 2.31: Plots showing evolution of error over time for 7 separate real robot experiments. The y axis is in log scale and the x axis is time in seconds. The lines represent the error over time for each of the random restarts and different configurations of the robot.

2.7.4 Discussion

In robotics, proper extrinsic calibration are often mission critical and the effects of this are apparent. For instance, in the DARPA robotics challenge, a fall could result in throwing off the extrinsic calibration. This in turn affects other tasks such as the tele-operation, and perception. Online techniques that have the ability to self calibrate can help mitigate such issues to a certain extent. However, this would be more useful if the robot could perform a self calibration routine before performing any operation.

The laborious and human error prone task of calibration requires a researcher to collect data for a camera given a calibration object manually. This is often random motion but does presuppose some knowledge of the calibration procedure with some level of experience, and so may not be purely random. As seen from the results however, the proposed algorithm performs better than merely a random walk approach that thought of as a novice human performing manual calibration, but more so in accordance of an expert who's each motion serves to reduce a specific uncertainty.

When calibration is performed one of the points of note is that the person performing the calibration will attempt to collect data at as many unique pose transforms between the sensor and the calibration targets. This intuitively can be seen in the plots from the trajectories generated in the simulation that the robot chooses to reduce its uncertainty.

Although a human has an intuition of what kind of motion helps generate better results for the camera calibration problem, this is not equivalent in having a numerical estimate of the variance in the change in error between steps. This often means that an over abundance of data is collected to ensure the quality of calibration. Performing this task manually every time a configuration change occurs on the robot is a time consuming task as well as challenging from a repeatability and reproducibility standpoint.

In the proposed algorithm, we use the idea of a look ahead to see what the consequence of an action is, and this allows us to estimate the parameter very close to the ground truth in fewer steps. Some of the constraints of this type of setup are that the amount of information available in the world is limited to what we can observe and what we have observed. This means that we can have an initial belief state estimate where the world is completely unknown and the belief state can be updated over time. The number of possible belief states is large and searching through all possible situations is not possible. This restricts the amount of look ahead that will actually help the algorithm. This highlights one of the main differences between our work and similar past research. We believe that this research will greatly benefit the mobile robot community as a whole and allow us to reduce the time taken on calibration routines between experiments.

One of the interesting observations, while performing the experiments was that there were certain emergent behaviors. For instance, the robot initially prefers taking paths that are cyclic and this could be attributed to a notion of closing the loop. This is an interesting behavior as it relates the active planning to intuition.

Acknowledgment

This chapter appears in the following publications:

- Alexander J. B. Trevor, John G. Rogers III, Carlos Nieto-Granda, and Henrik I. Christensen. Applying domain knowledge to SLAM using virtual measurements. *In IEEE International Conference on Robotics and Automation (ICRA)*, pages 5389–5394. Anchorage, Alaska, USA, May 3-7, 2010.

- Alex Trevor, John Rogers, Carlos Nieto-Granda, and Henrik Iskov Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3D. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop: Semantic Mapping and Autonomous Knowledge Acquisition. Taipei, Taiwan, October 18-22, 2010.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. SLAM with expectation maximization for moveable object tracking. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 2077–2082. Taipei, Taiwan, October 18-22, 2010.

- A. Trevor, J. Rogers, C. Nieto, and H. Christensen. Feature-based mapping with grounded landmark and place labels. *In the Robotics Science & Systems (RSS) Workshop on Grounding Human-Robot Dialog for Spatial Tasks*. Los Angeles, CA, USA. June, 2011.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. Simultaneous localization and mapping with learned object recognition and semantic data association. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 1264–1270. San Francisco, CA. USA, October 9-14, 2011.

- John G. Rogers, Stuart Young, Jason M. Gregory, Carlos Nieto-Granda, and H. I. Christensen. Robot mapping in large-scale mixed indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

- John Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, Alexander Cunningham, Manohar Paluri, Nathan Michael, Frank Dellaert, Henrik I. Christensen, and Vijay Kumar. Effects of sensory precision on mobile robot localization and mapping. *In the 11th International Symposium on Experimental Robotics (ISER)*, pages 433–446. New Delhi and Agra, India, December18-21, 2010.

- Varun Murali, Carlos Nieto-Granda, Siddharth Choudhary, and Henrik I. Christensen. Active planning based extrinsic calibration of exteroceptive sensors in unknown environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Daejeon, South Korea, October 9-14, 2016.

I would like to thank my co-authors Alexander J. B. Trevor, John G. Rogers III, Varun Murali, Siddharth Choudhary, Alexander Cunningham, Manohar Paluri, Stuart Young, Jason M. Gregory, Prof. Henrik I. Christensen, Prof. Nathan Michael, Prof. Frank Dellaert and Vijay Kumar.

Chapter 3

Semantic Segmentation of the Environment

3.1 Introduction

Humans are constantly trying to make their lives easier. Service robots capable of operating in human environments have the potential to improve daily life by assisting humans in a variety of tasks. Endowing these robots with the ability to understand and reason about spatial regions such as individual rooms, as well as understanding the semantic labels of such spaces, could facilitate tasks such as navigation and mobile manipulation in human environments.

Human environments are typically partitioned into discrete spaces, such as offices, corridors, living rooms, etc. Such a partitioning allows humans to organize and enable their everyday activities, and these spaces typically have specific purposes and labels. Service robots that understand the partitioning of human environments can utilize this information to better assist humans in everyday tasks. For example, if a robot is given the command “fetch the red mug from the kitchen”, having an understanding of the location and extent of the region considered “kitchen” is beneficial.

In the last years, we have seen important developments in service and assistive robots for domestic applications and tasks. These works focus on the understanding of the environment using semantic information in order to create a synergistic interaction between humans and robots. Dellaert and Bruemmer[DB04] proposed extending FastSLAM to add semantic information of the environment to each particle's map. Several approaches have been presented for map partitioning, using topological and geometric representations of the environment. For example, Oberländer [Obe+08] proposed a SLAM algorithm based on FastSLAM 2.0 [Mon+03] that maps features representing regions with a semantic type, topological properties, and an approximate geometric extent. The resulting maps enable spatial reasoning on a semantic level and provide abstract information allowing efficient semantic planning and a convenient interface for human-machine interaction. Thrun [Thr98] integrated grid-based maps to learn the environment using artificial neural networks and naïve Bayesian integration to generate a topological map by partitioning the latter into coherent regions.

Another body of work focuses on extracting semantic spatial properties of the environment from 2D and 3D data. Donsung and Nevatia [KN94] introduced a new spatial representation, *s-map*, for an indoor navigation robot. This map represents the locations of visible 3D-surfaces of obstacles in a 2D space. O'Callaghan [ORD09] developed a new statistical modeling technique for building occupancy maps by providing both a continuous representation of the robot's surrounding and an associated predictive variance employing a Gaussian process and Bayesian learning. Ekvall [EKJ07b] applied an automatic strategy for map partitioning based on detecting borders between rooms and narrow opening to denote doors or gateways using different types of features (lines, points, SIFT). *Rhino* [Bur+98] is an example of a service robot which integrates localization, mapping, collision avoidance, planning, and various modules concerned with user interaction telepresence giving tours on a museum. *BIRON*, a mobile Home Tour Robot [Spe+06], uses integrated vision based localization, a modular architecture and a spoken dialog system for on-line labeling and interaction about different locations in a real, fully furnished home environment where it was able to learn the names of different rooms. The approach presented by Topp and Christensen [TC06] and [TC10], provides a separation of regions that relate to a user's view on the environment and detection of transitions between them. They assumed an interactive setup for the specification of regions and showed the applicability of their method in terms of distinctiveness for space segmentation and in terms of localization purposes. Also, semantic place categorization using visual features has been addressed in [PMC08; SI07]. Many robotics researchers focused on place recognition tasks or on the problem of scene recognition in computer vision [OT01; QT09].

Ekvall *et al.* [EKJ07a] demonstrated a strategy for integrating spatial and semantic knowledge using SLAM and object recognition based on Receptive Cooccurrence Histograms. Espinace *et al.* [Esp+10] presented an indoor scene recognition system based on a generative probabilistic hierarchical model to associate objects to scenes. The performance of the object classifiers is improved by including geometrical information obtained from a 3D range sensor that facilitates a focus of attention mechanism. Kollar *et al.* [KR09] utilizes the notion of object-object and object-scene context to reason about the geometric structure of the environment to predict the location of the objects. These approaches only identify the place based in the specific objects detected and the hierarchical model that is used to link the objects with the place.

Quattoni and Torralba [QT09] introduced a purely vision-based place recognition system which improves the performance of the global gist descriptor by detecting prototypical scene regions. The size, shape, and location of up to ten object prototypes has to be labeled and learned in advance. The labeling part is especially work-intense while the approach with several almost fixed regions can only find objects in typical views on the scene. However, it is not suited well for robotic applications. Since we want to deal with flexible positions of typical objects, we apply a visual attention mechanism that finds important regions automatically.

There has also been work on recognizing known places under various changes (e.g. furniture, weather conditions, walking people) [Pro+06; Ull+08]. Their approach combines local SIFT [Low04] features and global Composed Receptive Field Histograms (CRFH) [LL04] features.

Mozos *et al.* [Moz+07a; MB06] proposed a place categorization method for building topological maps using AdaBoost[FS95] on a plethora of simple features extracted from laser range data. They improved their semantic labels with an object recognition system that found known objects in omnidirectional camera images. In [WCR09] , Wu *et al.* presented a solution to visual place categorization based on a global visual descriptor, the spatial Census Transform Histogram (CENTRIST), and Bayesian filtering [WR11].

3.2 Gaussian Regional Analysis

As robots have to cooperative with humans it is advantageous that they have a shared representation of the space, preferably a model that is simple for the human to use as part of commanding the robot and understanding feedback. Semantic mapping literature has focused on developing robotic mapping techniques capable of functionally supporting these types of interactions. To perform these tasks, one of the strategies that is used is to portray the relationship between a place and the knowledge that is associated with it e.g.(functionality, objective location), is semantic mapping. Kuipers [Kui+00] proposed the *Spatial Semantic Hierarchy* (SSH), which is a qualitative and quantitative model of knowledge of large-scale space consisting of multiple interacting representations. This map also informs the robot of the control strategy that should be used to traverse between locations in the map. This representation is based on the relationship of objects, actions and the dependencies from the environment. More recently, Beeson *et al.* [Bee+07] provided a more specific framework representation of spatial knowledge in small scale space. This framework is focused on the robot's sensory horizon e.g.(global and local symbolic, and metrical reasoning of the space), but also human interaction.

Existing approaches for robot indoor navigation build an occupancy grid map using range data from its sensors. These maps, however, only provide geometric information such as obstacles and open areas in the environment without a semantic understanding of it. Martínez-Mozos and Rottmann [Moz+07b] [Rot+05] introduce a semantic understanding of the environment creating a conceptual representation referring to functional properties of typical indoor environments. Providing semantic information enables a mobile robot to more efficiently accomplish a variety of tasks such as human-robot interaction, path-planning, and localization. Ekvall [EKJ07b] integrated an augmented SLAM map with information based on object recognition, providing a richer representation of the environment in a service robot scenario.

3.2.1 Approach

Our goal was to design a system capable of reasoning about spaces. In contrast to work such as [Moz+07b], which builds a topological map on top of a metric map, we instead provide a continuous classification of the metric map into semantically labeled regions.

The semantic map layer of our system is a multivariate probability distribution on the coordinates of our metric map to a set of semantic labels. This multivariate distribution is modeled as a Gaussian model. Each of the Gaussians in the model is based on the robot's sensor data when it was provided a label by a human guide. Each spatial region is represented using one or more Gaussians in our metric map's coordinate frame. So, a region with label L and n Gaussians, each with mean μ and covariance Σ, is represented as:

$$Region = \{L, \{\{\mu_1, \Sigma_1\}, \{\mu_2, \Sigma_2\}, ..., \{\mu_n, \Sigma_n\}\}\}$$

A semantic map is then just a collection of such regions, so a semantic map with m regions would be represented as:

$$Map = \{region_1, region_2, ..., region_m\}$$

Our system builds these maps partitions of our metric maps through human guidance. The human takes the robot on a tour of the space (either by driving the robot manually, or using a person following behavior), and teaches the robot typing the appropriate label for the space that it is currently in. The regional analysis technique is to take a laser scan measurement, fit a Gaussian to the resulting points, and store the mean and covariance in the map along with the label provided by the human.

Using this semantic map partition, the robot can be asked for its belief of the name of the region it is currently occupying. This is done by evaluating the Mahalanobis distance of the robot's current pose x close by labels coded as Gaussian region models (Equation 3.1), and choosing the region that is closest using this metric.

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{3.1}$$

This map representation allows for probabilistic classification of the map by region label. Additionally, while navigating through the environment, the robot continuously checks its position with respect to the semantic map partition. If it is not sufficiently confident (more than a certain threshold) that it is in a region with a known label, it prompts the user to input the name of the current region.

Once the robot has a semantic map partition, users can request that the robot navigate to one of the regions, such as "living room". The robot can then find the region in the map with label "living room", and calculate the Mahalanobis distance from its current position to the mean of each Gaussian in the region.

The robot selects the closest of these as the goal, and sends this to its path planner in order to autonomously navigate to that region. While traveling, the robot continuously calculates its confidence of which region it is, and stops when it is confident that it is more likely to be in the goal region than any other region as follows:

$$\textit{Distance to goal} < \frac{1}{4} * \textit{Nearest distance to the non-goal}$$

This results in the robot entering a region, but not attempting to move to the region's center. Additionally, if the robot enters the region with the desired label at any point while navigating to its goal point, perhaps because the path to the closest point was blocked, it will recognize this and stop once it is sufficiently in the goal region.

In this work, we only use the semantic map partition for navigation tasks; however, we believe this map representation has a number of other applications, such as searching for objects. For example, if we have a mobile manipulation platform and ask the robot to "get the mug from the kitchen", our map representation can be used to give a spatial region of our metric map which should be search, by finding the area that is labeled as kitchen with at least a certain confidence level.

When a human is leading a robot on such a tour and instructs it that it is in a new location with label *L*, our approach is to then take the point cloud generated by our laser range finder, fit a Gaussian to the points in the point cloud and associate it with that label. If the robot was positioned such that it had a good view of the extent of the room or area with its laser scanner, the resulting Gaussian's mean and variance will give us a reasonable idea as to the room's location and extent. Our system currently requires at minimum a single Gaussian to represent each label's area, but also allows for multiple such Gaussians to be taught with a single label, which allows us to approximate arbitrarily shaped rooms.

Reasoning about the extent of spaces in this way provides several benefits. If we wish to make a request of the robot, such as "move to the kitchen and search for a mug", it now has some idea of the extent of the area it must search that is considered to be part of "the kitchen", whereas with a waypoint, it would not. For navigation tasks, this means that we can ask the robot to move to a room, and it will move to the nearest point within the room. Also, since the Gaussians provide probabilistic classification, we can also use our model to determine the likelihood that we are within a certain labeled space.

3.2.2 Experiments

Our approach has been implemented and evaluated in several experiments. We designed two simulated environments in Stage[GVH03][*] in which the robot can be taught locations and navigate between them. Preliminary experiments were also performed using our Segway RMP-200 mobile platform to verify our technique on a real robot platform.

Simulation Environment

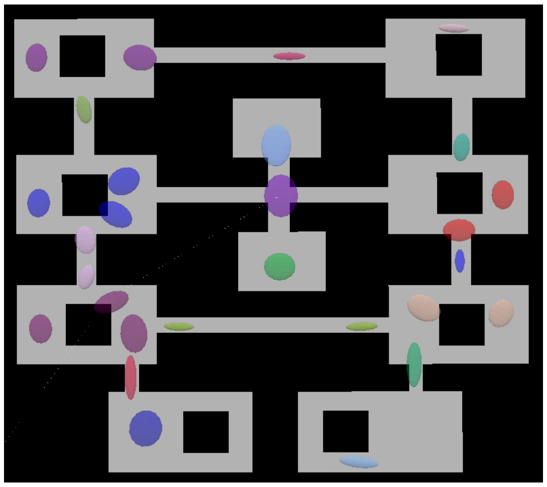


Figure 3.1: A visualization of the Gaussian regions representing the rooms and hallways. All the Gaussians are colored in different colors for identification. This figure is best viewed in color.

In this simulated experiment, we tested the effectiveness of our method by providing labels for each of the rooms and hallways. The first experiment, shown in Figure 3.1, consisted of labeling each room and each hallway and navigating between them in order to test the system's

[*]Stage is a 2D multiple-robot simulator from the Player project. http://playerstage.sourceforge.net

effectiveness at accepting this type of navigation command.

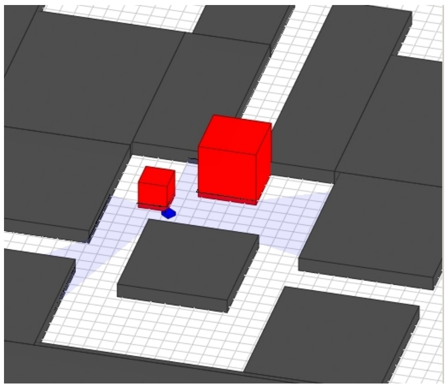


Figure 3.2: The robot's path is obstructed with a simulated block. The robot is the smaller blue object and the obstacles are the large red cubes.

The robot is able to successfully navigate with the calculated trajectory, avoiding obstacles. One of the tests was to move from a room labeled "room 9" to "room 4" in the map as appears in Figure 3.1. Shown in Figure 3.2, the shortest path was obstructed with a simulated block and the robot replanned a new trajectory to reach the goal as can be seen in Fig 3.3. The laser hits on the obstacle that blocks the shortest path can be seen near the robot in Figure 3.3.

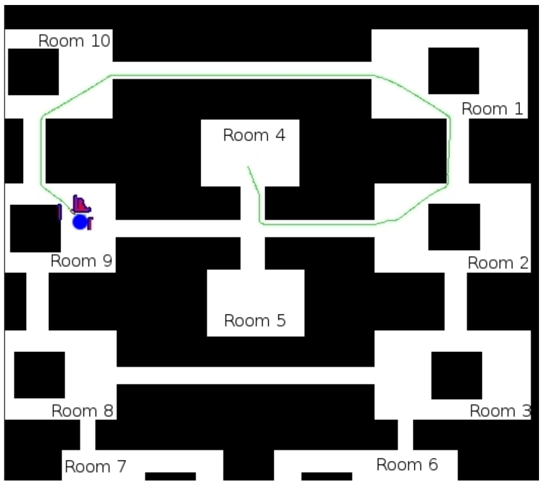


Figure 3.3: Robot path replanned to navigate from room '9' to central hallway and arrived in room '4'. The blue circle represents the robot, the green line the robot path and in red the obstacles detected by the laser scan.

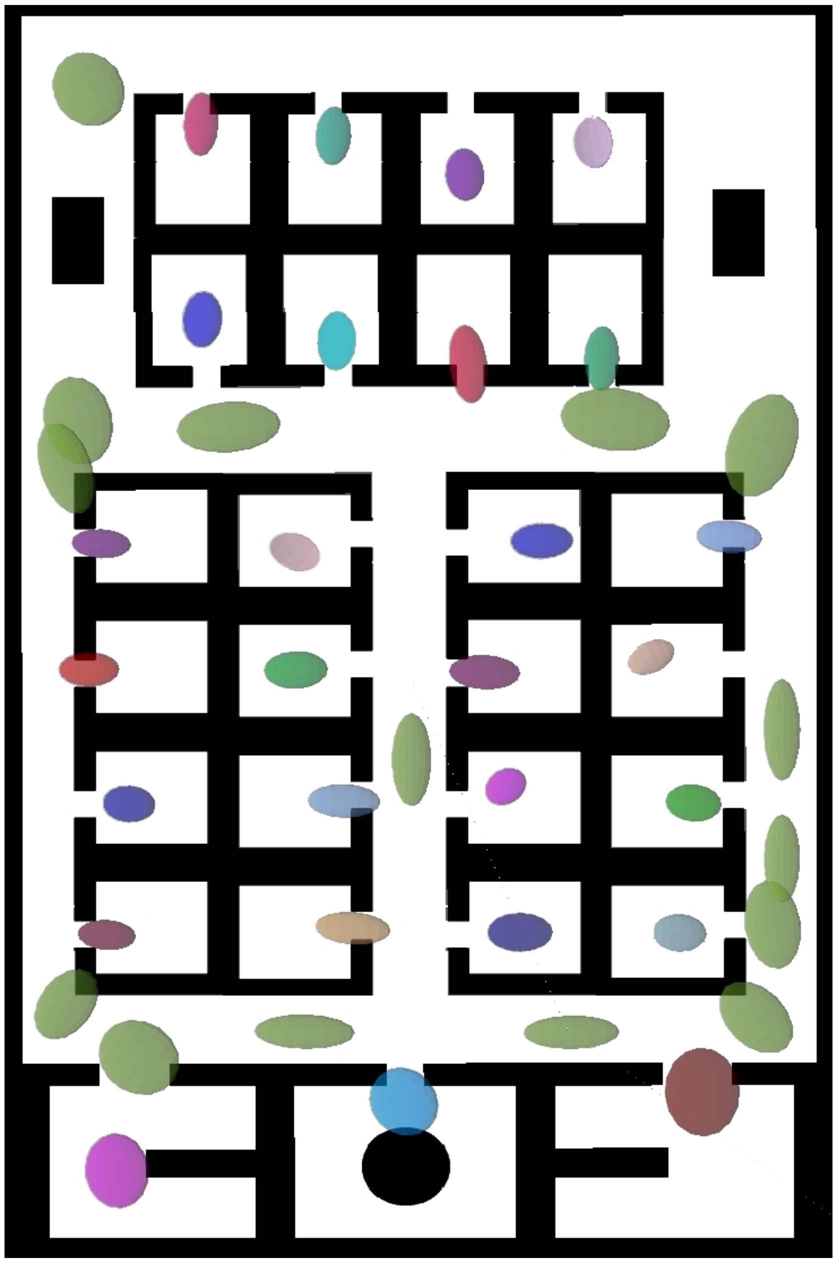


Figure 3.4: The second simulated office environment used for our experiments. Colored ellipses represent the Gaussians in our model, and different colors represent different spaces. This figure is best viewed in color.

For the second experiment, we labeled twenty seven rooms and a hallway in the map, and left one unknown area unlabeled, as can be seen in the upper right corner of Figure 3.4. The main purpose was to simulate an office environment, where the transition region between rooms is a hallway. The robot is able to continuously provide the current location and navigate from one region to another. Based on this map classification, we created several different scenarios to test our system. One of the test scenarios, shown in Fig 3.5, involved the robot navigating between two regions with different labels, a room and a hallway, represented by Gaussians with means very close to each other. The robot was requested to navigate from the room to the hallway and back.

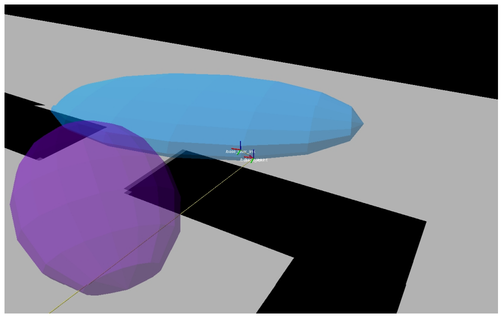


Figure 3.5: Visualization of the robot navigating between to regions in the environment.

Our system calculated the Mahalanobis distance to find the closest Gaussian region, but the regions are very close to each other that the robot only turn and move only a short distance.

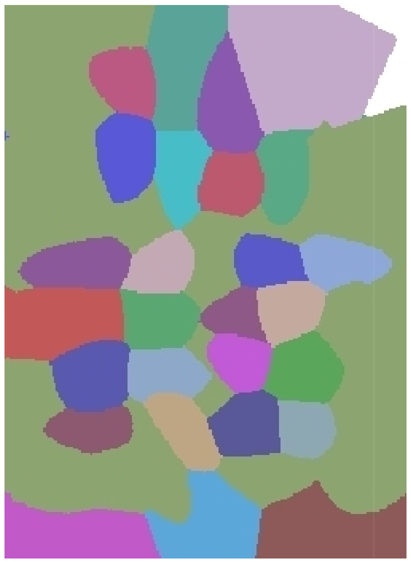


Figure 3.6: A visualization of the decision boundaries of the regions representing the rooms and hallways on the map shown in 3.4. All the Gaussians are colored in different colors for identification. This figure is best viewed in color.

This demonstrates that our system will cause the robot to move into a region until it has a certain confidence level that it is in the region, rather than stopping on the equiprobable decision boundary between the regions as shown in Fig 3.6. Another scenario, shown in Figure 3.7 is to drive the robot through a hallway to an unknown area which has never been assigned any label. When the robot reached a location that was not likely to be part of any previously labeled region, it displayed *"I do not know where I am."* and requested that the user provide the current location's label displaying *"Please tell me where I am"*.

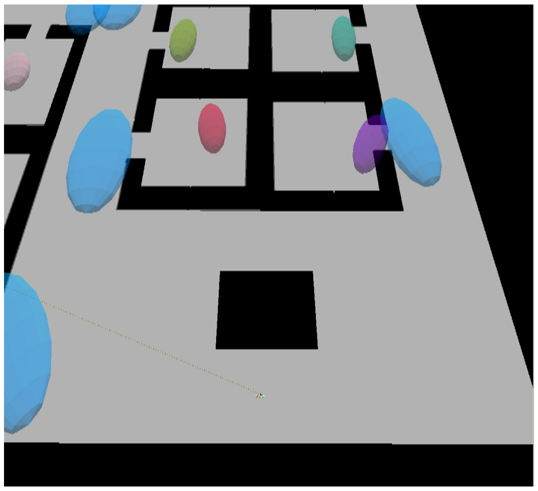


Figure 3.7: The robot is driven in an unknown area which has not been assigned any label.

Several more scenarios consisted of teaching the robot with different orientations and locations inside of the rooms. Localizing the robot between rooms and hallways worked better when the user taught new locations to the robot in the middle of the room as opposed to in the doorways because the laser hits were more representative of the room's extent.

Finally, we tested the robot by starting it with no semantic map information, so that it would prompt the user immediately for a label, which the human guide provided. The robot then navigated around the environment, and would stop whenever it was not confident that it knew the label for the current pose. Upon being provided with a label for the current location by the human guide, the tour resumed until it again needed a new label.

This resulting map is shown in Figure 3.8. This experiment demonstrates our method's

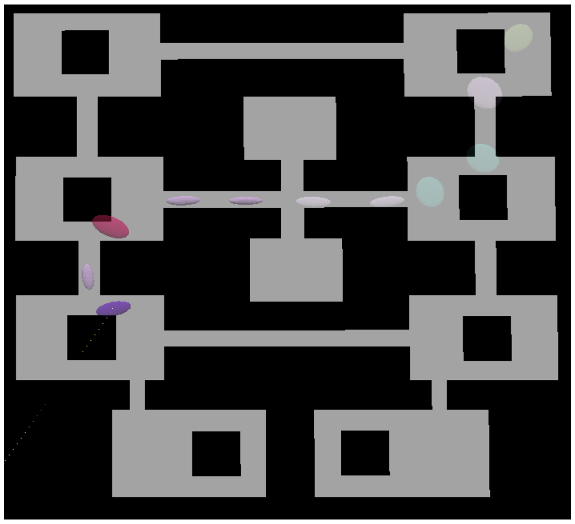


Figure 3.8: Result from starting the robot with no known locations in the semantic map, and prompting the user when the robot was not confident of the appropriate label.

effectiveness at determining when a new label is required. The robot began in the left middle room, and ended in the upper right.

Real Environment

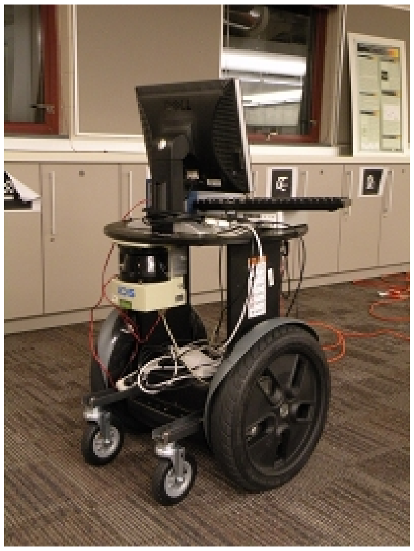


Figure 3.9: Our robot platform used in our experiments.

Our approach has been implemented on a Segway RMP-200 mobile platform (Fig 3.9). It is equipped with a SICK LMS-291 laser scanner, which is used for localization, mapping and obstacle avoidance, and is controlled by an on-board Mac mini computer (2.26GHz/Core 2 Duo).

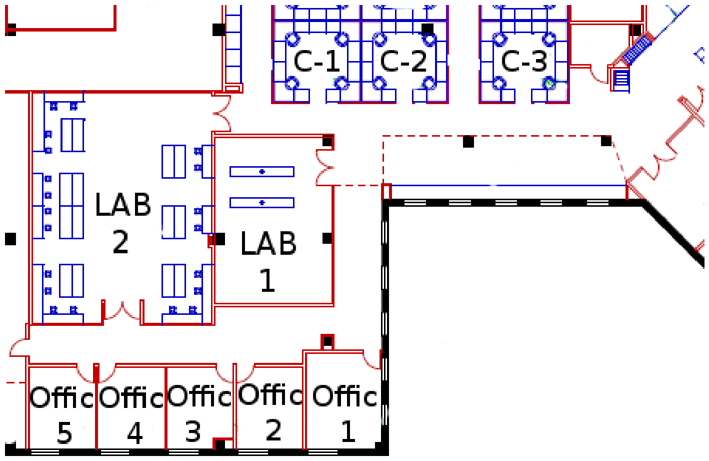


Figure 3.10: Robotics and Intelligent Machines Laboratory Map.

We conducted an experiment in our real office environment (Figure 3.10), teaching the robot with 5 different rooms and several different points in the hall. The first step was to drive our robot through the environment while collecting laser data and odometry. The SLAM gmapping module included with ROS[Qui+09b], which is based on the Rao-Blackwellized particle filter technique by Grisetti *et. al* [GSB07], was then used to build a metric grid map of the environment.

The map shown in Figure 3.11 was then used for localization in our experiments. Beginning with the map, the human tour started from the hallway to one of the three cubicles. When the human guide stopped, the robot was provided with the name of the location. During the tour, the robot could be queried for its current location, then the robot would calculate the Mahalanobis distance to the Gaussians regions and report the label of the nearest one. If the robot pose was not near to a previously labelled location, the robot report the location as “unknown”. Also, the robot can be asked to move to a specific known location, for example move from “*C-3*” to “*LAB 1*”.

Then, the robot calculates a safe trajectory to the room using the global map and continuously runs a local planner to avoid obstacles throughout the environment. When the robot successfully completed the task, it reported that it arrived at the current location's label.

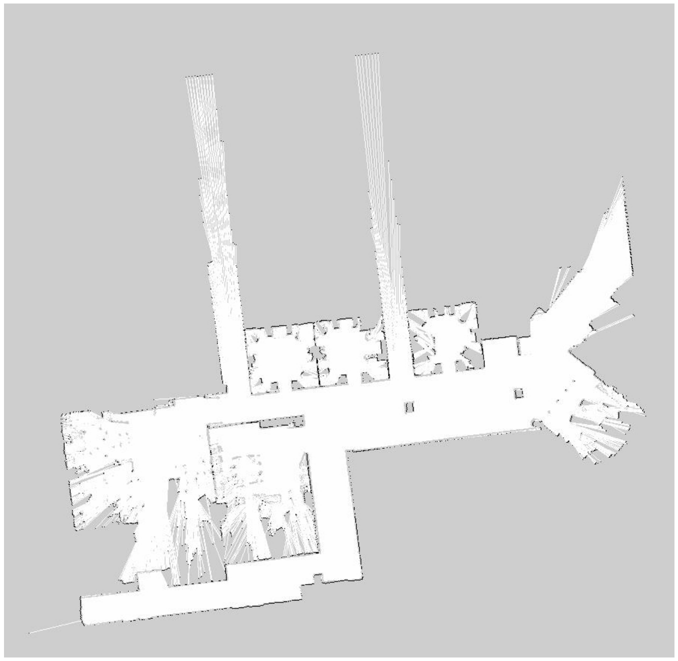


Figure 3.11: Generated Occupancy Grid Map for localization used in our experiments.

3.2.3 Discussion

We presented a technique for partitioning metric maps into labeled spatial regions using a Gaussian model. This representation enables the robot to perform navigation tasks in the map, as demonstrated by our experiments both in simulation and on a real robot platform. Also, this technique allows constraining the search space when processing tasks using automatic labeling; for example, if the robot detects a coffee machine and a microwave, then the region should be labeled as a kitchen. We will discuss in this chapter how to do the automatic semantic labeling of a specific region.

3.3 Region Decomposition

3.3.1 Approach

The region decomposition stage breaks the map into areas which are determined by distance and line of sight (LOS) to a specific point in a region. This point is called the region anchor. In indoor environments, we hypothesize that distance and line of sight are good heuristics for delineating regions. Region decomposition is a well studied problem in robotics and often reduces to a variant of the Voronoi diagram computation. In [LSB10], the authors propose an algorithm to update a Voronoi decomposition based on input from a dynamic occupancy grid representation of the environment. In [OF10] Voronoi information is used to extract higher-level semantic information about the environment in order to identify features such as hallways, doors, and rooms. This strategy also operatver a *k*-means with a (LOS) awareness. Our approach continues in this vein and adapts Boris Lau's code to serve a prior for the clustering algorithm. This prior is used to sort and determine potential anchor points for creating the regions. As a result the decomposition is different from a Voronoi based decomposition in that smaller rooms and regions can expand into hallways. Larger rooms are not form continuous regions and hallways are also divided up by distance. In a Voronoi based decomposition, the region would be bounded by the intersection with the hallway. However, this decomposition provides for situations where the portion of the hallway might be more strongly correlated to the entrance of that room than the a section of hallway which is farther away.

In the decomposition algorithm, $\mathcal{A}$ possible regions are considered given a Map $\mathcal{M}$. The cells of the map that are considered part of the voronoi decomposition in Boris Lau's code are the possible anchor points $\mathcal{A}$. The algorithm uses the initial position within the map to perform a brushfire computation on $\mathcal{A}$. As a result, points which are not burned are dropped from $\mathcal{A}$. This forces the decomposition to be over a continuous space. Afterwards $\mathcal{A}$ is sorted by distance from each point in $\mathcal{A}$ to its the nearest obstacle.

```
Require: Initial map $\mathcal{M}$, Initial pose $p_{init}$, Number of Regions $\mathcal{K}$
 1: $\mathcal{P} \leftarrow$ Brushfire$(\mathcal{M}, p_{init})$
 2: Assign points within $\mathcal{P}$ to Regions $\mathcal{R}$
 3: while $\exists \mathcal{R}_j \in \mathcal{R}^p$ do
 4:    Calculate centroids for $\mathcal{R}$
 5:    for $\mathcal{P}$ do
 6:       if Points are assigned to this region then
 7:          determine the point's distance to the region center
 8:          if The point is above the threshold distance or not in LOS then
 9:             Remove point from the region
10:             Add this point to the list of unregioned points
11:          end if
12:       end if
13:       for $\mathcal{R}$ do
14:          if Point is closer to region $\mathcal{R}$ centroid and is in LOS then
15:             Move the point to the closer LOS region
16:             Note that a point exchange has taken place
17:          end if
18:       end for
19:    end for
20:    Add unregioned points to regions without points.
21: end while
```

Algorithm 4: ContourSeekingExploration

In general, spatial correlation of RSS will depend on wavelength and varies according to environment geometry. In this paper, we will assume that a fixed characteristic length ρ is known a priori to be valid across the entire environment and will be used as a parameter for our region decomposition algorithm. Given a partially known occupancy grid representation of the free space in the environment, the k-means algorithm is used to find a decomposition of space with k regions where k is given by

$$k = \frac{M\delta_m^2}{\pi\rho^2}. \tag{3.2}$$

M is the total number of open cells in the occupancy grid $\mathcal{M}$ and δ_m is the resolution of the grid. Initialization of the k regions are chosen randomly within the free space. The basic k-means algorithm will perturb region centers until the Voronoi cells induced by the k centers are approximately the same size, i.e., with characteristic length ρ.

3.3.2 Sampling Regions

We want to partition the environment into regions such that we can obtain good estimates of the RSS average within each region, being sufficiently large so that the RSS average will vary across neighboring regions, and not so small to require a large overall amount of sampling and computation. The choice of region size is informed by the evolution of the nominal RSS value, and the local decorrelation of the RSS measurements due to fast fading and shadowing. The RSS average trend generally obeys $d^{-\alpha}$ decay with range d, where α varies depending on the environment with $2 \leq \alpha \leq 5$ typical. On the other hand, the fast fading results in rapid spatial variation. Consider the Rayleigh fading model for local RSS measurements, which has spatial autocorrelation given by

$$R(\delta) = aJ_0^2(2\pi\delta/\lambda) \tag{3.3}$$

where δ is the distance between locations, a is a constant, J_0 is the zero-order Bessel function of the first kind, and λ is the carrier wavelength. This indicates that samples taken 0.38λ apart have zero correlation, and samples taken at larger distances have small correlation. This heuristic is well-supported in rich scattering environments by many measurement campaigns, especially at microwave frequencies [LJB07]. However, the spacing necessary for decorrelation is variable depending on the propagation environment, with generally longer correlation in line of sight conditions. In practice, non-LOS indoor microwave measurements taken one or two wavelengths apart are generally viewed as experiencing independent fades. In other cases, such as hallways with LOS, the distance required for highly decorrelated RSS samples can increase by an order of magnitude (10 to 20 wavelengths, say), and the correlation will tend to drop off less rapidly with distance, e.g., see [LJB07]. An interesting special case occurs with a fixed ground node and an elevated mobile node; here the spatial correlation varies slowly in a trajectory along the LOS direction, whereas it varies more rapidly in a trajectory orthogonal to the LOS direction.

3.3.3 Experiments and Results

The robots used in the point-to-point wireless connectivity experiments are shown in Fig. 3.12. The *Packbot* is a ground platform equipped with a skid-steer drive system, onboard computation, and 802.11 wireless communication and Microstrain 3DM-GX2 inertial measurement unit (IMU). These systems are managed by an on-board computer. Each Packbot is equiped with a Hokuyo UTM-30LX scanning laser range finder with 60 m range. In addition, a *Zigbee* standard radio is mounted on each Packbot. This radio is used solely for measuring RSS between Packbot 1 and 2. Packbot1 is equipped with a Zigbee with a chip antenna and UTM-30LX. Packbot2 is equipped with a Zigbee with a chip antenna and UTM-30LX-EW. These are minor configuration differences since both robots use the same methods for wireless Zigbee commuunication and localization enabled by a laser range finder.

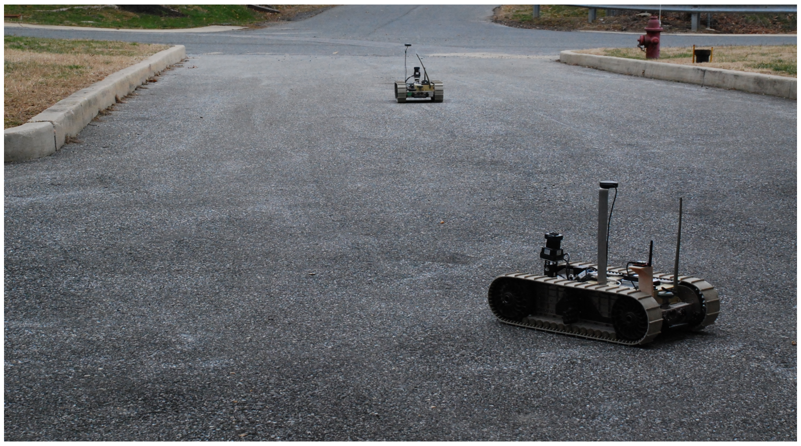


Figure 3.12: Packbots 1 and 2 equipped with Zigbee radios and Hokuyo laser scanners

Autonomous behaviors such as probabilistic localization and navigation are provided by leveraging the open source *Robotics Operating System* (ROS). This enables the robot to be able to localize in 2d map and navigate throughout a defined 2D environment. Rather than focus on the complicated problem of finding optimal sampling trajectories for a kinematically constrained vehicle in this work, we randomly visit different points within a prescribed region. When it is not possible to plan a path to this goal, the Packbot samples by rotating in place. The ZigXbee radio is mounted 0.23 m in front of the robot's center of rotation in order to take advantage of this rotation.

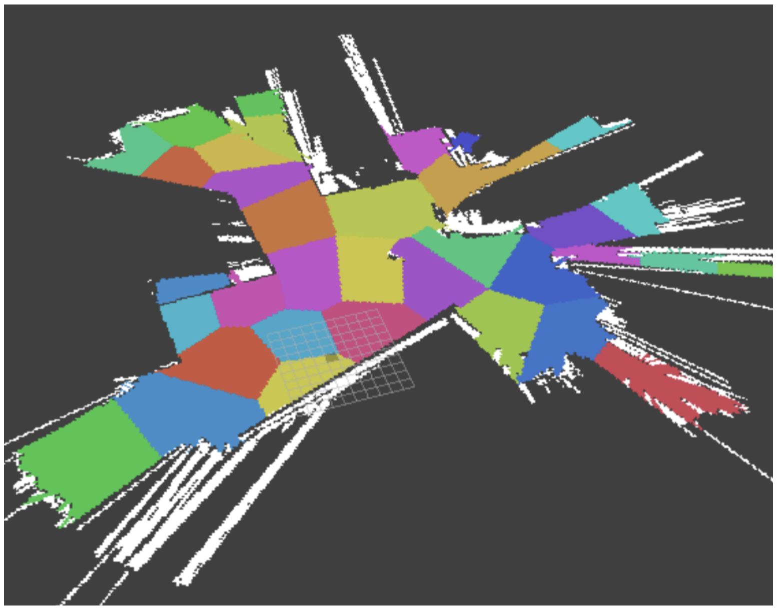


Figure 3.13: Region descomposition of the environment.

The testing environment is a typical office space. Furniture and walls provide are closely space creating multipath interference and the potential for shadowing regions. The size of the environment was chosen to show the full dynamic range of the ZigXbee transmitter-receiver pair when set at maximum power. The map used in testing was marked with imaginary obstacles for the sake of convenience.

3.4 Place Categorization

Scene recognition and understanding has been an important area of research in the Robotics & Computer Vision community for more than a decade now. Programming robots to identify their surroundings is integral to building autonomous systems for aiding humans in house-hold environments. In the past, many robotics researchers focused on place recognition tasks [PMC08; SI07] or on the problem of scene recognition in computer vision [OT01; QT09].

Recently, there have been several approaches to Scene Recognition using Neural Networks. Liao *et al.* [Lia+16; Lia+17] used Convolutional Neural Networks (CNNs) to recognize the environment based on object occurrence for semantic reasoning, but their system information and mapping results are not provided. Sun *et al.* [Sun+18] proposed a Unified Convolutional Neural Network which performs Scene Recognition and Object Detection. Luo *et al.* [LC18] developed a semantic mapping framework utilizing spatial room segmentation, CNNs trained for object recognition, and a hybrid map provided by a customized service robot. Niko *et al.* [Sün+16] proposed a transferable and expandable place categorization and semantic mapping system that requires no environment-specific training. Mancini *et al.* [Man+18] addressed Domain Generalization (DG) in the context of semantic place categorization. They also provide results of state-of-the-art algorithms on the VPC Dataset [WCR09] that we compare to. However, most of these results do not test their algorithms on a wide variety of platforms.

Kostavelis *et al.* [KG15] provided a survey of previous work in semantic mapping using robots in the last decade. According to their study, scene annotation augments topological maps based on human input or visual information of the environment. Bormann *et al.* [Bor+16] pointed out that the most popular approaches in room segmentation involve segmenting floor plans based on spatial regions.

An essential aspect of any spatial region is the presence of specific objects in it. Some examples include a bed in a bedroom, a stove in a kitchen, a sofa in a living room, etc. Niko *et al.* [Sän+18] formulated the following three reasoning challenges that address the semantics and geometry of a scene and the objects therein, both separately and jointly: 1) Reasoning About Object and Scene Semantics, 2) Reasoning About Object and Scene Geometry, and 3) Joint Reasoning about Semantics and Geometry. The content in this thesis focuses on the first reasoning challenge and uses Convolutional Neural Networks (CNNs) as feature extractors for both scenes and objects. The goal is to design a system that allows a robot to identify the area where it is located using visual information in a manner similar to how a human being would.

3.4.1 Approach

We consider a set of five different models, abbreviated as Diverse scEne Detection methods in Unseen Challenging Environments (DEDUCE), for place categorization. Each model is derived from two base modules, one based on the PlacesCNN [Zho+17] and the other being an Object Detector-YOLOv3 [RF18]. The classification model can be formulated as a supervised learning problem. Given a set of labeled training data $\mathcal{X}^{tr} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2)...(\mathbf{x}_N, \mathbf{y}_N)\}$, where $\mathbf{x}_i$ corresponds to the data samples and $\mathbf{y}_i$ to the scene labels, the classifier should learn the discriminative probability model

$$p(\hat{\mathbf{y}}_j|\Phi(\mathcal{X}^{tr})) \tag{3.4}$$

where $\hat{\mathbf{y}}_j$ corresponds to the j-th predicted scene label and $\Phi = \{\phi_1, \phi_2...\phi_t\}$ are the set of different feature representations obtained from the $\mathbf{x}_i$. This trained model should be able to correctly classify a set of unlabelled test samples $\mathcal{X}^{te} = \{\mathbf{x}_1, \mathbf{x}_2...\mathbf{x}_M\}$. It is to be noted that while the goal of each of our five models is to perform place categorization, it is the Φ which varies across them. We now describe the two base modules, and how our five models are derived and trained from them. The complete network architecture is given in Figure 3.14.

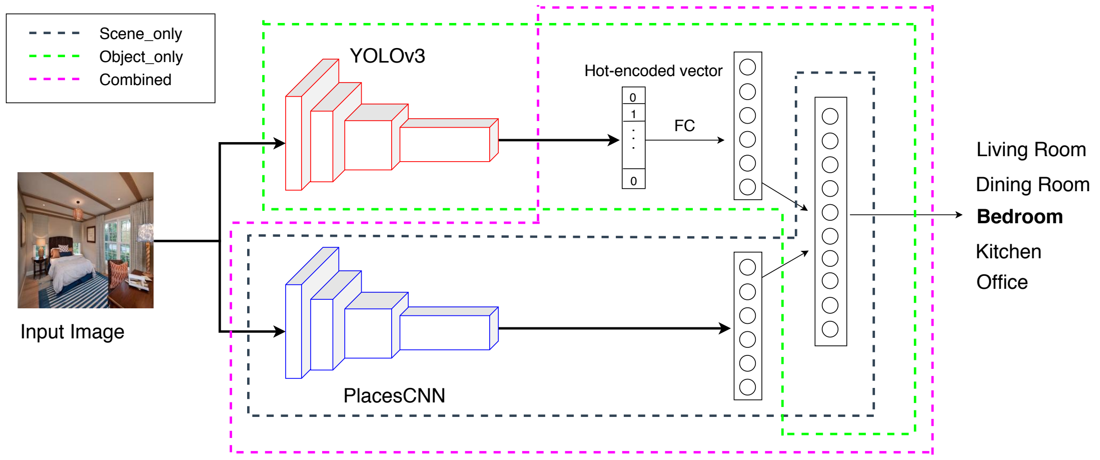


Figure 3.14: Model Architecture. The highlighted regions represent the portion of the network which was trained for the respective models.

3.4.2 Scene Recognition

For scene recognition, we use the PlacesCNN model. The base architecture is that of Resnet-18 [He+16] which has been pre-trained on the ImageNet dataset [Den+09] and then fine-tuned on the Places365 dataset [Zho+17]. We choose seven classes out of the total 365 classes, which are integral to the recognition of indoor home/office environments - Bathroom, Bedroom, Corridor, Dining room, Living room, Kitchen and Office. We use the official training and validation split provided for our work. The training set consists of 5000 labelled images for each scene-class, while the test set contains 100 images for each scene.

3.4.3 Object Detection

Object detection is a domain that has benefited immensely from the developments in deep learning. Recent years have seen people develop many algorithms for object detection, some of which include YOLO [Red+16b; RF17; RF18], SSD [Liu+16], Mask RCNN [He+17], Cascade RCNN [CV18] and RetinaNet [Lin+17]. We work with the YOLOv3 [RF18] detector here, mainly because of its speed, which makes real-time processing possible. It is a Fully Convolutional Network (FCN), and employs the Darknet-53 architecture which has 53 convolution layers, consisting of successive 3x3 and 1x1 convolutional layers with some shortcut connections. The network used here has been pre-trained to detect the 80 classes of the MS-COCO dataset [Lin+14].

3.4.4 Place Categorization models

Scene Only

The first model which we use consists of only the pre-trained and fine-tuned PlacesCNN with a simple Linear Classifier on top of it. This model accounts for a holistic representation of a scene, without specifically being trained to detect objects. Thus, the feature vector for this model is given by $\Phi_{scene} = \phi_s$.

Object Only

The second model acts a Scene classifier using only the information of detected objects. There is no separate training performed here to identify the individual scene attributes. For this purpose, we create a codebook of the most common COCO-objects seen in all the seven scenes. This is shown in Table 3.1. It is to be noted that every object has been associated to only one scene, thereby making it a *landmark*. For this model, the feature representation is given by $\Phi_{obj} = \phi_{\{obj\}}$ where $\{obj\}$ is the set of objects detected in the image.

Table 3.1: Top landmark objects (non-human) for the seven different scene classes

Bathroom	Toilet	Sink	-	-
Bedroom	Bed	-	-	-
Corridor	-	-	-	-
Dining Room	Dining Table	Chair	Wine Glass	Bowl
Kitchen	Oven	Microwave	Refrigerator	-
Living Room	Sofa	Vase	-	-
Office	TV-Monitor	Laptop	Keyboard	Mouse

Scene+Attention

In this model, we compute the activation maps for the given image of a scene, and using those, we try to visualize where the network has its focus during scene classification. From the output of the final block convolutional layer (layer 4) of the WideResnet architecture [ZK16], we get the 14x14 feature blobs which retain the spatial information corresponding to the whole image. Our model is similar to the *soft* attention mechanism of [Xu+15] in the sense that here too, we assign the weights to be the output of a softmax layer, thereby associating a probability distribution to it. However, since we are not classifying based on a sequence of images, we do not employ a recurrent network to compute the sequential features. Instead, we simply utilize the weights of the final FC layer and take its dot product with the feature blobs to obtain the heatmap. The final step is to up-sample this 14x14 heatmap to the input image size, and then overlay it on top to obtain the activation mask $m(x_n)$ of the input image x_n. Therefore, the feature representation for this model is $\Phi_{attn.} = \phi_{m(x_n)}$. The basic architecture is given in Figure 3.15.

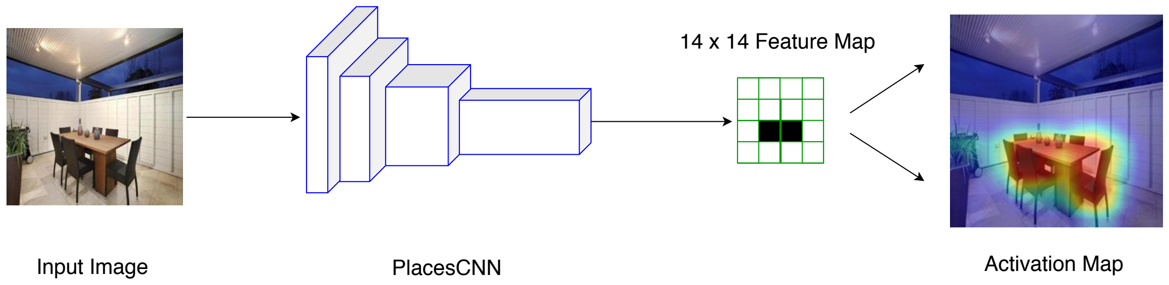


Figure 3.15: Architecture for Generation of the Activation Map. The 14x14 feature maps obtained from the block layer 4 of WideResNet are combined with the weights from the final FC layer, and then their dot product id up-sampled to the image size and overlaid on top to get the activation maps

Combined

In this model, we use the PlacesCNN mentioned above as a feature extractor to give the semantics of a scene. In addition, the YOLO detector gives us the information regarding the objects present in the image. Given an image of a scene, our model creates a hot-encoded vector of 80 dimensions, corresponding to the object classes of MS-COCO, with only the indices of the detected objects set to 1. We then concatenate this vector along with that of the output of the scene feature extractor, and train a Linear Classifier on top of it. Since we combine the two different features of scene and objects, the feature representation here is given by $\Phi_{comb.} = \{\phi_s, \phi_{\{obj\}}\}$.

Scene+N-best objects

Our final model is similar to the above in the sense that here also, we use both the PlacesCNN and the YOLO detector. However, this model does not need to be retrained again and so, it is significantly faster. For this model, we place a certain confidence threshold on the scene detector, and only when the probability of classification is below this threshold, we search for the information about specific objects in the scene (as obtained from Table 3.1). The reason for introducing this as a new model is two-fold. Firstly, we eliminate the scenario of looking at every object present since it is often redundant, given the semantics of the scene. Secondly, this is similar to how we as human beings operate when we come across an unknown scene. The feature representation for this model is given by $\Phi_{N-best} = \{\phi_s, \phi_{\{N-obj\}}\}$.

3.4.5 Experiments and Results

We evaluated our five models described above on a number of platforms. In this section, we first describe our training procedure, and then talk about the different experiment settings used for evaluation.

Training Procedure

As mentioned in Section 3.4.1, the base architecture for our scene classifier is the ResNet-18 architecture. The data pre-processing and training process is similar to [Zho+17]. We used the Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 0.1, momentum of 0.9, and a weight decay of 10^{-4}. For the Φ_{scene} and the $\Phi_{attn.}$ models, the training was performed for 90 epochs with the learning rate being decreased by a factor of 10 every 30 epochs. The $\Phi_{comb.}$ model converged much faster and so, it was only trained for 9 epochs, with the learning rate reduced by 10 times after every 3 epochs. For all the 3 training procedures, the cross-entropy loss function was optimized, which minimizes the cost function given by

$$J(\hat{\mathbf{y}}_j, \mathbf{y}_j) = -\frac{1}{N}(\sum_{j=1}^{N} \mathbf{y}_j \odot log(\hat{\mathbf{y}}_j)) \tag{3.5}$$

The training process was carried out on a NVIDIA Titan Xp GPU using the PyTorch framework. The performance of the five DEDUCE algorithms on the test set of Places365 is shown in Table 3.2 for the seven classes chosen.

Table 3.2: Accuracy in percentage of DEDUCE on Places365 dataset

Scenes	Φ_{scene}	Φ_{obj}	$\Phi_{attn.}$	$\Phi_{comb.}$	Φ_{N-best}
Dining room	79	**94**	75	79	80
Bedroom	90	74	90	90	**91**
Bathroom	**92**	65	**92**	91	**92**
Corridor	94	90	**99**	96	94
Living Room	**84**	25	68	80	**84**
Office	85	29	76	**94**	83
Kitchen	**87**	62	70	**87**	**87**
Avg	87.3	62.6	81.4	**88.1**	87.3

Experiment Settings

In order to check the robustness of our models, we further evaluated their performance on two state-of-the-art still-image datasets.

SUN Dataset

The SUN-RGBD dataset [SLX15] is one of the most challenging scene understanding datasets in existence. It consists of 3,784 images using Kinect v2 and 1,159 images using Intel RealSense cameras. In addition, there are 1,449 images from the NYUDepth V2 [Sil+12], and 554 manually selected realistic scene images from the Berkeley B3DO Dataset [Jan+13], both captured by Kinect v1. Finally, it has 3,389 manually selected distinguished frames without significant motion blur from the SUN3D videos [XOT13] captured by Asus Xtion. Out of this, we sample the seven classes of importance and use the official test split to evaluate our models. We only consider the RGB images for this work since our training data doesn't have depth information. The performance is summarized in Table 3.3.

Upon comparison with Table 3.2, which contains the results on the Places365 dataset where our models were fine-tuned, a number of observations can be made which are consistent for both the datasets. Firstly, the $\Phi_{comb.}$ model performs the best. This is intuitive since here, the scene classification is done using the combined training of both the information about the scene attributes and the object identity. Secondly, the Φ_{obj} model works the best for the *Dining*

Table 3.3: Accuracy in percentage of DEDUCE on SUN dataset

Scenes	Φ_{scene}	Φ_{obj}	$\Phi_{attn.}$	$\Phi_{comb.}$	Φ_{N-best}
Dining room	65.2	**83.7**	53.3	67.4	72.8
Bedroom	43.7	36.5	**48.9**	**48.9**	47.3
Bathroom	94.5	87.0	**97.3**	96.6	95.2
Corridor	44.4	**67.6**	**67.6**	44.4	41.7
Living Room	58.8	24.0	43.6	**59.2**	58.8
Office	84.0	12.6	75.8	**90.6**	80.6
Kitchen	77.1	63.5	63.9	**83.8**	77.4
Avg	66.8	53.6	64.3	**70.1**	67.7

Room class, even though its overall performance is the worst. This trend can be attributed to the fact that dining rooms can be easily identified by the presence of specific objects, whereas the scene attributes might throw in some confusion (for instance when the kitchen/living room is partially visible in the image of a dining room). Thirdly, for *Corridor*, the performance of the $\Phi_{attn.}$ model is best for both the datasets. This supports the fact that in order to classify a scene like a corridor, viewing only a small portion of the image close to the vanishing point is sufficient. Finally, the Φ_{N-best} model performs just as good or better than the Φ_{scene} model. This proves that presence of objects does indeed improve the scene classification. For the best performance using the Φ_{N-best} model, the threshold was set to 0.5 for the *Places* dataset while it was 0.6 for the *SUN* dataset. The reason for the higher confidence on scene attributes for *Places* dataset is most likely due to the fact that the scene classifier itself was fine tuned on it.

VPC Dataset

The Visual Place Categorization dataset [WCR09] consists of videos captured autonomously using a HD camcorder (JVC GR-HD1) mounted on a rolling tripod. The data has been collected from 6 different home environments, and three different floor types. The advantage of this dataset is that the collected data closely mimics that of the motion of a robot - instead of focusing on captured frames or objects/furniture in the rooms, the operator recording the data just traversed across all the areas in a room while avoiding collision with obstacles. For comparison with the state-of-the-art algorithms, we test our methods only on the five classes which are present in all the homes - Bathroom, Bedroom, Dining room, Living room and Kitchen. Table 3.4 contains the results for the individual home environments for these five classes. For the AlexNet [KSH12] and ResNet [He+16] models, we adopt the same training procedure as in [Man+18]. It can be seen from the table that our models perform better than the rest in all but one of the home environments and much better in the overall performance.

Table 3.4: VPC Dataset: Average Accuracy across the 6 home environments

Networks	H1	H2	H3	H4	H5	H6	avg.
AlexNet	49.8	53.4	49.2	64.4	41.0	43.4	50.2
AlexNet+BN	54.5	54.6	55.6	69.7	41.8	45.9	53.7
AlexNet+WBN	54.7	51.9	61.8	70.6	43.9	46.5	54.9
AlexNet+WBN*	53.5	54.6	55.7	68.1	44.3	49.9	54.3
ResNet	55.8	47.4	64.0	69.9	42.8	50.4	55.0
ResNet+WBN	55.7	49.5	**64.7**	70.2	42.1	52.0	55.7
ResNet+WBN*	56.8	50.9	64.1	69.3	45.1	51.6	56.5
Ours (Φ_{scene})	**63.7**	57.3	63.7	**71.4**	60.2	65.9	63.7
Ours ($\Phi_{comb.}$)	**63.7**	**60.7**	64.5	70.7	**65.7**	**68.8**	**65.7**

Table 3.5 further compares our models with all other baseline algorithms tested on the VPC dataset. The reported accuracies are the average over all the six home environments. We first consider the methods described in [WCR09], which use SIFT and CENTRIST features with a Nearest Neighbor Classifier, and also exploit temporal information between images by coupling them with Bayesian Filtering (BF). Next, we look at the approach of [FT12] where Histogram of Oriented Uniform Patterns (HOUP) is used as input to the same classifier. [YW12] proposed the method of using object templates for visual place categorization, and reported results for Global configurations approach with Bayesian Filtering (G+BF), and that combined with the object templates (G+O(SIFT)+BF). Ushering the deep learning era, AlexNet [KSH12] and ResNet [He+16] architectures give better results, both with their base models, as well as the Batch Normalized (BN) and the Weighted Batch Normalized versions [Man+18]. However, comparisons with our Φ_{scene} and $\Phi_{comb.}$ models show that our methods beat all the other results

Table 3.5: VPC Dataset: Comparison with State Of The Art

Method	[WCR09]				[FT12]	[YW12]		AlexNet			ResNet		**Ours**	
Config.	SIFT	SIFT+BF	CE	CE+BF	HOUP	G+BF	G+O(SIFT)+BF	Base	BN	WBN*	BN	WBN*	**Scene-only**	**Combined**
Acc.	35.0	38.6	41.9	45.6	45.9	47.9	50	50.2	53.7	54.3	55.0	56.5	**63.7**	**65.7**

Table 3.6: VPC Dataset: Φ_{scene} model results for individual homes

	Home 1			Home 2			Home 3	Home 4		Home 5		Home 6		**Avg. (ours)**	Avg. [FT12]
	0	1	2	0	1	2	1	1	2	1	2	0	1		
Dining Room	-	71.5	-	-	43.5	-	72.4	49.4	-	19.6	-	-	52.0	**51.4**	15.5
Bedroom	-	32.0	43.1	28.8	-	41.1	41.7	60.7	45.7	-	51.5	43.5	50.3	43.8	**73.0**
Bathroom	83.4	63.6	50.2	85.0	-	86.5	79.0	96.1	91.6	96.3	59.6	84.3	67.0	**78.6**	68.7
Living Room	-	97.6	-	-	37.4	-	72.9	79.0	-	76.2	-	-	77.6	**73.5**	25.9
Kitchen	-	40.0	-	-	72.5	-	52.3	85.7	-	67.5	-	-	92.8	**68.5**	46.6
Avg.	**83.4**	**61.0**	**46.7**	**56.9**	**51.1**	**63.8**	**63.7**	**74.2**	**68.6**	**64.9**	**55.5**	**63.9**	**68.0**		

Table 3.7: VPC Dataset: $\Phi_{comb.}$ model results for individual homes

	Home 1			Home 2			Home 3	Home 4		Home 5		Home 6		**Avg.(ours)**	Avg. [FT12]
	0	1	2	0	1	2	1	1	2	1	2	0	1		
Dining Room	-	83.9	-	-	47.7	-	79.8	48.8	-	25.00	-	-	47.1	**55.4**	15.5
Bedroom	-	32.2	37.4	31.9	-	49.6	53.0	66.2	41.7	-	56.6	51.8	58.8	47.9	**73.0**
Bathroom	82.4	58.9	48.3	86.4	-	82.1	79.9	91.8	90.5	96.6	63.3	83.0	64.9	**77.3**	68.7
Living Room	-	97.6	-	-	49.7	-	54.2	82.0	-	81.5	-	-	88.1	**75.5**	25.9
Kitchen	-	56.0	-	-	73.7	-	55.6	87.9	-	83.2	-	-	92.3	**74.8**	46.6
Avg.	**82.4**	**65.7**	**42.9**	**59.1**	**57.0**	**65.9**	**64.5**	**75.4**	**66.1**	**71.6**	**59.9**	**67.4**	**70.3**		

by significant margins.

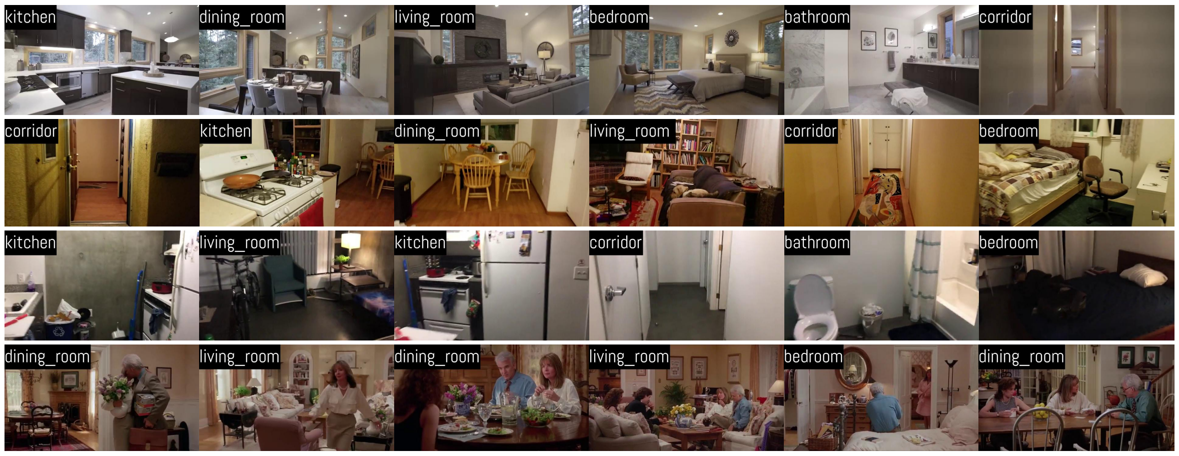


Figure 3.16: Detection Results on Real-World Videos. Top row corresponds to the video of a Real-Estate model house. The next 2 rows are from the houses of the authors and their friends. The bottom row is obtained from a house in the movie *"Father of the Bride"*.

Real-World Scene Recognition

In order to test the robustness of our DEDUCE models, we expand the domain of test cases beyond the aforementioned still image data sets. Figure 3.16 does this by showing the results of scene recognition on real-world data recorded using hand-held cameras. We employ the Φ_{N-best} model for these cases due to its ability to mimic the natural behavior of humans, whereby an initial prediction is made based on the scene attributes, and if unsure, more information related to specific objects is gathered in order to update/re-inforce the initial prediction. The top row corresponds to the tour of a semi-furnished real estate home obtained from YouTube which only has the relevant objects in the scene. Although it is not a sequential tour of the house, it does contain all the rooms. Also, professional photographers captured this video and hence, the image quality and white-balance of the camera is pretty good. The next-2 rows pose a more challenging case as they correspond to homes currently inhabited by people. We consider two examples of these houses, one which is a standard bungalow residence, while the other being a student apartment. From experience, the bungalow is a much cleaner home, whereas student apartments are prone to presence of cluttered objects and overlapping scene boundaries. Moreover, the videos were recorded by the inhabitants using their cellphone cameras. This inherently brings motion blur into the picture, specially during scene transitions. Finally, the last row depicts the settings of a house from the movie *"Father of the Bride"*. This ensures that our model is robust enough to classify scenes even when the focus of the recording is on people instead of the background settings. All the detection results mentioned in this chapter are available as individual videos in the following link `https://goo.gl/sYyVZ2`.

3.4.6 Semantic Mapping

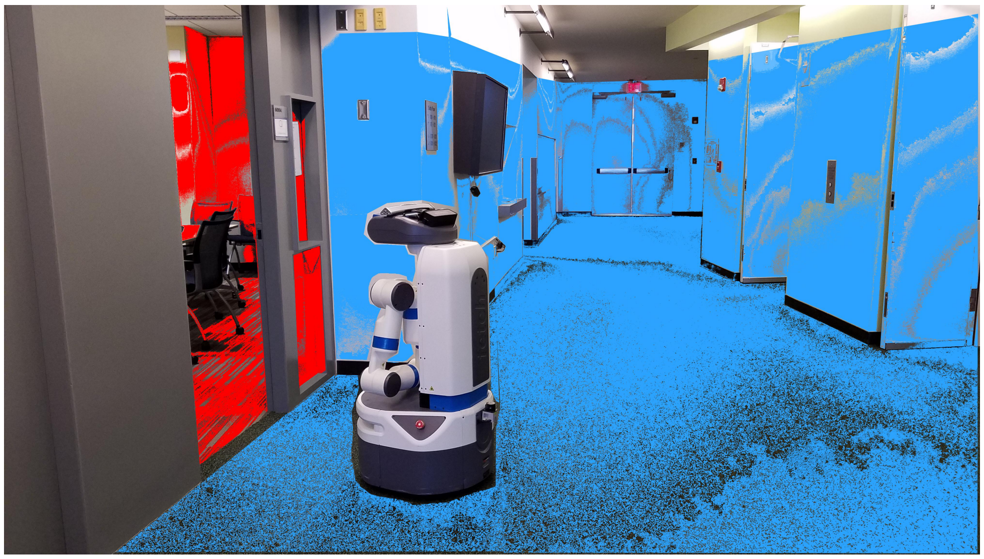
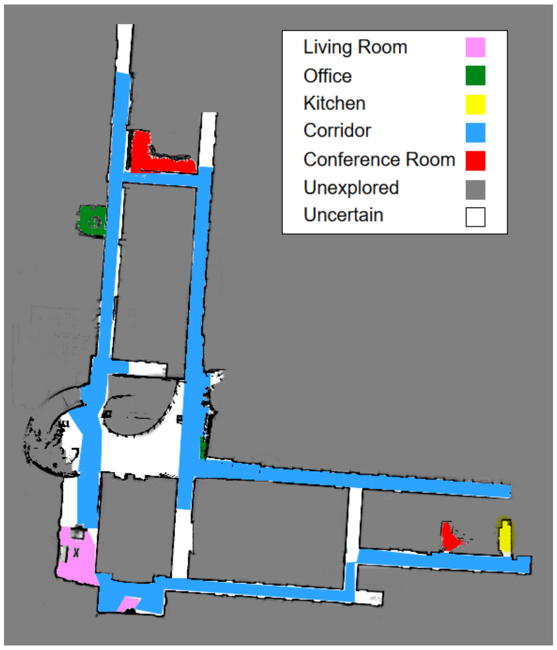


Figure 3.17: Visualization of the semantic mapping performed while the Fetch robot is navigating through the environment.

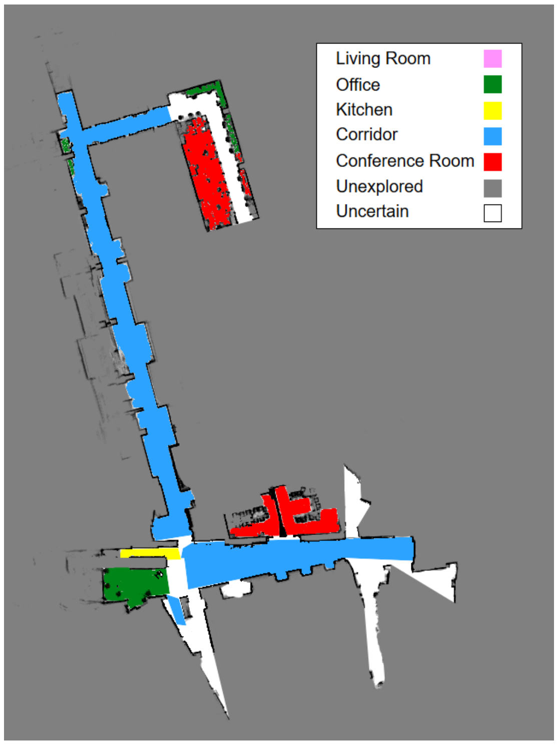
The experimental setting for semantic mapping involves running our algorithm on a mobile robot platform in two different environments. The platform is a Fetch Mobile Manipulator and Freight Mobile Robot Base by Fetch Robotics[†]. Figure 3.17 shows the robot performing scene classification in one of the environments.

[†]`https://fetchrobotics.com/robotics-platforms/`

The semantic maps for the experiments were constructed using our mapping system Omnimapper [TRC14a] described in chapter 2. It utilizes the GTsam library in order to optimize a graph of measurements between robot poses along a trajectory, and between robot poses and various landmarks in the environment. The measurements of simple objects like points, lines, and planes are data associated with mapped landmarks using the joint compatibility branch and bound (JCBB) technique [NT01b]. The regions for color segmentation are acquired by the Gaussian Region algorithm of [Nie+10b]. However, in our gaussian region approach described in section 3.2 and [Nie+10b], the map partitions were built through human guidance, whereby the robot was taken on a tour of the space (either by driving the robot manually, or using a person following behavior) and the respective scene labels were taught to it. This is in contrast to our approach, where the labels are learned from our visual place categorization system. Thus, the robot is itself capable of identifying the scenes without any human guide. We used the Φ_{N-best} model for this task, and retrained the scene classifier to exclude the *Bedroom*, *Dining Room* & *Bathroom* scenes, and instead include *Conference Room* as it is more likely to occur in an academic building environment. Figure 3.18a shows the navigation of the robot in the Computer Science and Engineering (CSE) Building. Our system was able to classify the seven regions of the floor map. However, there are some regions detected by Omnimapper using the laser range finder. These are painted in white to denote their invisibility to the camera. The second test environment is the Contexual Robotics Institute (CRI) building, which has a very different floor map in comparison to CSE. The result of the run made here is shown in Figure 3.18b.



(a) Semantic Map of the CSE building.



(b) Semantic Map of the CRI.

Figure 3.18: Place categorization experiments with mobile robots. Each color represents one of the seven classes of the visual place categorization that our system classified.

3.4.7 Discussion

In our approach we considered five different models for performing place categorization, which are derived mainly from two base modules - a scene recognizer, and an object detector. We demonstrated the effectiveness of our algorithms in a series of experiments, ranging from scene recognition in still-image data sets to real-world videos captured from different sources, and finally via the generation of labeled semantic maps using data gathered by multiple mobile robot platforms. We showed that (i) different models are favorable for different scenes (Table 3.2 and 3.3), and thus the ideal scene recognition system would likely be a combination of these five models, (ii) the proposed methods give successful results on many different types of video recordings, even when they are affected by object clutter, motion blur, and overlapping boundaries and (iii) our models are robust enough to be tested on data gathered by mobile robotic platforms on multiple building scenarios which are affected by occlusions and poor lighting conditions.

Acknowledgment

This chapter appears in the following publications:

- Anwesan Pal, Carlos Nieto-Granda and Henrik Christensen. DEDUCE: Diverse scEne Detection methods in Unseen Challenging Environments. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. Macau, China, November 4-8, 2019.

- C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Object guided autonomous exploration for mobile robots in indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XVI*, volume 9084, Baltimore, MD, April 2014.

- Carlos Nieto-Granda, John Rogers, Alexander J. B. Trevor, and Henrik Iskov Christensen. Semantic map partitioning in indoor environments using regional analysis. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 1451–1456. Taipei, Taiwan, October 18-22, 2010.

- Alex Trevor, John Rogers, Carlos Nieto-Granda,and Henrik Iskov Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3d. *In the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop: Semantic Mapping and Autonomous Knowledge Acquisition. Taipei, Taiwan, October 18-22, 2010.

I would like to thank my co-authors John G. Rogers III, Alexander J. B. Trevor, Siddharth Choudhary, Varun Murali, Jeff Twig, Anwesan Pal, and Prof. Henrik Christensen.

Chapter 4

Autonomous Exploration and Navigation

4.1 Introduction

Multi-robot mapping and exploration was addressed by Fox et al. [Fox+06] and Vincent et al. [Vin+08]. These papers build a map using up to three robots with a decision theoretic planner that trades off robot rendezvous operations with frontier exploration. These robots rendezvous to determine their relative pose transforms to provide constraints to recover the final map. In contrast, our approach does not require this rendezvous step because landmarks are globally data associated between each robot using a centralized map coordinator. The exploration strategy used is similar to our strategy called Reserves; however, we will not use a rendezvous step and do not require a decision-theoretic planner. In Olson et al. [Ols+12] the authors describe a system which controls a team of up to 14 mobile robots in an urban reconnaissance mission. In this system, a planning algorithm allocates robots to explore navigation goals in an unknown environment and build a map.

Robots are coordinated in team collaboration behavior to accomplish other goals of the MAGIC 2010 competition, but they do not move together in formations to maximize availability of additional robots. In Hollinger et al. [HSK10], the authors prove performance characteristics on a multi-robot collaboration strategy to perform adversary search. By representing the topological configuration of a map as a graph, the robots can guarantee that the adversarial search will prevent recontamination of previously cleared nodes with an arbitrarily sized team. In Joyeux et al. [Joy+09], the authors describe a distributed system for managing robot plans for performing high-level tasks. This architecture prevents conflicts between robot plans and can handle communication failures. These papers both present strategies and architectures for collaboration between robot agents to perform tasks.

In [Sim+00b] and [Bur+05], the authors present a coordination algorithm for multi-robot exploration and mapping which tries to maximize information gain through greedy assignment of robots to exploration frontiers. Since the allocation of one robot to an exploration frontier will diminish the predicted information gain to that frontier when allocating other robots, the team will tend to spread out. The authors recognize an important property that their system exhibits of minimizing interference between platforms; one of the key differences between the coordination strategies which we will evaluate in this paper is this property of platform interference. The authors in [Sim+00b] describe a collaboration strategy which is most similar to the *Reserves* strategy, in that idle robots remain until new exploration goals with sufficient predicted information gain require their attention. The authors of [Sim+00b] allude to potential future work where idle robots move to near the exploration frontier to try to be ready for new exploration goals.

Distributed estimation in multi robot systems is currently an active field of research, with special attention being paid to communication constraints [Pau+15], heterogeneous teams [Bai+11; Ind+12], estimation consistency [BWL09], and robust data association [Ind+14; Don+15]. Robotic literature offers distributed implementations of different estimation techniques, including Extended Kalman filters [RB02; ZR06], information filters [TL03], and particle filters [How06; Car+14]. More recently, the community reached a large consensus on the use of maximum likelihood (*ML*) estimation (maximum a-posteriori, in presence of priors), which, applied to trajectory estimation, is often referred to as *pose graph optimization* or *pose-based SLAM*. *ML* estimators circumvent well-known issues of Gaussian filters (e.g., build-up of linearization errors) and particle filters (e.g., particle depletion), and frame the estimation problem in terms of nonlinear optimization. In multi robot systems, *ML* trajectory estimation can be performed by collecting all measurements at a centralized inference engine, which performs the optimization [AN08; Kim+10; Bai+11]. Variants of these techniques invoke partial exchange of raw or preprocessed sensor data [Laz+11; Ind+14].

In many applications, however, it is not practical to collect all measurements at a single inference engine. When operating in a hostile environment, a single attack to the centralized inference engine (e.g., one of the robot) may threaten the operation of the entire team. Moreover, the centralized approach requires massive communication and large bandwidth. Furthermore, solving trajectory estimation over a large team of robots can be too demanding for a single computational unit. Finally, the centralized approach poses privacy concerns as it requires to collect all information in a single robot; if an enemy robot is able to deceive the other robots and convince them that it is part of the team, it can easily gather sensitive information (e.g., trajectory covered and places observed by every robot). These reasons triggered interest towards *distributed trajectory estimation*, in which the robots only exploit local communication, in order to reach a consensus on the trajectory estimate.

Cunningham et al. [CPD10] use Gaussian elimination, and develop an approach, called *DDFSAM* (Decentralized Data Fusion Smooth and Mapping), in which each robot exchanges a Gaussian marginal over the *separators* (i.e., the variables shared by multiple robots); the approach is further extended [CID13], to avoid storage of redundant data.

Grisetti et al.[GKN12] compute a good initial estimate for global alignment through a submapping approach. [ZHD13] propose an approximation for large-scale SLAM by solving for a sequence of submaps and joining them in a divide-and-conquer manner using linear least squares. [Sug+14] present an approximate SLAM approach based on hierarchical decomposition to reduce the memory consumption for pose graph optimization. While Gaussian elimination has become a popular approach it has two major shortcomings. First, the marginals to be exchanged among the robots are dense, and the communication cost is quadratic in the number of separators. This motivated the use of sparsification techniques to reduce the communication cost [Pau+15]. The second reason is that Gaussian elimination is performed on a linearized version of the problem, hence these approaches require good linearization points and complex bookkeeping to ensure consistency of the linearization points across the robots [CID13]. The need of a linearization point also characterizes gradient-based techniques [KB13]. In many practical problems, however, no initial guess is available, and one has to develop ad-hoc initialization techniques, e.g., [Ind+14].

4.2 Linguistic Composition of Semantic Maps and Hybrid Controllers

This framework provides an approach to generate robot policies by automatically combining Semantic Mapping and Hybrid Control. Semantic mapping and hybrid control are both effective approach within robotics. Semantic mapping produces detailed models of *unstructured environments* [Nie+10b; TNC10; TC10; ORD09; TRC12]; however, this approach provides no direct link to robot action. Hybrid models combine continuous and discrete robot dynamics to efficiently and verifiably represent *robot action* [DKS11; DS11a; DS12; Alu+93; Hen96; BK04]; however, there is no automatic method to produce control models for large, complicated *systems*. While superficially, it appears that semantic mapping and hybrid control are fundamentally different approaches, they are actually closely related. The topological graph of a semantic map and the discrete event system of a hybrid control model are both instances of *formal language*. Thus, we propose to combine the linguistic representations of semantic maps and robot action models to produce an efficient and verifiable control policy for mobile manipulation in unstructured environments.

This work focuses on the application domain of service robots in human environments. Previously, we developed new techniques for mapping using Semantic SLAM [Nie+10b; TNC10] and for hybrid systems using our Motion Grammar [DKS11; DS11a; DS12]. Here, we integrate these approaches to produce a combined robot-environment action model. Then, we apply established methods in supervisory control [CL08] to derive a robot control policy for a mobile manipulation task. This control design approach formally guarantees that the resultant policy satisfies the task specification. Finally, we demonstrate of this approach on a Segway RMP-200 mobile robot.

4.2.1 Approach

The method of this paper produces a robot control policy for unstructured environments by combining Simultaneous Localization and Mapping (SLAM) with Hybrid Control. We combine these two approaches through Formal Language. First, we produce a basic grammar for the robot's actions and generate the map of the environment via SLAM. Then we compose the action grammar and environment map using the Motion Grammar Calculus. Finally, we apply a supervisory controller to generate the policy for the robot. We now explain some background on formal language, define our hybrid systems model, the Motion Grammar, and summarize the SLAM technique.

4.2.2 Formal Language

Formal language is the underlying representation we use to combine mapping and hybrid control. Language and automata theory provide a rigorous method for reasoning about the discrete dynamics of a robotic system. A *formal language* is a set of strings. Strings are sequences of atomic symbols which we can use to describe discrete events, predicates, locations, or actions within our system. A *grammar* defines a formal language. We first briefly review some relevant points of language theory. For a thorough coverage of formal language and its applicability to robotic systems, please see [HU79; CL08; DS11a].

Definition 1 (Context-Free Grammar, CFG)

$G = (Z,V,P,S)$ where Z is a finite alphabet of symbols called *tokens*, V is a finite set of symbols called *nonterminals*, P is a finite set of mappings $V \mapsto (Z \cup V)^*$ called *productions*, and $S \in V$ is the *start symbol*.

The productions of a CFG are conventionally written in Backus-Naur form. This follows the form $A \to X_1X_2 \dots X_n$, where A is some nonterminal and $X_1 \dots X_n$ is a sequence of tokens and nonterminals. This indicates that A may *expand* to all strings represented by the right-hand side of the productions. The symbol ε is used to denote an empty string. For additional clarity, nonterminals may be represented between angle brackets $\langle\rangle$ and tokens between square brackets [].

Grammars have equivalent representations as *automata* which *recognize* the language of the grammar. This automata form provides a more convenient representation for some tasks, such as defining languages for maps in Section 4.2.6. The equivalence of grammars and automata means that we can freely choose whichever representation is most convenient. In the case of a Regular Grammar – where all productions are of the form $\langle\mathrm{A}\rangle \to [\mathrm{a}]\,\langle\mathrm{B}\rangle$, $\langle\mathrm{A}\rangle \to [\mathrm{a}]$, or $\langle\mathrm{A}\rangle \to \varepsilon$ – the equivalent automaton is a Finite Automaton (FA), similar to a Transition System with finite state. A CFG is equivalent to a Pushdown Automaton, which is an FA augmented with a stack; the addition of a stack provides the automaton with memory and can be intuitively understood as allowing it to count.

Definition 2 (Finite Automata, FA)

$M = (Q,Z,\delta,q_0,F)$, where Q is a finite set of *states*, Z is a finite alphabet of *tokens*, $\delta : Q \times Z \mapsto Q$ is the *transition function*, $q_0 \in Q$ is the *start state*, $F \in Q$ is the set of *accept states*.

Definition 3 (Acceptance and Recognition)

An automaton M *accepts* some string σ if M is in an accept state after reading the final element of σ. The set of all strings that M accepts is the *language* of M, $\mathfrak{L}_M$, and M is said to *recognize* $\mathfrak{L}_M$.

Regular Expressions [HU79] and Linear Temporal Logic (LTL) [BK+08] are two alternative notations for finite state languages. These representations are convenient forms for defining supervisory controllers as in Section 4.2.8. The basic Regular Expression operators are concatenation $\alpha\beta$, union $\alpha|\beta$, and Kleene-closure α^*. Some additional common Regular Expression notation includes $\neg\alpha$ which is the complement of α, the dot (.) which matches any token, and $\alpha?$ which is equivalent to $\alpha|\varepsilon$. Regular Expressions are equivalent to Finite Automata and Regular Grammars. LTL extends propositional logic with the binary operator *until* $\cup$ and unary prefix operators *eventually* $\Diamond$ and *always* $\Box$. LTL formula are equivalent to Büchi automata, which represent *infinite* length strings, termed ω-Regular languages. We can also write ω-Regular Expressions by extending classical Regular expressions with infinite repetition for some α given as α^ω. These additional notations are convenient representations for finite state supervisors.

Any string in a formal language can be represented as a *parse tree*. The root of the tree is the start symbol of the grammar. As the start symbol is recursively broken down into tokens and nonterminals according to the grammar syntax, the tree is built up according to the productions that are expanded. A production $A \rightarrow X_1 \ldots X_n$ will produce a piece of the parse tree with parent A and children $X_1 \ldots X_n$. The children of each node in the parse tree indicate which nonterminals or tokens that node *expands* to in a given string. Internal tree nodes are nonterminals, and tree leaves are tokens. The parse tree conveys the full syntactic structure of the string.

While grammars and automata describe the structure or *syntax* of strings in the language, something more is needed to describe the meaning or *semantics* of those strings. One approach for defining semantics is to extend a CFG with additional *semantic rules* that describe operations or actions to take at certain points within each production. Additional values computed by a semantic rule may be stored as *attributes*, which are parameters associated with each nonterminal or token, and then reused in other semantic rules. The resulting combination of a CFG with additional semantic rules is called a *Syntax-Directed Definition* (SDD) [Aho+07, p52].

4.2.3 The Motion Grammar

Next, we model robot action using the Motion Grammar (MG), giving an initial set of hybrid control actions the robot can perform. MG represents the operation of a robotic system as a Context-Free language, augmenting a Context-Free Grammar with additional variables to handle the continuous dynamics. We use this combined representation to describe the operation of the full robotic *system* [DKS11; DS11a].

Definition 4 The Motion Grammar is a tuple

$$\mathcal{G}_M = (Z, V, P, S, \mathcal{X}, \mathcal{Z}, \mathcal{U}, \eta, K)$$

where:

Z	set of events, or tokens
V	set of nonterminals
$P \subset V \times (Z \cup V \cup K)^*$	set of productions
$S \in V$	start symbol
$\mathcal{X} \subseteq \Re^m$	continuous state space
$\mathcal{Z} \subseteq \Re^n$	continuous observation space
$\mathcal{U} \subseteq \Re^p$	continuous input space
$\eta : \mathcal{Z} \times P \times \mathbb{N} \times \mapsto Z$	tokenizing function

$$K \subset \mathcal{X} \times \mathcal{U} \times \mathcal{Z} \mapsto \mathcal{X} \times \mathcal{U} \times \mathcal{Z} \qquad \text{semantic rules}$$

The Motion Grammar describes the *language* of the robotic system. The terminal symbols of this language are robot events and predicates, representing a discrete abstraction of the system path.

We use two properties to ensure the validity of a system modeled as a Motion Grammar: *completness* and *correctness*. Completeness ensures that our model $\mathcal{G}$ is a faithful representation of the physical system F. We define this property using the *simulation* relation, that all paths in F are also paths in $\mathcal{G}$. Correctness ensures that our model $\mathcal{G}$ satisfies some desired property S. We define correctness using the subset relation.

Definition 5

Given $\mathcal{G}_M$ and system F then $\mathsf{complete}\{\mathcal{G}\} \equiv F \preceq \mathcal{G}_M$

Definition 6

A Motion Grammar $\mathcal{G}$ is correct with respect to some specification S when all strings in the language of $\mathcal{G}$ are also in S: $\mathsf{correct}\{\mathcal{G}, \mathsf{S}\} \equiv \mathfrak{L}(\mathcal{G}) \subseteq \mathfrak{L}(S)$

4.2.4 Semantic Simultaneous Localization and Mapping

Our mapping system identifies surfaces and connected free spaces in the world [Tre+10a; TRC12]. We use the surfaces, such as walls and tables, to localize the robot based on its relative position to these object. We represent free spaces as Gaussian regions in $\Re^3$ with mean at the center of the free space and standard deviation indicating the dimensions of the free space [Nie+10a]. Topological connections between these Gaussian regions indicate connected free spaces in the environment. For example, a door or hallway between two rooms would connect the Gaussian regions for those rooms.

We then extend the metric and topological information of the map surfaces and connected Gaussians with additional semantic information by labeling each of the Gaussian regions. These *Semantic Maps* provide useful information for navigation and localization of the robot.

In addition, the semantic content of the map permits higher-level reasoning about the spatial regions of the environment. We exploit this semantic information in our composition of the map with a grammar for robot action.

4.2.5 Composing Maps and Grammars

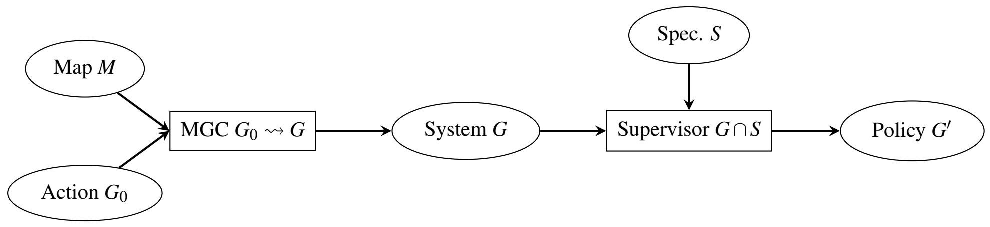
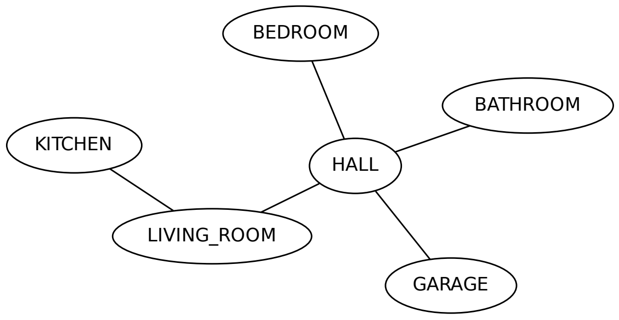


Figure 4.1: Sequence of operations to generate policy.

We produce the control policy for the robot by composing a semantic map and a base action grammar, following Figure 4.1. We will explain this approach using the example map for the Georgia Tech Aware home, Figure 4.2(a), and the base grammar for mobile manipulation, Figure 4.2(b). First, we convert the map graph into a grammar for the map language. Then, we compose the map grammar and the action grammar using the Motion Grammar Calculus (MGC) to model the robotic system operating within the mapped environment. Finally, we produce a task policy by applying a supervisory controller to this system model.

4.2.6 Map Languages

To better analyze the semantic map, we first represent this map using formal language. The Gaussian free space regions of the map are arranged in a graph, indicating connectivity between these regions. The graph for the Aware Home is Figure 4.2(a). This graph is equivalent to a Regular Language representing the set of all traces through the map.



(a) Semantic Map M

$$\begin{array}{rcl} \langle \mathrm{S} \rangle & \rightarrow & [\mathrm{room}]\,\langle \mathrm{S} \rangle \\ & | & [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{S}' \rangle \\ \langle \mathrm{S}' \rangle & \rightarrow & [\mathrm{room}]\,\langle \mathrm{S}' \rangle \\ & | & [\mathrm{place}]\,\langle \mathrm{S} \rangle \end{array}$$

(b) Base Grammar G_0

Figure 4.2: Example of Semantic Map M and base manipulation grammar G_0. This map represents the Georgia Tech Aware Home.

Definition 7

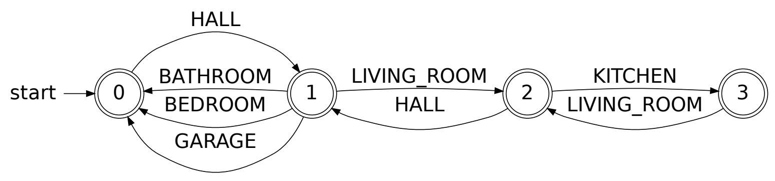
Let Map $M = (N, V)$, where N is a finite set of location symbols, and $V \subset N \times N$ is the set of adjacent symbols $n_i \rightarrow n_j$.

We can transform any Map M into a regular grammar. We note that when analyzing Finite Automata, the language symbols are typically given along transitions [HU79; Aho+07] wheres in a map, location symbols mark a state. For regular languages, these two conventions – terminal language symbols on states and terminal language symbols on edges – are equivalent.

```
Input: $Q$ ;                                    // Initial States
Input: $E : Q \times Q$ ;                       // Initial Edges
Output: $Q'$ ;                                  // Final States
Output: $Z'$ ;                                  // Edge Symbols
Output: $E' : Q' \times Z' \times Q'$ ;         // Final Edges
$Z' = Q$;
$Q' = E$ ;
$E' = \emptyset$;
forall $q \in Q$ do
    forall $(e_i = Q \rightarrow q) \in E$ do
        forall $(e_j = q \rightarrow Q) \in E$ do
            $E' = E' \cup e_i \xrightarrow{q} e_j$
        end
    end
end
```

Algorithm 5: State to Edge Symbols

Algorithm 5 transforms the state terminal map to an edge terminal automaton. Then, we can directly convert this automaton to a Regular Grammar.



$$
\begin{array}{lcl}
\langle 0 \rangle & \rightarrow & [\text{hall}]\,\langle 1 \rangle \\
\langle 1 \rangle & \rightarrow & [\text{bathroom}]\,\langle 0 \rangle \\
 & | & [\text{bedroom}]\,\langle 0 \rangle \\
 & | & [\text{garage}]\,\langle 0 \rangle \\
 & | & [\text{livingroom}]\,\langle 2 \rangle \\
\langle 2 \rangle & \rightarrow & [\text{hall}]\,\langle 1 \rangle \\
 & | & [\text{kitchen}]\,\langle 3 \rangle \\
\langle 3 \rangle & \rightarrow & [\text{livingroom}]\,\langle 2 \rangle
\end{array}
$$

Figure 4.3: Representing maps with formal language.

We demonstrate the conversion for the map in Figure 4.2(a). First, we apply Algorithm Algorithm 5 to produce a FSM from the map graph. Since the output of this algorithm is a Nondeterminisic Finite Automaton with more than the minimum necessary number of states, we convert the NFA to a DFA [Aho+07, p152] and minimize the number of DFA states with Hopcroft's Algorithm [Aho+07, p180]. This result is Figure 4.3(a). Note that in this example, we save two states over the original map in Figure 4.2(a). Finally, we convert the FSM to the Regular grammar in Figure 4.3(b).

4.2.7 Composition using the Motion Grammar Calculus

In order to semantically merge the robot and environment models, we apply our Motion Grammar Calculus (MGC). MGC is a set of rewrite rules for hybrid systems modeled in the Motion Grammar [DS12]. According to these rules, we extend our action grammar with each map symbol while maintaining only those transitions allowed by the map. While supervisory control can only operate to restrict system G using existing symbols, the MGC crucially describes how to introduce *new symbols* into G. There are two relevant rewrite rules from the MGC that we use here.

Transform 1 (Symbol Splitting)

Given some $\zeta_0 = [\mathrm{x} \in \mathcal{R}_0] \in Z$, create tokens $\zeta_1 = [\mathrm{x} \in \mathcal{R}_1]$ and $\zeta_2 = [\mathrm{x} \in \mathcal{R}_2]$ such that $\mathcal{R}_1 \cup \mathcal{R}_2 = \mathcal{R}_0 \wedge \mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset$ and update token set $Z' = Z - \zeta_0 \cup \{\zeta_1, \zeta_2\}$. The new nonterminal set is $V' = V \cup \{A_0, A_1, A_2, A_3, A_4\}$. The new production set is $P' = P - \{(A \to \alpha_1 \zeta_0 \kappa \alpha_2) \in P\} \cup$
$\{(A \to \alpha_1 A_0), (A_0 \to A_1 | A_2) \, : \, (A \to \alpha_1 \zeta_0 \kappa \alpha_2) \in P\}$
$\cup \{(A_1 \to \zeta_1 \kappa A_3), (A_2 \to \zeta_2 \kappa A_4) \, : \, (A \to \alpha_1 \zeta_0 \kappa \alpha_2) \in P\}$
$\cup \{(A_3 \to A_2 | \alpha_2), (A_4 \to A_1 | \alpha_2) \, : \, (A \to \alpha_1 \zeta_0 \kappa \alpha_2) \in P\}$.

Transform 2 (Adjacency Pruning)

For $p_1 = A \rightarrow r_A \kappa_A B$, $B \rightarrow \beta_1 | \dots | \beta_n$, if r_A is not adjacent to $\mathcal{R}_0(\beta_n)$ WLOG, then $P' = P - p_1 \cup \{A \rightarrow r_A \kappa_A B'\} \cup \{B' \rightarrow \beta_1 | \dots | \beta_{n-1}\}$

By applying these transforms, we can introduce the map symbols into the action grammar while preserving the validity of the model. Each derivation step maintains the *completeness* of the model according to the path of the hybrid system. By assuming that the initial model is complete, this ensures that all derived models are also complete. For the remainder of the MGC and proofs of its correctness, please see [DS12].

In addition to these two transforms, we also use the $\mathsf{first}()$ and $\mathsf{follow}()$ sets [Aho+07] to define initial and adjacent symbols. The $\mathsf{first}()$ set defines all terminals which may begin some derivation of a grammar symbol. The $\mathsf{follow}()$ set defines all terminals which may appear immediately to the right of some symbol in a grammatical derivation [Aho+07][p221].

Definition 8 (First Set)

Define $\mathsf{first}(X)$ for some grammar symbol X to be the set of terminals which may begin strings derived from X.

Definition 9 (Follow Set)

Define $\mathsf{follow}(X)$ for grammar symbol X to be the set of terminals a that can appear immediately to the right of X in some sentential form. Note that for map grammars such as Figure 4.3(b), the follow set for each terminal symbol is equivalent to the adjacent nodes in the map graph Figure 4.2(a).

Proposition 1

Given a grammar G representing some map M, $\mathsf{follow}(z)$ of some terminal symbol z of G represents the set of all map locations adjacent to z.

```
Input: $(Z_M, V_M, P_M, S_M)$ ;                                    // Map Grammar
Input: $(Z_A, V_A, P_A, S_A)$ ;                                 // Action Grammar
Output: $(Z, V, P, S)$ ;                                      // Combined Grammar
$(Z, V, P, S) \leftarrow (Z_A, V_A, P_A, S_A)$ ;
/* Add map symbols by splitting first($S_A$)                                 */
$z_0 = \mathsf{first}(S_A)$;
forall $z \in Z_M$ do
    $(Z, V, P, S) \leftarrow$ Transform 1 to split $z_0$ into $z$ and $z_0$
end
/* Prune non-adjacent map symbols                                            */
forall $z_1 \in Z_M$ do
    forall $z_2 \in Z_M$ do
        if $z_2 \notin \mathsf{follow}(z_1)$ then
            $(Z, V, P, S) \leftarrow (Z, V, P, S) \cap \overline{\mathfrak{L}\{.^{*}z_1 Z_A{}^{*} z_2.^{*}\}}$ ;
        end
    end
end
```

Algorithm 6: Composing Map and Action Grammars

Algorithm 6 describes how we apply these transforms to compose the Map and Action grammars. First, we introduce all map symbols into the action grammar by repeatedly splitting the initial terminal symbol of the action grammar by direct application of Transform 1. Next, we prune out productions indicating transitions between non-adjacent map locations. To prune these productions, we apply Transform 2 by intersecting the grammar with sets of allowable transitions. The disallowed transitions are indicated by the regular expression $L = (.^{*}z_1 Z_A{}^{*} z_2.^{*})$ in line 8 of Algorithm 6. The complement of this regular expression defines all paths which do not move directly from z_1 to z_2. Since z_1 and z_2 are non-adjacent, intersecting with $\overline{L}$ will preserve only paths which do not contain the disallowed transition. The result is a grammar which contains the original action model and all permissible transitions from the semantic map. We apply Algorithm 6 to combine the map grammar(Figure 4.3), with the base grammar for mobile manipulation (Figure 4.2). In this process, the initial nonterminal of the base grammar, [room], is repeatedly split into all the symbols of the semantic map. Then all transitions between non-adjacent map symbols are pruned away. This produces the combined grammar of Figure 4.4(a).

$$
\begin{array}{rcl}
\langle \mathrm{S}_0 \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \\
\langle \mathrm{H} \rangle & \rightarrow & [\mathrm{r}]\,\langle \mathrm{R} \rangle \mid [\mathrm{b}]\,\langle \mathrm{B} \rangle \mid [\mathrm{o}]\,\langle \mathrm{O} \rangle \\
 & \mid & [\mathrm{d}]\,\langle \mathrm{D} \rangle \mid [\mathrm{l}]\,\langle \mathrm{L} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{H}' \rangle \\
\langle \mathrm{B} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{B}' \rangle \\
\langle \mathrm{O} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{O}' \rangle \\
\langle \mathrm{R} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{R}' \rangle \\
\langle \mathrm{D} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{D}' \rangle \\
\langle \mathrm{L} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \mid [\mathrm{k}]\,\langle \mathrm{K} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{L}' \rangle \\
\langle \mathrm{K} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L} \rangle \mid [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{K}' \rangle \\
\langle \mathrm{H}' \rangle & \rightarrow & [\mathrm{r}]\,\langle \mathrm{R}' \rangle \mid [\mathrm{b}]\,\langle \mathrm{B}' \rangle \mid [\mathrm{o}]\,\langle \mathrm{O}' \rangle \\
 & \mid & [\mathrm{d}]\,\langle \mathrm{D}' \rangle \mid [\mathrm{l}]\,\langle \mathrm{L}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{H} \rangle \\
\langle \mathrm{B}' \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{B} \rangle \\
\langle \mathrm{O} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{O} \rangle \\
\langle \mathrm{R} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{R} \rangle \\
\langle \mathrm{D} \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{D} \rangle \\
\langle \mathrm{L}' \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \mid [\mathrm{k}]\,\langle \mathrm{K}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{L} \rangle \\
\langle \mathrm{K}' \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L}' \rangle \mid [\mathrm{place}]\,\langle \mathrm{K}' \rangle
\end{array}
$$

(a) Uncontrolled: G

- Let $\mathscr{R} = \{[\mathrm{h}], [\mathrm{r}], [\mathrm{o}], [\mathrm{d}], [\mathrm{l}]\}$

- Pick object in kitchen:
  $S_0 = .^*\,[\mathrm{k}]\,(\neg\mathscr{R})^*\,[\mathrm{pick}]\,.^*$

- Place object in bedroom:
  $S_1 = .^*\,[\mathrm{b}]\,[\mathrm{place}]\,.^*$

- Don't put the object anywhere else:
  $S_0 = \neg\,(.^*\neg\,[\mathrm{b}]\,[\mathrm{place}]\,.^*)$

- Move the object only once:
  $S_2 = (\neg\,[\mathrm{place}])^*\,[\mathrm{place}]\,\neg([\mathrm{pick}])^*$

- Let $\mathscr{X} = (\neg\,[\mathrm{x}])^*\,[\mathrm{x}]\,(\neg\,[\mathrm{x}])^*$

- Don't revisit rooms:
  $S_3 = \bigcap_{[\mathrm{x}]\in\mathscr{R}} \mathscr{X}\,(([\mathrm{pick}] \mid [\mathrm{place}])\,\mathscr{X})^*$

- End in the hallway: $S_4 = .^*\,[\mathrm{h}]\,\$$

(b) Supervisor: S

$$
\begin{array}{rcl}
\langle \mathrm{S}_0 \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H} \rangle \\
\langle \mathrm{H} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L} \rangle \\
\langle \mathrm{L} \rangle & \rightarrow & [\mathrm{k}]\,\langle \mathrm{K} \rangle \\
\langle \mathrm{K} \rangle & \rightarrow & [\mathrm{object}]\,[\mathrm{pick}]\,\langle \mathrm{K}' \rangle \\
\langle \mathrm{K}' \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L}' \rangle \\
\langle \mathrm{L}' \rangle & \rightarrow & [\mathrm{h}]\,\langle \mathrm{H}' \rangle \\
\langle \mathrm{H}' \rangle & \rightarrow & [\mathrm{b}]\,\langle \mathrm{B}' \rangle \\
\langle \mathrm{B}' \rangle & \rightarrow & [\mathrm{place}]\,\langle \mathrm{B}'' \rangle \\
\langle \mathrm{B}'' \rangle & \rightarrow & [\mathrm{h}]
\end{array}
$$

(c) Controlled: G'

Figure 4.4: Grammars for the Uncontrolled and Controlled mobile manipulator in the Aware Home.

4.2.8 Supervisory Control

Finally, we use supervisory control to produce the policy G' from our system model G and task specification S, [CL08, p133]. This application of supervisory control will permit only those transitions of the model G which are also contained in specification S. We represent this as the intersection, $G' = G \cap S$ Given that G is Context-Free and S is Regular, we use the algorithm defined in [HU79, p135] to produce Context-Free G', ensuring that we can efficiently execute the policy given by G'. This algorithm operates on a Context-Free language model for system G and a Regular language specification for correct operation S with the assumption that we can *block* any undesirable transitions in G. The corrected system language, then, is $G' = G \cap S$, where G' is also Context-Free. We note in addition that to prune non-adjacent regions permitted by Transform 2 in Algorithm 6, we apply this same language intersection operation.

$$\begin{array}{rcl}
\langle \mathrm{S} \rangle & \rightarrow & [\mathrm{room}]\, \langle \mathrm{S} \rangle \\
 & | & [\mathrm{victim-alive}]\, \langle \mathrm{V} \rangle \\
 & | & [\mathrm{victim-dead}]\, \langle \mathrm{D} \rangle \\
\langle \mathrm{V} \rangle & \rightarrow & ([\mathrm{healthy}] \,|\, [\mathrm{injured}])\, \langle \mathrm{notify} \rangle \langle \mathrm{S} \rangle \\
\langle \mathrm{D} \rangle & \rightarrow & \langle \mathrm{notify} \rangle \langle \mathrm{S} \rangle
\end{array}$$

Figure 4.5: A potential base grammar for rescue robots.

We use supervisory control of the grammar in Figure 4.4(a) to perform the desired mobile manipulation task. To instruct the robot to bring an object from the kitchen to the human in the bedroom, we construct our supervisor according to the regular expressions in Figure 4.4(b). Thus, our controlled system is:

$$G' = G \cap \bigcap_{i=0}^{4} S_i = [\mathrm{h}]\,[\mathrm{l}]\,[\mathrm{k}]\,[\mathrm{object}]\,[\mathrm{pick}]\,[\mathrm{l}]\,[\mathrm{h}]\,[\mathrm{b}]\,[\mathrm{place}]\,[\mathrm{h}] \tag{4.1}$$

4.2.9 Experiments

We implemented this approach on a Segway RMP-200 mobile platform as shown in Figure 4.6. This platform is equipped with an ASUS Xtion PRO LIVE camera, providing RGBD information for plane and surface extraction and with a UTM-30LX Hokuyo laser used to label the spatial regions as Gaussian models. It includes a Schunk parallel jaw gripper to manipulate objects.

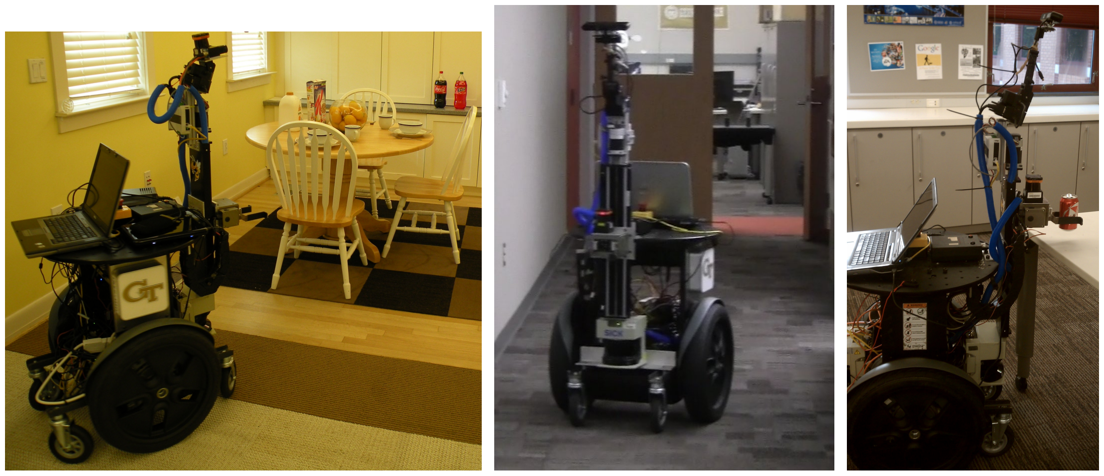

(a) Aware Home (b) RIM Center (c) Picking

Figure 4.6: Segway RMP-200 mobile platform in the Georgia Tech Aware, the RIM Center, and picking a soda can.

We conducted the experiments in the Georgia Tech Aware Home [Kie+08] and RIM center. For both of the home and office environments, we first drove the robot through each area collecting 3D point clouds, laser, and odometry. Our mapper extracts planes and surfaces in the environment, building the map and localizing the robot. During the navigation, the robot partitions the environment into Gaussian regions. This produces the Gaussian map in Figure 4.7.

Then, we annotate the Gaussian regions of the map with semantic labels. The result is a graph, shown previously for the Aware Home in Figure 4.2(a) and also for the RIM center in Figure 4.8. This resulting map is suitable for both human interpretation and automatic symbol manipulation.

Next, we apply the method described in Section 4.2.5 to generate the symbolic model for the robot in each of the environments. For the Aware home, this model is given in Figure 4.4(a), and for the RIM center in Figure 4.8.

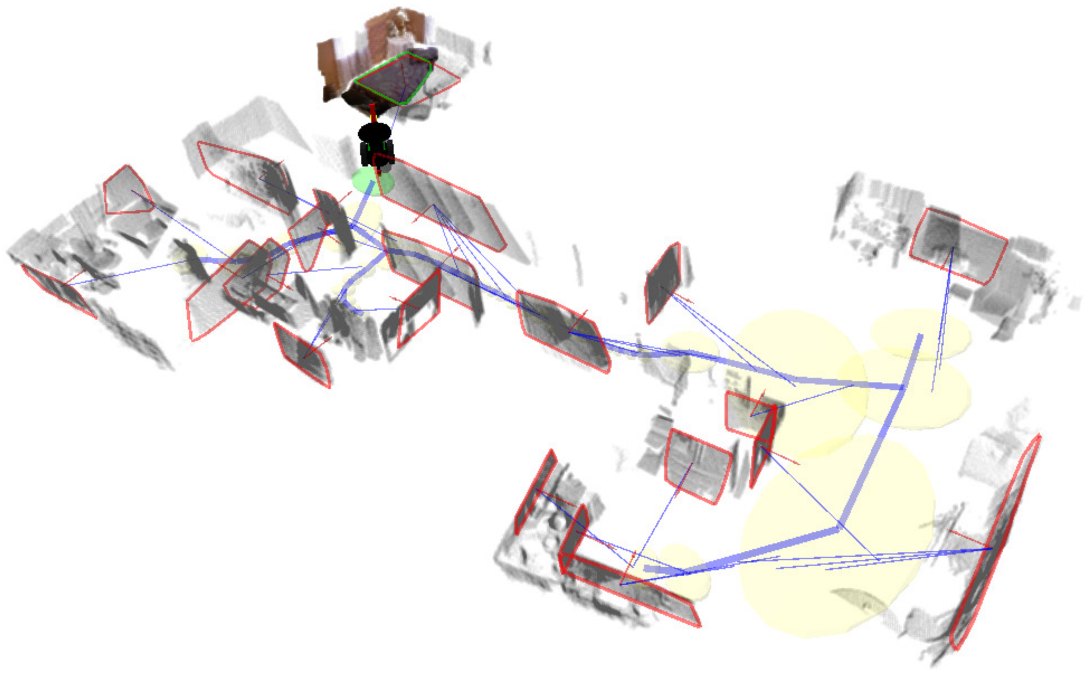
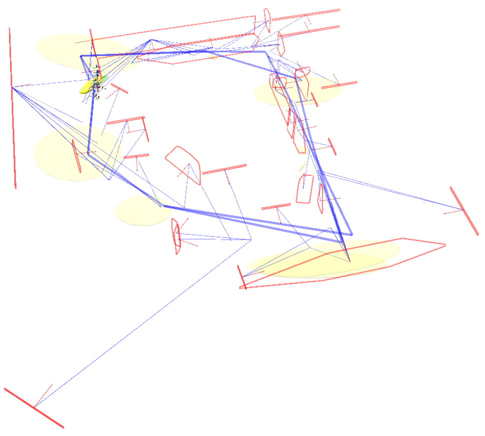
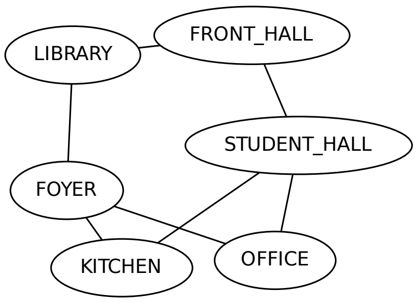


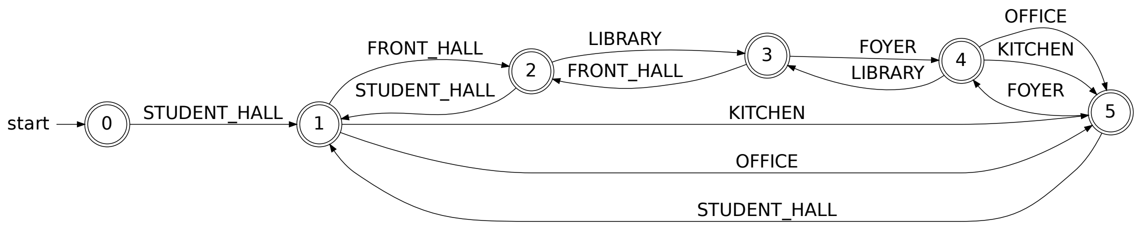
Figure 4.7: Generated Semantic Maps for the Aware Home. In the map, black shows 3D robot model, gray shows point clouds, yellow shows connected Gaussian regions (blue edges), and red shows the surfaces.

For the Aware Home, we asked the robot to perform the following task, *Collect a soda from the kitchen and bring it to the bedroom*, expressed as the specification in Figure 4.4(b). For the RIM Center, we apply a similar supervisor in Figure 4.9 to collect a soda from kitchen and bring it to library. The labeled location can be queried at the end of the tour. The main purpose of this labeling is to correlate the location of each room known to the robot with a symbol to use in specific tasks later requested by a human user.


(a) RIM Map]



(b) RIM Graph



(c) RIM FSM

Figure 4.8: Generated Semantic map of Georgia Tech RIM Center and the equivalent graph and Finite Automata forms.

The policy for the task in the RIM environment, Figure 4.11, is more complicated than for the Aware Home, Figure 4.4(c). This is because the RIM map contains multiple paths between all rooms. Thus, all these possible paths are captured in the control policy grammar. The result is the nine strings represented by the following regular expression,

$$G' = (k|rlfk|olfk)\,[\text{pick}]\,(fl|fosrl|srl)\,[\text{place}] \tag{4.2}$$

- Let $\mathscr{R} = \{[\mathrm{s}], [\mathrm{k}], [\mathrm{o}], [\mathrm{f}], [\mathrm{l}], [\mathrm{r}]\}$
- Pick object in kitchen:
  $S_0 = .^*\,[\mathrm{k}]\,(\neg\mathscr{R})^*\,[\mathrm{pick}]\,.^*$
- Place object in library:
  $S_1 = .^*\,[\mathrm{l}]\,[\mathrm{place}]\,\$$
- Move the object only once:
  $S_2 = (\neg\,[\mathrm{place}])^*\,[\mathrm{place}]\,\neg([\mathrm{pick}])^*$
- Let $\mathscr{X} = (\neg\,[\mathrm{x}])^*\,[\mathrm{x}]\,(\neg\,[\mathrm{x}])^*$
- Don't revisit rooms:
  $S_3 = \bigcap_{[\mathrm{x}]\in\mathscr{R}} \mathscr{X}\,(([\mathrm{pick}]\,|\,[\mathrm{place}])\,\mathscr{X})^*$

Supervisor: S

Figure 4.9: Supervisor for the Grammars for the Uncontrolled and Controlled mobile manipulator in the RIM Center.

These generated policies direct the robot along the path to complete the specified task. For the Aware Home, the robot fetches the object from the kitchen and delivers it to the bedroom, illustrated in Figure 4.10. This figure shows the path of the robot, both as a trajectory though the map and as the sequence of language symbols.

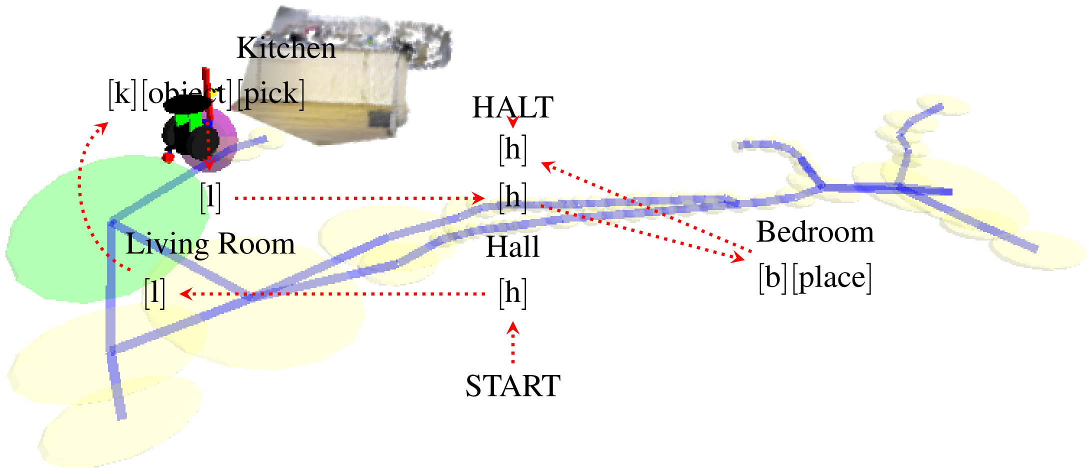


Figure 4.10: Path of the robot following controller in Figure 4.4(c) and Equation 4.1, shown as robot enters the living (green oval). Solid blue lines show the map connections between rooms, and dotted red lines show the robot path.

$$
\begin{array}{rcl}
\langle \mathrm{S_0} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S} \rangle \\
\langle \mathrm{S} \rangle & \rightarrow & [\mathrm{r}]\,\langle \mathrm{R} \rangle \mid [\mathrm{o}]\,\langle \mathrm{O} \rangle \mid [\mathrm{k}]\,\langle \mathrm{K} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{S'} \rangle \\
\langle \mathrm{O} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S} \rangle \mid [\mathrm{f}]\,\langle \mathrm{F} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{O'} \rangle \\
\langle \mathrm{K} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S} \rangle \mid [\mathrm{f}]\,\langle \mathrm{F} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{K'} \rangle \\
\langle \mathrm{K} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S} \rangle \mid [\mathrm{f}]\,\langle \mathrm{F} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{K'} \rangle \\
\langle \mathrm{F} \rangle & \rightarrow & [\mathrm{o}]\,\langle \mathrm{O} \rangle \mid [\mathrm{k}]\,\langle \mathrm{K} \rangle \mid [\mathrm{l}]\,\langle \mathrm{L} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{F'} \rangle \\
\langle \mathrm{L} \rangle & \rightarrow & [\mathrm{f}]\,\langle \mathrm{F} \rangle \mid [\mathrm{r}]\,\langle \mathrm{R} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{L'} \rangle \\
\langle \mathrm{R} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L} \rangle \mid [\mathrm{s}]\,\langle \mathrm{S} \rangle \mid [\mathrm{pick}]\,\langle \mathrm{R'} \rangle \\
\langle \mathrm{S'} \rangle & \rightarrow & [\mathrm{r}]\,\langle \mathrm{R'} \rangle \mid [\mathrm{o}]\,\langle \mathrm{O'} \rangle \mid [\mathrm{k}]\,\langle \mathrm{K'} \rangle \\
 & \mid & [\mathrm{place}]\,\langle \mathrm{S} \rangle \\
\langle \mathrm{O'} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S'} \rangle \mid [\mathrm{f}]\,\langle \mathrm{F'} \rangle \mid [\mathrm{place}]\,\langle \mathrm{O} \rangle \\
\langle \mathrm{K'} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S'} \rangle \mid [\mathrm{f}]\,\langle \mathrm{F'} \rangle \mid [\mathrm{place}]\,\langle \mathrm{K} \rangle \\
\langle \mathrm{F'} \rangle & \rightarrow & [\mathrm{o}]\,\langle \mathrm{O'} \rangle \mid [\mathrm{k}]\,\langle \mathrm{K'} \rangle \mid [\mathrm{l}]\,\langle \mathrm{L'} \rangle \\
 & \mid & [\mathrm{place}]\,\langle \mathrm{F} \rangle \\
\langle \mathrm{L'} \rangle & \rightarrow & [\mathrm{f}]\,\langle \mathrm{F'} \rangle \mid [\mathrm{r}]\,\langle \mathrm{R'} \rangle \mid [\mathrm{place}]\,\langle \mathrm{L} \rangle \\
\langle \mathrm{R'} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L'} \rangle \mid [\mathrm{s}]\,\langle \mathrm{S'} \rangle \mid [\mathrm{place}]\,\langle \mathrm{R} \rangle
\end{array}
$$

(a) Uncontrolled: G

$$
\begin{array}{rcl}
\langle \mathrm{S_0} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S} \rangle \\
\langle \mathrm{S} \rangle & \rightarrow & [\mathrm{k}]\,\langle \mathrm{K} \rangle \mid [\mathrm{r}]\,\langle \mathrm{R} \rangle \mid \qquad (4.3) \\
\multicolumn{3}{l}{[\mathrm{o}]\,\langle \mathrm{OL} \rangle} \\
\langle \mathrm{K} \rangle & \rightarrow & [\mathrm{pick}]\,\langle \mathrm{K'} \rangle \\
\langle \mathrm{R} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{OL} \rangle \\
\langle \mathrm{OL} \rangle & \rightarrow & [\mathrm{f}]\,\langle \mathrm{F} \rangle \\
\langle \mathrm{F} \rangle & \rightarrow & [\mathrm{k}]\,\langle \mathrm{K} \rangle \\
\langle \mathrm{K'} \rangle & \rightarrow & [\mathrm{f}]\,\langle \mathrm{F'} \rangle \mid [\mathrm{s}]\,\langle \mathrm{S'} \rangle \\
\langle \mathrm{F'} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L'} \rangle \mid [\mathrm{o}]\,[\mathrm{O'}] \\
\langle \mathrm{S'} \rangle & \rightarrow & [\mathrm{r}]\,\langle \mathrm{R'} \rangle \\
\langle \mathrm{O'} \rangle & \rightarrow & [\mathrm{s}]\,\langle \mathrm{S'} \rangle \\
\langle \mathrm{R'} \rangle & \rightarrow & [\mathrm{l}]\,\langle \mathrm{L'} \rangle \\
\langle \mathrm{L'} \rangle & \rightarrow & [\mathrm{place}]
\end{array}
$$

(b) Controlled: G'

Figure 4.11: Grammars for the Uncontrolled and Controlled mobile manipulator in the RIM Center. Notice how the policy captures all possible paths through the environment that satisfy the specification.

4.2.10 Discussion

In this approach, we combine a Semantic Map and a Motion Grammar using the Motion Grammar Calculus (MGC). This ensures the validity of our final system model because each transform of the MGC preserves *completeness* of the model. Then, applying a supervisory controller guarantees that the final policy is *correct* with regard to the specification. Thus, the overall approach is *correct-by-construction* in the sense that the final system model is guaranteed by the MGC to simulate our initial system, and the resultant policy satisfies the supervisory control specification.

The defining characteristic of this method is the uniform representation of the set of all robot paths as a language with an explicit grammar. This representation allows iterative development of the grammatical control policy by the progressive application of MGC transformations and supervisory control specifications. At each step of this derivation, the mechanical application of the MGC transforms and supervisory control ensures that we maintain a valid model of the system. Furthermore, because the policy for each task is itself a grammar, we can compose multiple individual task policies to produce a system to perform each of those tasks, all within the same grammatical framework. We expect these capabilities for incremental design and policy composition to be useful as we extend our work to multiple tasks and more complicated systems with larger grammars.

While search-based motion planning could perform some of the tasks in this work, there are certain advantages given by our linguistic formulation and use of supervisory control for policy generation. Random-sampling planners such as RRTs and PRMs assume a continuous search space, while our application domain includes discrete features for detecting and manipulating objects. General search based planning assumes an explicit goal state and produces a *plan* to reach that state. In contrast, the linguistic approach considers the set of acceptable *paths* and produces a *policy* to stay within that set of paths.

4.3 Coordination Strategies for Multi-robot Exploration and Mapping

Mobile robots are already widely used by first responders both in civilian and military operations. Today, these robot missions are usually performed by a person through tele-operation.

Such a mode of operation challenges the operator as the cognitive load is significant, which was presented in [Zhe+11]. There is consequently a desire to introduce some degree of autonomy to reduce the burden on the operator. For design of fully autonomous systems, there is a need to supply the system with a complete map of the environment or to endow the system with methods for automated mapping and exploration. Significant progress has been made on mapping and exploration with single robot systems, an thorough overview can be found in [DB06; BD06].

Moving from single robot systems to multi-robot teams poses a number of additional challenges. First of all the operator is posed with an added complexity in terms of controlling multiple entities at the same time, which is known to be a challenge [Zhe+11]. In addition, integration of maps generated by multiple robots into a coherent representation is also a challenge. Finally, there is a need to consider how the team-members can cooperate to accelerate the exploration of a previously unseen environment. There has been some progress reported on multi-robot mapping as presented in [Fox+06]. A number of methods for exploration of spaces have also been presented, see [Par08] for a recent summary of related research.

In our approach, we consider multi-robot exploration and mapping for larger teams. i.e. multi-robot systems with up to 9 team members. The main contribution for this work is an evaluation of different strategies for coordinating the efforts of a robot team during an exploration mission in an unknown environment. Three strategies are presented; these strategies will be referred to as *Reserves*, *Divide and Conquer*, and *Buddy System*. These strategies differ in how *proactive* extra members of the team are when they would otherwise not be needed for the exploration task, i.e. when all potential paths or frontiers are allocated to other team members. The first strategy, *Reserves*, is the least proactive. In this strategy, extra robots will wait in the starting area until they are needed. The second strategy, *Divide and Conquer*, is the most proactive. In this strategy, robots travel in as large a group as possible and split in half when new navigation goals (branches, junctions in corridors) are uncovered. The final strategy, *Buddy System*, represents a compromise between the other two strategies. In *Buddy System*, robots travel in teams of two until new navigation goals are detected and the team will split to follow both paths.

4.3.1 Approach

Mapping System

We use the library called *OmniMapper* described in section Figure 2.4 shown in 4.12. In this application, the mapper is using the *plane* mapping plugin described in Section 2.4.

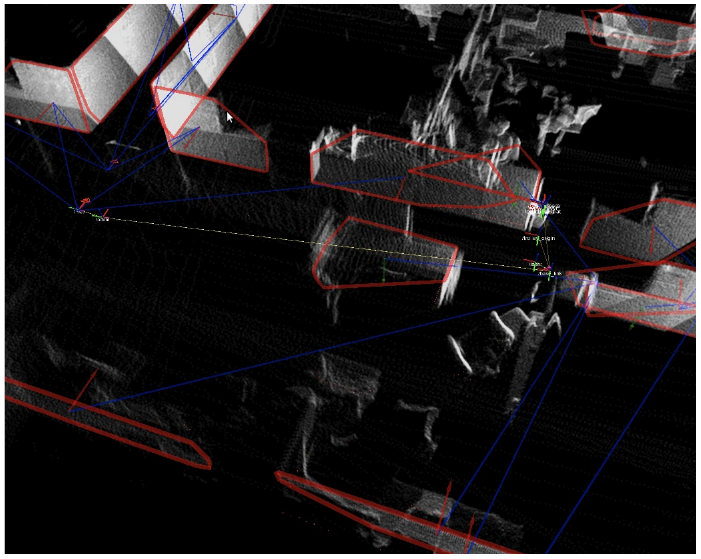


Figure 4.12: OmniMapper.

Each robot in the team builds a map locally with the *OmniMapper* and sends map data to the map coordinator. Each robot can incorporate new landmark measurements whenever it has moved far enough from the last pose where measurements were made. In the current implementation this is set to 10cm. When a robot finishes optimizing its local map with new landmark measurements, all relevant information needed by the map coordinator is packaged and transmitted.

The information which is needed by the map coordinator to incorporate a new piece of information from a team member consists of many components. First, the sensor measurement data is needed. In the current implementation, this consists of the extracted plane information consisting of a plane equation along with a convex hull of points along the perimeter of the plane. This represents a significant compression over an alternative scheme where all point-cloud data could be transmitted and processed at the master node. The relevant information which is sent from the robot to the remote master map coordinator consists of as little data as possible to minimize communication overhead. This information is collected into a map-chunk message. The extracted feature measurement is one component which is sent; this extracted feature is typically much simpler than the raw sensor data which it is based upon. Secondly, the team member's integrated odometry is transmitted. An odometric estimate is used because it is a smooth value and will not jump due to loop closures on the individual robot. For small motions, odometry is a good estimate of the robot's relative pose to the previous measurement taken. The individual robot's posterior estimate of where it thinks it is in its own local map frame is sent to the master map coordinator. The master uses this estimate toproduce a correction to establish the relationship between the global map frame and each of the local map frames. Since the master mapper knows the transformation between the global map frame and each robot's local map frame, it translates all motion commands into the robots local frames of reference before they are sent to the robots. This allows the master node to compute the odometric relative pose since the prior landmark measurement data was incorporated; this is used to insert a relative pose factor and also give initial conditions for data association. Finally, the team member's local map pose is transmitted. This is used by the master node to compute a map pose correction. This correction is sent back to the team member so that it knows it's relative pose in the global map frame. This knowledge is needed so that the team member can interpret exploration goals correctly.

The map coordinator maintains trajectories for each of the robots in the team. Measurements from each robot are merged into one global view of the landmarks. This is realized through a simple modification to the standard *OmniMapper* through duplication of data structures tracking indexing data and pose information used for interaction with GT-SAM into arrays. This implementation potentially allows for an unlimited number of team members to build a map together.

Each turtlebot in these experiments maps planar wall structures using a Microsoft Kinect sensor. Planar segments corresponding to walls are extracted from point clouds via a RANSAC [FB81] based algorithm [RC11]. This process was described in section 2.4.

The Kinect sensor on each robot has a narrow field-of-view which is not ideal for detecting exploration frontiers. To alleviate this problem, we incorporated a strategy by which each robot will rotate periodically to get a 360 degree view of its surroundings. This data is synchronized with robot odometry to synthesize a 360 degree laser scan. This synthesized laser scan is sent to the local mapper and forwarded to the global mapper. At the global mapper, it is linked to a trajectory pose element and used to populate an occupancy grid. This occupancy grid is re-computed after every map optimization so that a loop closure will result in a correct occupancy grid map. The frontier based exploration strategies detailed below use this occupancy grid to find the boundary between clear and unknown grid cells.

4.3.2 Exploration Strategy

Each robot team leader uses a frontier based exploration strategy similar to the one used in [Vin+08]. An exploration frontier is defined on a costmap cellular decomposition where each cell has one of three labels: *Clear*, *Obstacle*, and *Unknown*. The costmap is initialized as *Unknown*. Costmap cells are set to *Obstacle* corresponding to locations where the Kinect sensor detects an obstacle in the environment. The cells on a line between the obstacle cell and the robot's current location are set to *Clear*. Exploration frontiers are defined as *Clear* cells which are adjacent to at least one neighbor where the label is *Unknown*, illustrated in Figure 4.13.

The high level robot exploration goal allocation is centrally planned on the same workstation where the global map is constructed. There are many choices which can be made by the exploration planner when choosing which robot or group of robots should move towards an

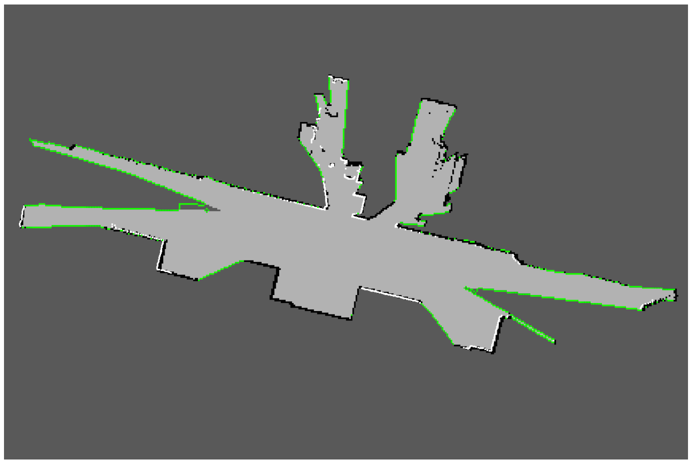


Figure 4.13: An example cost-map representing the exploration frontier. Green cells represent the exploration frontier between Unknown and Clear grid cells. Black cells have the Obstacle label.

exploration goal. We have chosen to employ a greedy strategy by which the nearest robot or team is allocated to a goal instead of a more sophisticated traveling-salesman type of algorithm. We believe that this is appropriate because the exploration goals will change as the robots move through the environment; re-planning will be required after each robot or team reaches an exploration goal.

4.3.3 Coordination Strategy

The coordination strategy used between robot agents as well as the number of robots are the independent variables in the experiments performed in this section. The coordination strategy refers to the proportion of robots which are dispatched to each exploration goal. On one extreme, a single robot can be sent to explore a new goal; at the other extreme all available robots can be sent to a new goal. Larger robot teams sent to a new exploration goal will improve availability of new agents at the location of new exploration goals are discovered. The larger group has spare robots which can be quickly allocated to explore new goals, such as those discovered when the team moves past a corridor intersection or t-junction. If the group of robots allocated to a navigation goal is too large, then the robots can interfere with each other due to local reactive control of multiple agents with respect to dynamic obstacles and limited space in corridors. The strategies selected for testing trade off *availability* (robots are close and able to explore branching structure quickly) with *non-interference* (robots do not get in each other's way).

The first coordination algorithm is called *Reserve*. In this algorithm shown in Figure 4.14, all unallocated robots remain a the starting locations until new exploration goals are uncovered. When a branching point is detected by an active robot, the closest reserve robot will be recruited into active status to explore the other path. This strategy has low availability because all of the reserve robots remain far away at the entrance; however, it has minimal interference because the exploring robots will usually be further away from other robots.

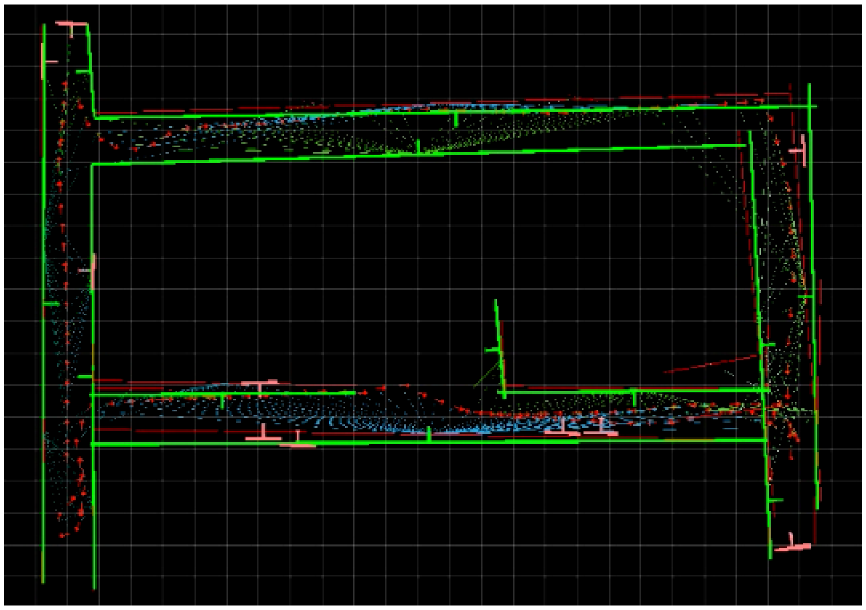


Figure 4.14: A map built by three robots using the *Reserve* cooperative mapping strategy.

The second coordination algorithm is *Divide and Conquer*. In this strategy, shown in Figure 4.15, the entire robot group follows the leader until a branching point is detected. The group splits in half, with the first $\frac{n}{2}$ robots following the original leader, robot $\frac{n}{2}+1$ is selected as the leader of the second group, and robots $\frac{n}{2}+2$ through n are now members of its squad. Once there are n squads with one robot, no further divide operations can be made and new exploration goals will only be allocated once a robot has reached a dead-end or looped back into a previously explored area. This algorithm maximizes availability, but potentially causes significant interference between robots.

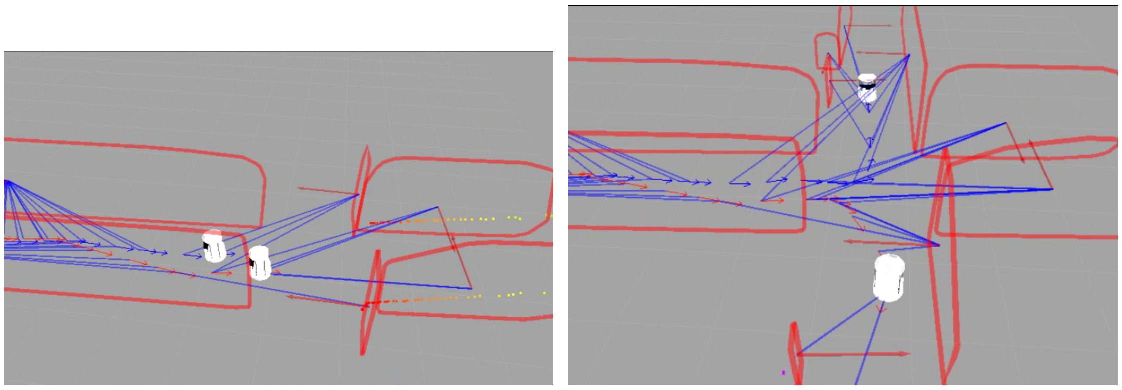
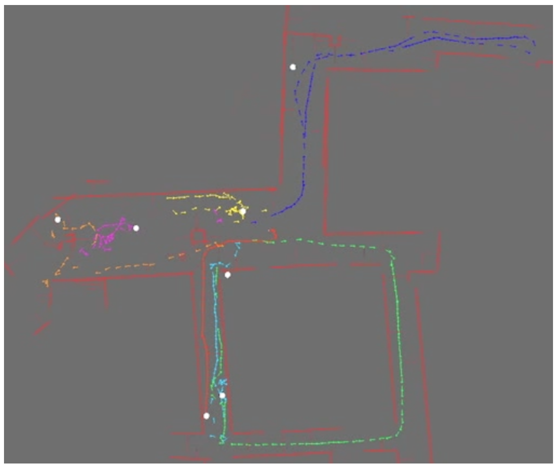

(a) Two robots approaching the intersection. (b) Two robots split and move past the intersection

Figure 4.15: An illustration of the *Divide and Conquer* exploration strategy. As the robots approach an intersection, the team must split and recruit new partner robots from the reserved units.

The third coordination algorithm is called the *Buddy System*. In this strategy, shown in Figure 4.16, robots are recruited from the reserve pool in teams of two. When a branching point is detected by a full team of two robots, the team will split into two and proceed along both paths. When these single robots detect additional split points, new teams of two robots will be allocated out of the reserve pool and they will explore this new goal and divide when another branching point is reached. This strategy uses small teams of robots which are able to maneuver around one another without too much interference, while maintaining good availability to respond quickly to explore new frontiers.

An example 3D map built by two robots as they approach a branch point can be seen in Figure 4.15(a). At this point, the robot team splits and each team member takes a separate path, as seen in Figure. 4.15(b). The map shown is built concurrently with local maps built on each robot. The global map is used to establish a global frame of reference for robot collaboration message coordinates.


(a) A map built by seven robots in an experiment using the *Reserve* cooperative mapping strategy.

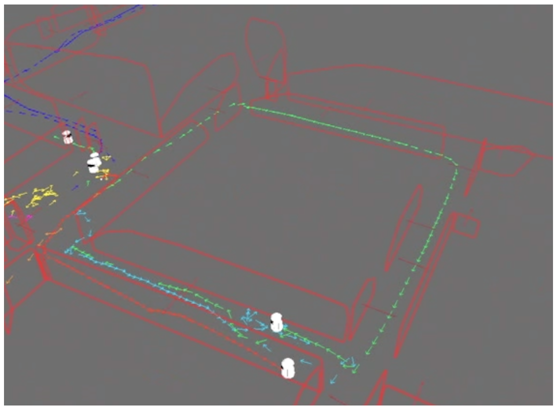

(b) The same map shown from a different angle to demonstrate 3D plane features which are used for map landmarks.

Figure 4.16: Global maps gathered by a team of seven mobile robots.

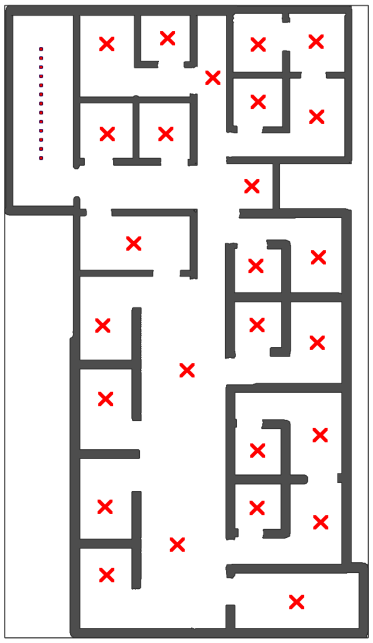
4.3.4 Experiments

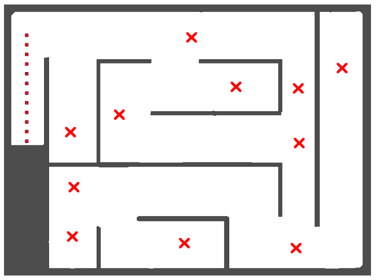
Simulation Experiments

We first evaluated our collaboration strategies in simu-lated environments with the Stage simulator (Gerkey et al.,2003) in which the robot can explore and navigate.We then performed live robot experiments in multiple environ-ments including an office building and a training facility. Results from live robot experiments will be presented in Section 4.

Three different scenarios were chosen for these simulation experiments. In each of these scenarios, the robots were introduced in a single entrance from an initial formation to simulate a breach entrance into a hostile environment. The first scenario is shown in Figure 4.17(a); this scenario is designed to be typical of a home or office environment. There is a combination of open space and smaller rooms. Many rooms are connected via main corridors, but some rooms are connected to each other through other side passages. Since this environment has a lot of maneuvering room and early opportunities for exploration in multiple directions, we expect that exploration will be accelerated as robot team size is increased. Availability of additional robots will also be important in this environment,since new exploration frontiers will be uncovered quickly as exploration proceeds; therefore, we expect the Divide and Conquer strategy to perform well. The second scenario, which is modeled after a simple maze, is shown in Figure 4.17(b). Again, the robot team is introduced from an initial formation at an entrance point to the maze. Since this is a relatively simple maze, it has allow branching factor and will not benefit large robot team sizes. Availability of additional robots will be helpful, but due to the limited number of points where additional robots are needed for exploring branching structure, Divide and Conquer should enjoy less of an advantage over the other strategies.The third scenario is shown in Figure 4.17(c). It consists of a series of moderately-sized rooms with large obstacles connected by long corridors. This structure is meant to represent an underground base where the rooms are placed down long corridors to provide separation from the entrance and protection from assault. In this environment, the robots will have to travel longer distances before branching points,so the availability of additional robots in Divide and Conquer,as well as Buddy System to a lesser extent, should be helpful for exploring the map quickly. Large team sizes may not be that helpful in this environment, since teams should be able to reform and continue the exploration after vising the rooms while other team members are still proceeding down the next section of corridor.

Simulation test runs were launched automatically from a script. Three computers were used for running simulations; to keep machine performance from affecting timing results, all runs for each scenario were run on one machine. A checking program was implemented which monitors the robots and compares their progress through the exploration procedure by refering to a set of ground-truth *navigation keypoints*. When a robot passes within a 1.5 meter threshold of a navigation keypoint, this event is recorded by the checking program. When all navigation keypoints have been visited by the robot team, the exploration task is complete and the checking program records results and allows the script to start the next test run.


(a) Simulated environment *S1*: Rooms world. This map is meant to represent a home or office environment.


(b) Simulated environment *S2*: Maze world. This is a maze type structure with a relatively low branching factor.

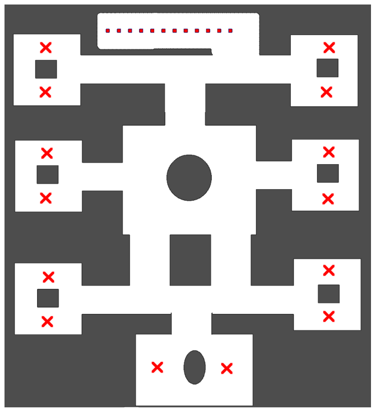
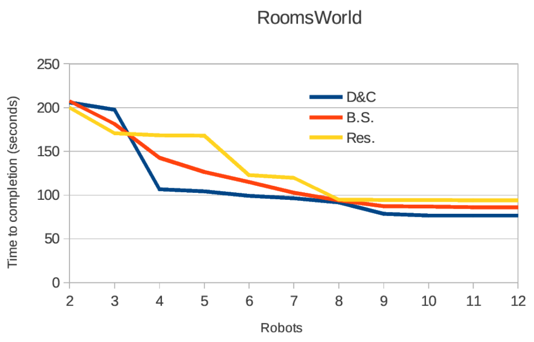

(c) Simulated environment *S3*: Lair world. This environment is meant to simulate an underground bunker with high branching factor and many corridors.

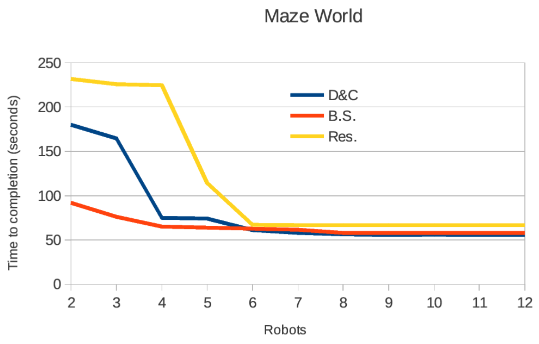
Figure 4.17: Simulated maps used to test coordination strategies on various types of environments within the Stage simulator. Robots are shown as a line of red dots in their initial starting area which represents a breach entrance. Navigation keypoints are illustrated with red 'X' marks.

This system of navigation keypoints is used to get a reasonable measure of when the robot team has completed the exploration and mapping task. Navigation keypoints are selected such that if all of them are uncovered, then the environment is fully explored, and vice-versa.

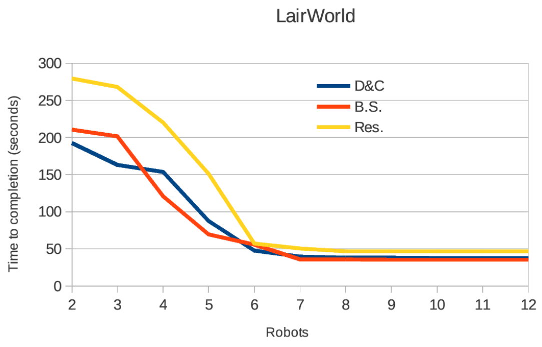
Total exploration time results for these three simulation scenarios are presented in Figure 4.18. Each of these graphs presents the time taken to fully explore the environment with a team size from 2 to 12 robots and each of the three strategies *Divide and Conquer*, *Buddy System*, and *Reserves*. Since there is no collaboration with a single robot, there isn't any difference between the performance of each strategy. For this reason, exploration timing for a team with a single robot was omitted.



(a) Results from environment *S1*: Rooms world



(b) Results from environment *S2*: Maze world



(c) Results from environment *S3*: Lair world

Figure 4.18: Times taken to fully explore each simulation environment with varying team size, by strategy. A map is fully explored when every location marked with a red 'X' from Figure 4.17 has a robot approach within a 1 meter radius.

In the *Rooms World* simulation scenario shown in Figure 4.17(a), the results can be seen in Figure 4.18(a). *Rooms World* is meant to represent a typical domestic or office environment with a mixture of larger and smaller spaces separated by both corridors as well as some direct connections. For all three strategies, the time taken to fully explore the map is reduced as the team size is increased initially, but then plateaus above 9 robots. We believe that this is due to the fact that this is still a relatively simple scenario and large teams of robots are not necessary to proceed through the map. In the scenarios *Divide and Conquer* and *Buddy System* we noticed that robot teams often stayed together as they proceeded around structures such as in the upper right and lower right corners of Figure 4.17(a). These rooms often did not present more than one exploration frontier at a time and therefore additional robots were not very helpful in pursuing alternate frontiers. *Divide and Conquer* seems to maintain its performance at larger team sizes better than *Reserves*. This indicates that in a typical domestic or office environment, the additional availability of additional robots in a *Divide and Conquer* strategy is helpful for exploration. *Divide and Conquer* outperforms other strategies for almost all robot team compositions. In this scenario, using more than 9 robots isn't helpful in terms of exploration time; however, this is the largest environment and the other two test scenarios will plateau with a smaller team size.

The results for the *Maze World* scenario illustrated in Figure 4.17(b) are shown in Figure 4.18(b). In this environment, additional robots do not have much of an advantage over smaller team sizes beyond 6 robots. This is likely due to the fact that though this is a maze environment, it does not exhibit significant branching factor; additional robots are not needed beyond the number of exploration frontiers which are open at a given time. As the robots explore this maze, the number of exploration frontiers which are open at a time is around this number, so it makes sense that additional robots beyond 4 would not be very helpful, as is seen with the more proactive strategies *Divide and Conquer* and *Buddy System*. *Divide and Conquer* and *Buddy System* appear to have an advantage in exploration time over *Reserves* with smaller team sizes. This map is relatively small, so robots are able to move to new frontiers quickly, and having additional robots available only saves this small amount of additional time. Due to the small branching factor in this scenario, high availability collaboration strategies like *Divide and Conquer* allocate resources more efficiently when they are scarce with smaller team sizes, but this advantage disappears as team sizes grow.

The results for the *Lair World* simulation environment from Figure 4.17(c) are shown in Figure 4.18(c). This environment is comprised of a series of corridors which attach rooms to a central room. In this simulation, additional robots seem to not improve performance beyond 7 robots. *Buddy System* outperforms other strategies, which indicates that each robot encounters a single branch point by the time that a new team is available to reform the teams. This result is also apparent in the results for *Maze World* shown in Figure 4.18(b).

The second series of simulated experiments repeats the previous experiments with much larger teams of 25 robots. An example of these experiments is shown in Figure 4.19.

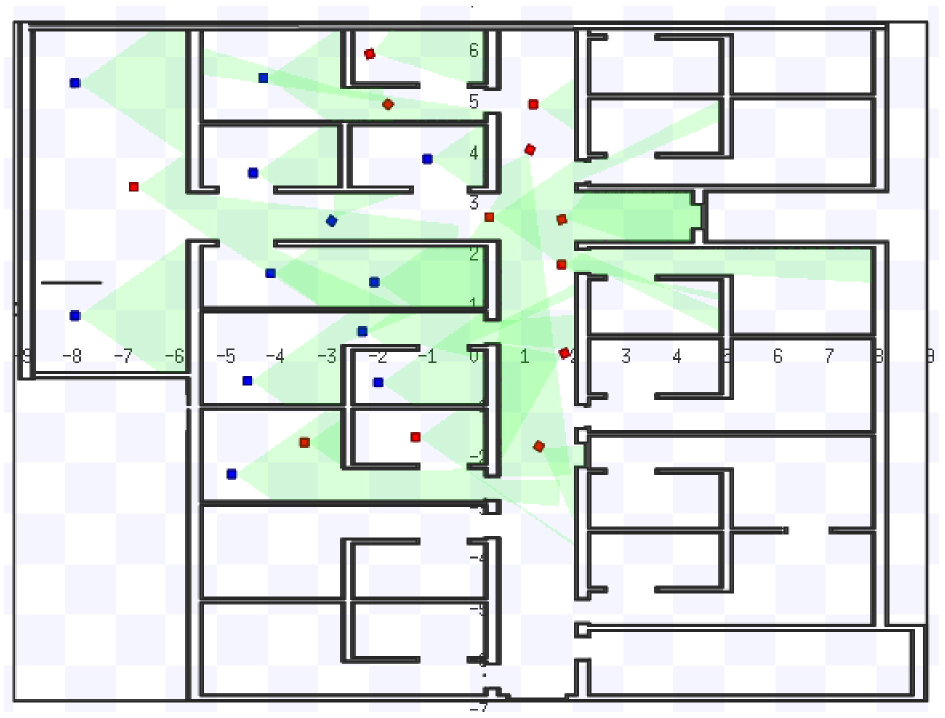


Figure 4.19: A simulation run using 25 robots on environment S3 with the strategy *Divide and Conquer*. Robots are shown as red and blue squares. The field of view of the sensors are shown in green.

This series of simulations has demonstrated that the performance of the collaboration strategies which we have presented is dependent on the topology and geometry of the environment. Strategies such as *Divide and Conquer* kepp additional robots available to explore branching structure; however, these strategies also generate more interference between robots. *Divide and Conquer*, and, to a lesser extent, *Buddy System* require robots to navigate close to each other. With such close approach of other robots, navigation is more cautious to avoid collision. In smaller environments, the risk of collision is higher and larger teams of robots interfere with each other more. Also, the utility of these types of strategies is dependent on the degree of branching structure exhibited by the environment during exploration.

Live robot experiments

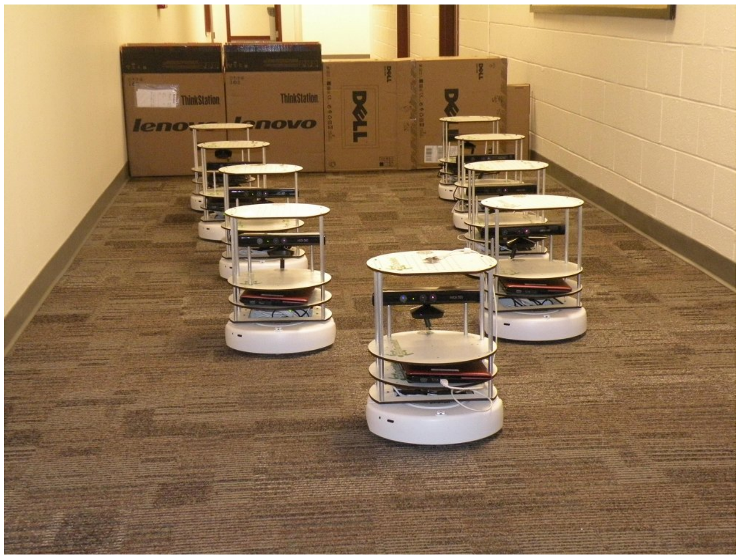


Figure 4.20: Nine TurtleBots used in these experiments

We evaluated the performance of various robot coordination strategies in the multi-robot exploration and mapping task. An example scenario for the *Divide and Conquer* cooperative mapping strategy can be seen in the panorama image in Figure 4.21. Live-robot experiments were performed in two settings: an office environment, and a training facility consisting of several buildings simulating a small village. At the time when the office experiments were performed, only the *Divide and Conquer* and *Reserves* strategies were available; the *Buddy System* strategy was not tested here. All three strategies were tested in many of the buildings in the simulated village training facility.

Office environment

In the first series of live robot experiments, we evaluated the first two strategies which were developed, the *Reserves* and *Divide and Conquer*. This series of experiments was designed to evaluate the performance of these two cooperative exploration strategies. A total of 6 runs were performed for each cooperation strategy, team size, and starting location. For each experiment

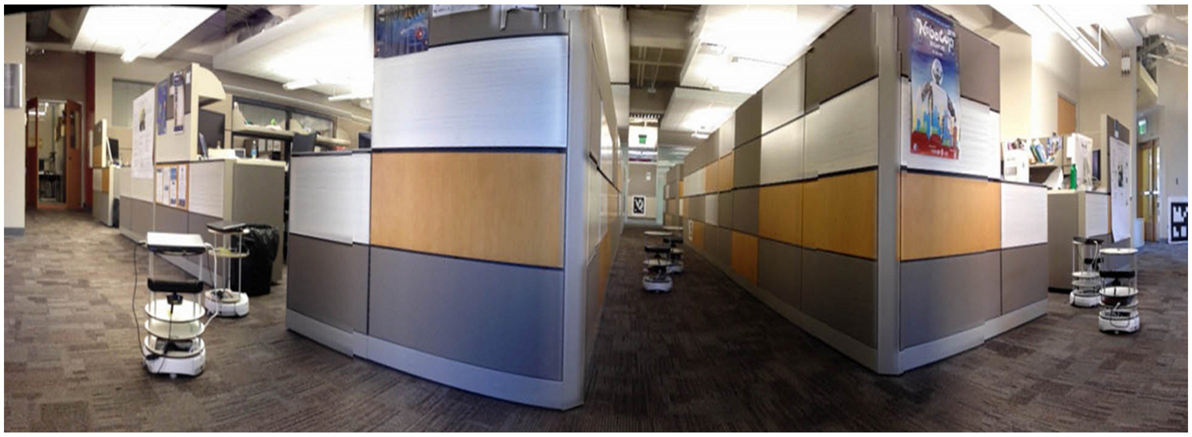


Figure 4.21: An example scenario for the experiments described in this paper. Three teams of two robots are exploring the branching hallway structure in an office environment. In this illustration, the robots are using the *Divide and Conquer* cooperative mapping strategy.

run, the *TurtleBot* team explored the environment from a wedge-shaped starting configuration, which can be seen in Figure 4.20. These experiments were performed in an office environment. In order to measure the exploration and mapping performance in each location, we chose specific starting locations which are labeled *Base1* and *Base2* in Figure 4.22. These initial configurations were chosen to represent a breaching action where mobile robots would be introduced into a hostile environment[*].

The first series of experiments demonstrate team performance based upon coverage in a mapping task on an unknown office environment. Robot team sizes were varied from 2 to 9 robots. A map built with 7 robots at TurtleBots using the *Reserve* strategy is seen in Figure 4.23(a). An image showing the same final global map from a side view demonstrates the 3D plane features in Figure 4.23(b).

Each of the collaboration strategy and robot team size experiments were performed from two starting locations. These starting locations are labeled *Base1* and *Base2* in Figure 4.22. A series of interesting locations was determined in advance by examining the building floor-plan; these points of interest are also marked in Figure 4.22. Each experiment run gets a score based

[*]Red lines indicate artificial barricades to restrict the initial exploration of the robot teams to simulate a breach entrance into a hostile environment. Blue squares indicate the location of navigation keypoints used for evaluating test runs, as described in the text.

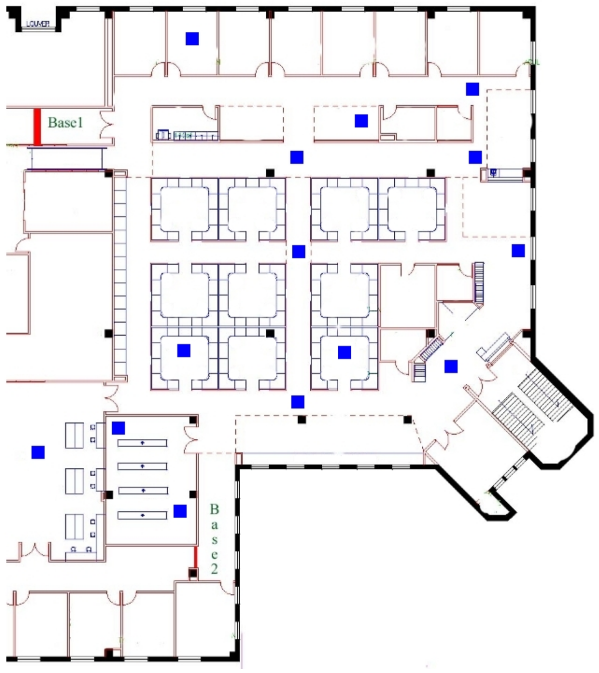
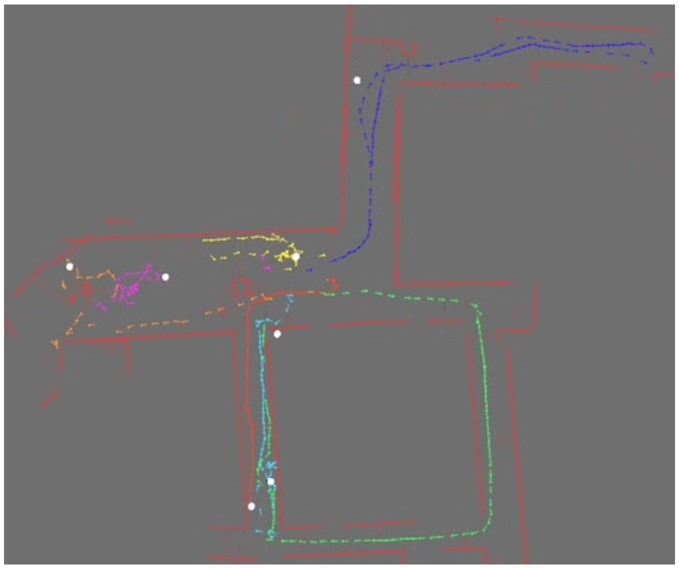


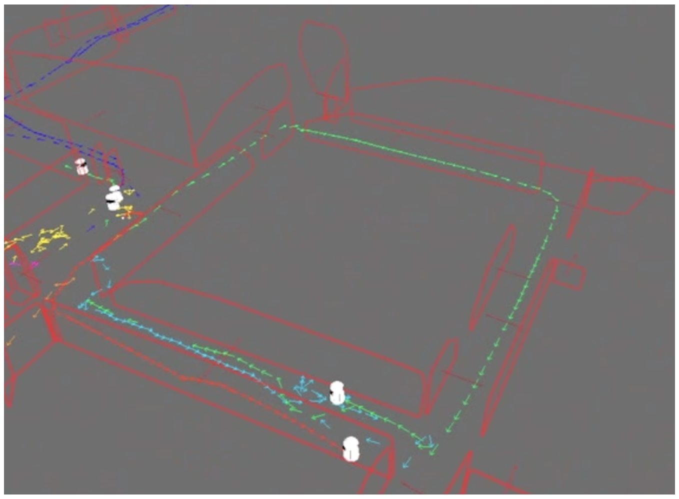
Figure 4.22: Our office environment where the experiments were performed. The areas labeled Base1 and Base2 are the initial position of the robots.

on how many of these points of interest are visited and mapped before a time limit is reached. This score represents the effectiveness of that algorithm and team size at exploring the entity of an unknown map.

In the first experiment series from *Base1* in Figure 4.22, both strategies achieve reduced exploration coverage per robot as the team size is increased, as can be seen in the graphs in Figure 4.24. In this starting location, there is limited space to maneuver, so both strategies generate significant interference between robots trying to move to their goals. In several instances, pairs of robots even crashed into each other due to the limited field-of-view of their sensors. We believe that the *Divide and Conquer* strategy results in Figure 4.24(b) indicate that the team was slightly more effective than the *Reserves* strategy in Figure 4.24(a). At the largest team size of 9 robots, the *Divide and Conquer* strategy usually visited one additional point-of-interest more than the *Reserves* strategy.



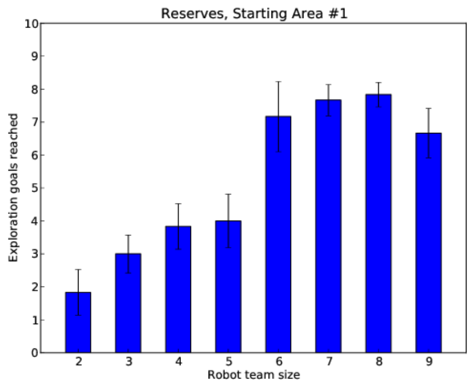
(a) A map built by seven robots in an experiment using the *Reserve* cooperative mapping strategy.



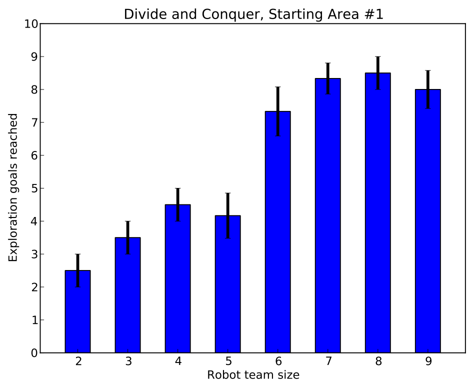
(b) The same map shown from a different angle to demonstrate 3D plane features which are used for map landmarks.

Figure 4.23: Global map gathered by a team of seven mobile robots using the Reserve strategy. In Figure (a) shows a top-down view after a loop closure is detected at the large loop. In Figure (b) is the same map viewed from an angle to show the extent of the 3D walls.

Additional qualitative impressions are that the *Divide and Conquer* strategy explored the points-of-interest that it reached more quickly than with the *Reserves* strategy. For both strategies, the best team size appears to be 8 robots in this starting location, with a sharp increase at 6 robots.



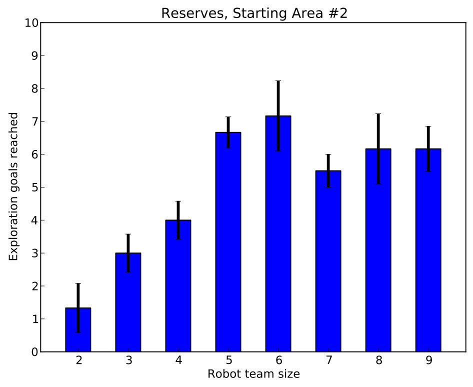
(a) Reserves



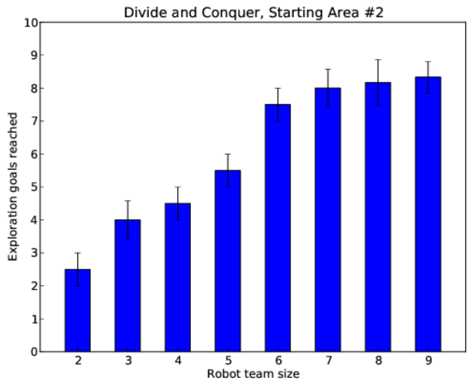
(b) Divide and Conquer

Figure 4.24: Results from the first starting area

In the second set of the first series of experiments, the robot teams were placed in the starting area labeled *Base2* in Figure 4.22. As in the first experiment, the per-robot performance of both strategies decreased as the number of robots were increased. This series of experiments demonstrates a marked improvement of the *Divide and Conquer* strategy over the *Reserves* strategy as can be seen in Figure 4.25. The *Divide and Conquer* strategy causes more robots to be making observations of exploration frontiers due to the fact that groups contain more than one robot. These additional observations of the frontier allow the *Divide and Conquer* strategy to find exploration frontiers faster than the *Reserves* strategy, and therefore explore more points-of-interest. The second experiment started from an area where there is more room to maneuver. This allowed the *Divide and Conquer* strategy to have less interference since the entire team moved together out of the starting area into the larger area before any divide operations were performed. The *Reserves* strategy still had to initially maneuver from the cramped starting location. As in the first experiment, the *Divide and Conquer* strategy qualitatively explored the environment faster than the *Reserves* strategy. The *Divide and Conquer* strategy behaved similarly with the first experiment, with increased performance with additional robots up to the limits of our study, while the *Reserves* strategy exhibited reduced performance due to interference with team sizes above 6.



(a) Reserves



(b) Divide and Conquer

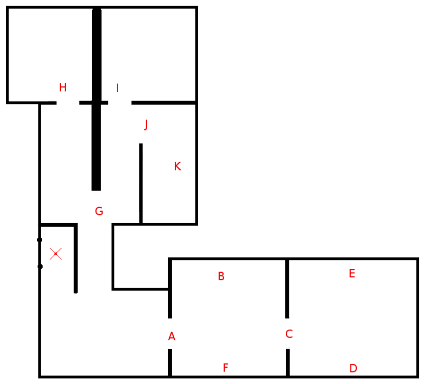
Figure 4.25: Results from the second starting area

Training facility

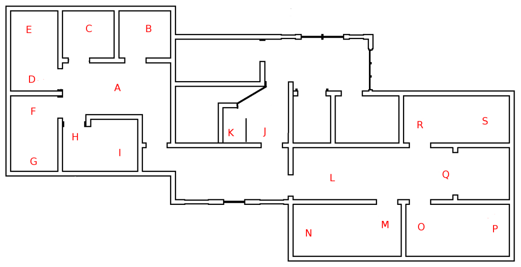
In the second series of live robot experiments, we evaluated all three collaboration strategies *Reserves*, *Divide and Conquer*, and the new strategy *Buddy System* in various buildings in a training facility designed to simulate an urban environment or village. Due to the remoteness of this training facility, we only brought five robots for testing. The robot team size is varied from three to five robots. We attempted to run each test three times, though some runs were not completed due to scheduling constraints.

The procedure used for the experiments at the training facility differs somewhat from the experiments at the office building. In these experiments, we evaluated robot exploration performance using the same metric as in the simulations from Section 4.3.4. In the experiments in the training facility, we allowed the teams to take as long as needed to explore every keypoint. The amount of time elapsed to fully explore each building is the metric used for comparing strategies in these experiments.

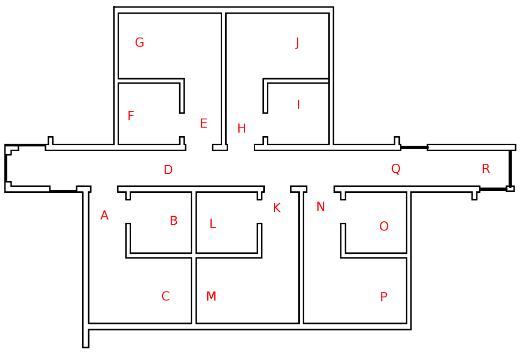
Architectural floor-plans from the buildings explored in this experiment are shown in Figures 4.26(a), 4.26(b) and 4.26(c). The set of keypoint locations were manually labeled by room separation. Some of these locations are difficult to reach given the physical capabilities of the robots. In Figure 4.27(a), the robot needs to navigate through a narrow hallway in order to explore the rest of the floor of building C. For example in Figure 4.26(b), the robot needs to reach the goal labeled *P* and *O* in order to complete the exploration of the room. These experiments were performed with teams of three, four and five robots. Figure 4.27(d) shows the initial configuration of one of the experiments ran in Building *C*. This initial configuration needs to be set in order to fit the robots close to a door. This is meant to simulate an exploration task for a rescue mission which starts by introducing the robots through a doorway into the building.



(a) Building *A*



(b) Building *B*



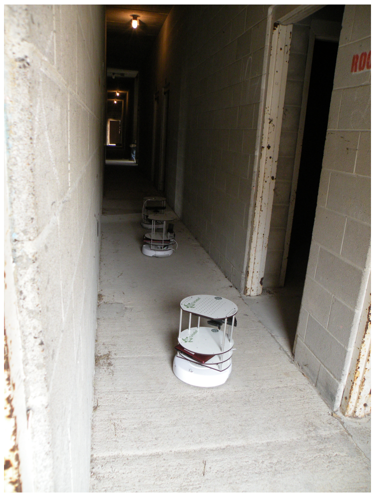
(c) Building *C*

Figure 4.26: Architectural floor plans for buildings used for robot exploration and mapping experiments. Navigation key points used for scoring runs are indicated by letters on the maps.

As shown in Figure 4.27(b), the robots explore and navigate in an environment which exhibits difficult lighting conditions; however, this is not a problem for the Microsoft Kinect sensor.

(a) Robots navigating trough narrow hallways in Building *B*

(b) Three robots exploring Building *B*


(c) Two robots exploring a narrow hallway on Building *C*

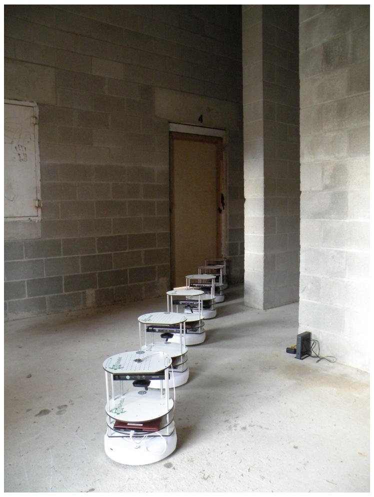

(d) The initial configuration of the robots in Building *A*, intended to simulate a breach entry from the door.

Figure 4.27: Robot teams running exploration and mapping experiments

4.3.5 Results

Our results are summarized in tables 4.1, 4.2, 4.3 show the average on time that each group of robot takes to explore the three buildings. In comparison with the simulation experiments, the implementation in real robots are subject to hardware failures and a hostile environment with debris and small potholes. In some of the experiments, particularly those which took place in building C, this caused a high degree of variability between runs. However, running the experiments in the other two environments provided a good estimation of the performance of each exploration strategy.

Table 4.1: Average time of each exploration strategy with 3 robots. Buddy System was not run in Building C. Statistics are gathered from three runs.

Strategy	Building A	Building B	Building C
Divide and Conquer	2233 ± 56	1718 ± 287	1127 ± 30
Reserves	$2526 \pm -$	1699 ± 58	1521 ± 241
Buddy System	1989 ± 58	1531 ± 86	$-$

Table 4.2: Average time of each exploration strategy with 4 robots. Buddy System was not run on Building C. All other statistics are gathered from three runs, except Reserves in Building C was only run once.

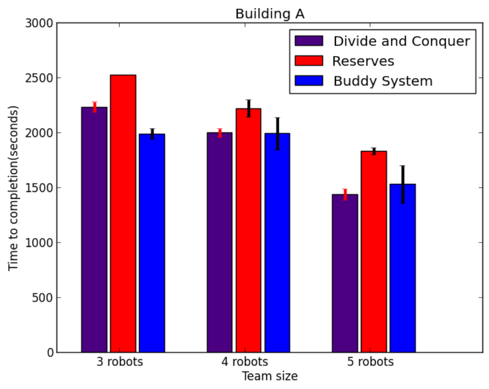
Strategy	Building A	Building B	Building C
Divide and Conquer	1997 ± 46	1188 ± 93	773 ± 31
Reserves	2221 ± 95	1271 ± 36	$1602 \pm -$
Buddy System	1991 ± 182	1237 ± 90	$-$

Table 4.3: Average time of each exploration strategy with 5 robots.

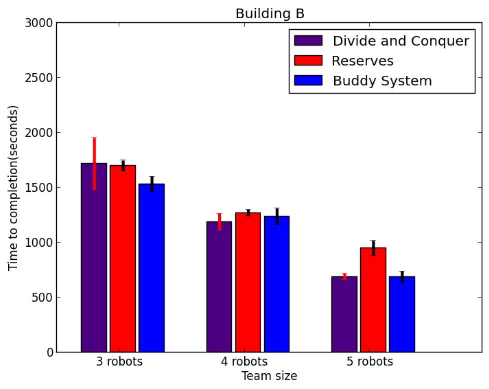
Strategy	Building A	Building B	Building C
Divide and Conquer	1437 ± 61	689 ± 35	920 ± 69
Reserves	1832 ± 38	947 ± 83	$1689 \pm -$
Buddy System	1529 ± 212	685 ± 66	$-$

We analyze each strategy in terms of the average time that the robots took to finish the exploration task given team size and collaboration strategy. As the results indicated in figures 4.28(a), 4.28(b), and 4.28(c), the strategies *Divide and Conquer* and *Buddy System* always performed better than the *Reserves* strategy. The performance also always increases as the team size is increased in buildings A and B.

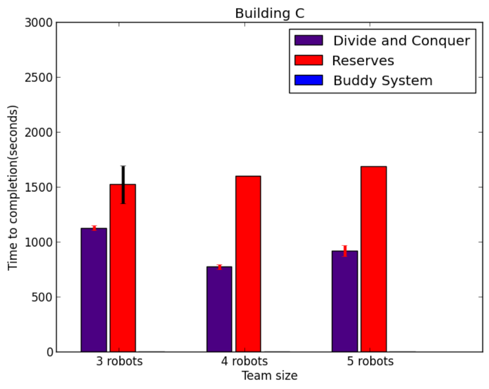
This results is consistent with the office building results presented in the first series of experiments. Those experiments showed decreased performance improvement only once the team size was increased beyond 6 robots, which is larger than the teams we were able to test



(a) Building A



(b) Building B



(c) Building C

Figure 4.28: Comparison of the strategies and team sizes the buildings at the test facility. Note that the *Buddy System* was not tested in Building C due to time constraints.

in this experiment series. The team performance is not improved in building C due to the poor conditions in that building including debris and concrete seams which inhibited progress.

4.3.6 Discussion

We have presented experiments which evaluate two collaboration strategies which can be used by teams of mobile robots to map and explore an unknown environment. We have also evaluated the impact of the number of robots on coverage in the exploration and mapping task.

The first collaboration strategy, called *Reserves* keeps a pool of unallocated robots at the starting location. A new robot is activated when there are more exploration frontiers than currently active robots. This strategy was intended to minimize the amount of interference between robot agents since robots would be far away from each other during exploration. The results from our experiments do not indicate that this strategy results in less interference than other strategies since performance decreases more when more robots are added in some environments. The *Reserves* strategy is significantly slower at exploring the environment than other strategies.

The second collaboration strategy, called *Divide and Conquer* has all available robots proceed in one large group. Once there are two exploration frontiers, at a corridor t-junction for example, the team will divide in half and each sub-team will follow one of the exploration frontiers. This process will be repeated with teams dividing in half each time they see branching structure in the environment. It was anticipated that this strategy would result in higher interference since robots would be maneuvering close together; however, the increased availability of robots near new exploration frontiers offsets this phenomenon.

Divide and Conquer appears to be a more effective strategy than *Reserves* for exploring and mapping an unknown environment. There are additional hybrid strategies which could now be considered such as the *Buddy System*, which modifies the *Reserves* strategy with teams of 2 robots instead of 1. We believe that this strategy will mitigate much of the slowness of the *Reserves* strategy while still minimizing interference.

4.4 Distributed mapping with privacy and communication constraints

The use of multiple cooperative robots has the potential to enable fast information gathering, and more efficient coverage and monitoring of large areas. For military applications, multi-robot systems promise more efficient operation and improved robustness to adversarial attacks. In civil applications (e.g., pollution monitoring, surveillance, search and rescue), the use of several inexpensive, heterogeneous, agile platforms is an appealing alternative to monolithic

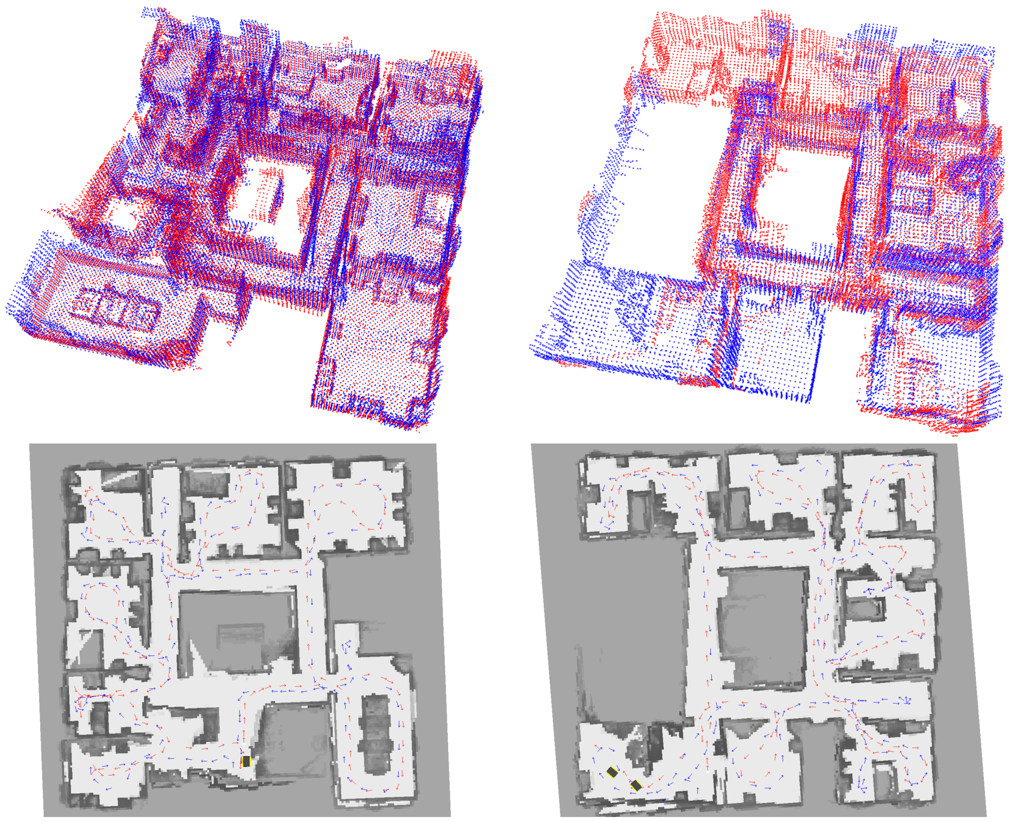


Figure 4.29: In our field experiments, distributed trajectory estimation enables 3D reconstruction of an entire building using two robots (red, blue). Each column of the figure shows the reconstructed point cloud of a floor (top), and the estimated trajectories overlaid on an occupancy grid map (bottom).

single robot systems.

The deployment of multi-robot systems in the real world poses many technical challenges, ranging from coordination and formation control, to task allocation and distributed sensor fusion. In this paper we tackle a specific instance of the sensor fusion problem. We consider the case in which a team of robots explores an unknown environment and each robot has to estimate its trajectory from its own sensor data and leveraging information exchange with the teammates. Trajectory estimation is relevant as it constitutes the backbone for many estimation and control tasks (e.g., geo-tagging sensor data, 3D map reconstruction, position-aware task allocation). Indeed, in our application, trajectory estimation enables distributed 3D reconstruction and localization (Figure 4.29).

We consider a realistic scenario, in which the robots can only communicate when they are within a given distance. Moreover, also during a rendezvous (i.e., when the robots are close enough to communicate) they cannot exchange a large amount of information, due to bandwidth constraints. Moreover, we aim at a technique that allows each robot to estimate its own trajectory, while asking for minimal knowledge of the trajectory of the teammates. This “privacy constraint” has a clear motivation in a military application: in case one robot is captured, it cannot provide sensitive information about the areas covered by the other robots in the team. Similarly, in civilian applications, one may want to improve the localization of a device (e.g., a smart phone) by exchanging information with other devices, while respecting users’ privacy. Ideally, we want our distributed mapping approach to scale to very large teams of robots. Our ultimate vision is to deploy a swarm of agile robots (e.g., micro aerial vehicles) that can coordinate by sharing minimal information and using on-board sensing and computation. This section takes a step in this direction and presents distributed mapping techniques that are shown to be extremely effective in large simulations (with up to 50 robots) and in real-world problem (with up to 4 robots).

4.4.1 Approach

We consider a distributed *ML* trajectory estimation problem in which the robots have to collaboratively estimate their trajectories while minimizing the amount of exchanged information. We focus on a fully 3D case, as this setup is of great interest in many robotics applications (e.g., navigation in uneven terrain, UAVs). We also consider a fully distributed setup, in which the robots can communicate and acquire relative measurements only during rendezvous events.

We present two key contributions to solve the distributed mapping problem. *The first contribution are a set of distributed algorithms that enable distributed inference at the estimation back-end.* Our approach can be understood as a distributed implementation of the chordal initialization discussed in [Car+15b]. The chordal initialization consists in approximating the

ML trajectory estimate by solving two quadratic optimization subproblems. The insight of the present work is that these quadratic subproblems can be solved in a distributed fashion,leveraging distributed linear system solvers. In particular, we investigate distributed implementations of the Jacobi Over-Relaxation and the Successive Over-Relaxation. These distributed solvers imply a communication burden that is linear in the number of rendezvous among the robots. Moreover, they do not rely on the availability of an accurate initial guess. The second contribution is *the use of high-level object-based models at the estimation front-end and as map representation*. Traditional approach for multi-robot mapping rely on feature-based maps which are composed of low level primitives like points and lines [Dav+07]. These maps become memory-intensive for long-term operation, contain a lot of redundant information (e.g., it is unnecessary to represent a planar surface with thousands of points), and lack the semantic information necessary for performing wider range of tasks (e.g., manipulation tasks, human-robot interaction).To solve these issues, we present an approach for multi-robot SLAM which uses object landmarks [Sal+13b] in a multi -robot mapping setup. We show that this approach further reduces the information exchange among robots, results in compact human-understandable maps, and has lower computational complexity as compared to low-level feature-based mapping. The third contribution is an *extensive experimental evaluation including realistic simulations in Gazebo and field tests in a military facility*. The experiments demonstrate that one of the proposed algorithms, namely the Distributed Gauss-Seidel method, provides accurate trajectory estimates,is more parsimonious, communication-wise, than related techniques, scales to large tea, and is robust to noise. Finally, our field tests show that the combined use of our distribute algorithms and object-based models reduces the communication requirements by several orders of magnitude and enables distributed mapping with large teams of robots.

4.4.2 Multi-Robot Pose Graph Optimization

We consider a multi-robot system and we denote each robot with a Greek letter, such that the set of robots is $\mathbf{\Omega} = \{\alpha, \beta, \gamma, \ldots\}$. The goal of each robot is to estimate its own trajectory using the available measurements, and leveraging occasional communication with other robots. The trajectory estimation problem and the nature of the available measurements are made formal in the rest of this section.

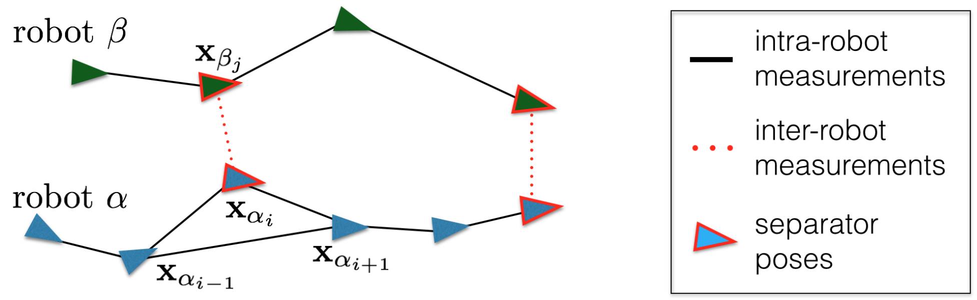


Figure 4.30: An instance of multi-robot trajectory estimation: two robots (α in blue, and β in dark green) traverse and unknown environment, collecting intra-robot measurements (solid black lines). During rendezvous, each robot can observe the pose of the other robot (dotted red lines). These are called inter-robot measurements and relate two *separators* (e.g., $\boldsymbol{x}_{\alpha_i}$, $\boldsymbol{x}_{\beta_j}$). The goal of the two robots is to compute the *ML* estimate of their trajectories.

We model each trajectory as a finite set of poses (triangles in Figure 4.30); the pose assumed by robot α at time i is denoted with $\boldsymbol{x}_{\alpha_i}$ (we use Roman letters to denote time indices). We are interested in a 3D setup, i.e., $\boldsymbol{x}_{\alpha_i} \in \mathrm{SE}(3)$; when convenient, we write $\boldsymbol{x}_{\alpha_i} = (\boldsymbol{R}_{\alpha_i}, \boldsymbol{t}_{\alpha_i})$, making explicit that each pose includes a rotation $\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3)$, and a position $\boldsymbol{t}_{\alpha_i} \in \mathbb{R}^3$. The trajectory of robot α is then denoted as $\boldsymbol{x}_\alpha = [\boldsymbol{x}_{\alpha_1}, \boldsymbol{x}_{\alpha_2}, \ldots]$.

Measurements.

We assume that each robot acquires relative pose measurements. In practice these are obtained by post-processing raw sensor data (e.g., scan matching on 3D laser scans). We consider two types of measurements: intra-robot and inter-robot measurements. The *intra-robot measurements* involve the poses of a single robot at different time instants; common examples of intra-robot measurements are odometry measurements (which constrain consecutive robot poses, e.g., $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{x}_{\alpha_{i+1}}$ in Figure 4.30) or loop closures (which constrain non-consecutive poses, e.g., $\boldsymbol{x}_{\alpha_{i-1}}$ and $\boldsymbol{x}_{\alpha_{i+1}}$ in Figure 4.30).

The *inter-robot measurements* are the ones relating the poses of different robots. For instance, during a rendezvous, robot α (whose local time is i), observes a second robot β (whose local time is j) and uses on-board sensors to measure the relative pose of the observed robot in its own reference frame. Therefore, robot α acquires an inter-robot measurement, describing the relative pose between $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{x}_{\beta_j}$ (red links in Figure 4.30). We use the term *separators* to refer to the poses involved in an inter-robot measurement.

While our classification of the measurements (inter vs intra) is based on the robots involved in the measurement process, all relative measurements can be framed within the same measurement model. Since all measurements correspond to noisy observation of the relative pose between a pair of poses, say $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{x}_{\beta_j}$, a general measurement model is:

$$\bar{\boldsymbol{z}}^{\alpha_i}_{\beta_j} \doteq (\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j}, \bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j}), \quad \text{with:} \begin{cases} \bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j} = (\boldsymbol{R}_{\alpha_i})^{\mathsf{T}} \boldsymbol{R}_{\beta_j} \boldsymbol{R}_{\varepsilon} \\ \bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j} = (\boldsymbol{R}_{\alpha_i})^{\mathsf{T}} (\boldsymbol{t}_{\beta_j} - \boldsymbol{t}_{\alpha_i}) + \boldsymbol{t}_{\varepsilon} \end{cases} \tag{4.4}$$

where the relative pose measurement $\bar{\boldsymbol{z}}^{\alpha_i}_{\beta_j}$ includes the relative rotation measurements $\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j}$, which describes the attitude $\boldsymbol{R}_{\beta_j}$ in the reference frame of robot α at time i, "plus" a random rotation $\boldsymbol{R}_{\varepsilon}$ (measurement noise), and the relative position measurement $\bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j}$, which describes the position $\boldsymbol{t}_{\beta_j}$ in the reference frame of robot α at time i, plus random noise $\boldsymbol{t}_{\varepsilon}$. According to our previous definition, intra robot measurements are in the form $\bar{\boldsymbol{z}}^{\alpha_i}_{\alpha_k}$, for some robot α and for $i \neq k$; inter-robot measurements are in the form $\bar{\boldsymbol{z}}^{\alpha_i}_{\beta_j}$ for two robots $\alpha \neq \beta$.

In the following, we denote with $\mathcal{E}^{\alpha}_{I}$ the set of intra-robot measurements for robot α, while we call $\mathcal{E}_I$ the set of intra-robot measurements for all robots in the team, i.e., $\mathcal{E}_I = \cup_{\alpha \in \boldsymbol{\Omega}} \mathcal{E}^{\alpha}_{I}$. The set of inter-robot measurements involving robot α is denoted with $\mathcal{E}^{\alpha}_{S}$ (S is the mnemonic for "separator"). The set of all inter-robot measurements is denoted with $\mathcal{E}_S$. The set of all available measurements is then $\mathcal{E} = \mathcal{E}_I \cup \mathcal{E}_S$.

Centralized *ML* Estimation

Let us collect all robot trajectories in a single (to-be-estimated) set of poses $\boldsymbol{x} = [\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta, \boldsymbol{x}_\gamma, \ldots]$. The *ML* estimate for $\boldsymbol{x}$ is defined as the maximum of the measurement likelihood:

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x}} \prod_{(\alpha_i,\beta_j)\in\mathcal{E}} \mathcal{L}(\bar{\boldsymbol{z}}^{\alpha_i}_{\beta_j} \mid \boldsymbol{x}) \tag{4.5}$$

where we took the standard assumption of independent measurements. The expression of the likelihood function depends on the distribution of the measurements noise, i.e., $\boldsymbol{R}_\varepsilon, \boldsymbol{t}_\varepsilon$ in Equation 4.4.

We follow the path of [Car+15a] and we assume that translation noise is distributed according to a zero-mean Gaussian with information matrix $\omega_t^2\mathbf{I}_3$, while the rotation noise x is distributed according to a Von-Mises distribution with concentration parameter ω_R^2. Under these assumptions, it is possible to demonstrate [Car+15a] that the $\boldsymbol{L}$ estimate $\boldsymbol{x} \doteq \{(\boldsymbol{R}_{\alpha_i}, \boldsymbol{t}_{\alpha_i}), \forall\alpha \in \Omega, \forall i\}$ can be computed as solution of the following optimization problem:

$$\min_{\substack{\boldsymbol{t}_{\alpha_i}\in\mathbb{R}^3 \\ \boldsymbol{R}_{\alpha_i}\in\mathrm{SO}(3) \\ \forall\alpha\in\boldsymbol{\Omega},\forall i}} \sum_{(\alpha_i,\beta_j)\in\mathcal{E}} \omega_t^2 \left\| \boldsymbol{t}_{\beta_j} - \boldsymbol{t}_{\alpha_i} - \boldsymbol{R}_{\alpha_i}\bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j} \right\|^2 + \frac{\omega_R^2}{2} \left\| \boldsymbol{R}_{\beta_j} - \boldsymbol{R}_{\alpha_i}\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j} \right\|_{\mathrm{F}}^2 \tag{4.6}$$

The peculiarity of problem 4.6 is the use of the *chordal distance* $\|\boldsymbol{R}_{\beta_j} - \boldsymbol{R}_{\alpha_i}\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j}\|_{\mathrm{F}}$ to quantify rotation errors, while the majority of related works in robotics uses the *geodesic distance* [Car+15b].

A centralized approach to solve the multi-robot PGO problem 4.6 works as follows. A robot collects all measurements $\mathcal{E}$. Then, the optimization problem 4.6 is solved using iterative optimization on manifold [Del12a] or fast approximations [Car+15b]. In this work, we consider the more interesting case in which it is not possible to collect all measurements at a centralized estimator. This problem can be formally stated as follows.

Problem (Distributed Trajectory Estimation)**.** *Design an algorithm that each robot* α *can execute during a rendezvous with a subset of other robots* $\Omega_r \subseteq \Omega \setminus \{\alpha\}$*, and that*

- *takes as input: (i) the intra-robot measurements* $\mathcal{E}_I^\alpha$ *and (ii) the subset of inter-robot measurements* $\mathcal{E}_S^\alpha$*, (iii) partial estimates of the trajectories of robots* $\beta \in \Omega_r$*;*
- *returns as output: the ML estimate* $\boldsymbol{x}_\alpha^\star$*, which is such that* $\boldsymbol{x}^\star = [\boldsymbol{x}_\alpha^\star, \boldsymbol{x}_\beta^\star, \boldsymbol{x}_\gamma^\star, \ldots]$ *is a minimizer of problem 4.6.*

Note that, while the measurements $\mathcal{E}_I^\alpha$ and $\mathcal{E}_S^\alpha$ are known by robot α, gathering the estimates from robots $\beta \in \Omega_r$ requires communication, hence we want our distributed algorithm to exchange a very small portion of the trajectory estimates.

Centralized Two-Stage PGO

The present work is based on two key observations. The first one is that the optimization problem 4.6 has a quadratic objective; what makes problem 4.6 hard is the presence of non-convex constraints, i.e., $\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3)$.

The second key observation is that each of this two stages can be solved in distributed fashion, exploiting existing distributed linear system solvers. We propose the use of a Distributed Jacobi algorithm. To help readability, we start with a centralized description of the approach, which is an adaptation of the chordal initialization of [Car+15b] to the multi-robot case.

Two-Stage Trajectory Estimation: Centralized Description

The approach proceeds in two stages. In the first stage, one solves for the unknown rotations of the robots by solving a relaxed problem. Then, one recovers the full poses via a single GN iteration. The two stages are detailed in the following:

Stage 1: rotation initialization via relaxation and projection

The first stage computes a good estimate of the rotations of all robots by solving the following rotation subproblem:

$$\min_{\substack{\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3) \\ \forall \alpha \in \boldsymbol{\Omega}, \forall i}} \sum_{(\alpha_i, \beta_j) \in \mathcal{E}} \omega_R^2 \left\| \boldsymbol{R}_{\beta_j} - \boldsymbol{R}_{\alpha_i} \bar{\boldsymbol{R}}_{\beta_j}^{\alpha_i} \right\|_{\mathrm{F}}^2 \tag{4.7}$$

which amounts to estimating the rotations of all robots in the team by considering only the relative rotation measurements (the second summand in problem 4.6). While problem 4.7 is nonconvex (due to the nonconvex constraints $\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3)$), many algorithms to approximate its solution are available in literature. Here we use the approach proposed in [MP07] and reviewed in [Car+15b]. The approach first solves the quadratic relaxation obtained by dropping the constraints $\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3)$, and then projects the relaxed solution to $\mathrm{SO}(3)$. In formulas, the quadratic relaxation is:

$$\min_{\boldsymbol{R}_{\alpha_i}, \forall \alpha \in \boldsymbol{\Omega}, \forall i} \sum_{(\alpha_i, \beta_j) \in \mathcal{E}} \omega_R^2 \left\| \boldsymbol{R}_{\beta_j} - \boldsymbol{R}_{\alpha_i} \bar{\boldsymbol{R}}_{\beta_j}^{\alpha_i} \right\|_{\mathrm{F}}^2 \tag{4.8}$$

which simply rewrites problem 4.7 without the constraints. Since problem 4.8 is quadratic in the unknown rotations $\boldsymbol{R}_{\alpha_i}, \forall \alpha \in \boldsymbol{\Omega}, \forall i$, we can rewrite it as:

$$\min_{\boldsymbol{r}} \|\boldsymbol{A}_r \boldsymbol{r} - \boldsymbol{b}_r\|^2 \tag{4.9}$$

where we stacked all the entries of the unknown rotation matrices $\boldsymbol{R}_{\alpha_i}, \forall \alpha \in \boldsymbol{\Omega}, \forall i$ into a single vector $\boldsymbol{r}$, and we built the (known) matrix $\boldsymbol{A}_r$ and (known) vector $\boldsymbol{b}_r$ accordingly (the presence of a nonzero vector $\boldsymbol{b}_r$ follows from setting one of the rotations to be the reference frame, e.g., $\boldsymbol{R}_{\alpha_1} = \mathbf{I}_3$). Since equation 4.8 is a linear least-squares problem, its solution can be found by solving the normal equations:

$$(\boldsymbol{A}_r^\top \boldsymbol{A}_r)\boldsymbol{r} = \boldsymbol{A}_r^\top \boldsymbol{b}_r \tag{4.10}$$

Let us denote with $\check{\boldsymbol{r}}$ the solution of Equation 4.10. Rewriting $\check{\boldsymbol{r}}$ in matrix form, we obtain the matrices $\check{\boldsymbol{R}}_{\alpha_i}, \forall\alpha \in \boldsymbol{\Omega}, \forall i$. Since these rotations were obtained from a relaxation of Equation 4.7, they are not guaranteed to satisfy the constraints $\boldsymbol{R}_{\alpha_i} \in \mathrm{SO}(3)$; therefore the approach [MP07] projects them to SO(3), and gets the rotation estimate $\hat{\boldsymbol{R}}_{\alpha_i} = \texttt{project}(\check{\boldsymbol{R}}_{\alpha_i}), \forall\alpha \in \boldsymbol{\Omega}, \forall i$. The projection only requires to perform an SVD of $\check{\boldsymbol{R}}_{\alpha_i}$ and can be performed independently for each rotation [Car+15b].

Stage 2: full pose recovery via single GN iteration.

In the previous stage we obtained an estimate for the rotations $\hat{\boldsymbol{R}}_{\alpha_i}, \forall\alpha \in \boldsymbol{\Omega}, \forall i$. In this stage we use this estimate to reparametrize problem 4.6. In particular, we rewrite each unknown rotation $\boldsymbol{R}_{\alpha_i}$ as the known estimate $\hat{\boldsymbol{R}}_{\alpha_i}$ "plus" an unknown perturbation; in formulas, we rewrite each rotation as $\boldsymbol{R}_{\alpha_i} = \hat{\boldsymbol{R}}_{\alpha_i}\mathrm{Exp}(\boldsymbol{\theta}_{\alpha_i})$, where $\mathrm{Exp}(\cdot)$ is the exponential map for SO(3), and $\boldsymbol{\theta}_{\alpha_i} \in \mathbb{R}^3$ (this is our new parametrization for the rotations). With this parametrization, Equation 4.6 becomes:

$$\begin{aligned}\min_{\substack{\boldsymbol{t}_{\alpha_i},\boldsymbol{\theta}_{\alpha_i}\\ \forall\alpha\in\boldsymbol{\Omega},\forall i}} \sum_{(\alpha_i,\beta_j)\in\mathcal{E}} & \omega_t^2 \left\| \boldsymbol{t}_{\beta_j} - \boldsymbol{t}_{\alpha_i} - \hat{\boldsymbol{R}}_{\alpha_i}\mathrm{Exp}(\boldsymbol{\theta}_{\alpha_i})\bar{\boldsymbol{t}}_{\beta_j}^{\alpha_i} \right\|^2 \\ & + \frac{\omega_R^2}{2} \left\| \hat{\boldsymbol{R}}_{\beta_j}\mathrm{Exp}\left(\boldsymbol{\theta}_{\beta_j}\right) - \hat{\boldsymbol{R}}_{\alpha_i}\mathrm{Exp}(\boldsymbol{\theta}_{\alpha_i})\bar{\boldsymbol{R}}_{\beta_j}^{\alpha_i} \right\|_{\mathrm{F}}^2\end{aligned} \tag{4.11}$$

Note that the reparametrization allowed to drop the constraints (we are now trying to estimate vectors in $\mathbb{R}^3$), but moved the nonconvexity to the objective ($\mathrm{Exp}(\cdot)$ is nonlinear in its argument). In order to solve Equation 4.11, we take a quadratic approximation of the cost function. For this purpose we use the following first-order approximation of the exponential map:

$$\mathrm{Exp}\,(\boldsymbol{\theta}_{\alpha_i}) \simeq \mathbf{I}_3 + \boldsymbol{S}(\boldsymbol{\theta}_{\alpha_i}) \tag{4.12}$$

where $\boldsymbol{S}(\boldsymbol{\theta}_{\alpha_i})$ is a skew symmetric matrix whose entries are defined by the vector $\boldsymbol{\theta}_{\alpha_i}$.

Substituting Equation 4.12 into Equation 4.11 we get the desired quadratic approximation:

$$\begin{aligned} \min_{\substack{\boldsymbol{t}_{\alpha_i},\boldsymbol{\theta}_{\alpha_i}\in\mathbb{R}^3 \\ \forall\alpha\in\boldsymbol{\Omega},\forall i}} \sum_{(\alpha_i,\beta_j)\in\mathcal{E}} & \omega_t^2 \left\| \boldsymbol{t}_{\beta_j} - \boldsymbol{t}_{\alpha_i} - \hat{\boldsymbol{R}}_{\alpha_i}\bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j} - \hat{\boldsymbol{R}}_{\alpha_i}\boldsymbol{S}(\boldsymbol{\theta}_{\alpha_i})\bar{\boldsymbol{t}}^{\alpha_i}_{\beta_j} \right\|^2 \\ & + \frac{\omega_R^2}{2} \left\| \hat{\boldsymbol{R}}_{\beta_j} - \hat{\boldsymbol{R}}_{\alpha_i}\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j} + \hat{\boldsymbol{R}}_{\beta_j}\boldsymbol{S}(\boldsymbol{\theta}_{\beta_j}) - \hat{\boldsymbol{R}}_{\alpha_i}\boldsymbol{S}(\boldsymbol{\theta}_{\alpha_i})\bar{\boldsymbol{R}}^{\alpha_i}_{\beta_j} \right\|_{\mathrm{F}}^2 \end{aligned} \tag{4.13}$$

Rearranging the unknown $\boldsymbol{t}_{\alpha_i}, \boldsymbol{\theta}_{\alpha_i}$ of all robots into a single vector $\boldsymbol{p}$, we rewrite Equation 4.13 as a linear least-squares problem:

$$\min_{\boldsymbol{p}} \; \|\boldsymbol{A}_p\,\boldsymbol{p} - \boldsymbol{b}_p\|^2 \tag{4.14}$$

whose solution can be found by solving the linear system:

$$(\boldsymbol{A}_p^\top \boldsymbol{A}_p)\boldsymbol{p} = \boldsymbol{A}_p^\top \boldsymbol{b}_p \tag{4.15}$$

From the solution of Equation 4.15 we can build our trajectory estimate: the entries of $\boldsymbol{p}$ directly define the positions $\boldsymbol{t}_{\alpha_i}, \forall\alpha\in\boldsymbol{\Omega}, \forall i$; moreover, $\boldsymbol{p}$ includes the rotational corrections $\boldsymbol{\theta}_{\alpha_i}$, from which we get our rotation estimate as: $\boldsymbol{R}_{\alpha_i} = \hat{\boldsymbol{R}}_{\alpha_i}\mathrm{Exp}\,(\boldsymbol{\theta}_{\alpha_i})$.

Remark 1 (Advantage of Centralized Two-Stage Approach)**.** *The approach reviewed in this section has three advantages. First, as shown in [Car+15b], in common problem instances (i.e., for reasonable levels of measurement noise) it returns a solution that is very close to the ML estimate.*

Second, the approach only requires to solve two linear systems (the cost of projecting the rotations is negligible), hence it is computationally efficient. Finally, the approach does not require an initial guess (the initial relaxation produces a good initial guess); therefore, the approach is able to converge even when the initial trajectory estimate is inaccurate (in those instances, iterative optimization tends to fail [Car+15b]).

4.4.3 Dealing With Bandwidth Constraints I: Distributed Algorithms

Distributed Pose Graph Optimization

In this section we show that the two-stage approach described in Section 4.4.2 can be implemented in a decentralized fashion. Since the approach only requires solving two linear systems, every distributed linear system solver can be used as workhorse to split the computation among the robots. For instance, one could adapt the Gaussian elimination approach of [CPD10] to solve the systems 4.10, 4.15. In this section we propose an alternative approaches, based on the Distributed Jacobi Over-Relaxation and the Distributed Successive Over-Relaxation algorithms, and we discuss their advantages.

In both Equations 4.10 and 4.15 we need to solve a linear system where the unknown vector can be partitioned into subvectors, such that each subvector contains the variables associated to a single robot in the team. For instance, we can partition the vector $\boldsymbol{r}$ in 4.10, as $\boldsymbol{r} = [\boldsymbol{r}_\alpha, \boldsymbol{r}_\beta, \ldots]$, such that $\boldsymbol{r}_\alpha$ describes the rotations of robot α. Similarly, we can partition $\boldsymbol{p} = [\boldsymbol{p}_\alpha, \boldsymbol{p}_\beta, \ldots]$ in Equation 4.15, such that $\boldsymbol{p}_\alpha$ describes the trajectory of α.

Therefore, Equations 4.10 and 4.15 can be framed in the general form:

$$\boldsymbol{H}\boldsymbol{y} = \boldsymbol{g} \Leftrightarrow \begin{bmatrix} \boldsymbol{H}_{\alpha\alpha} & \boldsymbol{H}_{\alpha\beta} & \cdots \\ \boldsymbol{H}_{\beta\alpha} & \boldsymbol{H}_{\beta\beta} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_{\alpha} \\ \boldsymbol{y}_{\beta} \\ \vdots \end{bmatrix} = \begin{bmatrix} \boldsymbol{g}_{\alpha} \\ \boldsymbol{g}_{\beta} \\ \vdots \end{bmatrix} \tag{4.16}$$

where we want to compute the vector $\boldsymbol{y} = [\boldsymbol{y}_{\alpha}, \boldsymbol{y}_{\beta}, \ldots]$ given $\boldsymbol{H}$ and $\boldsymbol{g}$; on the right of Equation 4.16 we partitioned the square matrix $\boldsymbol{H}$ and the vector $\boldsymbol{g}$ according to the block-structure of $\boldsymbol{y}$.

In order to introduce the distributed Jacobi algorithm, we first observe that the linear system 4.16 can be rewritten as:

$$\sum_{\delta \in \boldsymbol{\Omega}} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_{\delta} = \boldsymbol{g}_{\alpha} \qquad \forall \alpha \in \boldsymbol{\Omega}$$

Taking the contribution of $\boldsymbol{y}_{\alpha}$ out of the sum, we get:

$$\boldsymbol{H}_{\alpha\alpha}\boldsymbol{y}_{\alpha} = -\sum_{\delta \in \boldsymbol{\Omega} \setminus \{\alpha\}} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_{\delta} + \boldsymbol{g}_{\alpha} \qquad \forall \alpha \in \boldsymbol{\Omega} \tag{4.17}$$

The set of equations 4.17 is the same as the original system 4.16, but clearly exposes the contribution of the variables associated to each robot. The equations 4.17 constitute the basis for the *Successive Over-Relaxation* (SOR) and the *Jacobi Over-Relaxation* (JOR) methods that we describe in the following sections.

Distributed Jacobi Over-Relaxation (JOR):

The distributed JOR algorithm [BT89] starts at an arbitrary initial estimate $\boldsymbol{y}^{(0)} = [\boldsymbol{y}_{\alpha}^{(0)}, \boldsymbol{y}_{\beta}^{(0)}, \ldots]$ and solves the linear system 4.16 by repeating the following iterations:

$$\boldsymbol{y}_{\alpha}^{(k+1)} = (1-\gamma)\boldsymbol{y}_{\alpha}^{(k)} + (\gamma)\boldsymbol{H}_{\alpha\alpha}^{-1}\left(-\sum_{\delta \in \boldsymbol{\Omega} \setminus \{\alpha\}} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_{\delta}^{(k)} + \boldsymbol{g}_{\alpha}\right) \quad \forall \alpha \in \boldsymbol{\Omega} \tag{4.18}$$

where γ is the *relaxation factor*. Intuitively, at each iteration robot α attempts to solve Equation 4.17 (the second summand in Equation 4.18 is the solution of Equation 4.17 with the estimates of the other robots kept fixed), while remaining close to the previous estimate $\boldsymbol{y}_\alpha^{(k)}$ (first summand in 4.18). If the iterations 4.18 converge to a fixed point, say $\boldsymbol{y}_\alpha$ $\forall\alpha$, then the resulting estimate solves the linear system 4.17 exactly [BT89, page 131]. To prove this fact we only need to rewrite Equation 4.18 after convergence:

$$\boldsymbol{y}_\alpha = (1-\gamma)\boldsymbol{y}_\alpha + (\gamma)\boldsymbol{H}_{\alpha\alpha}^{-1}\left(-\sum_{\delta\in\boldsymbol{\Omega}\setminus\{\alpha\}} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_\delta + \boldsymbol{g}_\alpha\right)$$

which can be easily seen to be identical to Equation 4.17.

In our multi-robot problem, the distributed JOR algorithm can be understood in a simple way: at each iteration, each robot estimates its own variables ($\boldsymbol{y}_\alpha^{(k+1)}$) by assuming that the ones of the other robots are constant ($\boldsymbol{y}_\delta^{(k)}$); iterating this procedure, the robots reach an agreement on the estimates, and converge to the solution of Equation 4.16. Using the distributed JOR approach, the robots solve Equations 4.10 and 4.15 in a distributed manner. When $\gamma = 1$, the distributed JOR method is also known as the *distributed Jacobi* (DGS) method.

We already mentioned that when the iterations 4.18 converge, then they return the exact solution of the linear system. So a natural question is: *when do the Jacobi iteration converge?* A general answer is given by the following proposition.

Proposition 2 (**Convergence of JOR** [BT89]). *Consider the linear systems 4.16 and define the block diagonal matrix* $\boldsymbol{D} \doteq \operatorname{diag}\left(\boldsymbol{H}_{\alpha\alpha}, \boldsymbol{H}_{\beta\beta}, \ldots\right)$. *Moreover, define the matrix:*

$$\boldsymbol{M} = (1-\gamma)\mathbf{I} - \gamma\boldsymbol{D}^{-1}(\boldsymbol{H}-\boldsymbol{D}) \tag{4.19}$$

where $\mathbf{I}$ *is the identity matrix of suitable size. Then, the JOR iterations 4.18 converge from any initial estimate if and only if* $\rho(\boldsymbol{M}) < 1$, *where* $\rho(\cdot)$ *denotes the spectral radius (maximum of* *absolute value of the eigenvalues) of a matrix.*

The proposition is the same of Proposition 6.1 in [BT89] (the condition that $\mathbf{I} - \boldsymbol{M}$ is invertible is guaranteed to hold as noted in the footnote on page 144 of [BT89]).

It is non-trivial to establish whether our linear systems 4.10 and 4.15 satisfy the condition of Proposition 2. In the experimental section, we empirically observe that the Jacobi iterations indeed converge whenever $\gamma \leq 1$. For the SOR algorithm, presented in the next section, instead, we can provide stronger theoretical convergence guarantees.

Distributed Successive Over-Relaxation (SOR)

The distributed SOR algorithm [BT89] starts at an arbitrary initial estimate $\boldsymbol{y}^{(0)} = [\boldsymbol{y}_\alpha^{(0)}, \boldsymbol{y}_\beta^{(0)}, \ldots]$ and, at iteration k, applies the following update rule, for each $\alpha \in \boldsymbol{\Omega}$:

$$\boldsymbol{y}_\alpha^{(k+1)} = (1-\gamma)\boldsymbol{y}_\alpha^{(k)} + (\gamma)\boldsymbol{H}_{\alpha\alpha}^{-1}\left(-\sum_{\delta \in \boldsymbol{\Omega}_\alpha^+} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_\delta^{(k+1)} - \sum_{\delta \in \boldsymbol{\Omega}_\alpha^-} \boldsymbol{H}_{\alpha\delta}\boldsymbol{y}_\delta^{(k)} + \boldsymbol{g}_\alpha \right) \tag{4.20}$$

where γ is the *relaxation factor*, $\boldsymbol{\Omega}_\alpha^+$ is the set of robots that already computed the $(k+1)$-th estimate, while $\boldsymbol{\Omega}_\alpha^-$ is the set of robots that still have to perform the update 4.20, excluding node α (intuitively: each robot uses the latest estimate). As for the JOR algorithm, by comparing Equations 4.20 and 4.17, we see that if the sequence produced by the iterations 4.20 converges to a fixed point, then such point satisfies 4.17, and indeed solves the original linear system 4.16. When $\gamma = 1$, the distributed SOR method is known as the *distributed Gauss-Seidel* (DGS) method.

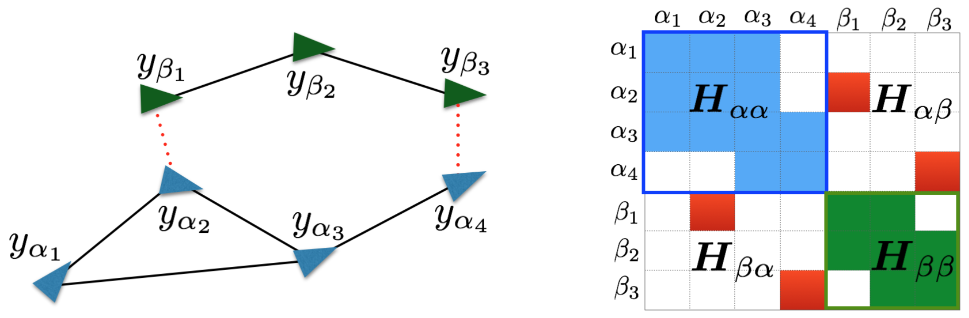
The following proposition, whose proof trivially follows from [BT89, Proposition 6.10, p. 154] (and the fact that the involved matrices are positive definite), establishes when the distributed SOR algorithm converges to the desired solution.

Proposition 3 (Convergence of SOR). *The SOR iterations 4.20 applied to the linear systems 4.10 and 4.15 converge to the solution of the corresponding linear system (from any* *initial estimate) whenever* $\gamma \in (0,2)$, *while the iterations do no converge to the correct solution whenever* $\gamma \notin (0,2)$.

According to [BT89, Proposition 6.10, p. 154], for $\gamma \notin (0,2)$, the SOR iterations 4.20 do not converge to the solution of the linear system in general, hence also in practice, we restrict the choice of γ in the open interval $(0,2)$. In the experimental section, we show that the choice $\gamma = 1$ ensures the fastest convergence.

Communication Requirements for JOR and SOR

In this section we observe that to execute the JOR and SOR iterations 4.18 and 4.20, robot α only needs its intra and inter-robot measurements $\mathcal{E}_I^\alpha$ and $\mathcal{E}_S^\alpha$, and an estimate of the separators, involved in the inter-robot measurements in $\mathcal{E}_S^\alpha$. For instance, in the graph of Figure 4.31 robot α only needs the estimates of $\boldsymbol{y}_{\beta_1}$ and $\boldsymbol{y}_{\beta_3}$, while does not require any knowledge about the other poses of β.



y_{β_1}
y_{β_2}
y_{β_3}
y_{α_1}
y_{α_2}
y_{α_3}
y_{α_4}
α_1 α_2 α_3 α_4 β_1 β_2 β_3
$\boldsymbol{H}_{\alpha\alpha}$
$\boldsymbol{H}_{\alpha\beta}$
$\boldsymbol{H}_{\beta\alpha}$
$\boldsymbol{H}_{\beta\beta}$

Figure 4.31: Example: (left) trajectory estimation problem and (right) corresponding block structure of the matrix $\boldsymbol{H}$.

To understand this fact, we note that both Equations 4.10 and 4.15 model an estimation problem from pairwise relative measurements. It is well known that the matrix $\boldsymbol{H}$ (sometimes called the *Hessian* [Del05a]) underlying these problems has a block structure defined by the Laplacian matrix of the underlying graph [BH07]. For instance, Figure 4.31 (right) shows the block sparsity of the matrix $\boldsymbol{H}$ describing the graph on the left: off-diagonal block-elements in position (α_i, β_j) are non zero if and only if there is an edge (i.e., a measurement) between α_i and

β_j.

By exploiting the block sparsity of $\boldsymbol{H}$, we can further simplify the JOR 4.18 iterations as:

$$\boldsymbol{y}_\alpha^{(k+1)} = (1-\gamma)\boldsymbol{y}_\alpha^{(k)} + (\gamma)\boldsymbol{H}_{\alpha\alpha}^{-1}\left(-\sum_{(\alpha_i,\delta_j)\in\mathcal{E}_S^\alpha} \boldsymbol{H}_{\alpha_i\delta_j}\boldsymbol{y}_{\delta_j}^{(k)} + \boldsymbol{g}_\alpha\right),\ \forall\alpha\in\boldsymbol{\Omega} \tag{4.21}$$

where we simply removed the contributions of the zero blocks from the sum in Equation 4.18.

Similarly we can simplify the SOR 4.20 iterations as:

$$\boldsymbol{y}_\alpha^{(k+1)} = (1-\gamma)\boldsymbol{y}_\alpha^{(k)} + (\gamma)\boldsymbol{H}_{\alpha\alpha}^{-1}\left(-\sum_{(\alpha_i,\delta_j)\in\mathcal{E}_S^{\alpha+}} \boldsymbol{H}_{\alpha_i\delta_j}\boldsymbol{y}_{\delta_j}^{(k+1)} - \sum_{(\alpha_i,\delta_j)\in\mathcal{E}_S^{\alpha-}} \boldsymbol{H}_{\alpha_i\delta_j}\boldsymbol{y}_{\delta_j}^{(k)} + \boldsymbol{g}_\alpha\right) \tag{4.22}$$

where we removed the contributions of the zero blocks from the sum in 4.20; the sets $\mathcal{E}_S^{\alpha+}$ and $\mathcal{E}_S^{\alpha-}$ satisfy $\mathcal{E}_S^{\alpha+}\cup\mathcal{E}_S^{\alpha-}=\mathcal{E}_S^\alpha$, and are such that $\mathcal{E}_S^{\alpha+}$ includes the inter-robot measurements involving robots which already performed the $(k+1)$-th iteration, while $\mathcal{E}_S^{\alpha-}$ is the set of measurements involving robots that have not performed the iteration yet (as before: each robot simply uses its latest estimate).

Equations 4.21 and 4.22 highlight that the JOR and SOR iterations (at robot α) only require the estimates for poses involved in its inter-robot measurements $\mathcal{E}_S^\alpha$. Therefore both JOR and SOR involve almost no "privacy violation": every other robot β in the team does not need to communicate any other information about its own trajectory, but only sends an estimate of its rendezvous poses.

The DGS algorithm starts from the insight that we can rewrite the linear system 4.16, by splitting the block diagonal terms of $\boldsymbol{H}$ from the off-diagonal terms: We describe the DGS algorithm to solve a generic Matrices are *compatible with the graph*: nonzero blocks only for edges in the graph. Essentially, we solve the linear systems of Section 4.4.2 using the distributed Jacobi algorithm. Note that we only need to solve two linear systems to get a very good solution, while with DDF-SAM we may need to perform multiple GN iterations and we need complex bookkeeping of the linearization points. description from [Ber97], page 131,143 Consider a linear system

$$Ax = b \tag{4.23}$$

with $x \in \mathbb{R}^{dn}$ and $A \in \mathbb{R}^{dn \times dn}$. Considering a partition of the vector x into n sub-vectors $x_i \in \mathbb{R}^d$ (this describe the trajectory of each robot). Partition the matrix A accordingly. Therefore the linear equation above can be written as:

$$\sum_{j=1}^{n} A_{ij} x_j = b_i \tag{4.24}$$

where A_{ij} is a block of A. Assuming A_{ii} is invertible for each $i = 1, \ldots, n$, we can solve for x_i:

$$x_i = -A_{ii}^{-1} [\sum_{j \neq i} A_{ij} x_j - b_i] \tag{4.25}$$

The Jacobi algorithm is an iterative solution for the linear system, and works by iterating the following computation:

$$x_i(t+1) = -A_{ii}^{-1} [\sum_{j \neq i} A_{ij} x_j(t) - b_i] \tag{4.26}$$

what makes it interesting in our context is that, we can simplify the iterations as:

$$x_i(t+1) = -A_{ii}^{-1} [\sum_{j \in \mathcal{N}_i} A_{ij} x_j(t) - b_i] \tag{4.27}$$

where $\mathcal{N}_i$ are the neighbors of robot i. The simplification follows from the block sparsity of the matrix A. If the sequence converges, then it must satisfy Equation 4.25, hence solving the original linear system.

4.4.4 Flagged Initialization

As we will see in the experimental section and according to Proposition 3, the JOR and SOR approaches converge from any initial condition when γ is chosen appropriately. However, starting from a "good" initial condition can reduce the number of iterations to converge, and in turns reduces the communication burden (each iteration 4.21 or 4.22 requires the robots to exchange their estimate of the separators).

In this work, we follow the path of [BH05a] and adopt a *flagged initialization*. A flagged initialization scheme only alters the first JOR or SOR iteration as follows. Before the first iteration, all robots are marked as "uninitialized". Robot α performs its iteration 4.21 or 4.22 without considering the inter-robot measurements, i.e., Equations 4.21 and 4.22 become $\boldsymbol{y}_{\alpha}^{(k+1)} = \boldsymbol{H}_{\alpha\alpha}^{-1}\boldsymbol{g}_{\alpha}$; then the robot α marks itself as "initialized". When the robot β performs its iteration, it includes only the separators from the robots that are initialized; after performing the JOR or SOR iteration, also β marks itself as initialized. Repeating this procedure, all robots become initialized after performing the first iteration. The following iterations then proceed according to the standard JOR 4.21 or SOR 4.22 update. [BH05a] show a significant improvement in convergence using flagged initialization. As discussed in the experiments, flagged initialization is also advantageous in our distributed pose graph optimization problem.

4.4.5 Dealing With Bandwidth Constraints II: Compressing Sensor Data via Object-based Representations

The second contribution of this paper is the use of high-level object-based models at the estimation front-end and as a map representation for multi-robot SLAM. Object-based abstractions are crucial to further reduce the memory storage and the information exchange among the robots.

Previous approaches for multi-robot mapping rely on feature-based maps which become memory-intensive for long-term operation, contain a large amount of redundant information, and lack the semantic understanding necessary to perform a wider range of tasks (e.g., manipulation, human-robot interaction). To solve these issues, we present an approach for multi-robot SLAM which uses object landmarks [Sal+13b] in a multi-robot mapping setup.

Section 4.4.5 introduces the additional mathematical notation and formalizes the problem of distributed object-based SLAM. Section 4.4.6 presents the implementation details of our distributed object-based SLAM system.

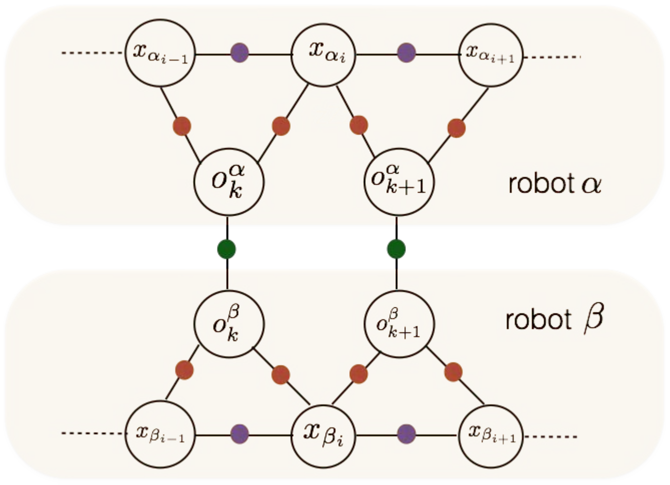


Figure 4.32: Factor graph representation of Multi-Robot Object based SLAM. $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{x}_{\beta_i}$ denote the poses assumed by robot α and β at time i respectively. The pose of the k^{th} object as estimated by robot α and β is denoted with $\boldsymbol{o}_k^\alpha$ and $\boldsymbol{o}_k^\beta$ respectively. Green dots shows inter-robot factors whereas orange and purple dots shows intra-robot factors.

Distributed Object-based SLAM

We consider a multi-robot system as defined in Section 4.4.2. Each robot, in addition to estimating its own trajectory using local measurements and occasional communication with other robots, also estimates the pose of a set of objects in the environment. We model each trajectory as a finite set of poses; the trajectory of robot α is $\boldsymbol{x}_\alpha = [\boldsymbol{x}_{\alpha_1}, \boldsymbol{x}_{\alpha_2}, \ldots]$. In addition, we denote with $\boldsymbol{o}_k^\alpha \in \mathrm{SE}(3)$ the pose of the k^{th} object as estimated by of robot α (Figure 4.32).

Measurements Similar to distributed pose graph optimization (Section 4.4.2), we assume that each robot acquires two types of relative pose measurements: intra-robot and inter-robot measurements. The *intra-robot measurements* consist of the odometry measurements, which constrain consecutive robot poses (e.g., $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{x}_{\alpha_{i+1}}$ in Figure 4.32), and object measurements which constrains robot poses with the corresponding visible object landmarks (e.g., $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{o}_k^{\alpha}$ in Figure 4.32). Contrarily to Section 4.4.2, the *inter-robot measurements* relate the object poses observed by different robots. During a rendezvous between robot α and robot β, each robot shares the label and pose of detected object landmarks with the other robot. Then, for each object observed by both robots, the teammates add an inter-robot measurements, enforcing the object pose estimate to be consistent across the teammates.

For instance, if $\boldsymbol{o}_k^{\beta}$ and $\boldsymbol{o}_k^{\alpha}$ in Figure 4.32 model the pose of the same object, then the two poses should be identical in the global coordinate frame. For this reason, inter-robot measurement between a pair of associated object poses is the identity.

The intra-robot object measurements follow the same measurements model of Equation 4.4. For instance, if the robot α at time i and at pose $\boldsymbol{x}_{\alpha_i}$ observes an object at pose $\boldsymbol{o}_k^{\alpha}$, then the corresponding measurement $\bar{\boldsymbol{z}}_{\boldsymbol{o}_k^{\alpha}}^{\boldsymbol{x}_{\alpha_i}}$ measures the relative pose between $\boldsymbol{x}_{\alpha_i}$ and $\boldsymbol{o}_k^{\alpha}$. Similarly we denote inter-robot measurement between object poses $\boldsymbol{o}_k^{\alpha}$ and $\boldsymbol{o}_k^{\beta}$ as $\bar{\boldsymbol{z}}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}}$. In order to ensure that the object pose estimate is consistent across teammates, we define the inter-robot measurement model $\bar{\boldsymbol{z}}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}}$ as:

$$\bar{\boldsymbol{z}}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}} \doteq (\mathbf{I}, \mathbf{0}), \quad \text{with:} \begin{cases} \boldsymbol{R}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}} = (\boldsymbol{R}_{\boldsymbol{o}_k^{\alpha}})^{\mathsf{T}} \boldsymbol{R}_{\boldsymbol{o}_k^{\beta}} \boldsymbol{R}_{\varepsilon} = \mathbf{I} \\ \bar{\boldsymbol{t}}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}} = (\boldsymbol{R}_{\boldsymbol{o}_k^{\alpha}})^{\mathsf{T}} (\boldsymbol{t}_{\boldsymbol{o}_k^{\beta}} - \boldsymbol{t}_{\boldsymbol{o}_k^{\alpha}}) + \boldsymbol{t}_{\varepsilon} = \mathbf{0} \end{cases} \tag{4.28}$$

where the relative object pose measurement $\bar{\boldsymbol{z}}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}}$ includes the relative rotation measurements $\boldsymbol{R}_{\boldsymbol{o}_k^{\beta}}^{\boldsymbol{o}_k^{\alpha}} = \mathbf{I}$, which describes the attitude of the estimated object pose $\boldsymbol{o}_k^{\beta}$, $\boldsymbol{R}_{\boldsymbol{o}_j^{\beta}}$ with respect to the reference frame of robot α, "plus" a random rotation $\boldsymbol{R}_{\varepsilon}$ (estimation noise), and the relative position measurement $\bar{\boldsymbol{t}}^{\boldsymbol{o}_k^\alpha}_{\boldsymbol{o}_k^\beta} = \mathbf{0}$, which describes the position $\boldsymbol{t}_{\boldsymbol{o}_k^\beta}$ in the reference frame of robot α, plus random noise $\boldsymbol{t}_\varepsilon$.

In the following, we denote with $\mathcal{E}_I^\alpha$ the set of intra-robot odometry for robot α, while we call $\mathcal{E}_I$ the set of intra-robot odometry measurements for all robots in the team, i.e., $\mathcal{E}_I = \cup_{\alpha \in \boldsymbol{\Omega}} \mathcal{E}_I^\alpha$. Similarly the set of intra-robot object measurements for robot α is denoted as $\mathcal{E}_o^\alpha$, whereas the set of all intra-robot object measurements is denoted as $\mathcal{E}_o$. Similar to Section 4.4.2, the set of inter-robot measurements involving robot α is denoted with $\mathcal{E}_S^\alpha$.

The set of all inter-robot measurements is denoted with $\mathcal{E}_S$. The set of all available measurements is then $\mathcal{E} = \mathcal{E}_I \cup \mathcal{E}_O \cup \mathcal{E}_S$. Note that each robot only has access to its own intra and inter-robot measurements $\mathcal{E}_I^\alpha$, $\mathcal{E}_O^\alpha$ and $\mathcal{E}_S^\alpha$.

4.4.6 Centralized *ML* Estimation

ML **trajectory and objects estimation.** Let us collect all robot trajectories and object poses in a (to-be-estimated) set of robot poses $\boldsymbol{x} = [\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta, \boldsymbol{x}_\gamma, \ldots]$ and set of object poses $\boldsymbol{o} = [\boldsymbol{o}^\alpha, \boldsymbol{o}^\beta, \boldsymbol{o}^\gamma, \ldots]$. The *ML* estimate for $\boldsymbol{x}$ and $\boldsymbol{o}$ is defined as the maximum of the measurement likelihood:

$$\boldsymbol{x}^\star, \boldsymbol{o}^\star = \arg\max_{\boldsymbol{x},\boldsymbol{o}} \prod_{(\boldsymbol{x}_{\alpha_i}, \boldsymbol{x}_{\alpha_{i+1}}) \in \mathcal{E}_I} \underbrace{\mathcal{L}(\bar{\boldsymbol{z}}^{\boldsymbol{x}_{\alpha_i}}_{\boldsymbol{x}_{\alpha_{i+1}}} \mid \boldsymbol{x})}_{\text{odometry factors}} \prod_{(\boldsymbol{x}_{\alpha_i}, \boldsymbol{o}_k^\alpha) \in \mathcal{E}_O} \underbrace{\mathcal{L}(\bar{\boldsymbol{z}}^{\boldsymbol{x}_{\alpha_i}}_{\boldsymbol{o}_k^\alpha} \mid \boldsymbol{x}, \boldsymbol{o})}_{\text{intra-robot object-measurement factors}} \prod_{(\boldsymbol{o}_i^\alpha, \boldsymbol{o}_j^\beta) \in \mathcal{E}_S} \underbrace{\mathcal{L}(\bar{\boldsymbol{z}}^{\boldsymbol{o}_i^\alpha}_{\boldsymbol{o}_j^\beta} \mid \boldsymbol{x}, \boldsymbol{o})}_{\text{inter-robot object-object factors}} \tag{4.29}$$

where we used the same assumptions on measurement noise as in Section 4.4.2. Defining

$\mathcal{X} = \boldsymbol{x} \cup \boldsymbol{o}$, we rewrite Equation 4.29 as:

$$\mathcal{X}^{\star} = \arg\max_{\mathcal{X}} \prod_{(\alpha_i,\beta_j)\in\mathcal{E}} \mathcal{L}(\bar{\boldsymbol{z}}^{\alpha_i}_{\beta_j} \mid \mathcal{X}) \tag{4.30}$$

Since the optimization problem in Equation 4.30 has the same structure of the one in Equation 4.5, we follow the same steps to solve it in a distributed manner using the Distributed Gauss-Seidel method. The next section presents the implementation details of our distributed object-based SLAM system.

Object-based SLAM Implementation

Object detection and pose estimation Each robot collects RGBD data using a depth camera, and measures its ego-motion through wheel odometry. In our approach, each RGB frame (from RGBD) is passed to the YOLO object detector [Red+16a], which detects objects at 45 frames per second. Compared to object-proposal-based detectors, YOLO is fast, since it avoids the computation burden of extracting object proposals, and is less likely to produce false positives in the background. We fine-tune the YOLO detector on a subset of objects from the *BigBird* dataset ([Sin+14]). The training dataset contains the object images in a clean background taken from different viewpoints and labeled images of the same objects taken by a robot in an indoor environment. During testing, we use a probability threshold of 0.3 to avoid false detections.

Each detected object bounding box is segmented using the *organized point cloud segmentation* [Tre+13]. The segmented object is then matched to the 3D template of the detected object class to estimate its pose. We extract PFHRGB features [Rus+08] in the source (object segment) and target (object model) point clouds and register the two point clouds in a Sample Consensus Initial Alignment framework [Rus09]. If we have at least 12 inlier correspondences, GICP (generalized iterative closest point [SHT05] is performed to further refine the registration and the final transformation is used as the object pose estimate. If less than 12 inlier correspondences are found, the detection is considered to be a false positive and the corresponding measurement is discarded. In hindsight, this approach verifies the detection both semantically and geometrically.

Data Association If object pose estimation is successful, it is data-associated with other instances already present in the map by finding the object landmark having the same category label within 2σ distance of the newly detected object. If there are multiple objects with the same label within that distance, the newly detected object is matched to the nearest object instance. If there exists no object having the same label, a new object landmark is created.

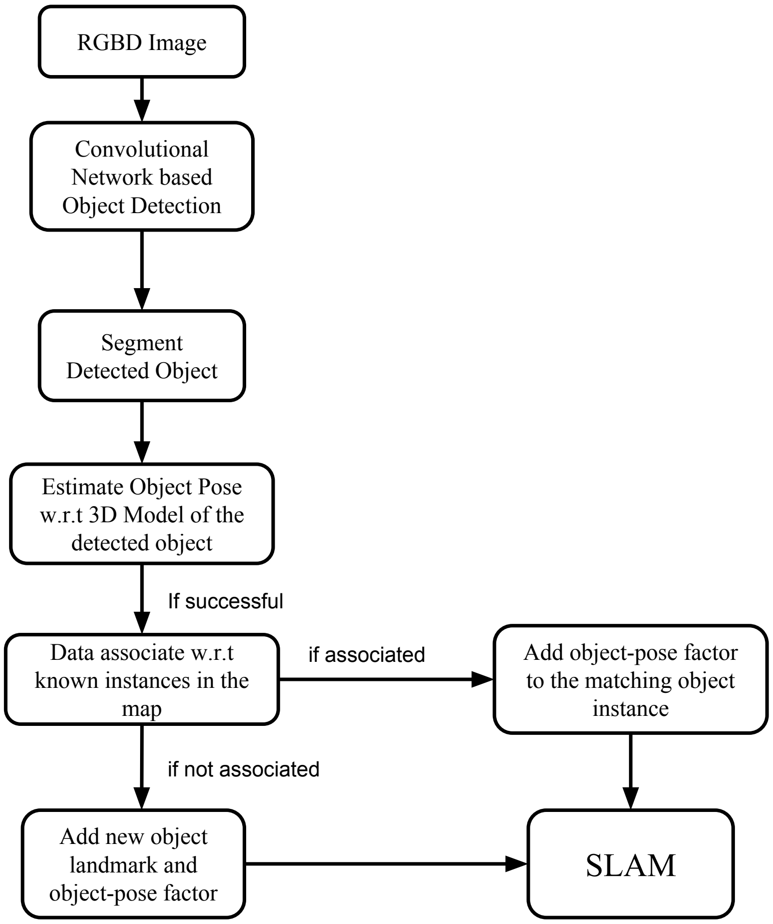

RGBD Image
Convolutional Network based Object Detection
Segment Detected Object
Estimate Object Pose w.r.t 3D Model of the detected object
If successful
Data associate w.r.t known instances in the map
if associated
Add object-pose factor to the matching object instance
if not associated
Add new object landmark and object-pose factor
SLAM

Figure 4.33: Flowchart of Object based SLAM

Before the first rendezvous event, each robot performs standard single-robot SLAM using OmniMapper [TRC12]. Both wheel odometry and relative pose measurements to the observed objects are fed to the SLAM back-end. A flowchart of the approach is given in Figure 4.33.

Robot Communication During a rendezvous between robots α and β, robot α communicates the category labels (class) and poses (in robot α's frame) of all the detected objects to robot β. We assume that the initial pose of each robot is known to all the robots, hence, given the initial pose of robot α, robot β is able to transform the communicated object poses from robot α's frame to its own frame.[†] For each object in the list communicated by robot α, robot β finds the nearest object in its map, having the same category label and within 2σ distance. If such an object exists, it is added to the list of *shared* objects: this is the set of objects seen by both robots. The list of shared objects contains pairs $(\boldsymbol{o}_k^\alpha, \boldsymbol{o}_l^\beta)$ and informs the robots that the poses $\boldsymbol{o}_k^\alpha$ and $\boldsymbol{o}_l^\beta$ correspond to the same physical object, observed by the two robots. For this reason, in the optimization we enforce the relative pose between $\boldsymbol{o}_k^\alpha$ and $\boldsymbol{o}_l^\beta$ to be zero.

We remark that, while before the first rendezvous the robots α and β have different reference frames, the object-object factors enforce both robots to have a single shared reference frame, facilitating future data association.

4.4.7 Experiments

We evaluate the distributed JOR and SOR along with DGS and DGS approaches (with and without using objects) in large-scale simulations (Sections 4.4.8 and 4.4.9) and field tests (Sections 4.4.10 and 4.4.11). The results demonstrate that (i) the DGS dominates the other algorithms considered in this paper in terms of convergence speed, (ii) the DGS algorithm is accurate, scalable, and robust to noise, (iii) the DGS requires less communication than techniques from related work (i.e., DDF-SAM), and (iv) in real applications, the combination of DGS and object-based mapping reduces the communication requirements by several orders of magnitude compared to approaches exchanging raw measurements.

[†]The knowledge of the initial pose is only used to facilitate data association but it is not actually used during pose graph optimization. We believe that this assumption can be easily relaxed but for space reasons we leave this task to future work.

4.4.8 Simulation Results: multi-robot Pose Graph Optimization

In this section, we characterize the performance of the proposed approaches in terms of convergence, scalability (in the number of robots and separators), and sensitivity to noise.

Simulation setup and performance metrics For our tests, we created simulation datasets in six different configurations with increasing number of robots: 4, 9, 16, 25, 36 and 49 robots. The robots are arranged in a 3D grid with each robot moving on a cube, as shown in Figure 4.34. When the robots are at contiguous corners, they can communicate (gray links).

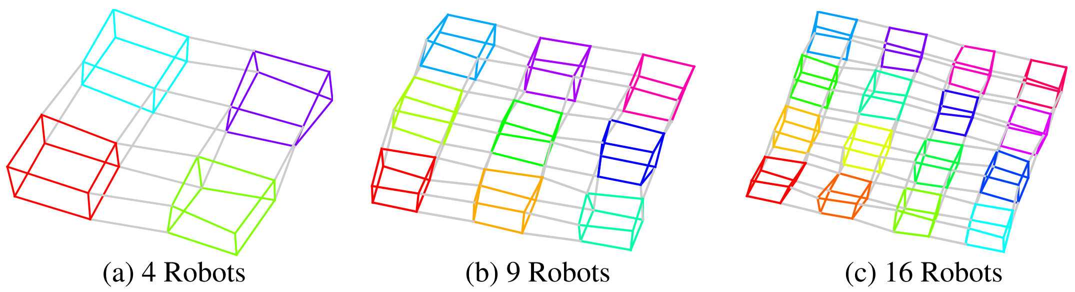

(a) 4 Robots (b) 9 Robots (c) 16 Robots

Figure 4.34: Simulated 3D datasets with different number of robots. Robots are shown in different colors. Gray links denote inter-robot measurements.

Unless specified otherwise, we generate measurement noise from a zero-mean Gaussian distribution with standard deviation $\sigma_R = 5^\circ$ for the rotations and $\sigma_t = 0.2$m for the translations. Results are averaged over 10 Monte Carlo runs.

In our problem, JOR or SOR are used to sequentially solve two linear systems, 4.10 and 4.15, which return the minimizers of 4.9 and 4.14, respectively. Defining, $m_r \doteq \min_{\boldsymbol{r}} \|\boldsymbol{A}_r\boldsymbol{r} - \boldsymbol{b}_r\|^2$, we use the following metric, named the *rotation estimation error*, to quantify the error in solving Equation 4.10:

$$e_r(k) = \|\boldsymbol{A}_r\boldsymbol{r}^{(k)} - \boldsymbol{b}_r\|^2 - m_r \tag{4.31}$$

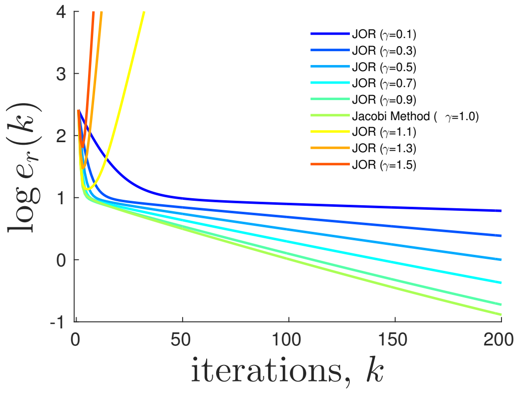
$e_r(k)$ quantifies how far is the current estimate $\boldsymbol{r}^{(k)}$ (at the k-th iteration) from the minimum of the quadratic cost. Similarly, we define the *pose estimation error* as:

$$e_p(k) = \|\boldsymbol{A}_p\boldsymbol{p}^{(k)} - \boldsymbol{b}_p\|^2 - m_p \tag{4.32}$$

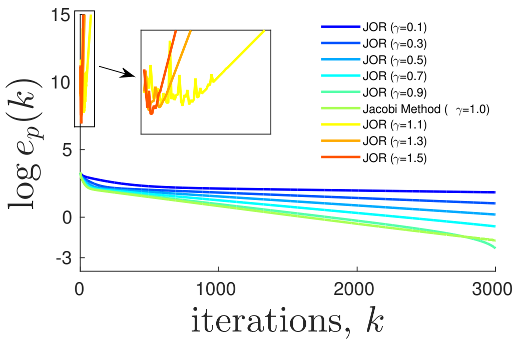
with $m_p \doteq \min_{\boldsymbol{p}} \; \|\boldsymbol{A}_p \, \boldsymbol{p} - \boldsymbol{b}_p\|^2$. Ideally, we want $e_r(k)$ and $e_p(k)$ to quickly converge to zero for increasing k.

Ultimately, the accuracy of the proposed approach depends on the number of iterations, hence we need to set a stopping condition for the JOR or SOR iterations. We use the following criterion: we stop the iterations if the change in the estimate is sufficiently small. More formally, the iterations stop when $\|\boldsymbol{r}^{(k+1)} - \boldsymbol{r}^{(k)}\| \leq \eta_r$ (similarly, for the second linear system $\|\boldsymbol{p}^{(k+1)} - \boldsymbol{p}^{(k)}\| \leq \eta_p$). We use $\eta_r = \eta_p = 10^{-1}$ as stopping condition unless specified otherwise.

Comparisons among the distributed algorithms In this section we consider the scenario with 49 robots. We start by studying the convergence properties of the JOR and SOR algorithms in isolation. Then we compare the two algorithms in terms of convergence speed. Figure 4.35 shows the rotation and the pose error versus the number of iterations for different choices of the parameter γ for the JOR algorithm. Figure 4.35a confirms the result of Proposition 2: JOR applied to the rotation subproblem converges as long as $\gamma \leq 1$. Figure 4.35a shows that for any $\gamma > 1$ the estimate diverges, while the critical value $\gamma = 1$ (corresponding to the DGS method) ensures the fastest convergence rate.



(a) Rotation Estimation



(b) Pose Estimation

Figure 4.35: JOR : convergence of (a) rotation estimation and (b) pose estimation for different values of γ (grid scenario, 49 robots). In the case of pose estimation, the gap between the initial values of $\gamma > 1$ and $\gamma \leq 1$ is due to the bad initialization provided by the rotation estimation for $\gamma > 1$.

Figure 4.36 shows the rotation and the pose error versus the number of iterations for different choices of the parameter $\gamma \in (0,2)$ for the SOR algorithm. The figure confirms the result of Proposition 3: the SOR algorithm converges for any choice of $\gamma \in (0,2)$.

Figure 4.36a shows that choices of γ close to 1 ensure fast convergence rates, while Figure 4.36b established $\gamma = 1$ (corresponding to the DGS method) as the parameter selection with faster convergence.

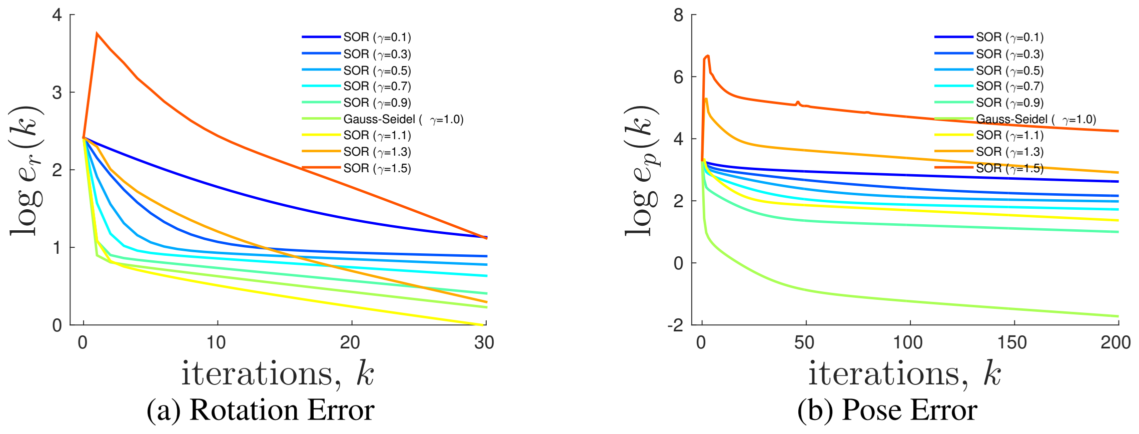
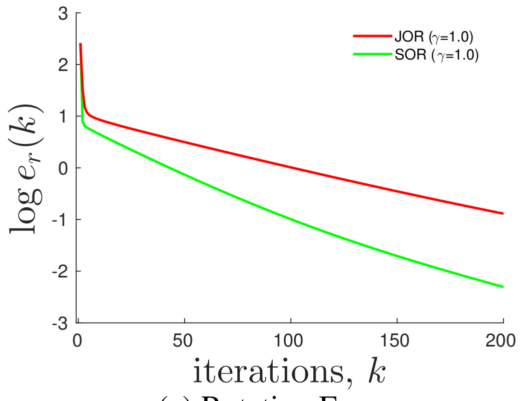


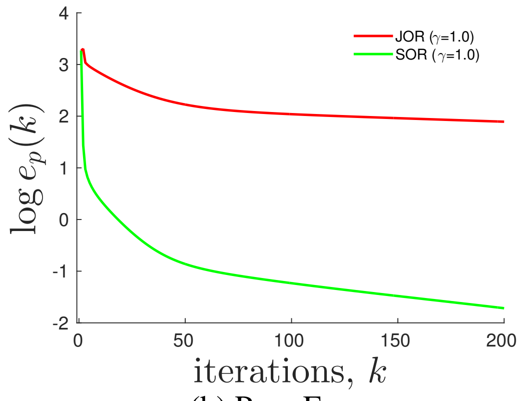
Figure 4.36: SOR : convergence of (a) rotation estimation and (b) pose estimation for different values of γ (grid scenario, 49 robots).

In summary, both JOR and SOR have top performance when $\gamma = 1$. Later in this section we show that $\gamma = 1$ is the best choice independently on the number of robots and the measurement noise.

Let us now compare JOR and SOR in terms of convergence. Figure 4.37 compares the convergence rate of SOR and JOR for both the rotation subproblem (Figure 4.37a) and the pose subproblem (Figure 4.37b). We set $\gamma = 1$ in JOR and SOR since we already observed that this choice ensures the best performance.

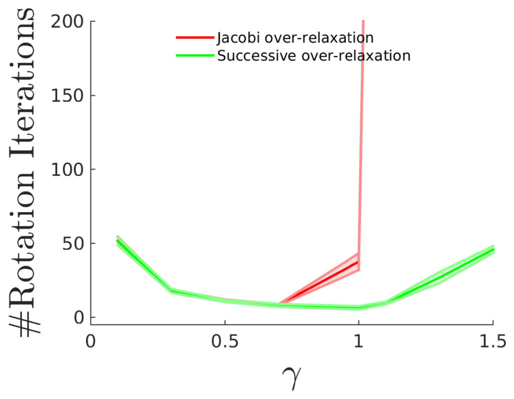


(a) Rotation Error

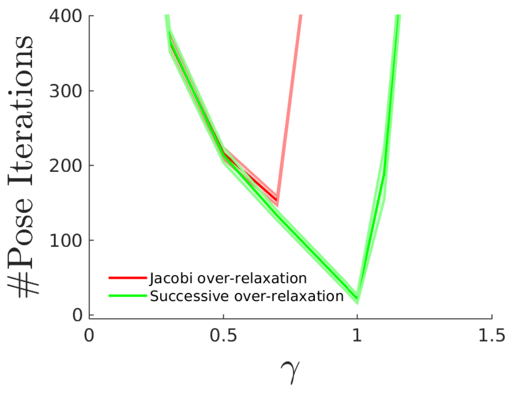


(b) Pose Error

Figure 4.37: JOR `vs` SOR : convergence of (a) rotation estimation and (b) pose estimation for the JOR and SOR algorithms with $\gamma = 1$ (grid scenario, 49 robots).



(a) Rotation Estimation



(b) Pose Estimation

Figure 4.38: JOR `vs` SOR : number of iterations required for (a) rotation estimation and (b) pose estimation for the JOR and SOR algorithms with $\gamma = 1$ (grid scenario, 49 robots). The average number of iterations is shown as a solid line, while the 1-sigma standard deviation is shown as a shaded area.

The figure confirms that SOR dominates JOR in both subproblems. Figure 4.38 shows the number of iterations for convergence (according to our stopping conditions) and for different choices of the parameter γ. Once again, the figure confirms that the SOR with $\gamma = 1$ is able to converge in the smallest number of iterations, requiring only few tens of iterations in both the rotation and the pose subproblem.

We conclude this section by showing that setting $\gamma = 1$ in SOR ensure faster convergence regardless the number of robots and the measurement noise. Figure 4.39 compares the number of

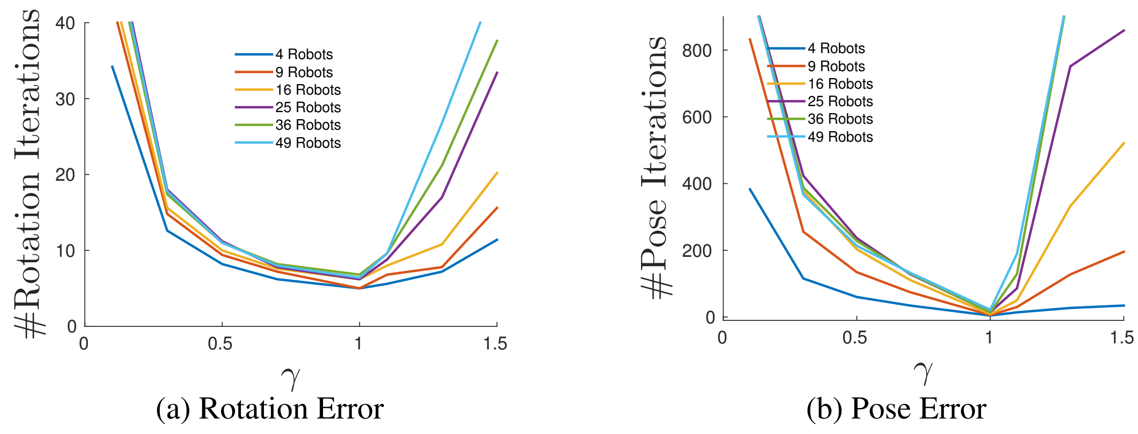


Figure 4.39: SOR : number of iterations required for (a) rotation estimation and (b) pose estimation in the SOR algorithm for different choices of γ and increasing number of robots.

iterations required to converge for increasing number of robots for varying γ values.

Similarly Figure 4.40 compares the number of iterations required to converge for increasing noise for varying γ value. We can see that in both the cases $\gamma = 1$ has the fastest convergence (required the least number of iterations) irrespective of the number of robots and measurement noise. Since SOR with $\gamma = 1$, i.e., the DGS method, is the top performer in all test conditions, in the rest of the paper we restrict our analysis to this algorithm.

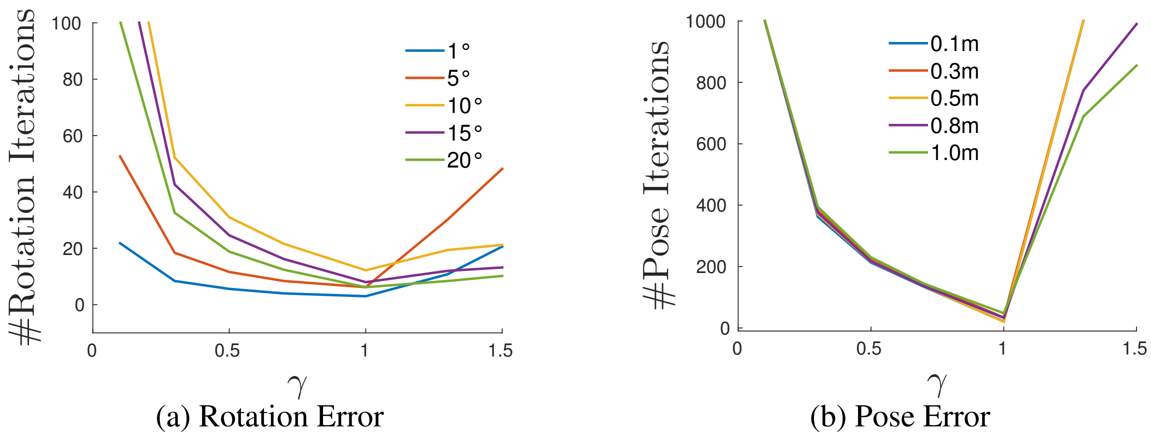
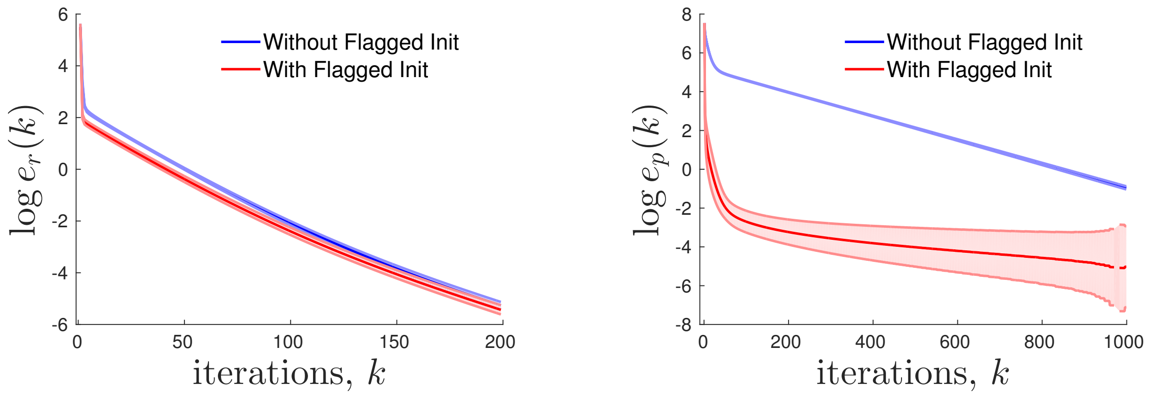


Figure 4.40: SOR : number of iterations required for (a) rotation estimation and (b) pose estimation in the SOR algorithm for different choices of γ and increasing measurement noise.

Flagged initialization In this paragraph we discuss the advantages of the flagged initialization. We compare the DGS method with flagged initialization against a naive initialization in which the variables ($\boldsymbol{r}^{(0)}$ and $\boldsymbol{p}^{(0)}$, respectively) are initialized to zero. The results, for the dataset with 49 robots, are shown in Figure 4.41.

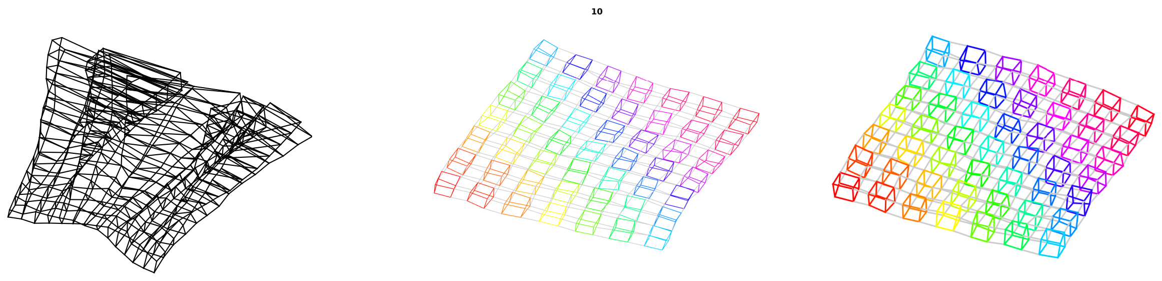


(a) Rotation Error (b) Pose Error

Figure 4.41: DGS : Comparison between flagged and non-flagged initialization on the grid scenario with 49 robots. Average estimation errors (solid line) and 1-sigma standard deviation (shaded area) are in log scale.

In both cases the estimation errors go to zero, but the convergence is faster when using the flagged initialization. The speed-up is significant for the second linear system (Figure 4.41b), for both linear systems. We noticed a similar advantage across all tested scenarios. Therefore, in the rest of the paper we always adopt the flagged initialization.

Stopping conditions and anytime flavor. This section provides extra insights on the convergence of the DGS method.



(a) Initial (b) 10 iterations (c) 1000 iterations

Figure 4.42: DGS : Trajectory estimates for the scenario with 49 robots. (a) Odometric estimate (not used in our approach and only given for visualization purposes), (b)-(c) DGS estimates after given number of iterations.

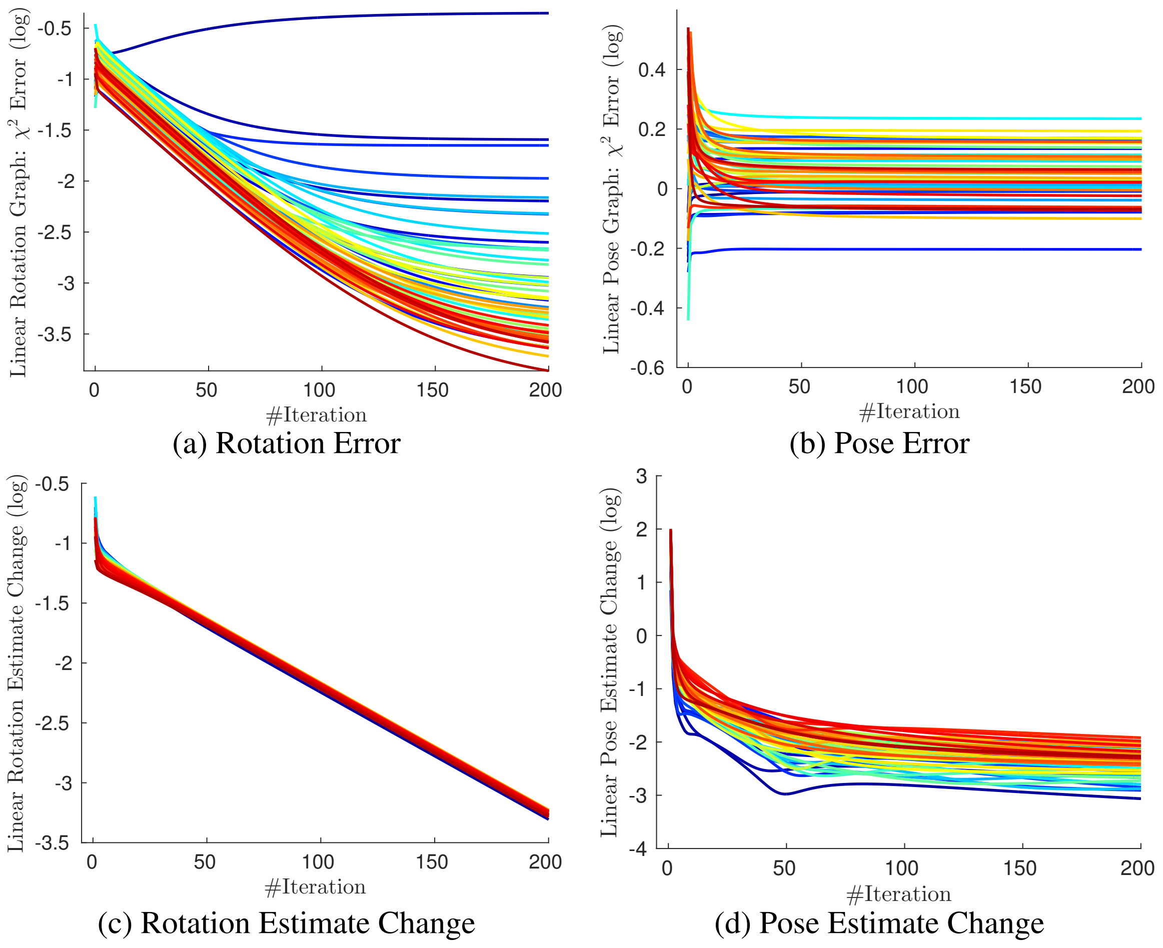


Figure 4.43: DGS : convergence statistics of rotation estimation and pose estimation for each robot (49 Robots). Robots are represented by different color lines.

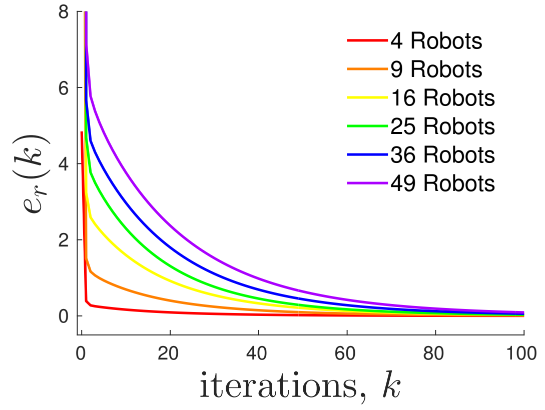
Figure 4.43a -b show the evolution of the rotation and pose error for *each* robot in the 49-robot grid: the error associated to each robot (i.e., to each subgraph corresponding to a robot trajectory) is not monotonically decreasing and the error for some robot can increase to bring down the overall error.

Figure 4.43c -d report the change in the rotation and pose estimate for individual robots. Estimate changes become negligible within few tens of iterations. As mentioned at the beginning of the section, we stop the DGS iterations when the estimate change is sufficiently small (below the thresholds η_r and η_p).

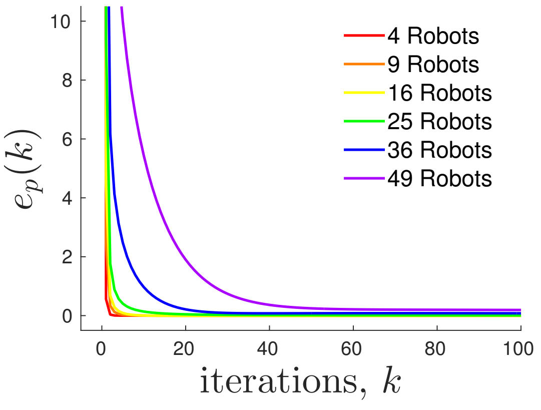
Figure 4.42 shows the estimated trajectory after 10 and 1000 iterations of the DGS algorithm for the 49-robot grid. The odometric estimate (Figure 4.42a) is shown for visualization purposes, while it is not used in our algorithm. We can see that the estimate after 10 iterations is already visually close to the estimate after 1000 iterations. The DGS algorithm has an anytime flavor: the trajectory estimates are already accurate after few iterations and asymptotically converge to the centralized estimate.

Scalability in the number of robots. Figure 4.44 shows the average rotation and pose errors for all the simulated datasets (4, 9, 16, 25, 36 and 49 robots). In all cases the errors quickly converge to zero. For large number or robots the convergence rate becomes slightly slower, while in all cases the errors is negligible in few tens of iterations. While so far we considered the errors for each subproblem ($e_r(k)$ and $e_p(k)$), we now investigate the overall accuracy of the DGS algorithm to solve our original problem 4.6.

We compare the proposed approach against the centralized two-stage approach of [Car+15b] and against a standard (centralized) Gauss-Newton (GN) method, available in gtsam ([Del12b]). We use the cost attained in problem 4.6 by each technique as accuracy metric (the lower the better).



(a) Rotation Error



(b) Pose Error

Figure 4.44: DGS : convergence for scenarios with increasing number of robots.

Table 4.4 reports the number of iterations and the cost attained in problem 4.6, for the compared techniques. The number of iterations is the sum of the number of iterations required to solve Equations 4.10 and 4.15. The cost of the DGS approach is given for two choices of the thresholds η_r and η_p.

Table 4.4: Number of iterations and cost attained in problem 4.6 by the DGS algorithm (for two choices of the stopping conditions), versus a centralized two-stage approach and a GN method. Results are shown for scenarios with *increasing number of robots*. Measurement noise is generated from a Gaussian distribution with standard deviation $\sigma_R = 5°$ for the rotations and $\sigma_t = 0.2$m for the translations. Results are averaged over 10 Monte Carlo runs.

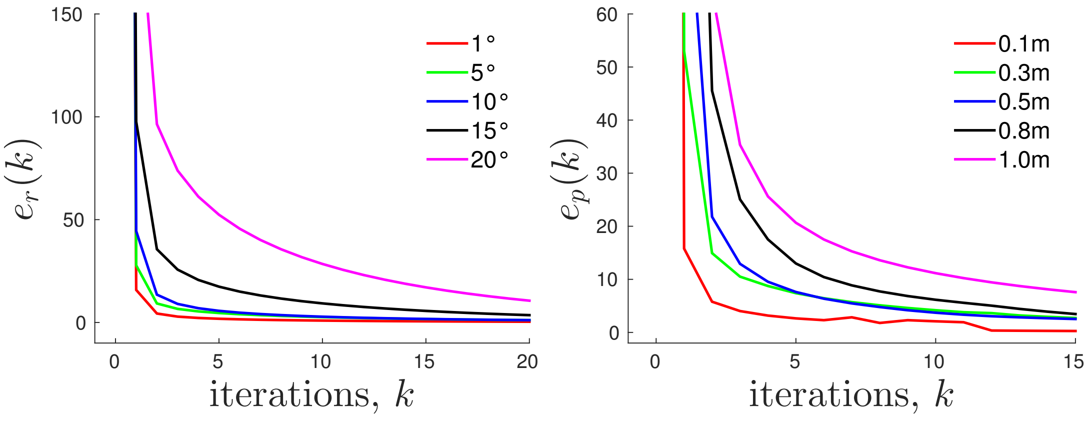
#Robots	Distributed Gauss-Seidel						Centralized	
	$\eta_r = \eta_p = 10^{-1}$			$\eta_r = \eta_p = 10^{-2}$			Two-Stage Cost	GN Cost
	#Iter	Cost	% Diff. w/ GN	#Iter	Cost	% Diff. w/ GN		
4	10	1.9	0	65	1.9	0	1.9	1.9
9	14	5.3	1.9	90	5.2	0	5.2	5.2
16	16	8.9	2.2	163	8.8	1.14	8.8	8.7
25	17	16.2	1.88	147	16.0	0.62	16.0	15.9
36	28	22.9	1.77	155	22.7	0.88	22.6	22.5
49	26	35.1	8.0	337	32.9	1.23	32.7	32.5

As already reported in [Car+15b], the last two columns of the table confirm that the centralized two-stage approach is practically as accurate as a GN method. When using a strict stopping condition ($\eta_r = \eta_p = 10^{-2}$), the DGS approach produces the same error as the centralized counterpart (difference smaller than 1%). Relaxing the stopping conditions to $\eta_r = \eta_p = 10^{-1}$ implies a consistent reduction in the number of iterations, at a small loss in accuracy (cost increase is only significant for the scenario with 49 robots).

In summary, the DGS algorithm (with $\eta_r = \eta_p = 10^{-1}$) ensures accurate estimation within few iterations, even for large teams.

Sensitivity to measurement noise. Figure 4.45 shows the average rotation and pose errors for increasing levels of noise in the scenario with 49 robots. Also in this case, while larger noise seems to imply longer convergence tails, the error becomes sufficiently small after few tens of iterations.

Table 4.5 evaluates the performance of the DGS method in solving problem 4.6 for increasing levels of noise, comparing it against the centralized two-stage approach of [Car+15b] and the Gauss-Newton method.



(a) Rotation Noise (b) Translation Noise

Figure 4.45: DGS : convergence for increasing levels of noise (scenario with 49 Robots). (a) Average rotation error for $\sigma_R = \{1, 5, 10, 15, 20\}^\circ$. (b) Average pose error for $\sigma_t = \{0.1, 0.3, 0.5, 0.8, 1.0\}$m.

The DGS approach is able to replicate the accuracy of the centralized two-stage approach, regardless the noise level, while the choice of thresholds $\eta_r = \eta_p = 10^{-1}$ ensures accurate estimation within few tens of iterations.

Table 4.5: Number of iterations and cost attained in problem 4.6 by the DGS algorithm (for two choices of the stopping conditions), versus a centralized two-stage approach and a GN method. Results are shown for *increasing measurement noise* in a scenario with 49 robots

Measurement noise		Distributed Gauss-Seidel						Centralized	
		$\eta_r = \eta_p = 10^{-1}$			$\eta_r = \eta_p = 10^{-2}$			Two-Stage	GN
$\sigma_r(^\circ)$	$\sigma_t(\text{m})$	#Iter	Cost	% Diff. w/ GN	#Iter	Cost	% Diff. w/ GN	Cost	Cost
1	0.05	8.5	2.1	16.0	51.0	1.8	0	1.8	1.8
5	0.1	21.8	14.8	6.47	197.8	14.0	0.71	14.0	13.9
10	0.2	35.6	58.4	4.28	277.7	56.6	1.07	56.6	56.0
15	0.3	39.8	130.5	3.57	236.8	128.4	1.90	129.3	126.0

Scalability in the number of separators

In order to evaluate the impact of the number of separators on convergence, we simulated two robots moving along parallel tracks for 10 time stamps. The number of communication links were varied from 1 (single communication) to 10 (communication at every time), hence the number of separators (for each robot) ranges from 1 to 10. Figure 4.46a shows the number of iterations required by the DGS algorithm ($\eta_r = \eta_p = 10^{-1}$), for increasing number of communication links.

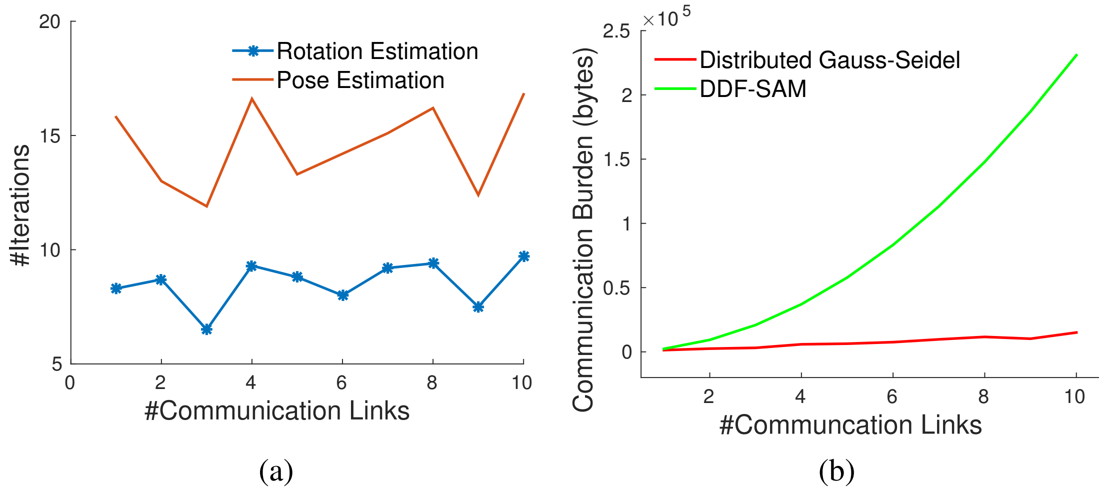


Figure 4.46: DGS vs DDF-SAM: (a) average number of iterations versus number of separators for the DGS algorithm. (b) communication burden (bytes of exchanged information) for DGS and DDF-SAM, for increasing number of separators.

The number of iterations is fairly insensitive to the number of communication links. Figure 4.46b compares the information exchanged in the DGS algorithm against a state-of-the-art algorithm, DDF-SAM ([CPD10]). In DDF-SAM, each robot sends $K_{\mathsf{GN}} \left[sB_p + (sB_p)^2 \right]$ bytes, where K_{GN} is the number of iterations required by a GN method applied to problem (4.6) (we consider the best case $K_{\mathsf{GN}} = 1$), s is the number of separators and B_p is the size of a pose in bytes. The quadratic term derives from the exchange of a dense marginal (mean and covariance).

In the DGS algorithm, each robots sends $K^r_{\text{DGS}}(sB_r) + K^p_{\text{DGS}}(sB_p)$ bytes, where K^r_{DGS} and K^p_{DGS} are the number of iterations required by the DGS algorithm to solve the linear systems (4.10) and (4.15), respectively, and B_r is the size of a rotation (in bytes).

We assume $B_r = 9$ doubles (72 bytes)[‡] and $B_p = 6$ doubles (48 bytes). The number of iterations K^r_{DGS} and K^p_{DGS} are the one plotted in Figure 4.46a . From Figure 4.46b we see that the communication burden of DDF-SAM quickly becomes unsustainable, while the linear increase in communication of the DGS algorithm implies large communication saving.

Realistic simulations in Gazebo

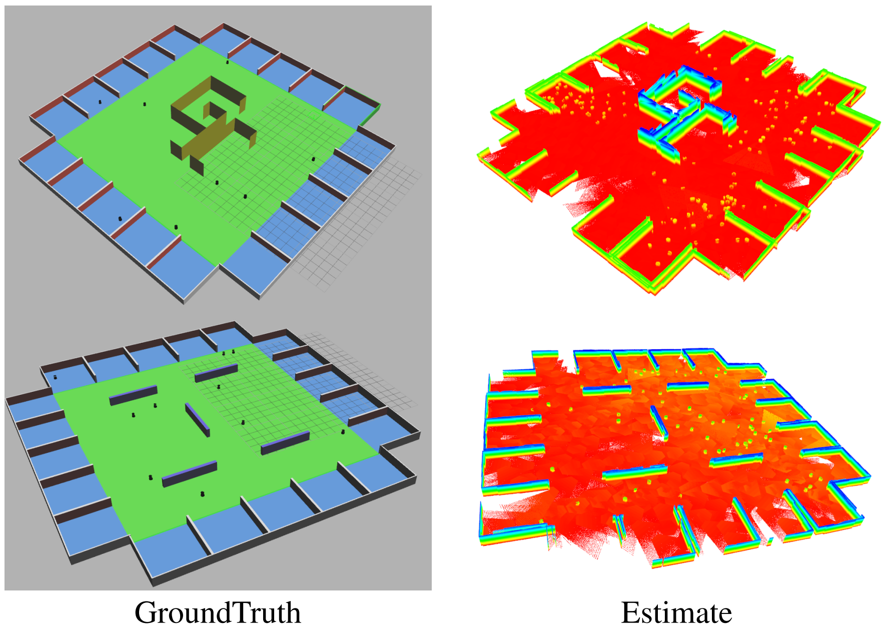
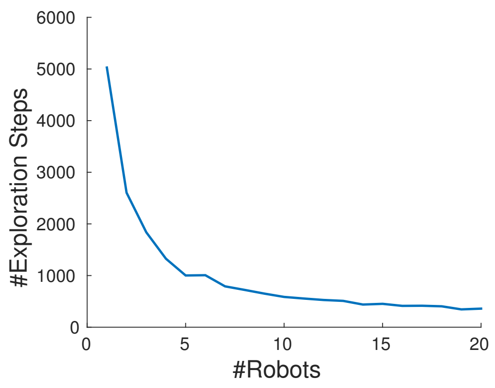


Figure 4.47: Gazebo tests: ground truth environments and aggregated point clouds corresponding to the DGS estimate.

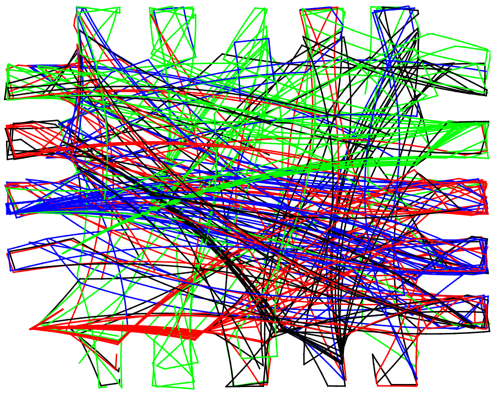
We tested our DGS -based approach in two scenarios in Gazebo simulations as shown in Figure 4.47. The robots start at fixed locations and explore the environment by moving according to a random walk. Each robot is equipped with a 3D laser range finder, which is used to intra-robot

[‡]In the linear system (4.10) we relax the orthogonality constraints hence we cannot parametrize the rotations with a minimal 3-parameter representation.

and inter-robot measurements via scan matching.



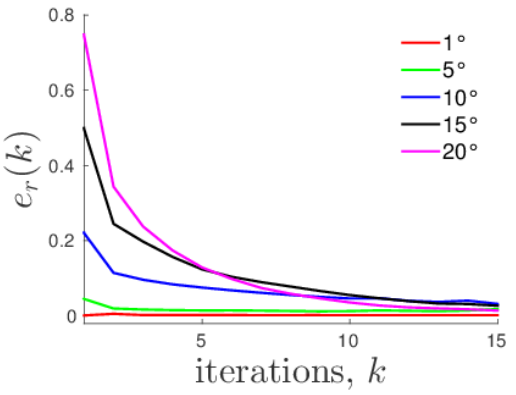
(a) Exploration steps



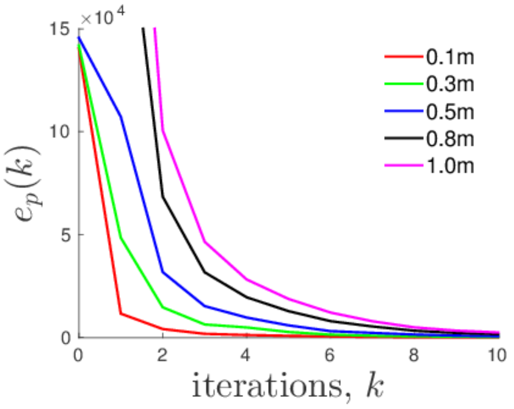
(b) Monte Carlo Runs

Figure 4.48: (a) Number of exploration steps required to explore a fixed-sized grid with increasing number of robots. (b) Samples of robot trajectories from our Gazebo-based Monte Carlo experiments.

In both scenarios, two robots communicate only when they are within close proximity of each other (0.5m in our tests). Results are average over 100 Monte-Carlo runs.



(a) Rotation Noise



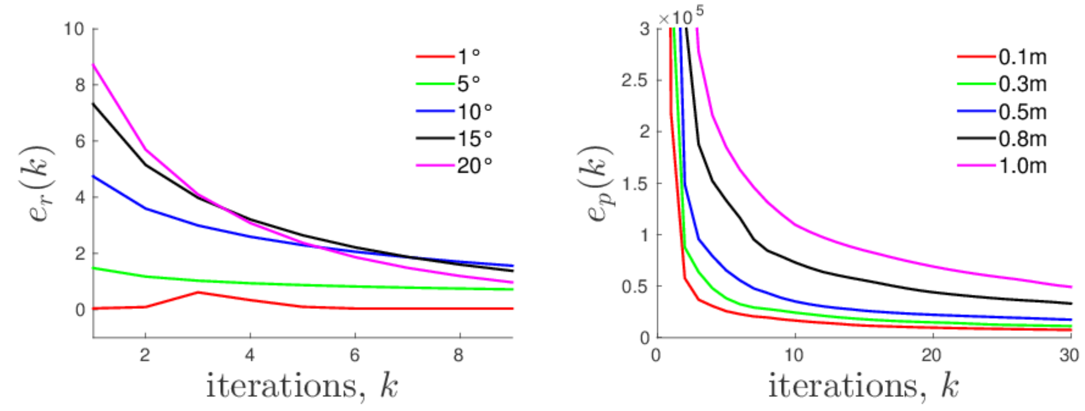
(b) Translation Noise

Figure 4.49: Convergence for increasing levels of noise (scenario with 2 Robots in Gazebo). (a) Average rotation estimation error for $\sigma_R = \{1, 5, 10, 15, 20\}^\circ$. (b) Average pose estimation error for $\sigma_t = \{0.1, 0.3, 0.5, 0.8, 1.0\}$m.

Figure 4.47 shows the aggregated point cloud corresponding to the DGS trajectory estimate, for one of the runs. The point cloud closely resembles the ground truth environment shown in the same figure.

Figure 4.48a shows that number of steps required to explore the whole environment quickly decreases with increasing number of robots.

This intuitive observation motivates our interest towards mapping techniques that can scale to large teams of robots. Figure 4.48b reports trajectory samples for different robots in our Monte Carlo analysis. Figures 4.49 and 4.50 show the average errors $e_r(k)$ and $e_p(k)$ for increasing levels of noise and for the scenario with 2 and 4 robots respectively.



(a) Rotation Noise (b) Translation Noise

Figure 4.50: Convergence for increasing levels of noise (scenario with 4 Robots in Gazebo). (a) Average rotation estimation error for $\sigma_R = \{1, 5, 10, 15, 20\}°$. (b) Average pose estimation error for $\sigma_t = \{0.1, 0.3, 0.5, 0.8, 1.0\}$m.

4.4.9 Simulation Results: Multi-Robot Object based SLAM

In this section we characterize the performance of the DGS algorithms associated with our object-based model described in Section 4.4.5. We test the resulting multi-robot object-based SLAM approach in terms of scalability in the number of robots and sensitivity to noise.

Simulation setup and performance metrics

We consider two scenarios, the `25 Chairs` and the `House`, which we simulated in Gazebo. In the `25 Chairs` scenario, we placed 25 chairs as objects on a grid, with each chair placed at a random angle. In the `House` scenario, we placed furniture as objects in order to simulate a living room environment.

Figure 4.51 shows the two scenarios. Unless specified otherwise, we generate measurement noise from a zero-mean Gaussian distribution with standard deviation $\sigma_R = 5^\circ$ for the rotations and $\sigma_t = 0.2$m for the translations. Six robots are used by default. Results are averaged over 10 Monte Carlo runs. We use the *absolute translation error* (ATE*) and *absolute rotation error* (ARE*) of the robot and landmark poses with respect to the centralized estimate as error metric.

These two metrics are formally defined below.

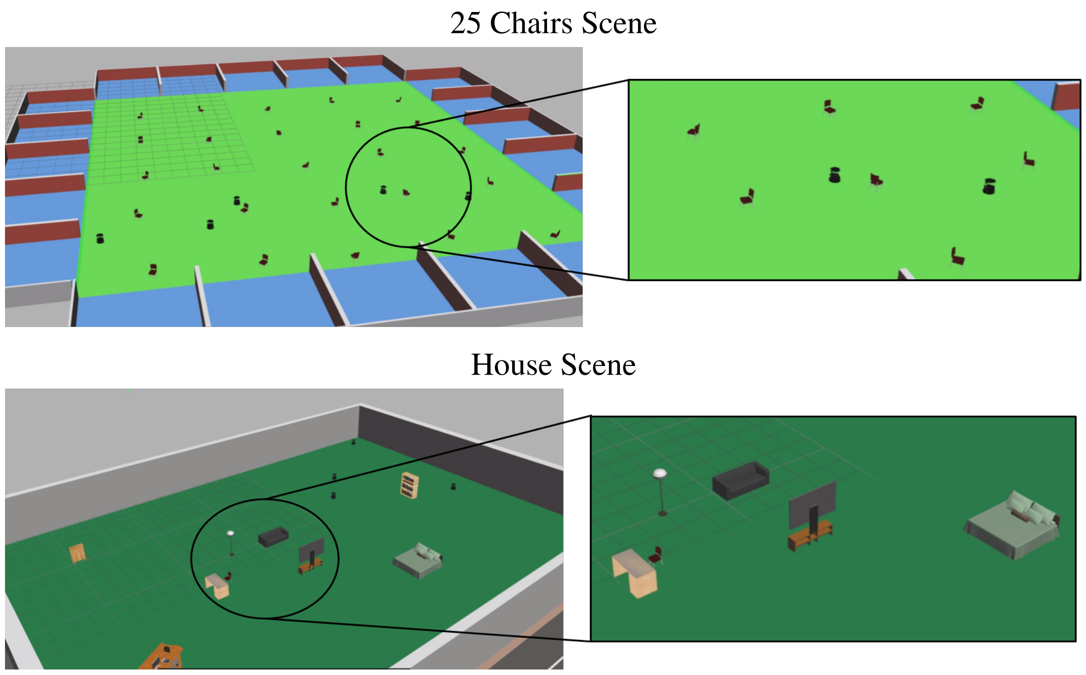


Figure 4.51: Multi robot object-based SLAM in Gazebo: the `25 Chairs` and `House` scenarios simulated in Gazebo.

Absolute Translation Error (ATE*)

Similar to the formulation by Sturm et al. [Stu+12], the average translation error measures the absolute distance between the trajectory and object poses estimated by our approach versus the centralized GN method. The ATE* is defined as:

$$ATE^* = \left(\frac{1}{\sum_{\alpha \in \Omega} n_\alpha} \sum_{\alpha \in \Omega} \sum_{i=1}^{n_\alpha} \| \boldsymbol{t}_{\alpha_i} - \boldsymbol{t}^*_{\alpha_i} \|^2 \right)^{\frac{1}{2}} \tag{4.33}$$

where $\boldsymbol{t}_{\alpha_i}$ is the position estimate for robot α at time i, $\boldsymbol{t}^*_{\alpha_i}$ is the corresponding estimate from GN, and n_α is the number of poses in the trajectory of α. A similar definition holds for the object positions where X^i is the i^{th} trajectory pose or object location estimated using our approach and X_c^i is the i^{th} corresponding pose estimated using the centralized Gauss-Newton method, $trans(x,y)$ measures the euclidean distance between x and y.

Absolute Rotation Error (ARE*)

The average rotation error is computed by evaluating the angular mismatch between the (trajectory and objects) rotations produced by the proposed approach versus a centralized GN method:

$$ARE^* = \left(\frac{1}{\sum_{\alpha \in \Omega} n_\alpha} \sum_{\alpha \in \Omega} \sum_{i=1}^{n_\alpha} \| \mathrm{Log}\left((\boldsymbol{R}^*_{\alpha_i})^\top \boldsymbol{R}_{\alpha_i} \right) \|^2 \right)^{\frac{1}{2}} \tag{4.34}$$

where $\boldsymbol{R}_{\alpha_i}$ is the rotation estimate for robot α at time i, $\boldsymbol{R}^*_{\alpha_i}$ is the corresponding estimate from GN. A similar definition holds for the object rotations.

Accuracy in the number of robots

Figure 4.52 compares the object locations and trajectories estimated using multi-robot mapping and centralized mapping for the two scenarios. Videos showing the map building for the two scenarios are available at:

`https://youtu.be/nXJamypPvVY` and `https://youtu.be/nYm2sSHuGjo`.

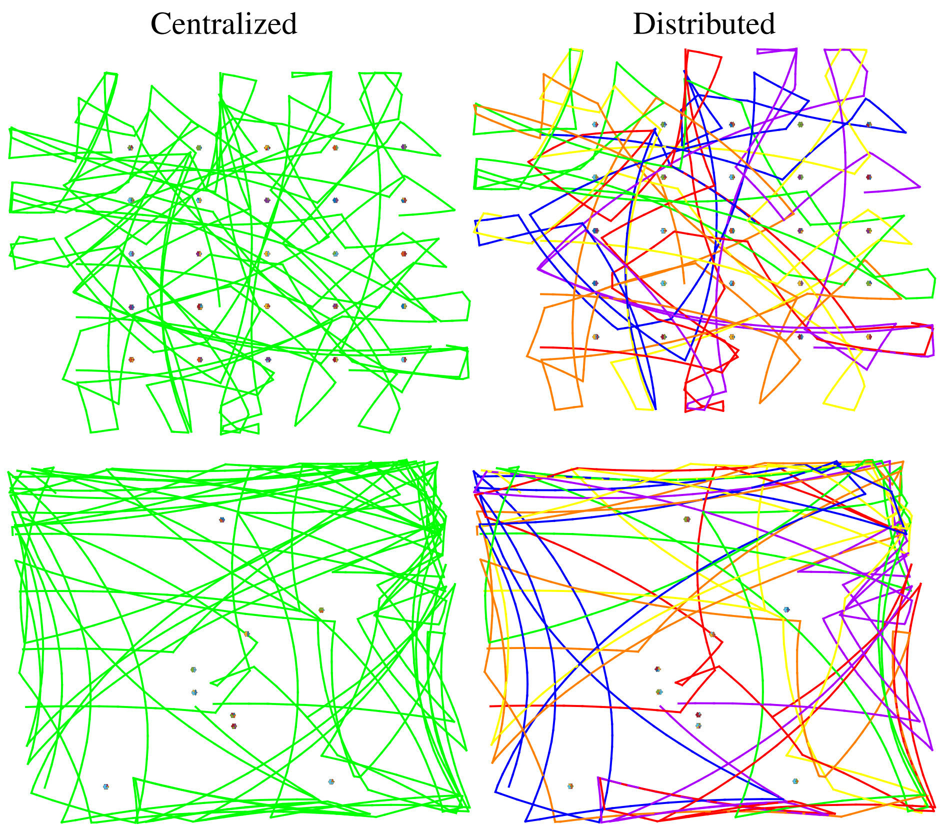


Figure 4.52: Trajectories of the six robots and object locations (shows as red dots) estimated using the centralized GN method and the proposed DGS method for the `25 Chairs` (top) and `House` scenarios (bottom).

Table 4.6 reports the number of iterations and our accuracy metrics (cost, ATE*, ARE*) for increasing number of robots. The table confirms that the distributed approach is nearly as accurate as the centralized Gauss-Newton method and the number of iterations does not increase with increasing number of robots, making our approach scalable to large teams. Usually, few tens of iterations suffice to reach an accurate estimate. Note that even when the cost of the DGS method is slightly higher than GN, the actual mismatch in the pose estimates is negligible (in the order of millimeters for positions and less than a degree for rotations). Therefore, our approach is scalable in the number of robots.

Table 4.6: Number of iterations, cost, ATE* and ARE* of our approach compared to the centralized Gauss-Newton method for *increasing number of robots*. ATE* and ARE* are measured using $\eta = 10^{-1}$ as stopping condition. Measurement noise is generated from a Gaussian distribution with standard deviation $\sigma_R = 5^{\circ}$ for the rotations and $\sigma_t = 0.2$m for the translations. Results are averaged over 10 Monte Carlo runs.

#Robots	Distributed Gauss-Seidel				Centralized	ATE* (m)		ARE* (deg)	
	$\eta = 10^{-1}$		$\eta = 10^{-2}$		GN	Poses	Lmrks.	Poses	Lmrks.
	#Iter	Cost	#Iter	Cost	Cost				
2	5.0	56.1	9.0	56.0	54.7	1.5e-03	8.0e-04	2.1e-01	2.8e-01
4	5.0	118.0	8.0	117.9	113.8	9.7e-04	7.5e-04	2.0e-01	2.8e-01
6	5.0	166.6	7.0	166.5	160.9	3.1e-03	2.1e-03	3.3e-01	4.0e-01

Sensitivity to measurement noise. We further test the accuracy of our approach by evaluating the number of iterations, the cost, the ATE* and ARE* for increasing levels of noise in `25 Chairs` scenario with 6 robots. Table 4.7 shows that our approach is able to replicate the accuracy of the centralized Gauss-Newton method, regardless of the noise level.

Table 4.7: Number of iterations, cost, ATE* and ARE* of our approach compared to a centralized Gauss-Newton method for *increasing measurement noise* in `25 Chairs` scenario with 6 robots. ATE* and ARE* are measured using $\eta = 10^{-1}$ as stopping condition.

Measurement		Distributed Gauss-Seidel				Centralized	ATE* (m)		ARE* (deg)	
noise		$\eta = 10^{-1}$		$\eta = 10^{-2}$		GN	Poses	Lmrks.	Poses	Lmrks.
$\sigma_r(^{\circ})$	$\sigma_t(\mathrm{m})$	#Iter	Cost	#Iter	Cost	Cost				
1	0.05	5.0	12.7	6.0	12.7	12.5	1.8e-04	1.3e-04	7.5e-02	9.0e-02
5	0.1	5.0	166.6	7.0	166.5	160.9	3.1e-03	2.1e-03	3.3e-01	4.0e-01
10	0.2	5.0	666.2	8.0	665.9	643.4	1.3e-02	8.8e-03	6.7e-01	8.1e-01
15	0.3	6.0	1498.3	10.0	1497.8	1447.2	3.0e-02	2.1e-02	1.0e+00	1.2e+00

4.4.10 Field Experiments: Multi-Robot Pose Graph Optimization

We tested the DGS approach on field data collected by two to four Jackal robots (Figure 4.53), moving in a military training facility. Each robot collects 3D scans using Velodyne 32E, and uses IMU and wheel odometry to measure its ego-motion [§]. 3D scans are used to compute inter-robot measurements (via ICP [TRC14a]) during rendezvous. We evaluated our approach in multiple buildings in the military training facility.

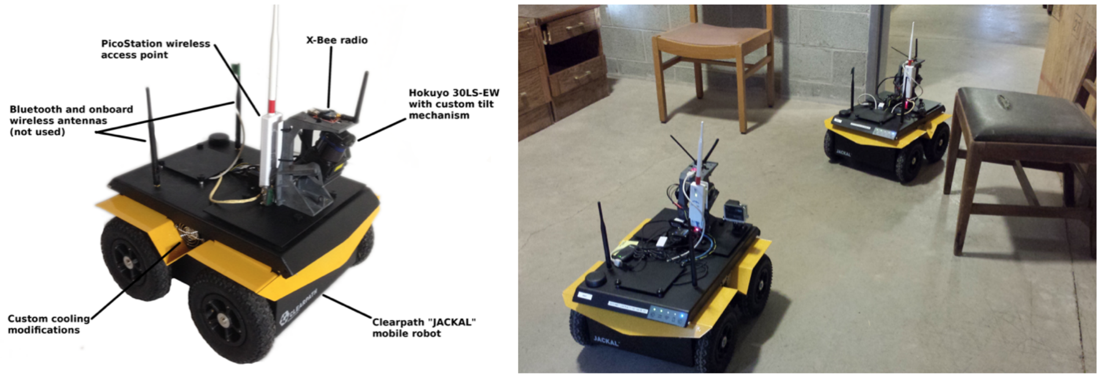

PicoStation wireless access point
X-Bee radio
Bluetooth and onboard wireless antennas (not used)
Hokuyo 30LS-EW with custom tilt mechanism
Custom cooling modifications
Clearpath "JACKAL" mobile robot

Figure 4.53: (Left) Clearpath Jackal robot used for the field tests: platform and sensor layout; (right) snapshot of the test facility with the two Jackal robots.

Figures 4.54, 4.55, and 4.56 show the aggregated 3D point clouds (left), the estimates trajectories (center), and the aggregated occupancy grid map (right) over multiple runs.

The central part of the figures compares the DGS estimate against the DDF-SAM estimate and the corresponding centralized estimate. Note that the test scenarios cover a broad set of operating conditions. For instance Figure 4.55 corresponds to experiments with 4 robots moving in indoor environment, while Figure 4.54 corresponds to tests performed in a mixed indoor-outdoor scenario (with robots moving on gravel when outdoor, Figure 4.57). The four tests of Figure 4.56 correspond to early results with 2 robots for which we do not have a comparison against DDF-SAM.

[§]`https://github.com/CognitiveRobotics/omnimapper`

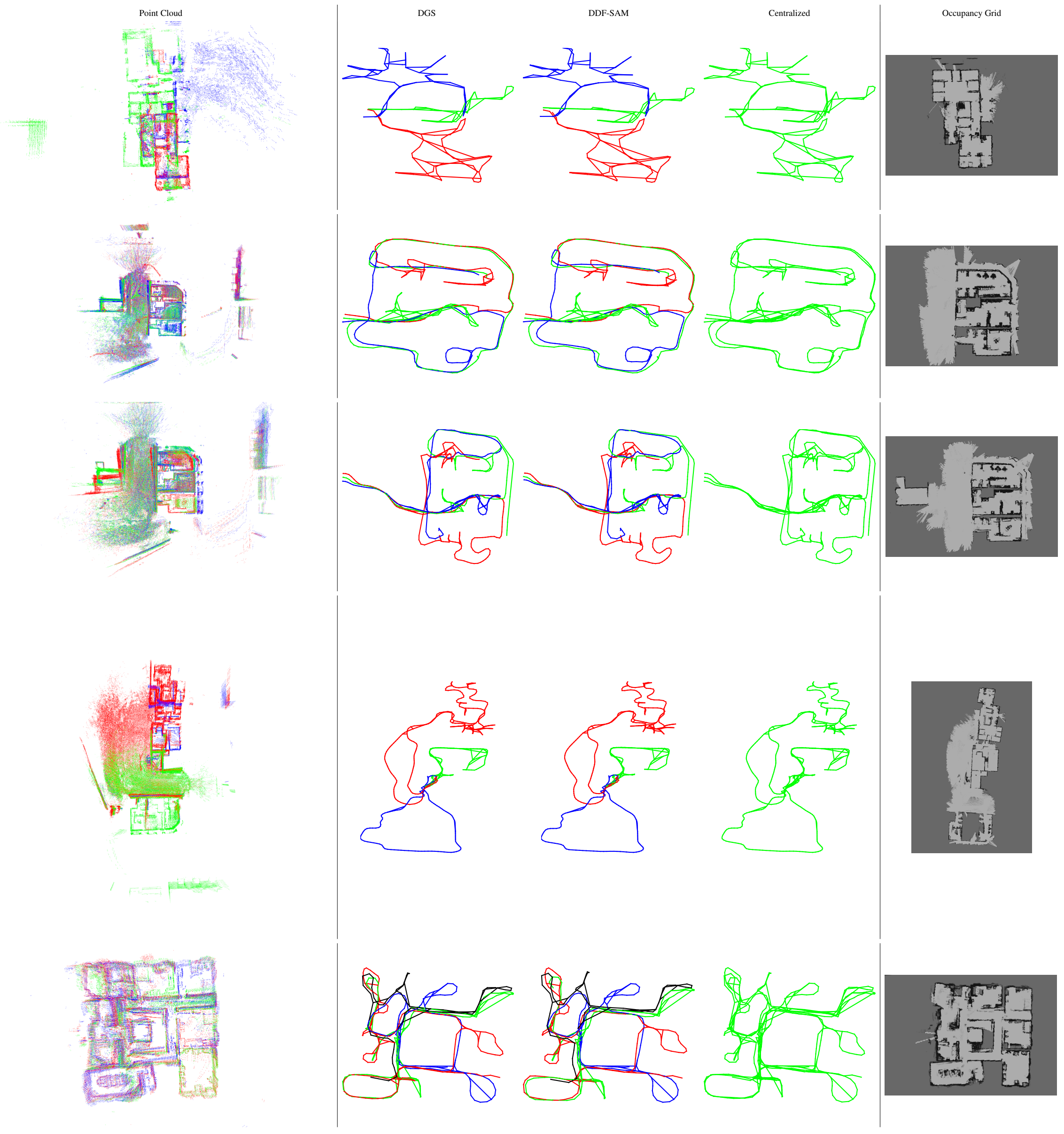


Figure 4.54: Mixed indoor-outdoor scenarios: (Left) aggregated point cloud obtained from the DGS trajectory estimate. (Center) estimated trajectories for DGS , GN and DDF-SAM (robots shown in red, blue, green and black for the distributed techniques). (Right) overall occupancy grid map obtained from the DGS trajectory estimate.

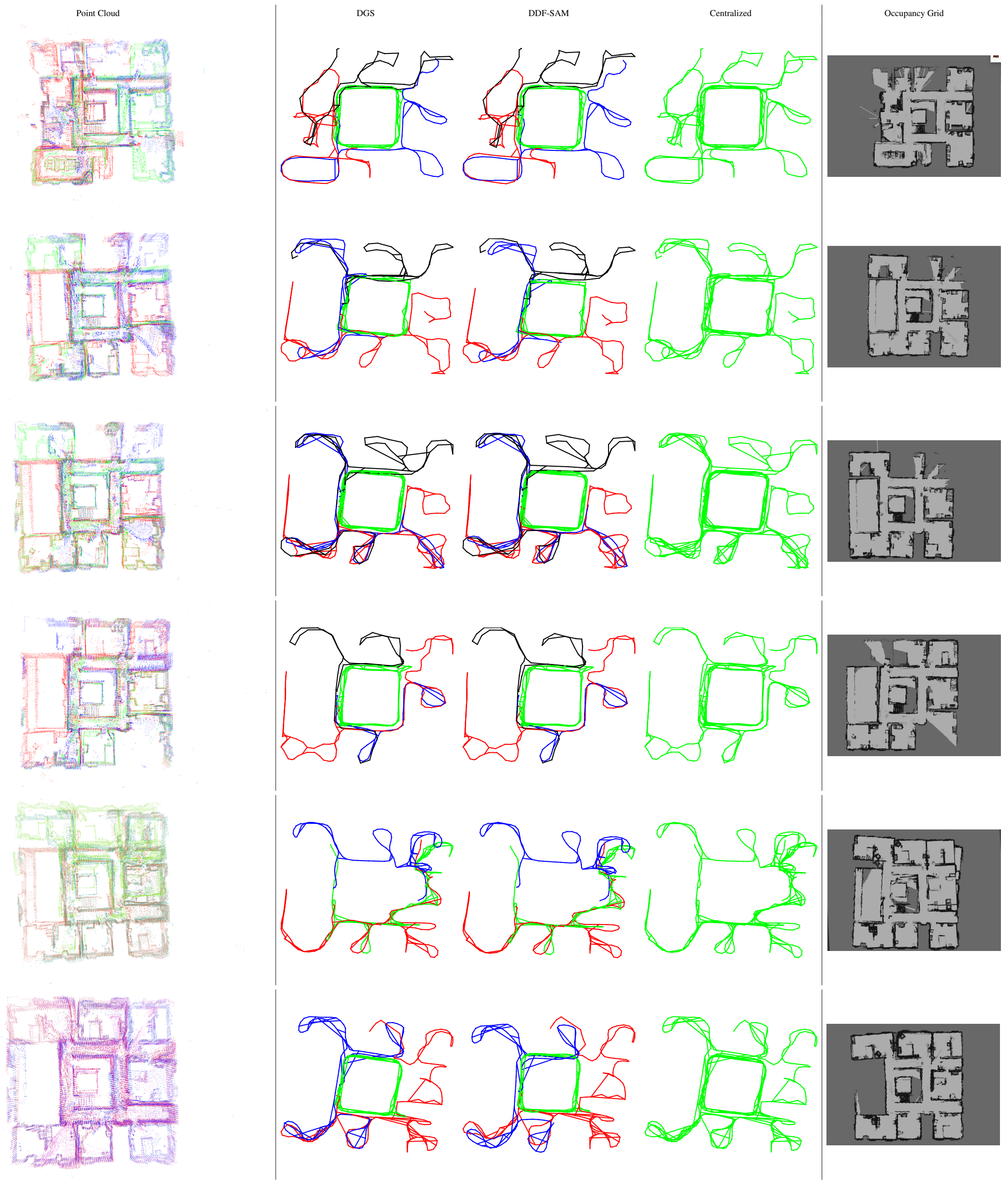


Figure 4.55: Indoor scenarios: (Left) aggregated point cloud obtained from the DGS trajectory estimate. (Center) estimated trajectories for DGS , GN and DDF-SAM (robots shown in red, blue, green and black for the distributed techniques). (Right) overall occupancy grid map obtained from the DGS trajectory estimate of the scenario obtained from the DGS trajectory estimate.

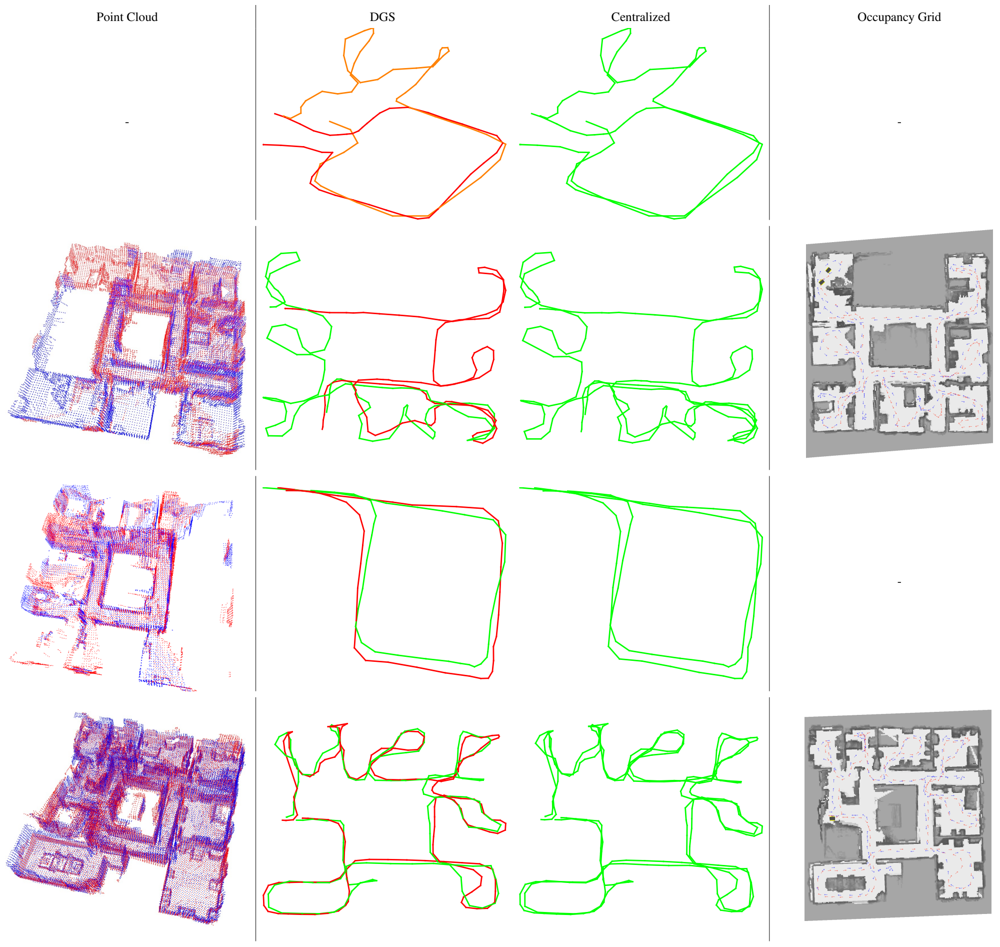


Figure 4.56: Early tests with 2 robots: (Left) aggregated point cloud obtained from the DGS trajectory estimate. (Center) estimated trajectories for DGS and GN. (Right) overall occupancy grid map obtained from the DGS trajectory estimate.

Quantitative results are given in Table 4.8, which reports the cost attained by the DGS algorithm as compared to the centralized GN cost and DDF-SAM. Number of iterations, ATE* and ARE* are also shown.

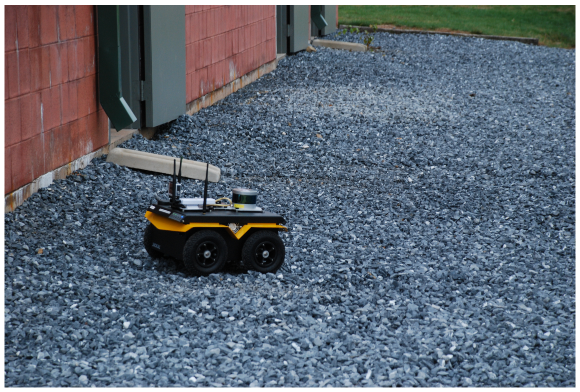
Each line of the table shows statistics for each of the 15 field tests in the military training facility. The first four rows (tests 0 to 3) correspond to tests performed in a mixed indoor-outdoor scenario (Figure 4.54).

Table 4.8: Performance of DGS on field data as compared to the centralized GN method and DDF-SAM. Number of iterations, ATE* and ARE* with respect to centralized Gauss-Newton estimate are also shown.

#Test	#vertices	#edges	#links	Distributed Gauss-Seidel								Centralized		DDF-SAM
				$\eta_r = \eta_p = 10^{-1}$				$\eta_r = \eta_p = 10^{-2}$				Two-Stage	GN	
				#Iter	Cost	ATE*	ARE*	#Iter	Cost	ATE*	ARE*	Cost	Cost	Cost
0	194	253	16	12	1.42	0.21	1.63	197	1.40	0.07	0.67	1.40	1.40	4.86
1	511	804	134	10	0.95	1.22	6.64	431	0.91	1.18	6.37	0.89	0.89	6.88
2	547	890	155	21	1.99	1.03	4.74	426	1.95	1.08	4.79	1.93	1.93	12.54
3	657	798	47	176	0.32	0.68	2.40	446	0.32	0.69	2.06	0.32	0.32	2.39
4	510	915	179	23	10.89	1.10	6.69	782	10.57	0.71	4.53	10.51	10.50	37.94
5	418	782	151	13	3.02	0.51	5.75	475	2.92	0.39	4.32	2.91	2.90	18.31
6	439	720	108	26	9.28	0.68	5.08	704	9.12	0.31	2.39	9.10	9.07	72.76
7	582	1152	228	10	3.91	0.31	3.40	579	3.78	0.26	2.43	3.78	3.78	16.38
8	404	824	183	11	1.92	0.13	1.78	410	1.89	0.12	1.25	1.89	1.89	6.82
9	496	732	86	41	4.30	0.54	4.20	504	4.29	0.45	3.04	4.28	4.27	21.53
10	525	923	147	15	5.56	0.39	3.93	577	5.43	0.23	2.04	5.43	5.40	19.59
11	103	107	3	71	0.85	1.58	14.44	328	0.84	0.27	2.18	0.84	0.84	-
12	227	325	50	16	0.79	1.11	10.44	511	0.71	0.80	7.02	0.68	0.68	-
13	77	127	26	10	0.33	0.34	2.23	78	0.26	0.21	1.25	0.26	0.26	-
14	322	490	85	28	1.42	0.83	5.05	606	1.07	0.47	2.10	1.04	1.04	-

The next seven rows (tests 4 to 10) correspond to tests performed with 4 robots in an indoor environment. The last four rows (tests 11 to 14) correspond to early results with 2 robots. Higher ATE* and ARE* in the first few rows is due to the fact that the robots move on gravel in outdoors which introduces larger odometric errors. Consistently with what we observed in the previous sections, larger measurement errors may induce the DGS algorithm to perform more iterations to reach consensus (e.g., test 3).

The columns “#vertices” and “#edges” describe the size of the overall factor graph (including all robots), while the column “#links” reports the total number of rendezvous events. In all the tests DDF-SAM performed worse than DGS which is shown by higher cost attained by DDF-SAM as compared to DGS .


(a) Clearpath Jackal robot moving on gravel

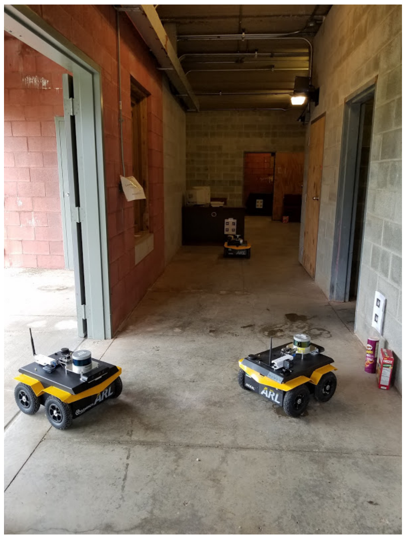

(b) Multi-robot team exploring and mapping inside buildings.

Figure 4.57: Experiments performed at the MOUT.

This is because DDF-SAM requires good linearlization points to generate condensed graphs and instead bad linearization points during communication can introduce linearlization errors resulting in higher cost. Figure 4.58 shows the corresponding histogram visualization comparing the cost attained by the DGS algorithm and centralized Two-Stage and GN algorithm.

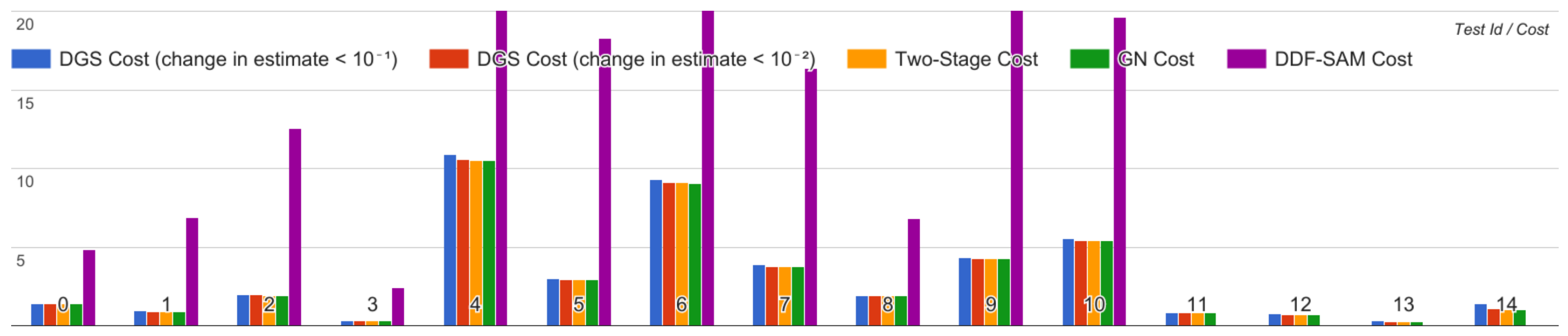


Figure 4.58: Histogram visualization comparing the cost attained by DGS algorithm on field data as compared to the centralized Two-Stage, GN and DDF-SAM method. It shows that all the proposed approaches are close to GN, while DDF-SAM has worse performance. The length of y-axis (cost) is limited to 20 for visualization purposes. Additional quantitative results are given in Table 4.8.

4.4.11 Field Experiments: Multi-Robot Object-based SLAM

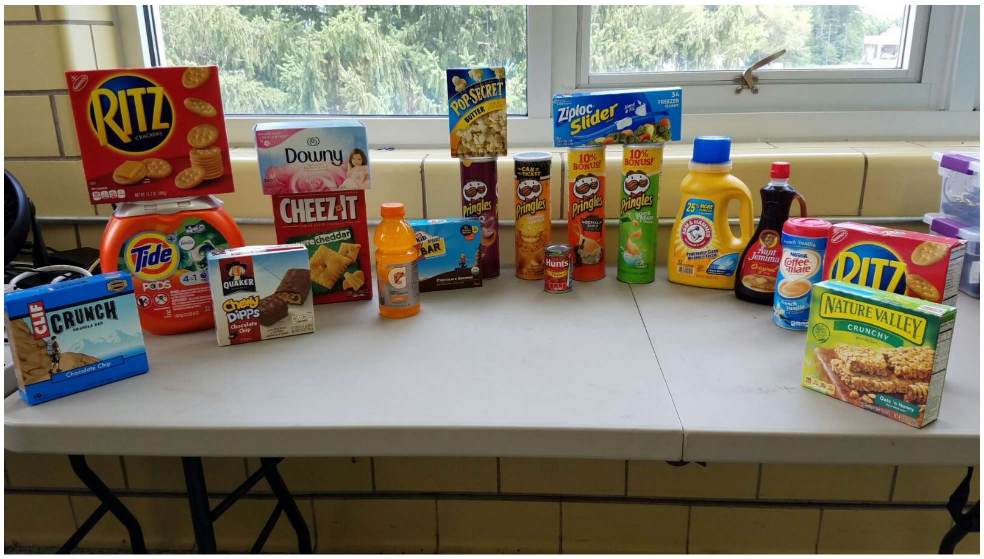


Figure 4.59: Objects from the BigBird dataset used in the field experiments of Section 4.4.11.

We test the combination of the DGS method and our object-based model on field data collected by two Jackal robots (Figure 4.60) moving in a military training facility. We scattered the environment with a set of objects (Figure 4.59) from the BigBird dataset ([Sin+14]). Each robot is equipped with an Asus Xtion RGB-D sensor and uses wheel odometry to measure its ego-motion. We use the RGB-D sensor for object detection and object pose estimation.

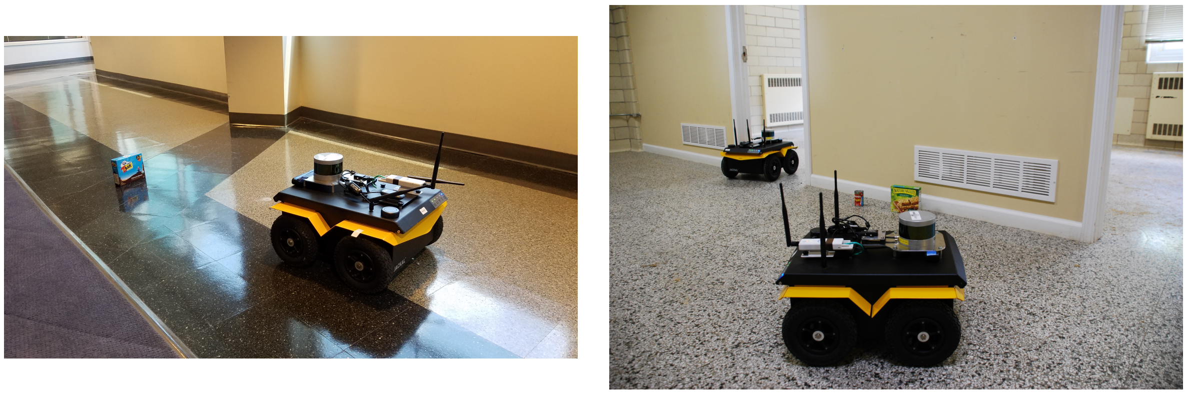


Figure 4.60: Snapshot of the test facility, the two Clearpath Jackal robots, and the objects used for object-based SLAM for the tests of Section 4.4.11.

We evaluated our approach in two different scenarios, the `stadium` and the `house`. We did two runs inside the `stadium` (`stadium-1` & `stadium-2`) and one run in the `house` with objects randomly spread along the robot trajectories. The `stadium` datasets were collected in an indoor basketball stadium with the robot trajectories bounded in a roughly rectangular area. The `house` dataset was collected in the living room and kitchen area of a house.

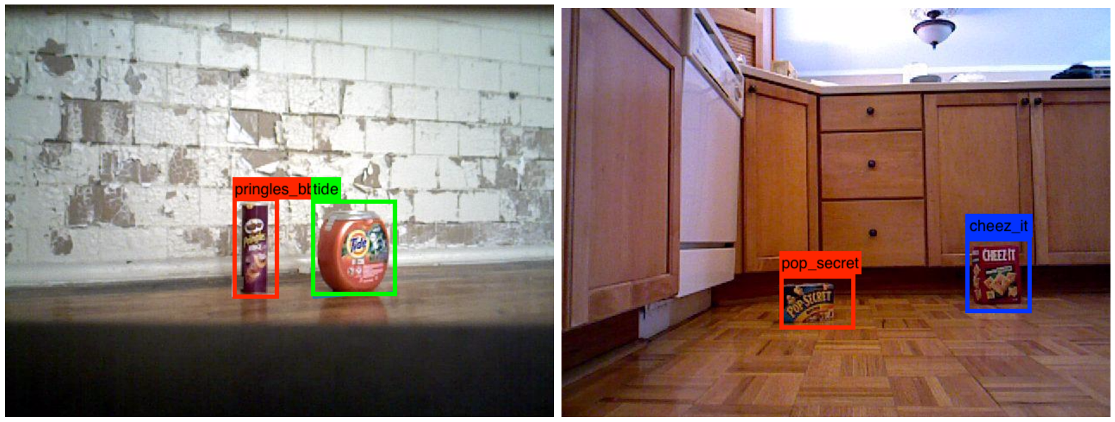


Figure 4.61: Snapshot of the YOLO object detection in two difference scenes: (left) `stadium` dataset, (right) `house` dataset.

Object detection. We used 12 objects from the BigBird dataset in all three runs. The two-stage process of object detection (semantic verification) followed by pose estimation (geometric verification) ensured that we do not add false positive detections. Similarly to the standard GN method, our current distributed optimization technique (DGS) is not robust to outliers. The detection thresholds can be further relaxed when using robust pose graph optimization techniques.

In the first run (`stadium-1`), 6 objects were added to the map out of the 12 objects present in the environment. Similarly, 5 objects were detected in `stadium-2` and `house`. Figure 4.61 shows a snapshot of the bounding box of the detected object in three different scenes. Videos showing YOLO object detection results on other datasets are available at `https://youtu.be/urZiIJK2IYk` and `https://youtu.be/-F6JpVmOrc0`.

Memory Requirements. Table 4.9 compares the average memory requirement per robot to store a dense point cloud map (PCD) with respect to storing a object-based map (Obj) in our real tests.

Table 4.9: Memory and communication requirements for our object based approach (Obj) as compared to Point cloud based approach (PCD) on field data.

Scenario	Avg. Per-Robot Memory Req. (MB)		Avg. Comm. Bandwidth Req. (MB)	
	PCD	Obj	PCD	Obj
Stadium-1	1.2e+03	1.9e+00	1.9e+01	1.5e-05
Stadium-2	1.4e+03	1.9e+00	1.4e+01	1.1e-05
House	2.1e+03	1.9e+00	1.6e+01	1.3e-05

Per-robot memory requirement in the case of dense point cloud is computed as $n_f KC$ where n_f is the number of frames, K is the number of points per frame and C is the memory required to store each point. In the case of object level map, it is computed as $n_o PC$ where n_o is the number of object models and P is the average number of points in each object model. Table 4.9 shows that, as expected, the per-robot memory requirement is orders of magnitude smaller with our object-based map as compared to point-cloud-based maps.

Communication Bandwidth Requirements. Table 4.9 also compares the average communication requirements in the case of transmission of dense point clouds and object-based models. When using point clouds, the robots are required sending at least one RGB-D frame at every rendezvous to estimate their relative pose. So the average communication for dense point cloud map is computed as $n_c KC$ where n_c is the number of rendezvous, K is the number of points per frame and C is the memory required to send each point. Communication in the case of our object-based map requires sending object category and object pose; a upper bound can be computed as $n_o L$ where n_o is the number of objects and L is the memory required to store category label and pose of an object. Table 4.9 confirms that our approach provides a remarkable advantage in terms of communication burden as it requires transmitting 6 orders of magnitude less than a point-cloud-based approach.

Accuracy. Figure 4.62 shows the trajectories of the two robots in three runs and the object pose estimates. The figure compares our approach and the corresponding centralized GN estimate. Quantitative results are given in Table 4.10, which reports the cost attained by our approach, the number of iterations, ATE*, ARE* as compared to the centralized GN approach.

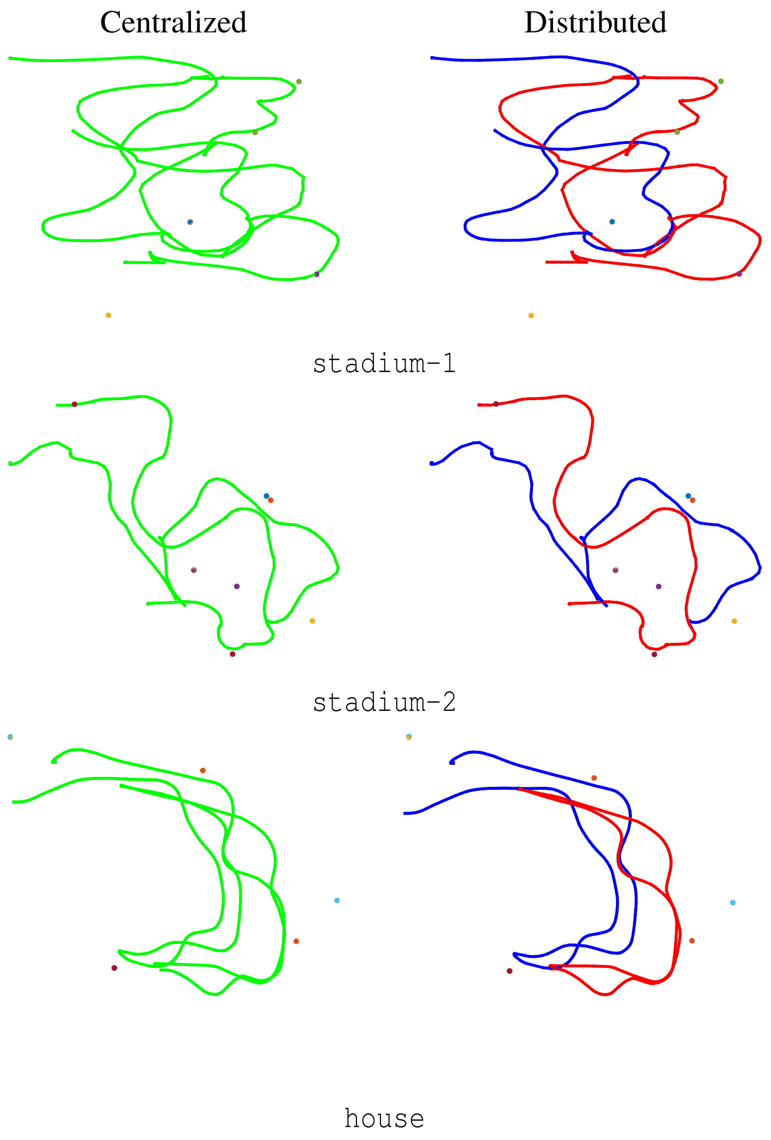


Figure 4.62: Real tests: estimated trajectories and object poses for our approach and for the centralized GN method. Trajectories of the two robots are shown as red and blue lines, while objects are shown as colored dots.

Table 4.10: Number of iterations, cost, ATE* and ARE* of our approach as compared to centralized Gauss-Newton method for Field data

Scenario	Initial	Distributed Gauss-Seidel				Centralized	ATE* (m)		ARE* (deg)	
		$\eta_r = \eta_p = 10^{-1}$		$\eta_r = \eta_p = 10^{-2}$		GN	Poses	Lmrks.	Poses	Lmrks.
	Cost	#Iter	Cost	#Iter	Cost	Cost				
Stadium-1	120.73	5.0	1.1e-09	5.0	1.1e-09	1.6e-10	1.9e-10	1.9e-10	1.4e-03	1.2e-04
Stadium-2	310.24	5.0	4.5e-12	8.0	4.4e-12	3.5e-13	2.1e-03	2.2e-03	1.2e-02	1.4e-02
House	43.59	5.0	1.1e-03	6.0	1.0e-03	8.4e-04	4.4e-02	6.2e-02	4.3e-01	4.9e-01

The table confirms that our distributed approach converges in few iterations and is practically as accurate as the centralized GN method; in particular the mismatch between the DGS and the GN estimates is in the order of millimeters for the position estimates and tenth of degrees for the rotation estimates. Note that for these indoor experiments the wheel odometry is fairly accurate, since the robot moves on wooden or tiled floor. This results in better performance for the proposed technique and in small costs in GN. The initial cost, instead, is large mostly because of the error in the initial alignment between the two robots.

4.4.12 Discussion

We investigate distributed algorithms to estimate the 3D trajectories of multiple cooperative robots from relative pose measurements. Our first contribution is the design of a 2-stage approach for distributed pose estimation and propose a number of algorithmic variants. One of these algorithms, the Distributed Gauss-Seidel (DGS) method, is shown to have excellent performance in practice: (i) its communication burden scale linearly in the number of separators and respect agents' privacy, (ii) it is robust to noise and the resulting estimates are sufficiently accurate after few communication rounds, (iii) the approach is simple to implement and scales well to large teams. We demonstrated the effectiveness of the DGS approach in extensive simulations and field tests.

Our second contribution is to extend the DGS approach to use objects as landmarks for multi-robot mapping. We show that using object-based abstractions in a multi-robot setup further reduces the memory requirement and the information exchange among teammates. We demonstrate our multi-robot object-based mapping approach in Gazebo simulations and in field tests performed in a military training facility.

We are currently extending the approach proposed in this paper in several directions. First, our current approach for object-based mapping assumes that a model of each observed objects is known in advance. However it can be challenging to store a large number of object models, and to account for intra-class variations. As a future work, we plan to extend our approach to the case where object models are not previously known (at an instance level) and instead object shapes are jointly optimized within our SLAM framework.

Second, our current approach is based on a nonlinear least squares formulation which is not robust to gross outliers. Therefore future work will focus on designing more general algorithms that are robust to spurious measurements.

Third, we plan to extend our experimental evaluation to flying robots. While we demonstrated the effectiveness of our approach in large teams of ground robots, we believe that the next grand challenge is to enable coordination and distributed mapping in swarms of agile micro aerial vehicles with limited communication and computation resources.

Acknowledgment

This chapter appears in the following publications:

- Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Henrik I. Christensen, and Frank Dellaert. Distributed mapping with privacy and communication constraints: Light weight algorithms and object-based models. *In the International Journal of Robotics Research (IJRR)*, 36(12):1286–1311, 2017.
- Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Zhen Liu, Henrik I. Christensen, and Frank Dellaert. Multi-robot object-based SLAM. *In the 14th International Symposium on Experimental Robotics (ISER)*. Tokyo, Japan, October 3-6, 2016.
- Siddharth Choudhary, Luca Carlone, Carlos Nieto-Granda, John Rogers, Henrik I. Christensen, and Frank Dellaert. Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed gauss-seidel approach. *In the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5261–5268. Stockholm, Sweden, May 16-21, 2016.
- C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. Object guided autonomous exploration for mobile robots in indoor and outdoor environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XVI*, volume 9084, Baltimore, MD, April 2014.
- Carlos Nieto-Granda, John G. Rogers, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the International Journal of Robotics Research (IJRR)*, 33(4):519–533, February 2014.
- C. Nieto-Granda, John G. Rogers, and H. I. Christensen. Multi-robot exploration strategies for tactical tasks in urban environments. *In SPIE Defense + Security conference. Unmanned Systems Technology XV*, Baltimore, MD, June 2013.

- Neil Dantam, Carlos Nieto-Granda, Henrik I. Christensen, and Mike Stilman. Linguistic composition of semantic maps and hybrid controllers. *In the 12th International Symposium on Experimental Robotics (ISER)*. Québec City, Canada, October 3-6, 2012.

- John Rogers, Carlos Nieto-Granda, and Henrik I. Christensen. Coordination strategies for multi-robot exploration and mapping. *In the 12th International Symposium on Experimental Robotics (ISER)*, pages 231–243. Québec City, Canada, October 3-6, 2012.

I would like to thank my co-authors John G. Rogers III, Siddarth Choudhary, Varun Murali, Neil Dantam, Nick Fung, Zhen Liu, Jeff Twiff, Prof. Luca Carlone, Prof. Henrik I. Christensen, Prof. Mike Stilman,[†] and Prof. Frank Dellaert.

[†]Prof. Mike Stilman, deceased, was with the Institute of Robotics and Intelligent Machines at Georgia Institute of Technology, Atlanta, Georgia.

Chapter 5

Information-Based Exploration

5.1 Introduction

Single-vehicle informative sampling work for spatial field modeling has been pioneered by [Sin+07; KSG08; Low+09]. They showed that using information-theoretic measures as part of the path planning approach can significantly improve modeling performance. Low et al. [Low+09] modified this initial off-line planning approach to run as adaptive sampling. In adaptive sampling, the robot plans the next steps during the mission, based on the last measurements that were taken. Singh et al. [Sin+07] used sequential allocation, where a centralized path planner planned paths for each robot in sequence, and all robots ran informative sampling in parallel. Ouyang et al. [Ouy+14] used active coordination in a decentralized approach. Robots would coordinate their actions, but only when there were other robots within their robot's planning neighborhood.

Typical multi-robot coordination approaches used in other domains, e.g. exploration and mapping with ground robots, include auction-based methods [Zlo+02; Sim+00a; SX04] and spatial segregation, typically through Voronoi partitioning [SSR12; Mar+15; PFS13].

Nieto-Granda et al. [NRC14] used three heuristics for multi-robot coordination:

1. additional robots wait until they are explicitly needed.
2. robots travel in groups and split when appropriate.
3. robots move in teams of two and split when needed.

The results indicated that a divide & conquer strategy is more effective than keeping robots waiting in reserve. The multi-robot strategies used in adaptive sampling are mostly considering only local collaboration or requiring a central planner. We do not want to use any central components in our system to avoid having a single point of failure. Furthermore, we would prefer strategies that consider the global space, be less susceptible to local minima, and account for the exploration-exploitation trade-off. Some commonly used methods of coordination, such as auction-based methods, require a fair amount of communication between the robots. We want to keep communication between robots to an absolute minimum, given the constraints of the acoustic communication channel, and limit time spent on communication, versus sampling.

In this work, we show that the multi-robot approach's modeling performance can be improved through coordinating actions between robots. This method prevents the scenario where robots choose the same waypoints, based on similar models of the world. We use an approach that combines Voronoi partitioning with a communication strategy for data sharing between robots to achieve a decentralized multi-robot coordination approach.

Previous approaches using Voronoi partitioning include the works of Patten, Schwager, and Soltero [PFS13; Sch+14; SSR14]. Patten et al. [PFS13], and Schwager et al. [Sch+14] used a single initial Voronoi partitioning which is used to divide the sample space across the robots. Patten et al. then had each robot run a traveling salesman problem (TSP) tour over sampling locations. These sampling locations have been greedily selected based on mutual information from within the Voronoi region.

Schwager et al. and Soltero et al. [Sch+14; SSR14] discussed coverage control approaches. In both cases, the density function to be estimated is modeled using a linear combination of a set of basis functions. Schwager et al. [Sch+14] had each robot also run a TSP tour inside their partition, but overall sampling locations, to create the model. The estimated density function is then used to calculate the weighted voronoi centroids, to which robots are deployed for sensor coverage.

In [SSR14], robots either already have the density function or run a lawnmower survey to obtain it. Robots then run paths whose waypoints are Voronoi generators. The waypoints move based on a control law that considers the density function, towards the (weighted) centroid of the area. The paths thus change shape to monitor the interesting regions of the survey area only.Both of these approaches run two stages: model estimation, followed by static sensor deployment, or persistent monitoring. Note that in this work, we focus only on the field estimation, and creating an informative model. We do not assume to know the density function beforehand and are interested in using online and adaptive approaches for modeling it. Patten et al. [PFS13] used a similar approach, where Voronoi partitioning is used to split the survey area across multiple robots. They then have each robot find a subset of sampling locations using mutual information and calculate a TSP tour for sampling. This approach is different in that a Voronoi partitioning is only calculated once, and similar to [Sin+07], the sampling locations, and TSP tour are also calculated only once at the start of the mission, assuming a reasonable model of the environment is already available.

In our research, we are interested in an online, adaptive approach to sampling, which reacts to changes in the model as the robot sample. In [Sch+14], they use Voronoi partitioning for coverage control, where an environment is initially divided into (Voronoi) partitions, using distributed K-means. The robots run a TSP tour inside their partitions, to estimate the density function. This density function is then used to calculate the Voronoi centroids, to which the robots are deployed for sensor coverage. In [SSR14], a control law is laid out, which is used for waypoint control.

After estimating the density function using a lawnmower survey, the waypoints in the robots' trajectories move based on a control law. This applies the density function for the Voronoi partition centered on the waypoint.

5.2 Multi-robot coordination for Informative Adaptive Sampling

Autonomous vehicles are cost and time-efficient systems for exploration and mapping of unknown environments. We are interested in the problem of modeling these unknown environments efficiently. For example, modeling algae abundance in lake environments. Note that this is, in essence, a problem of learning a spatial field. Previous works have shown the benefits of informative and adaptive sampling over traditional methods [Sin+07; Low+09; TWP11]. Multi-robot approaches can reduce the time required to explore or map an area. Our focus is on improving the multi-robot system efficiency and modeling performance through the addition of decentralized multi-robot coordination approaches.

We investigate a multi-robot coordination approach for decentralized, informative, adaptive sampling with autonomous underwater vehicles. We want a decentralized approach for the robustness of the system, such that there is no single point of failure, and to allow for anytime prediction; any robot at any point in time should have a reasonable model of the whole environment. Furthermore, we want an approach that keeps the required amount of communication to a minimum. Especially in underwater environments, communication is severely limited, and we cannot assume continuous access to a reliable communication channel.

To decentralize the approach, each robot maintains its own model of the environment. At specific points in time, robots can share measurements with each other, and add these to their own models. Each robot uses its own model, together with the model's entropy, to decide which locations to sample next.

We combine this with dynamic Voronoi partitioning, to decrease overlap in actions and decisions between vehicles, effectively coordinating actions between the vehicles, with minimal communication overhead. The Voronoi partitioning is re-calculated after every data-sharing event, such that the partitioning changes with the uncertainty in the model, i.e., where the vehicles need to sample. In such, we create a decentralized, multi-robot coordination approach for informative, adaptive sampling of unknown environments.

5.2.1 Approach

We developed a dynamic Voronoi partitioning approach for decentralized adaptive sampling. The vehicles create a model of the environment using Gaussian Process regression and running adaptive sampling utilizing the posterior map entropy of the model. To differentiate vehicle actions, each vehicle runs Voronoi partitioning over possible sampling locations. The posterior map entropy is used here as a density function for shifting the centroids of the Voronoi partitions towards interesting regions. While the vehicles are sampling, at depth, they can not share data. However, they can request data sharing events, at which time data is shared, and the Voronoi partitions are re-calculated. This section will briefly explain each of these subparts of our dynamic Voronoi partitioning[*] for vehicle coordination in informative adaptive sampling.

5.2.2 Gaussian Process regression & path planning

A common technique for spatial modeling, known in geostatistics as Kriging, is Gaussian Process (GP) regression [RW06]. GP regression is a non-parametric modeling technique, where the GP is completely specified by its mean function and its covariance matrix. While the GP

[*]Note: we call our approach 'dynamic Voronoi partitioning' because we re-calculate the (weighted) Voronoi partitions after every data-sharing event. Our approach is different from Bakolas and Tsiotras' 'dynamic Voronoi diagram' [BT10], where a Voronoi partition is re-calculated given expected time to reach Voronoi centers or 'terminal positions.'

is a non-parametric model, its performance is affected by hyperparameters. These are typically estimated from the data using gradient-based optimization [RW06]. In our experiments, we use a zero-mean prior and an isotropic squared exponential covariance function for the GP [KCS16]. Furthermore, we use resilient backpropagation (*Rprop*) for finding hyperparameters that maximize the likelihood of the model [BR13].

To model biological data, we use log-Gaussian Processes (ℓGP), as in [Low+09]: Let Y_x denote an ℓGP, modeling the sensor value y_x at location $x \in \mathcal{X}$, where $\mathcal{X} \subset \mathbb{R}^2$. Let $Z_x = \log_e Y_x$, denote a GP. Then we can create the ℓGP using GP regression by utilizing the fact that $z_x = \log_e y_x$. The GP's posterior mean and variance, $\mu_{Z_x|d_i}$ and $\sigma^2_{Z_x|d_i}$ (for sampled data d_i), are used to calculate the posterior mean and variance of the ℓGP [Low+09]:

$$\mu_{Y_x|d_i} = \exp\{\mu_{Z_x|d_i} + \sigma^2_{Z_x|d_i}/2\} \tag{5.1}$$

$$\sigma^2_{Y_x|d_i} = \mu^2_{Y_x|d_i}(\exp\{\sigma^2_{Z_x|d_i}\} - 1) \tag{5.2}$$

Each vehicle thus creates an ℓGP model on-board, as it is sampling the environment.

The goal of the vehicle's actions is to reduce the uncertainty in the model. The vehicle executes movements using a waypoint behavior. We take a greedy approach to finding future sampling locations, based on the vehicle's ℓGP model. As optimization criteria, we use posterior map entropy for the log-GP, as derived in [Low+09]:

$$\mathbb{H}\left[\mathbf{Y}_{x_{i+1}}|d_i\right] = \log\sqrt{2\pi e \sigma^2_{Z_{x_{i+1}}|d_i}} + \mu_{Z_{x_{i+1}}|d_i} \tag{5.3}$$

where d_i is the already sampled data. Thus the vehicle chooses as its next waypoint an unvisited location with the highest posterior map entropy. For the log-GP, as follows from Equation 5.3, this means locations with high posterior variance (to reduce it) and high expected sensor values (interesting areas). When the AUVs start sampling, the model will initially be uniform. To this end, the vehicles' initial waypoint is a random location in the survey area. The vehicles make straight-line movements to waypoints. It is assumed that there are no obstacles within the survey area. Those could be handled via an obstacle avoidance behavior.

5.2.3 Dynamic Voronoi partitioning

For action coordination between vehicles, we use dynamic Voronoi partitioning. This allows us to coordinate vehicle movements without the need for constant communication. The vehicles independently estimate Voronoi regions to limit their prospective sampling locations. Note that this is not used as a control law for vehicle movement. Assuming all vehicles have a relatively recent estimate of each other's positions, they can each calculate the Voronoi partitioning for the survey area: Each vehicle considers all (unvisited) sampling locations and creates a subset with only those locations closest to itself. If the vehicles would not know the positions of all other vehicles, one could use decentralized Voronoi partitioning, e.g., [Cor+02].

For these initial Voronoi partitions, we then consider the density function (i.e., the posterior map entropy), and calculate the weighted centroid for each partition. The weighted centroid is determined by considering all unvisited sampling locations in the partition and their posterior map entropy. Each location is weighted by its entropy (Equation 5.3), and the sums of all weighted latitude and longitude values are divided by the sum of all weights to find the latitude and longitude of the partition's weighted centroid. These weighted centroids are used as the Voronoi generators for the second round of Voronoi partitioning. The resulting partitions are then used in the path planning algorithm (Section 5.2.2).

5.2.4 Data sharing

Due to the limited bandwidth and potentially reduced throughput of underwater acoustic communications, it is not practical to have all vehicles broadcast their measurements underwater. However, we assume that the underwater communication channel is stable enough to get some messages through, e.g., for inter-vehicle collision avoidance, and for requesting surfacing events.

Each vehicle, at any time, can request a surfacing event. It broadcasts a message to the other vehicle(s) to request surfacing and meanwhile continues sampling. Vehicles receiving a request send an acknowledgment and start surfacing. Upon receipt of acknowledgments from all other vehicles, the initiating vehicle also surfaces. Once on the surface, the vehicles initiate a handshake protocol to ensure all other vehicles are at the surface and ready to start sharing data. After the handshake, all vehicles broadcast their (not yet transmitted) sensor measurements, through Wi-Fi communications. The received measurements from other vehicles are added to each vehicle's local ℓGP. At this point, each vehicle will recalculate the Voronoi partitioning (as per Section 5.2.3), and then re-commence with (at-depth) adaptive sampling.

There are many possible ways of determining when a surfacing event is required; e.g., the criterion could be time-based or information-based. In this work, we take the approach of requesting surfacing events when the vehicles detect that they are sampling close to the border of

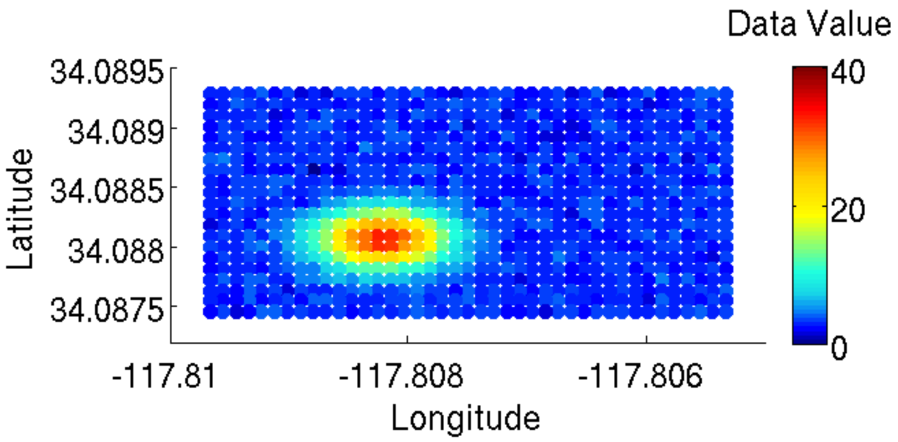


Figure 5.1: Scenario 1: single bloom

their Voronoi region. In such a case, the interesting areas are near the border of their region, and thus it is considered a good time to reconsider the partitioning. Note that vehicles can not request a surfacing event within five minutes of a previous surfacing event.

5.2.5 Experiments

We ran simulation experiments with two autonomous underwater vehicles, using the MOOS-IvP middleware [Ben+10]. The middleware includes a simple simulation of vehicle dynamics and PID control, as well as behavior-based autonomy. We use the following standard behaviors; loiter, waypoint, constant depth, and (inter-vehicle) collision avoidance. We use a constant speed of $1.5 m/s$, and constant depth of $5 m$. Our path planning approach updates the waypoint behavior.

To simulate algae blooms in a lake, we generated a 2-D grid space (400x200m, 10m spacing) with a 2-D Gaussian to represent a bloom, and additive Gaussian noise. For the data, we use a data value amplitude of 40, as a proxy for high Chlorophyll $\mu g/L$ values, with a noise amplitude of 10-20 percent. Within the MOOS-IvP simulation, each vehicle samples from this simulated data, and we add Gaussian noise ($\sigma = 1.5$) to model sensor noise [KCS16]. Note that the sensors take point measurements, there is no range to the sensors, and all vehicles have the

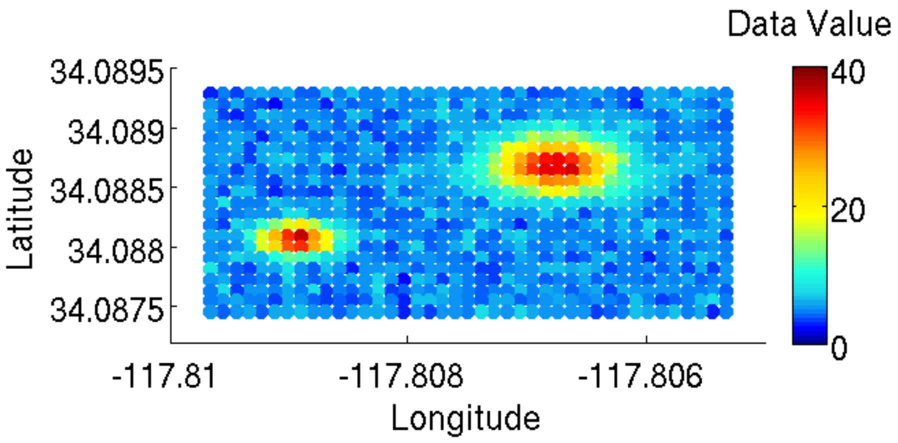


Figure 5.2: Scenario 2: dual bloom

same sensor. We simulate two different scenarios for the grid space; a single bloom with low noise, and two (differently sized) blooms in opposite corners with higher noise, as shown in Fig. 5.1 and 5.2.

To simulate communications between the vehicles, we use two forms of simulated communications: acoustic and Wi-Fi. For acoustic communications (acomms), the *Goby* acomms suite enables us to simulate the whole acomms stack; modem driver, medium access control, priority-based message queuing, and encoding and decoding of messages [Sch14]. We use a TDMA (Time Division Multiple Access) scheme for sharing the data channel between the vehicles. Each vehicle has a pre-assigned 3-second time slot, in which it can send one to two 32-byte messages. We limit the acomms range to $500m$, and are able to reduce the throughput of the channel to probabilistically drop messages, e.g. to drop 30% of the messages. For Wi-Fi communications, we simulate UDP communications, without restrictions on throughput or range[†].

In [KCS16], we showed the improvements in modeling performance when adding data sharing between vehicles. Vehicles sharing data performed better than those vehicles that did not share data and ran standard lawnmower surveys[‡] or adaptive sampling in parallel.

In this work, we compare the dynamic Voronoi partitioning approach for informative adaptive sampling with the data sharing approach, and run the following experiments:

- 2 AUVs, timed data sharing, no coordination,
- 2 AUVs, dynamic Voronoi partitioning,
- 3 AUVs, timed data sharing, no coordination,
- 3 AUVs, dynamic Voronoi partitioning.

[†]Assuming Wi-Fi via $2.4Ghz$ radio and good antennas, which we have empirically found capable of at least $500m$ (unobstructed) range, which covers the current experimental area.

[‡]A lawnmower survey is a typical coverage behavior where a vehicle travels back and forth across the survey area.

In the timed data sharing approach, the AUVs surface every ten minutes[§], to be able to share data. When all vehicles are on the surface, they share data after executing the handshake protocol (Section 5.2.4).

For every 2 AUV and 3 AUV experiments, we ran 10 and 15 simulations respectively, and we averaged over all results. Furthermore, we ran all these simulations over both scenarios (Fig. 5.1 and 5.2). The duration of each experiment was limited to the duration of simulations with the vehicles running lawnmower surveys. For example, to survey the area with three vehicles running high resolution, $20\,m$ track spacing, lawnmower surveys takes on average approximately 3500 seconds. Therefore, after the vehicles have run adaptive sampling for 3500 seconds, they are requested to return to the start location, for the final data sharing and hyperparameter optimization.

5.2.6 Results

The results from all experiments are evaluated using root mean squared error (RMSE) between the posterior mean from the ℓGP and the generated data, and using the negative log-likelihood (NLL). The RMSE captures the predictive mean performance of the ℓGP, and the NLL incorporates also the predictive variances. We store the model predictions throughout the mission, at 10 minute intervals ($600\,s$), for evaluation. Note that all figures in this section show results per these 10 minute time steps, and all changes therefore seem to occur at the same times. We run a first hyperparameter optimization at the first surfacing event. For the timed data sharing, this is at around 600 seconds into the mission. For the Voronoi mission, this is on average at 400-450 seconds into the mission. A second hyperparameter optimization is run at the end of the survey.

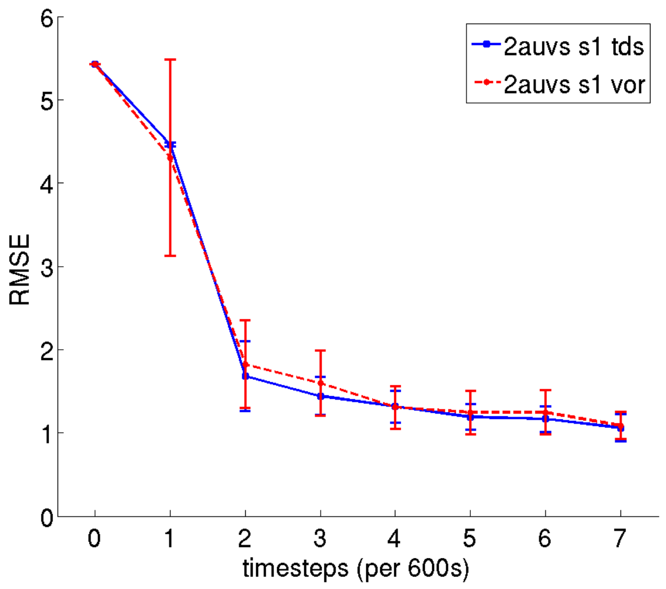


Figure 5.3: Scenario 1 (s1), one bloom: Average RMSE and one standard deviation (SD) error bars, averaged over all AUVs in the 2-AUV mission, for both timed data sharing (*tds*) and dynamic Voronoi partitioning (*vor*).

Two AUVs

We ran simulation with two AUVs, running either timed data sharing or dynamic Voronoi partitioning. Results are averaged over all vehicles, for 10 simulations per experiment and scenario.

Figure 5.3-5.6 show the results, in terms of RMSE and NLL, for two AUVs running adaptive sampling with only data sharing, versus coordination through dynamic Voronoi partitioning. Note that the final time step is after the final hyperparameter optimization, when the survey is finished, and can hence lead to a bigger reduction of RMSE or NLL. The vehicles running timed data sharing surfaced 5 times during the mission. The number of surfacing events for the dynamic Voronoi approach was 7-9 times.

Figure 5.3 and 5.4 show the RMSE and NLL for the first scenario, running with two AUVs. As can be seen, the performance for the two approaches is on average the same.

[§]This assumes that the vehicles have synchronized clocks, which is a given in our simulations, but should be paid attention to at field trials.

Figures 5.5 and 5.6 show the RMSE and NLL for the second scenario, running with two AUVs. As can be seen, the RMSE initially drops more quickly for the dynamic Voronoi partitioning, showing improved performance. Both methods reach similar values at the end of the mission. The performance is about the same in terms of NLL, but the dynamic Voronoi approach is in general more consistent in performance. Overall, for the experiments with 2 AUVs, we see that the modeling performance between the two methods is the same for the first scenario, but better with dynamic Voronoi partitioning in the second scenario.

Three AUVs

We also ran simulation with three AUVs, running timed data sharing and dynamic Voronoi partitioning. Results are averaged over all vehicles, for 15 simulations per experiment and scenario. Figures 5.7-5.10 show the averaged results for the simulations with three AUVs. Note that the final time step is after the final hyperparameter optimization, when the survey is finished, and can hence lead to a bigger reduction of RMSE or NLL. The vehicles running timed data sharing surfaced 4 times during the mission. The average number of surfacing events for the dynamic

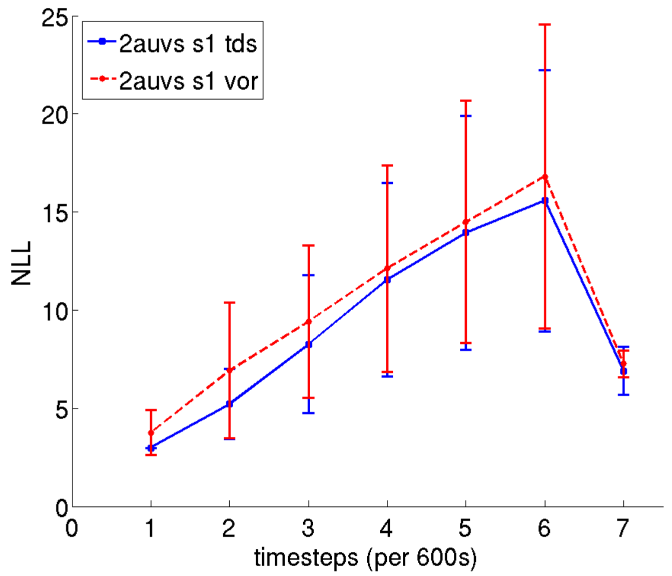


Figure 5.4: Scenario 1 (s1), two blooms: Average NLL and one SD error bars, averaged over all AUVs in the 2-AUV mission, for *tds* and *vor*.

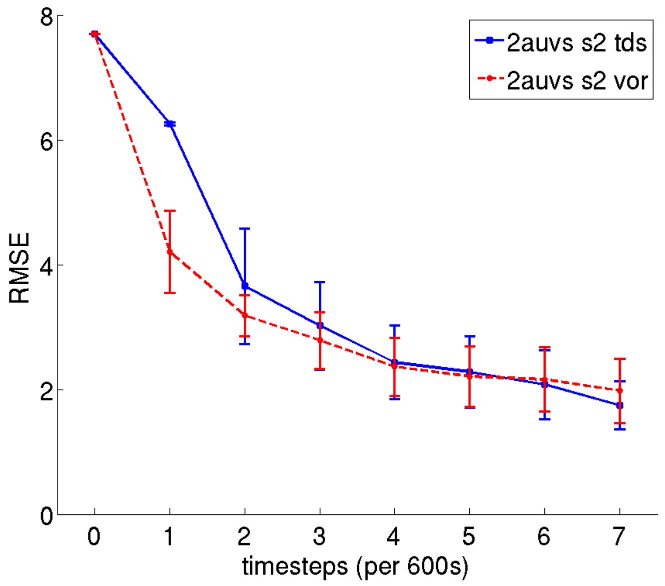


Figure 5.5: Scenario 2 (s2), two blooms: Average RMSE and one SD error bars, averaged over all AUVs in the 2-AUV mission, for both *tds* and *vor*.

Voronoi approach was 6-7 times.

Figures 5.7 and 5.8 show the RMSE and NLL for the first scenario with a single bloom. The figures show that with the coordination through Voronoi partitioning, the modeling perfor-

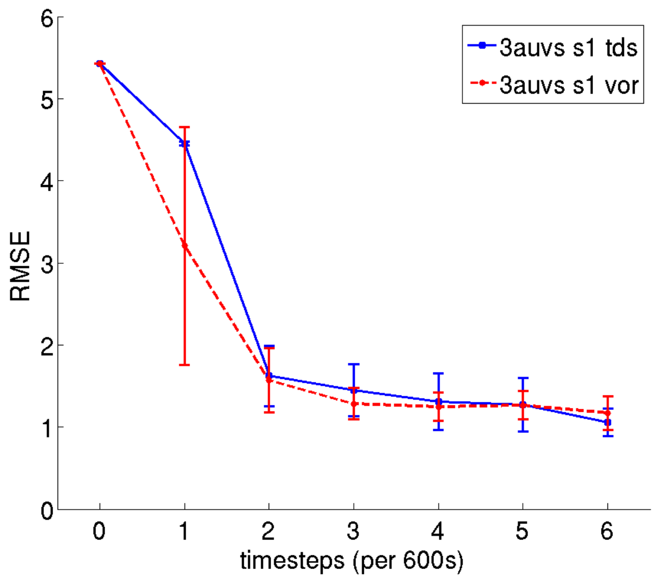


Figure 5.7: Scenario 1 (s1), one bloom: Average RMSE and one SD error bars, averaged over all AUVs in the 3-AUV mission, for both *tds* and *vor*.

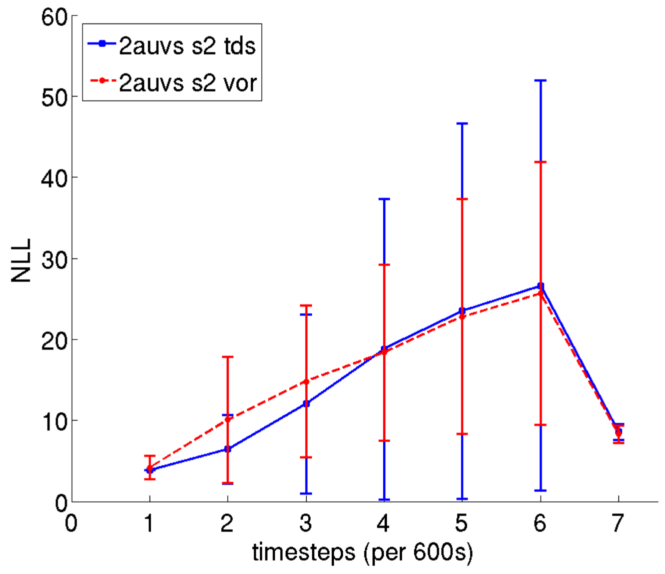


Figure 5.6: Scenario 2 (s2), two blooms: Average NLL and one SD error bars, averaged over all AUVs in the 2-AUV mission, for both *tds* and *vor*.

mance improves both in terms of RMSE and NLL, in particular over the first 3-4 time steps (1800-2400 seconds) of adaptive sampling.

Figures 5.9 and 5.10 show the RMSE and NLL for the second scenario with two blooms.

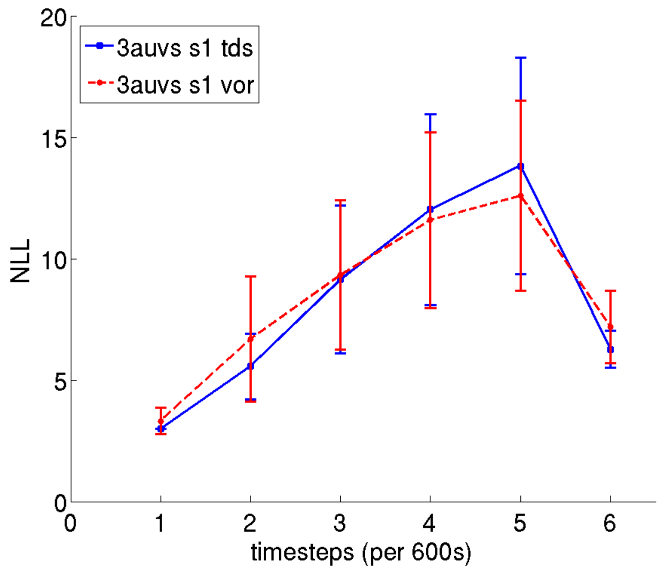


Figure 5.8: Scenario 1 (s1), one bloom: Average NLL and one SD error bars, averaged over all AUVs in the 3-AUV mission, for both *tds* and *vor*.

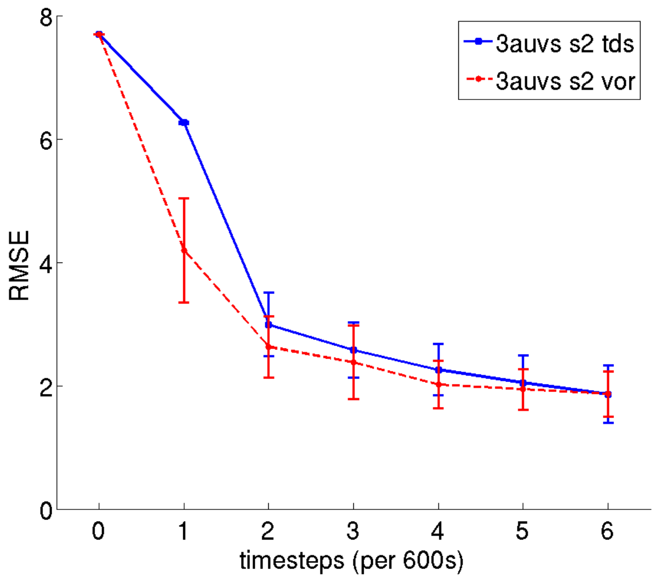


Figure 5.9: Scenario 2 (s2), two blooms: Average RMSE and one SD error bars, averaged over all AUVs in the 3-AUV mission, for both *tds* and *vor*.

Note that, due to the higher noise in the dual bloom scenario (see Figure 5.2), the RMSE and NLL are higher than for the single bloom scenario. Again we see that the performance is better when using coordination through Voronoi partitioning, instead of only data sharing. Overall, we see for the experiments with 3 AUVs that the use of dynamic Voronoi partitioning for vehicle coordination results in higher quality models, in particular for the second scenario with two blooms.

5.2.7 Discussion

The results show that the modeling performance improves with addition of a coordination approach, in multi-robot informative adaptive sampling. For the 2-AUV experiments, the performance is the same for the experiments on the first scenario. For the 3-AUV experiments, the performance improvement is also not as well defined for this scenario. This is likely because the single bloom scenario, with low noise, is a simple scenario, where the addition of vehicles, or coordination in the multi-robot approach, does not pay off.

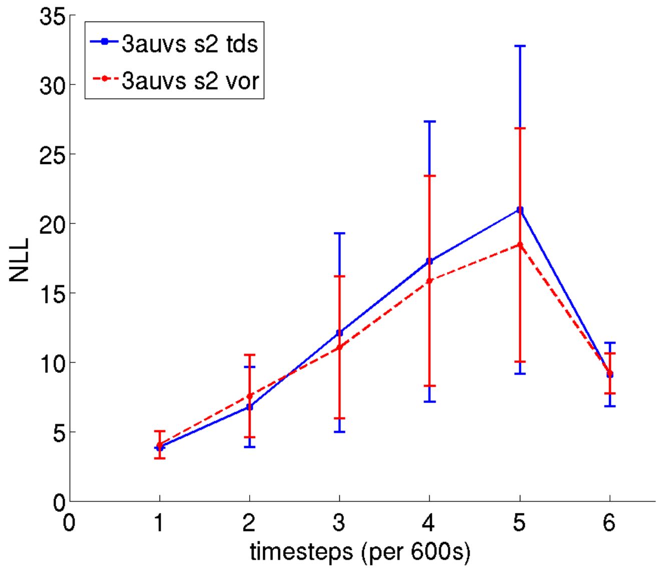


Figure 5.10: Scenario 2 (s2), two blooms: Average NLL and one SD error bars, averaged over all AUVs in the 3-AUV mission, for both *tds* and *vor*.

For the second scenario, the performance improvement is more pronounced. We see that in particular the RMSE improvement is bigger at the start of the mission, and overall the NLL performance is more consistent. The improvement is bigger at the beginning, because this is where the biggest improvement can be obtained. After ca. 3-4 time steps (1800-2400 seconds) of sampling, the performance starts to level off for all methods. This indicates that the model after this time span is about as good as it gets, and reaching the limit in possible performance given model and sensor noise. The addition of another hyperparameter optimization step at this time may aid in improving modeling performance. However, it is also an indicator that the mission could be ended earlier for all these adaptive sampling approaches.

We have not evaluated the overall time taken for the missions, because time was used to regulate mission duration, and no other stopping criteria were tested. All missions took exactly the same amount of time, and vehicles traveled approximately the same distance, given that the AUVs never stopped and ran at the same speed. If we were to let all robots run for infinity, every approach would end up with the same model performance. This is also evidenced by the final performance being similar, if not the same, for all methods.

We are thus looking for improvements in modeling performance at early stages of the mission. In [KCS16], we showed that the modeling performance for adaptive sampling is greatly improved in the early stages of the mission, after which the performance levels out, suggesting the adaptive mission could have been ended earlier. For the 3-AUV Voronoi approaches, we see also see improvements in the early stages, which mean that (a) we could potentially end the mission earlier, and (b) if we had to stop the mission at an earlier stage, we would have a better model. This type of ‘anytime prediction’ capability is very promising.

As mentioned in Section 5.2.6 and Section 5.2.6, the numbers of surfacing events for the dynamic Voronoi approach was a little higher for both the 2-AUV and 3-AUV experiments (7-9 versus 5, and 6-7 versus 4, respectively). For timed data sharing, the vehicles surfaced every 10 minutes ($600s$). For dynamic Voronoi partitioning, the trigger for surfacing events existed of a vehicle nearing the Voronoi border of their partition. The number of surfacing events thus depends on the size of the Voronoi partition, the location of the vehicle, and the location of informative unsampled areas in this partition. The number of surfacing events therefore is not optimized (i.e. minimized). It is limited however by a ‘blackout’ period of five minutes after each previous surfacing event, to prevent immediate re-surfacing, for example if a new Voronoi region border happens to be near a vehicle. It would be interesting to investigate different triggers for requesting surfacing events, and to develop a strategy for minimizing surfacing, while still maintaining the benefits of regular data sharing and anytime prediction capability. While our current research efforts focus on shallow waters and relatively shallow depths of features of interest, the minimization of surfacing events will be of greater importance for deep water operations.

Asynchronous surfacing strategies, via a central sharing vehicle/system, were not evaluated in this work either, because we wanted to avoid having a single point of failure on the surface or on shore, and we wanted to avoid the addition of a surfaced data mule for both cost and vehicle safety reasons.

Another approach for asynchronous surfacing would be to have vehicles share information in subsets or teams, and to share with those who do not have the data yet, whenever possible. This would require a lot more internal accounting and the capability of vehicles to share or negotiate what has (not) been transferred. While asynchronous sharing would remove the current system's ability to provide a good model from any vehicle at any time, it is an interesting avenue for future work. In particular when considering the application of adaptive sampling systems to environments where any form of communication can only be achieved in certain areas or at certain times.

In the future, we intend to explore different coordination approaches, as well as the effect on performance of the use of different path planning algorithms, and the application to larger areas. Furthermore, efforts are ongoing to demonstrate the adaptive sampling performance through field trials. Thus we also aim to create new scenarios for future simulations based on field data.

Overall, we have shown that informative adaptive sampling with multiple robots benefits from the addition of action coordination between vehicles. Previous works have focused mostly on coordinating within close range for local planning, or coordinating between vehicles by planning sequentially. We developed an approach where the vehicles can coordinate their actions, without limiting the coordination approach or the path planning to a local neighborhood.

Furthermore, our approach is completely decentralized, and creates a robust multi-robot system for adaptive sampling. In combination with the communications strategy, this makes the approach applicable for deployments in hazardous or communication-constrained environments. Our dynamic Voronoi partitioning technique for multi-robot coordination is thus an effective method in running decentralized, informative adaptive sampling with multiple robots, in unknown environments.

5.3 Information Gain Exploration - “Infosploration”

Teams of robots can potentially improve efficiency in persistent monitoring of environmental conditions, for monitoring of natural or human-made disasters, or search and rescue after such disasters.

For many of these scenarios, it is useful to create spatial models of the environment or environmental characteristics, such as temperature or algae distributions in the ocean, radiation levels after nuclear disasters, or radio/WiFi strength in buildings or disaster zones. One effective way of creating such spatial models is by using Gaussian Process (GP) regression, also known as Kriging in spatial statistics. When information-theoretic metrics, such as entropy or mutual information, are used on top of the GP model’s predictions to decide where to place sensors or robots, we speak of informative sampling, as further explained in Section 5.3.1. When informative sampling locations are determined after a model has been created, this is called off-line informative sampling, when the robot is creating a model. At the same time, sampling data, this is called on-line or adaptive informative sampling.

We address the efficient tasking of teams of robotic platforms to sample and map environmental properties. For this paper, we monitor radio signal strength indication (RSSI) in an indoor environment. Highly structured indoor environments represent a unique challenge in this kind of environmental modeling due to navigational complexity. This effect is magnified, where teams of robots must also take care to avoid blocking or crashing into one another due to narrow passageways or doorways.

We present a multi-robot coordination approach for informative adaptive sampling in structured environments. Specifically, our contributions include:

- identifying high priority points of interest according to an information gain rate adaptive metric
- an efficient partitioning of an environment for multi-robot cooperative missions

- a strategy for over segmenting a region to more deliberately coordinate multiple robots
- an implementation and experimental evaluation of these strategies on real world robotic platforms

5.3.1 Approach

We task multiple robotic platforms to model RSSI signals for an established environment. The system utilizes a priori knowledge of the environment, without knowledge of the RSSI topology. Up to four robots are tasked to traverse the area and measure the environment according to expected information gain per distance traveled. We utilize a high-level approach to coordination, in which the central tasking unit separates the robots by directing them to different areas of the environment.

Signal modeling by GP regression

We model RSSI using Gaussian Process (GP) regression, a standard method for spatial field modeling [RW06]. The GP approximates measurements at different locations with Gaussian functions based on a prior mean and covariance function $k(\cdot,\cdot)$, also known as the kernel. When a sample y at location $\mathbf{x}$ is added to the GP, the Gaussians at locations $\mathbf{x}'$ are updated based on the chosen kernel, where $\mathbf{x},\mathbf{x}' \in \mathbb{R}^2$.

For each possible sampling location $\mathbf{x}_*$ the GP has a predictive mean and predictive variance [RW06]:

$$\mu_{y_*}(\mathbf{x},\mathbf{x}_*,y) = k(\mathbf{x}_*,\mathbf{x})k(\mathbf{x},\mathbf{x})y \tag{5.4}$$

$$\sigma^2_{y_*}(\mathbf{x},\mathbf{x}_*) = k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{x})k(\mathbf{x},\mathbf{x})^{-1}k(\mathbf{x},\mathbf{x}_*) \tag{5.5}$$

where $\mathbf{x}$ and y are training locations and measurements, $\mathbf{x}_*$ are test locations, and $k(\cdot,\cdot)$ is the kernel, i.e. the covariance function. The predictive variance is especially useful in adaptive sampling because it gives an estimate of the confidence in the predicted mean and robots can choose to sample in locations with larger uncertainty (entropy).

The GP is initialized with a zero mean prior, and we use a combination of an isotropic squared exponential (SE) kernel, and white noise covariance function, for the kernel. The SE covariance function is given by [RW06]:

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}|\mathbf{x}-\mathbf{x}'|^2\right) \tag{5.6}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are two training sample locations, $\mathbf{x},\mathbf{x}' \in \mathbb{R}^2$, σ_f^2 is the signal variance (or amplitude), and l is the kernel's length scale. The signal variance and length scale are the GP's hyperparameters. We combine the SE kernel with a white noise kernel, to better model the expected noise in the data. This kernel has one hyperparameter: noise variance (σ_n^2). The hyperparameters can be estimated from data or set based on prior knowledge.

In this work, we use two GP models. The first GP model is created from real world RSSI data, which is collected by manually operating robotic sensors platforms in a series of controlled environments that contained several radio signal generators. The RSSI readings are collected and GP regression is used to create an RSSI model from the sensor data. This RSSI model is used as the ground truth in our evaluation. During subsequent simulation experiments, the simulated sensor data is generated from the trained 'real world GP model'. The second GP model concerns the GP models that are created on-board the robots, based on their sampled data, as part of the adaptive sampling procedure during all simulation runs. For real world experimentation, there is no ground truth model and instead we evaluate performance based on overall entropy of the environmental model.

Information Gain Utility

For any sampling location, the Gaussian Process model gives us a predictive mean and predictive variance which we can use to decide where to sample next. We use the differential entropy ($h(\mathbf{x})$) as a measurement of information. Because the RSSI is modeled as a Gaussian Process, we use the differential entropy of a normal distribution:

$$h(\mathbf{x}) = \ln(\sigma\sqrt{2\pi e}) \tag{5.7}$$

We note here that the differential entropy is independent of the mean of the normal distribution and depends only on the variance (σ^2).

Using the differential entropy for each point of interest, the system seeks to minimize the entropy of the map by tasking robots to points of high entropy. We calculate the utility of a point of interest ($\mathbf{x}_{poi}$) from a point of origin ($\mathbf{x}_0$) as:

$$U(\mathbf{x}_0, \mathbf{x}_{poi}) = \frac{\sum h(\mathbf{x}_1), h(\mathbf{x}_2), ..., h(\mathbf{x}_{poi})}{(dist[\mathbf{x}_0, \mathbf{x}_{poi}] + \alpha)} \tag{5.8}$$

where $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{poi}$ are the series of points that are traversed to achieve the shortest path, and α is a constant value parameter used to weight the travel distance according to the time the system needs to calculate a new travel path. This value considers that there is a cost involved with calculating where to go, such that choosing to visit nearby points is penalized. For these series of experiments, we use $\alpha = 1$. The value of $dist[x_a, x_b]$ refers to an estimation of the shortest path distance between two points, calculated in Section 5.3.2.

5.3.2 Multi-robot coordination and task allocation

For the multi-robot coordination problem, we seek to partition the sampling space into distinct regions and assign a robot to each partition. By separating the exploration space of each robot, we reduce the occurrence of repeated coverage as well as the amount of physical interference between robots. We use Equation 5.8 to calculate the utility of a specific path along a list of points. We then calculate the utility of a region as:

$$U(R_i) = \max_{(x_0, x_1, \ldots, x_n) \in R_i} \sum U(x_{p-1}, x_p) \tag{5.9}$$

in which we maximize the sum of the utility values of the points within a partitioned region over the traversal order of the points.

The optimal partition can be defined as a partitioning such that the utility of each partition is equal:

$$\forall_{R_i, R_j \in R} : U(R_i) = U(R_j) \tag{5.10}$$

Partitioning under this criteria would allow for each robot to explore a different region and to attain the same amount of information over time.

However, optimal partitioning is not practical in this context because the information gain is based on a model that is being trained in real time. As the model is updated, the amount of utility estimated for each partition will change. While the partitions could then be updated, this can be computationally expensive. Instead, we partition along the traversal cost of each segment. That is, each partition takes a similar amount of time to exhaustively traverse. Using travel cost as an approximation of utility gain has a number of advantages including that it is static and computationally easy to calculate. It is also a fair approximation because the entropy term only considers the GP's predictive variance, which is a measure of sampling location distances and not dependent on the measurements taken at sampling locations.

To create our partitioning, we form a graph structure representing the points of interest within the floor plan and perform a normalized graph cut. We formulate the graph structure through the establishment of each node as a point of interest and each edge as a line of sight neighbor connection, weighted by the physical distance. We then use the Floyd-Warshall algorithm to iteratively search through the weighted graph and find the shortest graph distance between each pair of vertices [Flo62]. We store these values in a distance matrix.

Each entry $C_{i,j}$ in the distance matrix is an approximation to the traversal cost between vertex i and vertex j, which are navigational points of interest sampled in the map. We define the matrix with entries $W_{i,j} = \frac{1}{C_{i,j}}$ to represent "spatial affinity", which means that point pairs will have large affinities if the robot can travel between them quickly. We then perform a normalized graph cut to partition the graph into segments in which the travel distances are balanced within each partition. Shi and Mallik described the normalized graph cut in [SM00]. Key details are provided here, but the reader should refer to their work for a complete treatment. The approach consists of finding the eigenvector corresponding to the second smallest eigenvalue of the matrix:

$$D^{-1}(D-W)D^{-1} \tag{5.11}$$

where D is the diagonal matrix with D_{ii} set as the sum of edges adjacent on vertex i in the affinity matrix W. Elements of the second smallest eigenvector that are positive are assigned to one group and negative are assigned to the other group in a binary partition. To create more partitions for larger robot teams, this algorithm is recursively applied to the previously computed partitions. When multiple potential partition candidates must be decided upon, as in the case of 3 robots where one of the two partitions from the previous step must be sub-partitioned, the one with the lowest normalized cut cost metric is selected.

There are several cases related to the number of partitions in relation to the number of robots. If the number of robots is equal to the number of partitions, i.e. *equal partitioning*, each robot is assigned its closest partition to explore. If there are more partitions than robots, i.e. *over partitioning*, a task assignment procedure is used to instruct each robot to sample an individual partition. This assignment will be revised over time as information is incorporated into the model. As the information gathering operation progresses, the entropy in the sampled partitions will be reduced, and robots will be reallocated to other partitions to improve the model.

The task assignment routine begins by adding up the entropy of each point of interest within a partition. These partitions are then sorted on the entropy measure and the highest entropy partitions are each assigned to a robot. In the case of *over segmenting*, some partitions will not initially be assigned to a robot for sampling. To ensure that robots continue to collect information, once the total entropy of their assigned partition falls to a ratio of 70% of the original value, the partition assignment procedure is repeated for that robot. At this point, the robot will be assigned to the region with the largest cumulative entropy in the model that is not currently assigned to a a robot. It is possible for this new assignment to be to the same partition if it has the largest entropy sum. Figure 5.11 shows partitions created for our test environments, showing fixed cuts for two to four robots (left three columns of images) and over segmentation using the normalized cut (rightmost column of images).

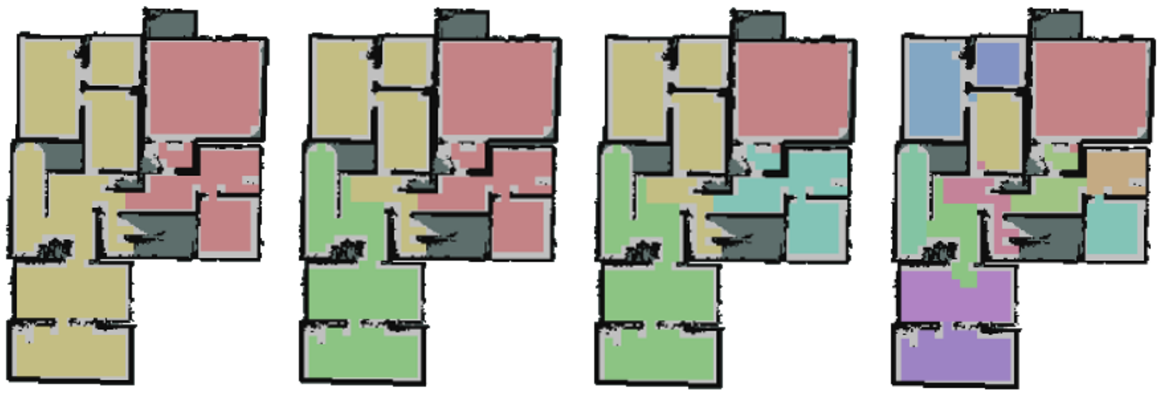


Figure 5.11: Partitions selected by normalized graph cuts on inverse path cost affinity matrix for between two and four robots for one of the test environments. From left to right are shown partitions for 2, 3, and 4 robots. On the right is an example of over segmenting the environment.

5.3.3 Experiments

Real world RSSI data

Real world RSSI data was collected in a controlled test environment using robotic sensor platforms, namely the Jackal robots, shown in Figure 5.18, with an accompanying sensor suite. The sensors include a laser range finder, X-Bee radio, and wireless access point. These robots were manually driven through a building environment that contained three static radio signal generators, each with unique identification. A robot is tele-operated along several trajectories within the building while RSSI, laser, and odometry data are recorded.

We create a model from this real world data to have a realistic sensor model, from which we can draw ground truth values in simulation. Alternatives to this approach include generating received signal strength measurements by a numerical electromagnetic propagation solver such as EM Cube [EMA]. To apply this type of simulation engine to our problem would require accurate modeling of electrical properties of materials and structures within the test environment, which is infeasible. It is also not clear how other phenomena such as radioactivity or especially chemical diffusion would be modeled by this process, though other types of numerical simulators could be applied to these other problems.

A simpler log-distance path loss model [CGK09] with parameters learned from measurements in the test environment could approximate the overall range-dependent component of RSSI, but would fail to adjust for line-of-sight and occlusion. This model would also be a poor approximation for a diffusion process such as may be measured with a chemical sensor. To build the simulation model, first, a map is generated from one of the test runs for each environment using OmniMapper [TRC14b]. Each run in that environment with a given configuration is then localized via Adaptive Monte-Carlo Localization (AMCL) [Fox+99] with recorded laser and odometry data.

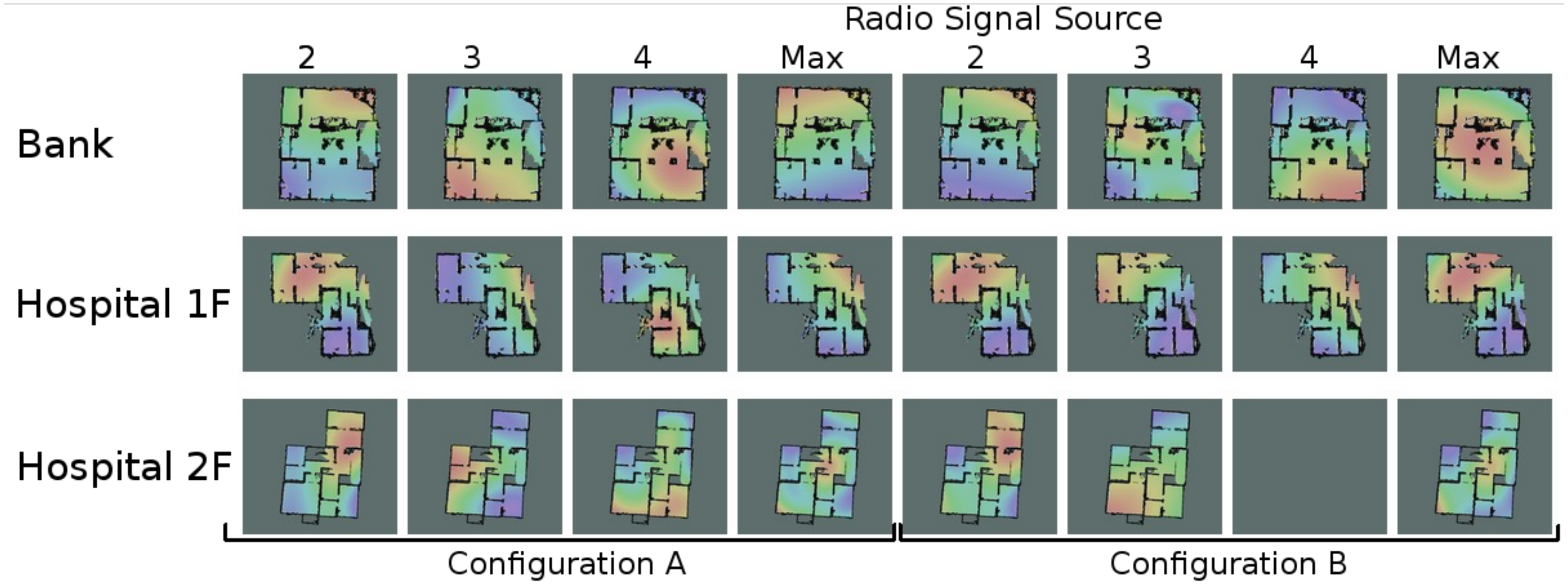


Figure 5.12: False color graphics of received signal strength indication (RSSI) learned from trajectories in real world environments. Red indicates a large value (close to transmitter) and dark blue/purple is a small value (far from transmitter and poor signal). Rows represent each of the three buildings, columns represent transmitter configurations. Upper labels (2,3,4) are radio transmitter ID numbers. "Max" indicates a model built with the maximum of the three transmitters. Lower labels "Configuration A" and "Configuration B" indicate the two placement configurations of the transmitters. The training data for each configuration was taken simultaneously and split into these separate models. The models are used to generate measurements in simulation, and as ground truth for analysis of simulation runs. The model for Hospital 2F Configuration 4B is missing due to battery failure on transmitter 4 during these data collection runs.

An initial pose estimate is used to speed up the convergence of the localization routine. While the localization routine is converged on a pose hypothesis, the corresponding RSSI measurements are recorded into a GP model using libGP [Blu]. This process is repeated for each trajectory with that given transmitter configuration. For each configuration, four separate models are built for additional evaluation options: one with just each of the three transmitters alone, and a final model which consists of the maximum RSSI value over all transmitters. A false-color image representing these learned ground truth models can be seen in Figure 5.12. Simulated RSSI values are generated from this model during a test run by sampling from our learned GP model at the robot's location.

Simulation experiments

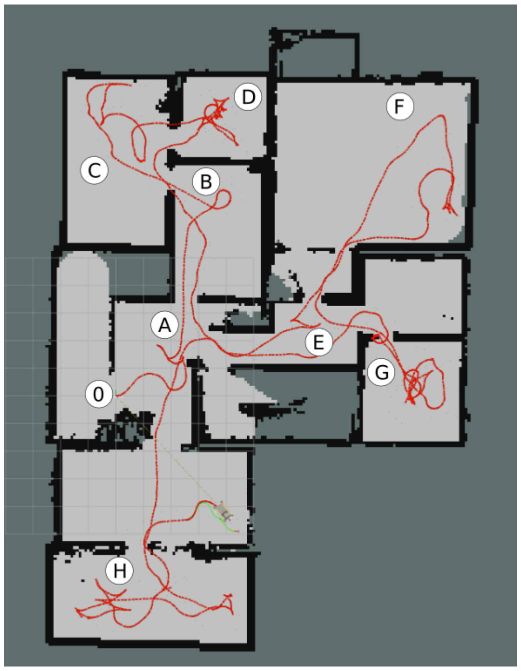
For the simulation experiments, the robots traverse the area and sample from the GP model trained from real world data in order to obtain simulated sensor measurements. A sensor measurement is taken from the GP for the corresponding real world location, according to the GP mean and variance, with noise sampled according to the ground truth model statistics, reflecting the variability in sensor measurements observed during the multiple training runs. We assume that the robots have a floor plan, but do not know the RSSI distribution or any information regarding the position of the transmitters. Localization is assumed to be accurate and for this simulation it is generated without noise. There is a centralized controller that decides on the task assignment of the robots and we assume full communication between all robots and the central server.

We simulate the following experiments:

- single robot nearest point - the robot traverses to the nearest unsampled point of interest
- multi robot information adaptive, uncoordinated - each robot seeks the point of interest with highest utility (information/cost)
- multi robot information adaptive, coordinated - the map is partitioned and each robot is

assigned its own area to explore

- multi robot information adaptive, coordinated over segmented - the map is partitioned into more segments than robots and each robot is assigned a segment to explore once its assigned segment has been measured



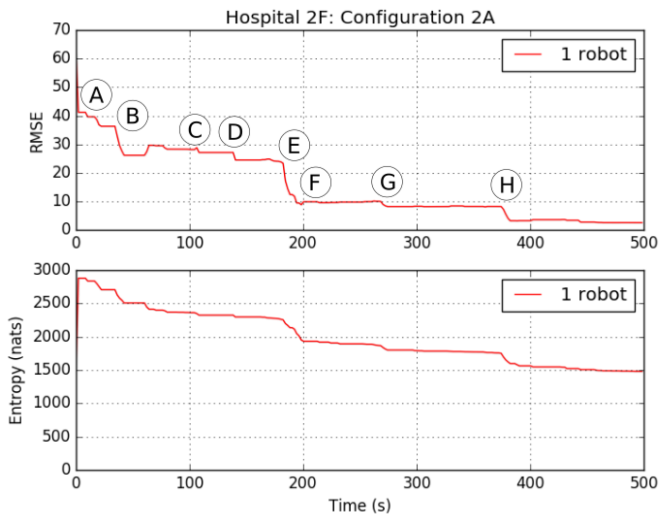


Figure 5.13: Annotated single robot sample run: The robot proceeds from point 0 (start) to A through H (end).

An annotated sample run alongside the corresponding RMSE and entropy measurements can be seen in Figure 5.13.

As the robot enters previously unexplored areas, we see a corresponding drop in overall entropy and a reduction in the overall error of the system. For example at point E, the robot progresses toward a large unexplored area and this results in a large reduction in error.

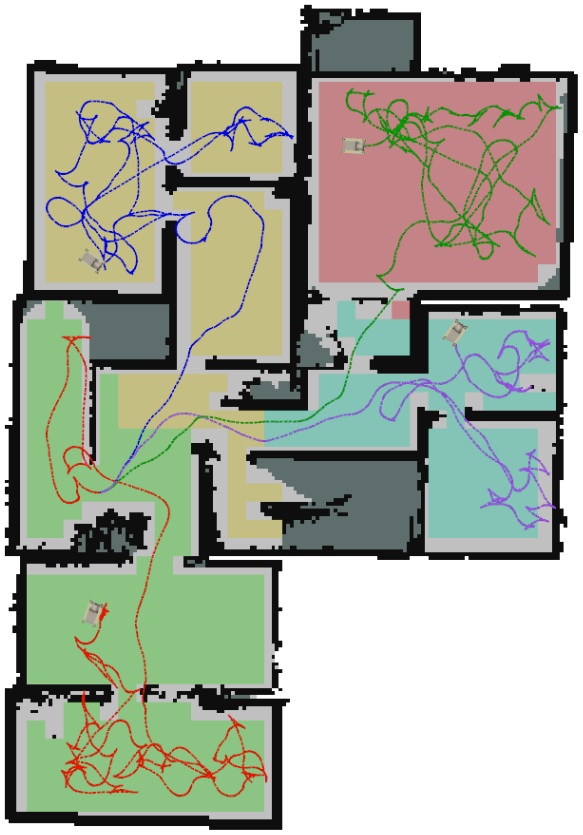


Figure 5.14: Sampling trajectory with four robots in one of our test environments. Partitions assigned to each robot are shown via shading open occupancy grid cells.

The experiments are carried out on three different floor plans labeled Bank, Hospital Floor 1, and Hospital Floor 2 corresponding to real world locations. The environments were seeded with 3 radio signal generators labeled 2, 3, and 4 (radio 1 is located on the exploration vehicle) placed within the environment under two different configurations: A and B. Including the "Max" radio configuration, in which all the radio signals are active, this results in 8 different radio signal configurations. This yields 24 different environments. The data from Hospital Floor 2, radio 4 in configuration B was lost due to a battery failure on radio 4.

A sample simulation run in Figure 5.14 shows the partitioned floor plan, model entropy and model error over time, as well as bar plots with overall entropy and error of the system at completion for different numbers of robots.

Table 5.1: Area under RMSE curves for the baseline (nearest-first) approach with up to four robots compared to the information adaptive, and information adaptive with coordination vs figures shown in Figure 5.15

	Baseline	dI-Adaptive	dI-Adaptive Partitioned
1 Robot	8701 ± 1010	5913 ± 1143	n/a
2 Robots	7662 ± 1833	4149 ± 853	3118 ± 343
3 Robots	6184 ± 1057	3183 ± 1013	2505 ± 184
4 Robots	5602 ± 469	3143 ± 1114	1675 ± 270

We also test the information rate adaptive approach against a baseline algorithm that simply traverses to the nearest neighboring unmeasured point. The results of this experiment can be seen in Table 5.1. We note here that the results for the 1 robot run for partitioned and unpartitioned space is not duplicated as it is the same scenario.

We evaluate the simulation results using the root-mean squared error (RMSE) between the ground truth (GP model of real world data) and the GP model created through sensor measurements by the simulated robot.

Figure 5.15 shows the area under the RMSE curves for the information adaptive approach with the area partitioned for the robots, for Hospital Floor 2 with radio configuration 2A. We can measure the area under the RMSE curve to evaluate how quickly the RMSE falls as the robots build their GP models. A small area under curve indicates that the error fell quickly, demonstrating an improvement on performance.

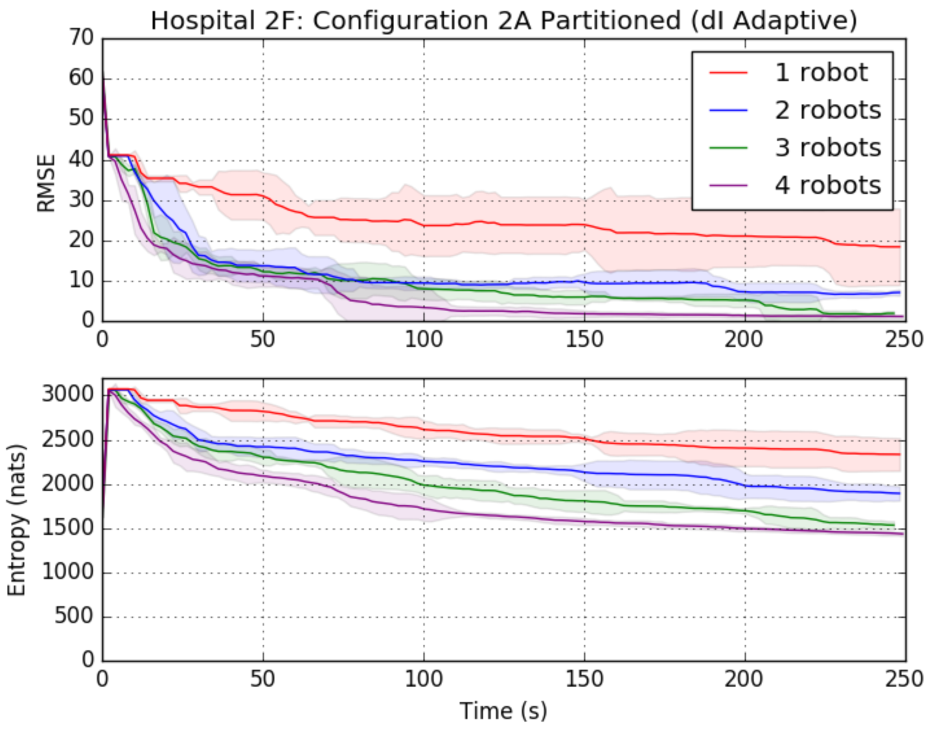


Figure 5.15: RMSE curves for up to four robots, using adaptive informative sampling and normalized graph cut-based partitioning for coordination, for Hospital Floor 2 configuration 2A. Averages are shown and two standard deviations are shaded from at least 5 simulations.

The simulation results show that the information rate adaptive approach performed better than the nearest neighbor baseline approach in all instances. Table 5.1 summarizes the final area under the RMSE curve against the baseline. As expected, larger robot teams are able to lower the system entropy faster. In addition, entropy drops significantly when adding our multi-robot coordination approach through area partitioning.

Table 5.2 shows the improvement in performance we see across all data sets for scenarios when the environment is segmented into the same number of partitions as robots versus when the robots are left uncoordinated. When considering each specific scenario, we find that on average the improvement is $30-35\%$.

For a clearer visual analysis, Figure 5.16 shows a more detailed graph with the area under the curve for a specific scenario, Hospital Floor 2 configuration 2A, comparing between partitioned versus uncoordinated multi-robot for a specific area and radio configuration.

Table 5.2: Area under RMSE curves, averaged across all maps and radio configurations, to compare the uncoordinated and coordinated (partitioned) information adaptive approaches.

	dI-Adaptive Uncoordinated	dI-Adaptive Partitioned	Improvement
1 Robot	4160.63	n/a	
2 Robots	2897.58	2076.17	0.717
3 Robots	2321.25	1495.61	0.644
4 Robots	2017.46	1404.52	0.696

From this result we see that overall modeling performance improves when we coordinate multiple robots through the partitioning of the sampling space. This improvement increases with the number of robots, showing the biggest improvement for the simulation with four robots. In addition, we explore the performance when operating under exact partitioning or over partitioning.

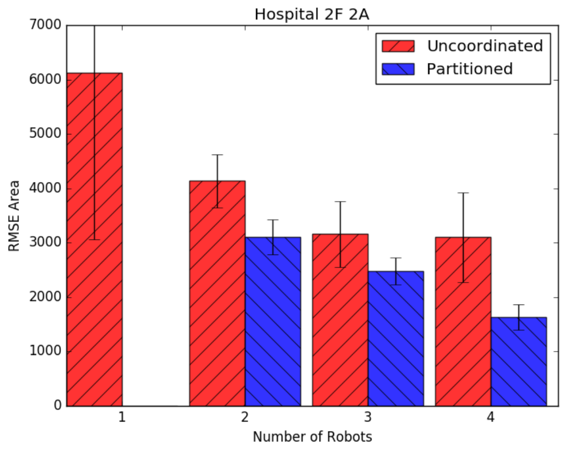


Figure 5.16: Comparison of the area under the curve between partitioned vs. uncoordinated results, on Hospital Floor 2, configuration 2A, for 1 to 4 robots.

Under this series of simulation experiments, Figure 5.17 shows that the coordinated (blue) approach generally performs better than the uncoordinated (red) multi-robot approach.

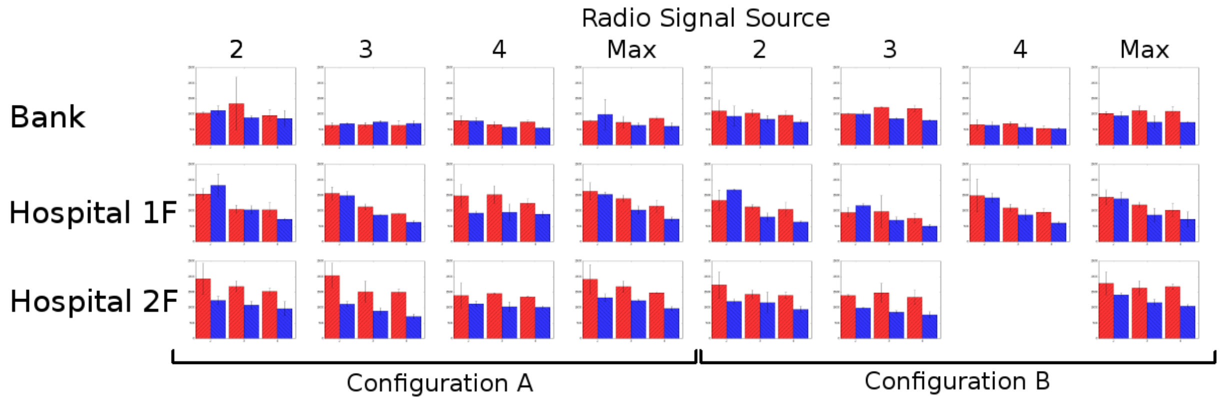


Figure 5.17: Montage of area under the RMSE curve results, across all environment configurations. Red bar plots are for equal number of partitions and robots, and blue bars are for the over segmented paritioning and task assignment. For each subgraph, the x axis shows the results for 2, 3, and 4 robots.

Live robot experiments

The system was then implemented on Jackal robots as seen in Figure 5.18. Experiments were performed on a number of different indoor environments on the campus of the University of California, San Diego.

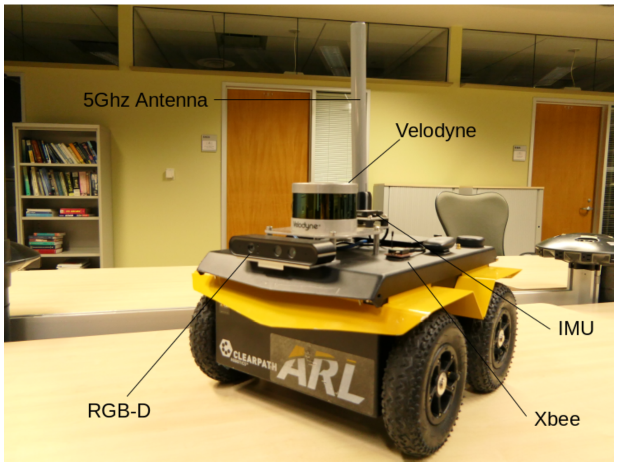


Figure 5.18: The Jackal robot used to gather RSSI data, indicating its sensors. The RSSI data was used to build the ground truth GP models.

Three radio signal generators were stationed through each environment and the robots were tasked from a central control station. We performed experiments with one, two, and three robots operating simultaneously. We gathered data comparing uncoordinated tasking of the robots to coordination with exact or over partitioned segments of the environment.

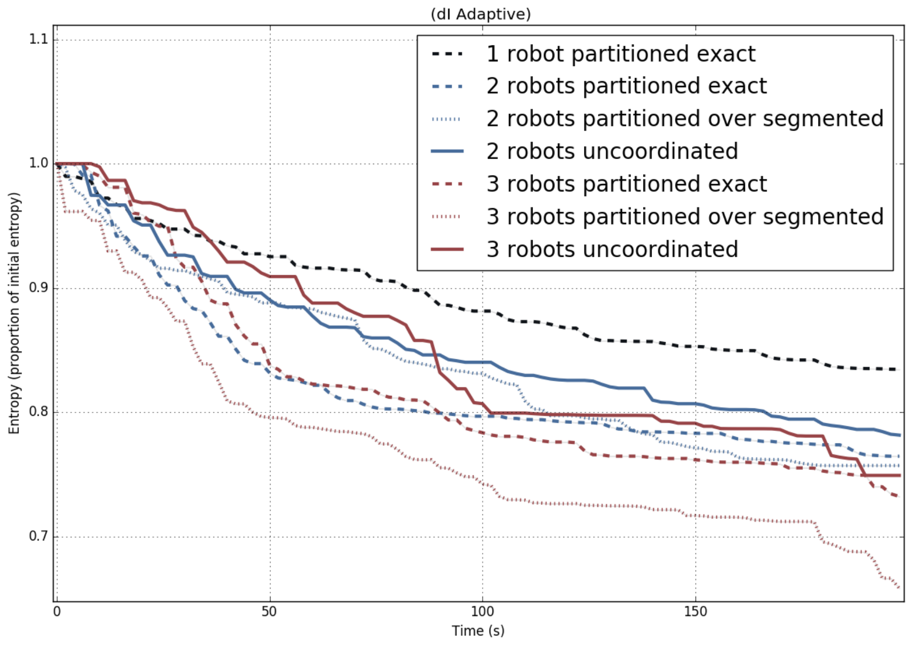


Figure 5.19: Comparison between 2 or 3 robots under uncoordinated tasking, tasking using exact partitioning, and tasking under over environments using information rate adaptive sampling for the first floor of Atkinson Hall (Figure 5.20) and the Computer Science & Engineering Building

The system was fielded on groups of up to three Jackal robots (see Figure 5.18) and experiments were run on the campus of the University of California, San Diego. Similar to the simulation experiments, radio transmitters were placed within the indoor environment and the robots attempted to build a GP model of the RSSI from the transmitters. Figure 5.19 shows the results from Atkinson Hall, 6th floor experimentation comparing the entropy of the environment over time under uncoordinated, exact, and over segmenting partitions strategies.

We see that with 3 robots, over segmenting resulted in a marked improvement in performance. We can see the partitioning for exact and over segmenting for the Atkinson Hall, 6th floor environment in Figure 5.20.

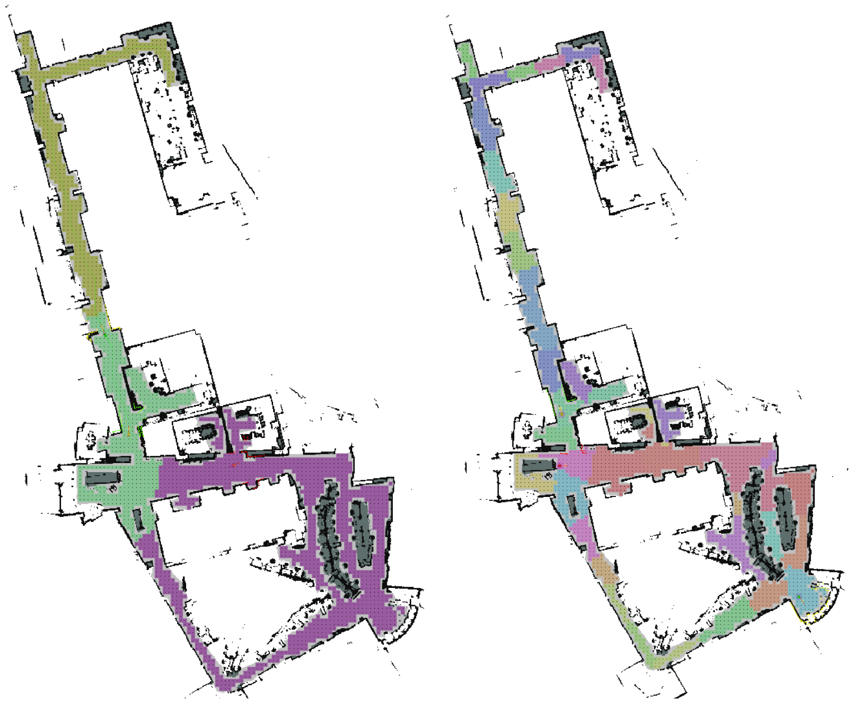


Figure 5.20: Comparison between exact and over segmenting partitions for 3 robots on the 6th floor of Atkinson Hall on the campus of the University of California, San Diego.

5.3.4 Discussion

We described a multi-robot coordination approach for adaptive informative sampling in structured environments. We utilized entropy reductions as an information gain metric and implemented a utility function designed to maximized expected information gained per distance travelled. We paired this utility with coordination through region segmentation for teams of robots and verified the approach through simulation and real world experimentation. Experimental data was gathered using teams of 1 to 4 robots, both without and with our coordination approach. We show that in simulation and in our initial real world testing, the coordination through region partitioning results in a faster rate of information gain reflecting in lower entropy in our GP model. Thus we have demonstrated the potential of our multi-robot coordination algorithm.

Acknowledgment

This chapter appears in the following publications:

- Nicholas Fung, John G. Rogers, Carlos Nieto-Granda, Henrik Christensen, Stephanie Kemna, and Gaurav sukhatme. Coordinating multi-robot systems through environment partitioning for adaptive informative sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, Canada, May 20-24, 2019.

- Carlos Nieto-Granda, John G. Rogers, Nicolas Fung, Stephanie Kenma, Henrik I. Christensen, and Gaurav Sukhatme. On-line coordination tasks for multi-robot systems using adaptive informative sampling. *In 2018 International Symposium on Experimental Robotics (ISER)*. Buenos Aires, Argentina, November 5-8, 2018.

- Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav Sukhatme. Multi-robot task allocation for collaborative adaptive informative sampling in structured environments. workshop on informative path planning and adaptive sampling. *In the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia, May 21-25, 2018.

- Stephanie Kemna, John G. Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav S. Sukhatme. Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments. In the IEEE International Conference on Robotics and Automation (ICRA). Singapore, May 29-June 3, 2017.

- Nick Fung, Carlos Nieto-Granda, Jason Gregory, and John Rogers. Autonomous exploration using an information gain metric. *In US Army Research Laboratory (ARL-TR-7638)*, Adelphi, MD. United States, March 2016.

I would like to thank my co-authors John G. Rogers III, Stephanie Kenma, Stuart Young, Jason Gregory, Nick Fung, Prof. Henrik I. Christensen, and Prof. Gaurav Sukhatme.

Chapter 6

Distributed Systems for Robust Exploration

6.1 Introduction

Search and rescue robots often need to find the source of a signal like radio-active material, heat signature, or gas leak. This problem is known as stochastic source seeking. To plan an optimal exploration trajectory, many researchers [ALP14] have suggested using information gain metrics like Mutual Information. However, in partially observable environments computing information gain of a trajectory becomes challenging if parts of the map are unknown.

Stochastic source seeking has been typically addressed in fully observable environments, with a single target [ALP14]. On the other hand, exploration in partially observable environment [Cho+17b] has dealt with exploration without a source seeking objective. However, the existing approaches that depend upon fully observable state do not generalize to all kinds of maps and obstacles. Specifically, we consider a U-maze (Fig 6.1), which has local optima with respect to greedy source seeking. The work closest to our approach is Shreshta et al. [Shr+19]. However, they attempt to predict large regions in the map, which are unlikely to generalize to new maps. Moreover, Shreshta et al. [Shr+19] use a flood-fill over the predicted map to estimate the information gain for each action. Flood filling can be brittle and lead to misleading information gains with even small holes in the map walls.

In this chapter, we present our state of the art architecture for multi-robot source seeking visual recognition. Our approach has been tested in simulation and real experiments at the University of California San Diego. Besides that, we provide a mathematical model of the problem and an algorithm solution to the source seeking problem.

6.2 Distributed Heterogeneous Multi-robot source seeking using information based sampling with visual recognition

6.2.1 Approach

We deployed a heterogeneous multi-robot team with UGVs and UAVs, whose mission is to locate, identify, and neutralize (defuse) an unknown number of IEDs hidden in the environment. The robots can measure radioactivity emitted by the IEDs. Besides that, the robots navigate and perform source seeking the task autonomously. The robots can request or provide service to use all the available resources to minimize the time to defuse all the IEDs. Also, each robot can negotiate with the other members of the team and re-compute the exploration regions.

6.2.2 Stochastic source seeking

Stochastic source seeking is the problem of searching a stochastic source $y_i \in \mathcal{Y}$ of a signal in an unknown environment $m \in \mathcal{M}$, using signal $z_{i,t} \in \mathbb{R}$ at time t when robot is at pose $x_t \in \mathcal{X}$. The problem of stochastic source seeking is typically addressed either using model-based or model-free methods. In this work, we focus on model-based methods because the many signal sources, like radio-activity, gas-leak, and heat, can be easily modeled or approximated.

Model-based methods assume that the signal model $\phi : \mathcal{Y} \times \mathcal{X} \times \mathcal{M} \to \mathbb{R}$ is given. We further assume that the signal sources can be in two states, active or inactive. Let $a_{i,t} = 1$ denote that the ith source is active. Let the signal sensed by the robot is given by:

$$z_t = \sum_{i=1}^{N} a_{i,t}\phi(y_i, x_t, m), \tag{6.1}$$

where $a_{i,t} \in \{0,1\}$ denotes whether the target i is active. A transition model is also known for the activation states of the signal-sources $a_{i,t+1} = g(a_{i,t}, x_t, m, u_t)$.

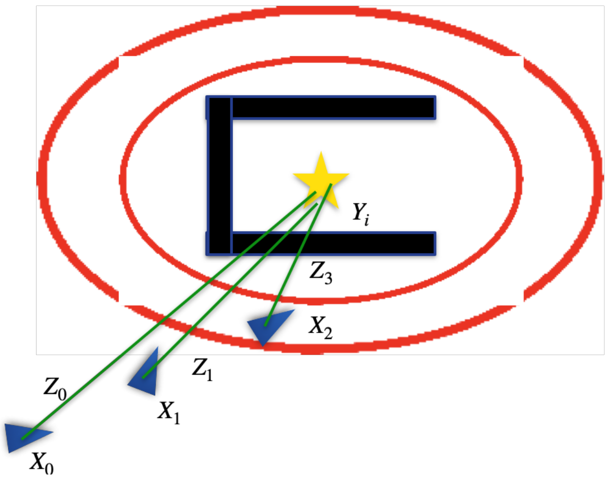


Figure 6.1: Our algorithm has a dual objective of exploration and source seeking. The robot is represented with a blue triangle and the target (source) is represented with a yellow star.

Problem (Model-based source seeking)**.** *Consider N targets, whose locations* $\{y_i\}_{i=1}^{N}$ *are unknown. We are given a robot with pose* x_t *that moves according to a given dynamics model* $x_{t+1} = f(x_t, u_t, m) + \eta_t$. *The problem is finding the policy* π *that deactivates all the targets in* *minimum time:*

$$\begin{aligned}
\pi^* &= \arg\min_{\pi} T \\
s.t \quad a_{i,T} &= 0 \quad \forall i \in \{1,\ldots,N\} \\
a_{i,t+1} &= g(a_{i,t}, x_t, m, u_t) \\
x_{t+1} &= f(x_t, u_t, m) + \eta_t \\
z_t &= \sum_{i=1}^{N} \phi(x_t, y_i, m) + \varepsilon_{i,t} \\
u_t &= \pi(z_{1:t})
\end{aligned} \tag{6.2}$$

6.2.3 Information gathering

Information gathering is the problem of exploring an unknown environment with the objective of collecting maximum information about the environment map. Given a robot with pose x_t at time t is present in an unknown environment m and observes lidar scans $l_t = \psi(x_t, m) + \omega_t$ according to a known observation model ψ and noise ω_t. Also assume that the dynamics of the robot are known.

Problem (Information Gathering)**.** *The Information Gathering problem is the problem of seeking to maximize the information about the unknown map m using the observations $l_{1:t}$:*

$$\begin{aligned}
\pi^* &= \arg\min_{\pi} \mathbb{H}(m \mid l_{1:t}, x_{1:t}) \\
s.t \quad x_{t+1} &= f(x_t, u_t, m) + \eta_t \\
l_t &= \psi(x_t, m) + \omega_t \\
u_t &= \pi(l_{1:t}),
\end{aligned} \tag{6.3}$$

where $\mathbb{H}(m \mid l_{1:t}, x_{1:t})$ is an information measure over the map m. Typically it is the Shannon entropy of the map random variable.

In this work, we combine the two problems, Problem 6.2.2 and Problem 6.2.3, to address them jointly. We also extend this problem to a heterogeneous multi-robot setting.

6.2.4 Distributed Heterogeneous Multi-Robot Source Seeking (DHMRSS)

We consider pairs of Heterogeneous robots with each pair containing a ground robot and an aerial vehicle. The ground robots have a large battery back-up and can cover large areas on the ground, while aerial vehicles that ride on top of ground robots have short battery back-up. However, they can reach places were ground robots cannot.

State transition model

Consider K pairs of robots with pose of the kth ground robot at time t denoted as $x_{k,t} \in \mathcal{X}_k$. Let the a binary variable $\chi_{k,t} \in \{0,1\}$ denote where the ground robot is carrying the aerial vehicle $\chi_{k,t} = 0$ or the aerial vehicle is flying $\chi_{k,t} = 1$. Let the pose of the aerial vehicle be denoted by $x^{(a)}_{k,t}$. Let the dynamics of flying aerial vehicle be $f^{(a)}_k(.)$ and the control signal be $u^{(a)}_{k,t}$. The dynamics of aerial vehicle is then given by:

$$x^{(a)}_{k,t+1} = \chi_{k,t} f^{(a)}_k(x^{(a)}_{k,t}, u^{(a)}_{k,t}, m) + (1-\chi_{k,t}) f_k(x^{(a)}_{k,t}, u_{k,t}, m) \tag{6.4}$$

$$\chi_{k,t+1} = \big(1 - [\|x^{(a)}_{k,t} - {}^a T_g x_{k,t}\| < \delta]\big)\chi_{k,t} + [\|x^{(a)}_{k,t} - {}^a T_g x_{k,t}\| < \delta] u^{(\chi)}_{k,t} + \eta^{(a)}_{k,t} \tag{6.5}$$

where $f_k(.)$ is the dynamics of the ground robot, u_k is the control signal for ground robot, $[\|x^{(a)}_{k,t} - {}^a T_g x_{k,t}\| < \delta]$ denotes whether the aerial vehicle is close enough to the ground robot to land and $u^{(\chi)}_{k,t}$ denotes control signal to land.

We consider an activation transition model that depends upon an object-detection algorithm, $\text{ObjDet}_i : I \to \{0,1\}$, where I is the space of Image input and the algorithm returns 1 on detecting the source and 0 otherwise. When the robot has detected the target object

$\text{ObjDet}_i(I_{k,t}) = 1$, then the target object i gets diffused automatically and it is no longer active $a_{i,t+1} = \min\{1 - \max_{k=1}^{K} \text{ObjDet}_i(I_{k,t}), a_{i,t}\}$. With slight abuse of notation, we include the image observation model as a part of object detection function $o_{i,k,t} = \text{ObjDet}_i(x_{k,t}, y_i, m)$ and write the updated model of activeness of target object as $a_{i,t+1} = \min\{1 - \max_{k=1}^{K} o_{i,k,t}, a_{i,t}\}$.

In this work, we consider a partially observable setting where we estimate map and robot pose using a typical SLAM pipeline that uses Bayes Filter to update the distribution over both the map m and the ground robot pose $x_{k,t}$,

$$\begin{aligned} bel_{t+1}(m, x_{k,t+1}) &:= p(m, x_{k,t} | l_{k,1:t}, u_{k,1:t-1}) \\ &= \frac{1}{Z} \mathbb{E}_{m, x_{k,t+1}} \left[p(l_{t+1} | m, x_{k,t+1}) p(x_{k,t+1} | x_{k,t}, u_{k,t}, m) \right] \in Bel, \end{aligned} \tag{6.6}$$

where $bel_0(m, x_{k,0}) := p(m, x_{k,0})$ is given as a prior over the map and initial robot location, Bel is the belief space and Z is the normalizing factor.

The full transition model can then be written as:

$$\underbrace{\begin{bmatrix} a_{i,t+1} \\ x_{k,t+1} \\ x^{(a)}_{k,t+1} \\ \chi_{k,t+1} \\ bel_{t+1}(m, x_{k,t+1}) \end{bmatrix}}_{s_{k,t+1}} = \underbrace{\begin{bmatrix} \min\{\max_{k=1}^{K} o_{i,k,t}, o^{(a)}_{i,k,t}, a_{i,t}\} \\ f_k(x_{k,t}, u_{k,t}, m) + \eta_{k,t} \\ \chi_{k,t} f^{(a)}_k(x^{(a)}_{k,t}, u^{(a)}_{k,t}, m) + (1 - \chi_{k,t}) f_k(x^{(a)}_{k,t}, u_{k,t}, m), \\ \left(1 - [\|x^{(a)}_{k,t} - {}^{a}T_g x_{k,t}\| < \delta]\right) \chi_{k,t} + [\|x^{(a)}_{k,t} - {}^{a}T_g x_{k,t}\| < \delta] u^{(\chi)}_{k,t} + \eta^{(a)}_{k,t} \\ \frac{1}{Z} \mathbb{E}_{m, x_{k,t}} \left[p(l_{k,t+1} | m, x_{k,t+1}) p(x_{k,t+1} | x_{k,t}, u_{k,t}, m) \right] \end{bmatrix}}_{\mathbf{F}(s_{k,t}, m, u_{k,t}, u^{(a)}_{k,t}, u^{(\chi)}_{k,t})} \tag{6.7}$$

Observation model

Each robot captures three kinds of observations: radio signal, object detections through vision sensor, and Lidar observations. First observation is the scalar radio-signal $z_{k,t} \in \mathbb{R}$, which is unaffected by the map. The second kind of observations are object detections using vision $o_{i,k,t} = \text{ObjDet}(x_{k,t}, y_i, m)$. The third is Lidar observations $l_{k,t}$ which help us detect obstacles run SLAM algorithm to estimate belief over map and pose $bel_t(m, x_{k,t})$. We can write the observation model as:

$$\underbrace{\begin{bmatrix} z_{k,t} \\ l_{k,t} \\ o_{i,k,t} \\ o^{(a)}_{i,k,t} \end{bmatrix}}_{\zeta_{k,t}} = \underbrace{\begin{bmatrix} \sum_{i=1}^{N} a_{i,t}\phi(x_{k,t}, y_i, m) + \varepsilon_{i,t} \\ \psi(x_{k,t}, m) + \omega_t \\ \text{ObjDet}(x_{k,t}, y_i, m) \\ \text{ObjDet}(x^{(a)}_{k,t}, y_i, m) \end{bmatrix}}_{\mathbf{O}(s_{k,t}, m)} \tag{6.8}$$

Communication model

We assume that the robot k can communicate with neighboring $B(x_{k,t}, \mathbf{x}_t)$ robots, receiving incoming messages $\alpha_{k,b,t} \in \mathcal{A}$ from robot $b \in B(x_{k,t}, \mathbf{x}_t)$ and sending broadcast outgoing messages as $\beta_{k,t} \in \mathcal{B}$. Let $\boldsymbol{\alpha}_{k,t} = \{\alpha_b\}_{b \in B(x_{k,t}, \mathbf{x}_t)}$ denote all incoming messages at time t for kth robot.

Problem (DHMRSS). *Under the transition model* (6.7) *and observation model* (6.8)*, we want to design a policy* $\pi_k : \mathcal{Z}^t \times \mathcal{A}^{Kt} \to \mathcal{U} \times \mathcal{U}^{(a)} \times \{0,1\} \times \mathcal{B}$ *for each robot* $k \in \{1, \dots, K\}$*. The kth robot takes an action* $u_{k,t}, u^{(a)}_{k,t}, u^{\chi}_{k,t}, \beta_{k,t} = \pi_k(\zeta_{k,1:t}, \boldsymbol{\alpha}_{k,1:t})$ *at each time step in order to* *deactivate all the signal sources in least time possible:*

$$
\begin{aligned}
\{\pi_k^*\}_{k=1}^K = \arg \min_{\{\pi_k\}_{k=1}^K} & \; T \\
s.t \quad a_{i,T} &= 0 \quad \forall i = \{1,\dots,N\} && \textit{// Terminal condition} \\
B_k &\geq \sum_{t=1}^{T-1} \chi_{k,t} && \textit{// Battery constraint} \\
s_{k,t+1} &= \mathbf{F}(s_{k,t}, m, u_{k,t}, u_{k,t}^{(a)}, u_{k,t}^{(\chi)}) && \textit{// State transition model} \\
\zeta_{k,t} &= \mathbf{O}(s_{k,t}, m) && \textit{// Observation model} \\
u_{k,t}, u_{k,t}^{(a)}, u_{k,t}^{(\chi)}, \beta_{k,t} &= \pi_k(\zeta_{k,1:t}, \boldsymbol{\alpha}_{k,1:t}) && \textit{// policy}
\end{aligned}
\tag{6.9}
$$

where B_k is the battery constraint for each of the aerial vehicle and $\hat{y}_{i,t}$ is the current estimate of the location of the target.

Note that Problem 6.2.4 is a combination of Problem 6.2.2 and Problem 6.2.3, but the objective of information gathering is implicit in the problem. Since we want to defuse *all* the signal-sources, we need to find all of them which requires exploration of the environment. We consider two specific instantiations of $bel_0(s_0)$ in Problem 6.2.4: one with uniform prior over the map space, and another with given map of the building but unknown locations of the furniture and obstacles.

Known vs Unknown map

In these experiments, we tested both scenarios:

a) The robots navigate a known environment.

b) The robots explore and navigate in an unknown environment.

```
Input: Robot state: $s_{k,t}$, Robot observation $\zeta_{k,t}$, Input messages $\boldsymbol{\alpha}_{k,t}$
Output: $u_{k,t}, u^{(a)}_{k,t}, u^{(\chi)}_{k,t}, \beta_{k,t}$
if $z_{k,t} \leq -70 dBm$ then
    $u_{k,t} \leftarrow$ Problem 6.2.3 ;          /* Explore for information gathering */
else
    $\hat{y}_{i,t}, \sigma_{i,t} \leftarrow$ EstimateSignalSourceLoc($z_{k,1:t}, x_{k,1:t}$)
    issuccess,$\hat{x}_{k,t+1:t+m} \leftarrow$ PlanTrajToTarget($x_{k,t}, \hat{y}_{i,t}$)
    $t_{\text{leaving}} \leftarrow t$ ;                    /* Record the time when leaving  */
    while issuccess AND $t < t_{leaving} + 1\ min$ do
        $u_{k,t} \leftarrow$ PathFollowing($\hat{x}_{k,t+1:t+m}, x_{k,t}$)
        if $ObjDet_i(I_{k,t}) = 1$ then
            $a_{i,t} = 0$ ;                          /* Deactivate signal source */
            break
        end
    end
    /* Ground robot needs drone for help                                 */
    $u^{(\chi)}_{k,t} \leftarrow 1$ ;                                        /* Take off */
    if issuccess then
        /* Ground robot timed out.                                       */
        $\hat{x}^{(a)}_{k,t+1:t+m} \leftarrow$ PlanExploratoryTrajectory($x^{(a)}_{k,t}, \hat{y}_{i,t}, \sigma_{i,t}$)
    else if MapRegionHasUnknownVoxels($bel_t(m, x_{k,t}), \hat{y}_{i,t}, \sigma_{i,t}$) then
        $\hat{x}^{(a)}_{k,t+1:t+m} \leftarrow$ PlanExploratoryTrajectory($x^{(a)}_{k,t}, \hat{y}_{i,t}, \sigma_{i,t}$)
    else
        issuccess,$\hat{x}^{(a)}_{k,t+1:t+m} \leftarrow$ PlanTrajToTargetAndBack($x^{(a)}_{k,t}, \hat{y}_{i,t}$)
    end
    Send desired trajectory $\hat{x}^{(a)}_{k,t+1:t+m}$ to aerial vehicle
    /* Aerial vehicle follows trajectory                                 */
    for $r \in \{t, \ldots, t+m-1\}$ do
        $u^{(a)}_{k,r} \leftarrow$ PathFollowing($\hat{x}^{(a)}_{k,r+1}, x_{k,r}$)
        if $ObjDet_i(I^{(a)}_{k,t}) = 1$ then
            $a_{i,t} = 0$ ;                          /* Deactivate signal source */
        end
    end
    $u^{(\chi)}_{k,t} \leftarrow 0$ ;                                /* Land when done */
end
```

Algorithm 7: Ground robot policy π_k to Identify and Defuse the Target

Known maps

As it is explained in Chapter 5, our technique Infosploration gives a solution of the partition of the map and how to manage the resources to look for the transmitters.

Unknown maps

In this case, the robots start together in a hallway and share their current position and a local map. Then, the robots move to specific frontiers and broadcast a message to the rest of the robots to avoid exploring the same regions. Besides, our coordination strategies *Divide and Conquer* is a trigger so the robots can explore in an efficient way of managing the exploration regions among the team members.

6.2.5 New Coordination Strategy

In [Nie+18], our robot team is capable of performing their tasks and accomplishing their mission. However, we did not take care of the management of resources. For example, when a robot finishes a task, it just continues mapping and navigating even if the assigned region for exploration was explored entirely. We extend our framework to provide the robots with the capability to broadcast two new messages: *a*) Task accomplished - Offering assistance. *b*) Requesting assistance to accomplish a task.

In our preliminary experiments, the time to explore a floor plan decreased comparing to our previous exploration strategy. However, the robots need to be able to communicate with the other members of the team. To enable the capability to communicate with the other robots even if one of them is far from the team, we developed a new module that allows the robot that finished the task to find other members. As we explained in Chapter 5, the segmentation of the environment is based on graph cuts. Therefore, we recompute the robots' initial knowledge to navigate in the trajectory or exploration regions that the other robots follow. One of the

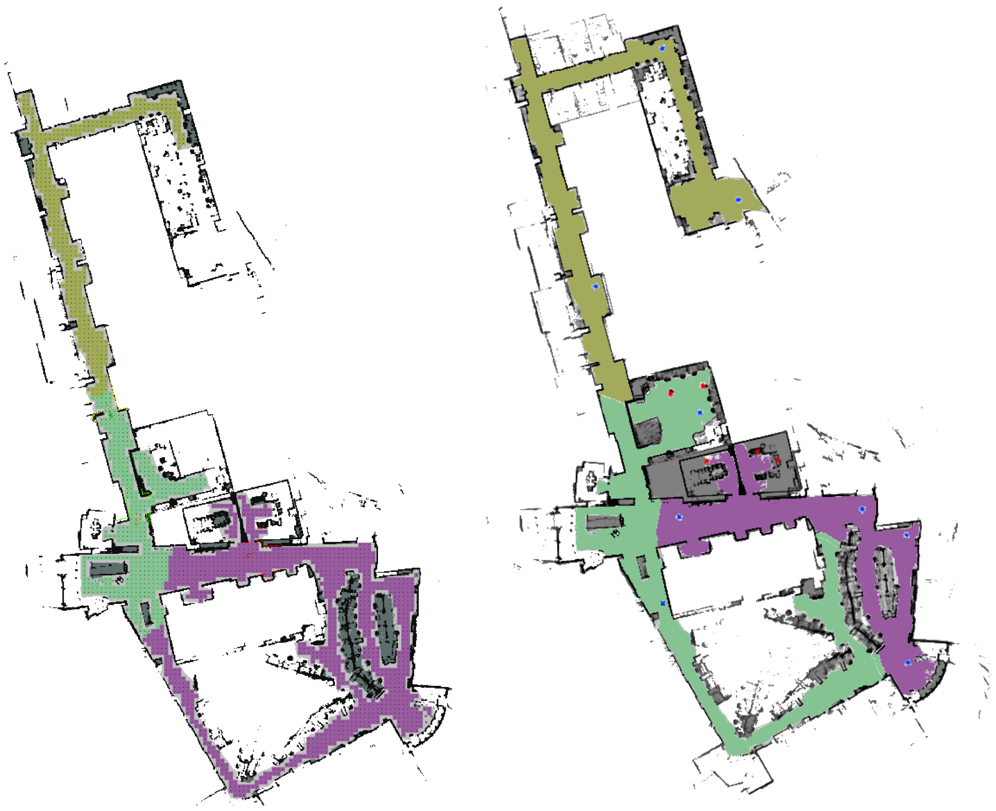


Figure 6.2: Comparison between the exact segmentation with the new dynamic reconfiguration after the robots agreed to explore regions from the other ones.

requirements to use this module is that the robot team has to know the map. The comparison between our previous and the current system is shown in Figure 6.2. In this scenario, the robot *A*, which exploration area is in color green, sent a broadcast message *Offering Assistance* to the robot *B* (exploration area color purple). Robot *A* and Robot *B* negotiated and recomputed their exploration areas, robot *A* continued exploring and searching for targets while robot *B* provides paths to the drone inside of the rooms.

6.2.6 Semantic Mapping: Adaptation of Gaussian Regions for Rooms and Hallways

In [Hea07], the US Marine Corps explained that common areas of IED emplacement include structure environments such as building, houses, and storage facilities. The explosive ordnance disposal team demonstrates that it is advantageous to know the rooms inside a building to predict the size and location of the explosive. In our architecture, we include this knowledge using our Place Categorization module explained in Section 3.4.4.

Shape Adaptation of Gaussian Regions

Our exploration module, called "Infosploration", explained in Section 5.3, allows the robots to explore an unknown environment using frontiers and identify rooms and hallways. We improve our Gaussian regions module to adapt to the shape of the rooms and hallways. This improvement allows the robots to label the rooms and enable a voxel exploration module called "Voxelsploration". The details of this application are found in [Nie19]. These regions are labeled by our place categorization system previously presented in Section 3.4.4.

Exploration Paths and Areas for Target Localization

In our previous work, Infosploration (Section 5.3 computes all the trajectories that the robot can travel considering the robot dynamics, power consumption, distance to the frontiers or regions, and the probability of the target's location. However, all the trajectories are computed in a 2D environment. Voxelsploration [Nie19] computes the paths in 3D, which allows us to have a heterogeneous team with different dynamics. Our system has been tested in 3 different platforms: packbots (capable of going upstairs), jackals (only navigate in one floor), and quadrotors (capable of navigating on different heights).

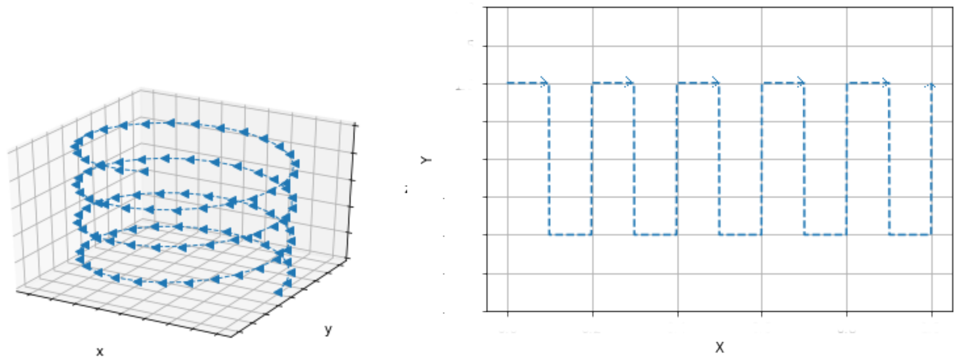


Figure 6.3: Visualization of two searching trajectories provided to the drone to localize a target.

Our new exploration module provides the capability to reach areas where the mobile robot cannot reach. Algorithm 8 explains the steps that our method follows to segment and decrease the search region on the environment. Figure 6.3 shows two different trajectories in which the UGV after navigating on the environment computes the probability of the target location and provides the search path that the drone has to follow to find the target.

```
Input: VoxelMap $region$, Robot Full Trajectory: $X_k$, Voxels Uncertaintiy: $tau$
if $UGV \longleftarrow \emptyset$ No reachable area then
    Compute a plane with high probability to find the target given by radio signal $z_t$;
    Create an exploration region enclosed of the computed planes $A_m$;
    if Robot Full Trajectory $X_k$ provides high values of radio signal $z_t$ then
        Map the trajectory $X_k$ to 3D and substract it as a polygon;
        from the $A_m \longrightarrow S_{area}$;
    end
    Run our planner to provides a trajectory that satisfy the full coverage of $S_{area}$;
end
Output: Searching Plan
```

Algorithm 8: Segmentation of Searching Areas

We have tested our framework on different floors and buildings from the University of California, San Diego. Each environment has a different level of complexity for the UGV and UAV navigation.

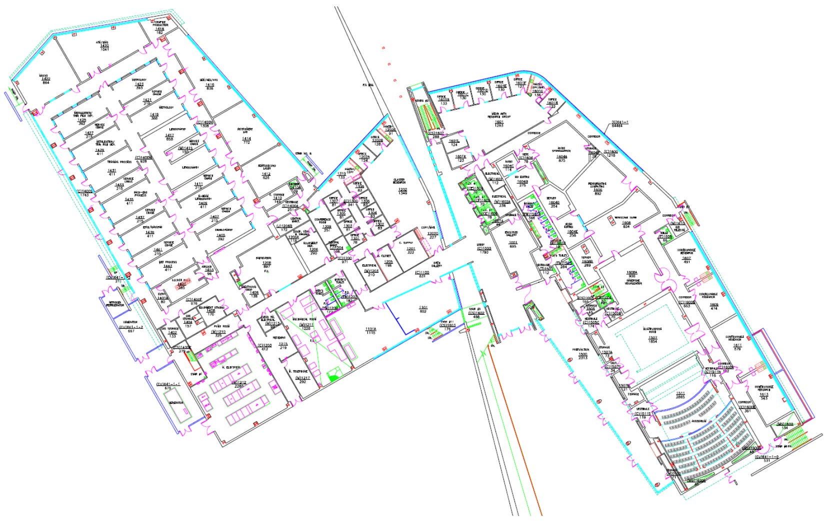


Figure 6.4: Floor plan of UCSD Atkinson Hall - 1st Floor.

In each experiment, the ZigXbees are tuned randomly, so the robot might listen to some from a far distance while others the robot has to be in the room to detect them. Besides, some of the rooms have obstacles that deny access to the UGV so it needs to request the assistance of the UAV. One of the environments that we explored is the Atkinson Hall First Floor. This scenario, shown in Fig 6.4, has different rooms with long hallways. Inside, most of the rooms have glass, and the semantic label provided is hallway, office, or lab. In this series of experiments, we only run the adaptable Gaussian Regions to determine if the robot was in another area different than a hallway. The size of this floor and the long hallways made a perfect exploration scenario. The communication among the robots is so constrained so that in most of the cases there is no communication.

6.2.7 Experiments

We deploy a team of ground robots (UGVs) paired with an aerial vehicle (UAV) to explore the environment. Each UGV runs its own mapping system (*Omnimapper*) and broadcasts messages among the team members. Each ground robot can listen to the radio signals. These signals are simulating radioactivity or gas emission in the area.

The robot behavior on this mission is the following:

```
Input: Robot pose: $X_k$, Radio signal: $z_t$
Output: Task
while $z_t = 0$ do
    Explore and look for targets
end
Find the location of the target
```

Algorithm 9: Robot tasks

When the task of the UGV is to find the target's location, it will navigate to the highest probability location of the signal and make visual identification of the target. However, there are many scenarios where the UGV will not be able to identify the target base on its dynamics, hardware, or reachability of the sensors. Then, the UGV will compute a trajectory with specific viewpoints that the UAV is requested to navigate to identify the target.

Simulation Experiments

We tested our framework in a virtual scenario using the ARL Unity-ROS-based simulation engine shown in figure 6.5. The robot team consisted of pairs of UGV, and UAV. Each team explored the environment looking for targets (a box with a sign in one of the faces).

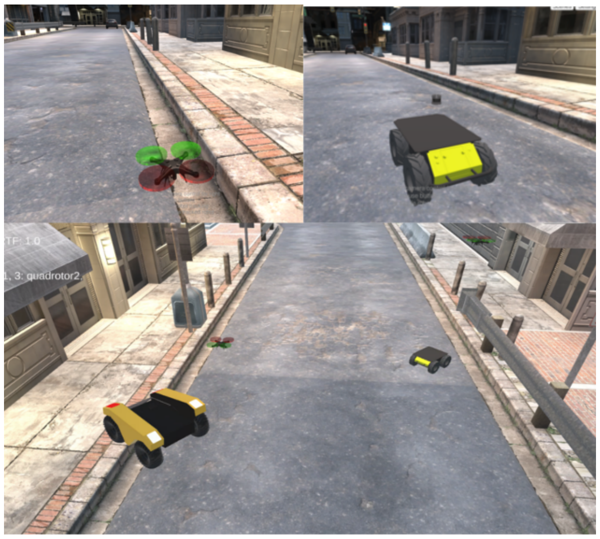


Figure 6.5: Our hetereogeneous UAV/UGV robot team used in the simulation experiments.

Each UGV is equipped with a Lidar and a camera, and each UAV has a RGB-d and monocular camera. The visualization of the trajectory of each UGV (blue and red), and each UAV trajectory (green) is shown in Figure 6.6. The IEDs are classified into two groups: only reached by UGV and only reached by UAV.

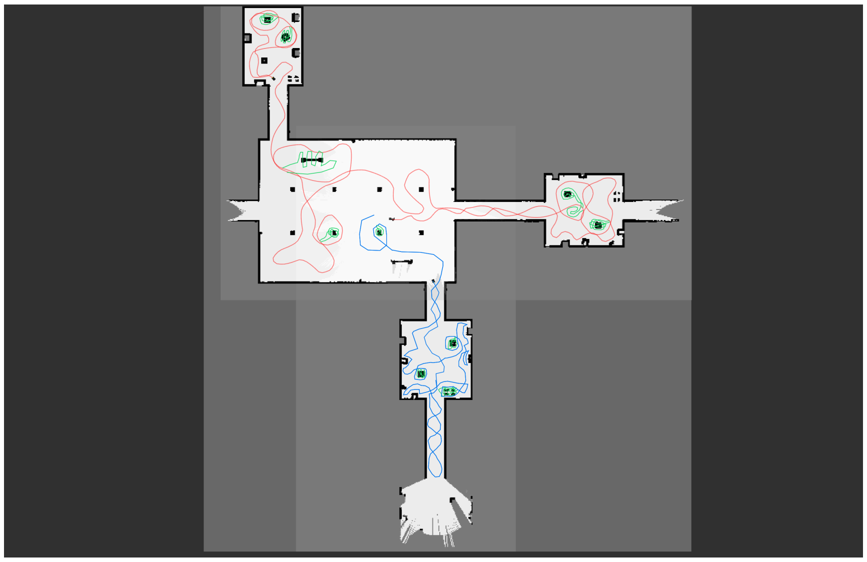


Figure 6.6: Visualization of the map explored in simulation by the UGV/UAV team.

Figure 6.7 shows how our framework using the new coordination strategy performed better than our previous approach. As expected, the new strategy keeps all the robots looking for the targets while they explore all the environment. The result is that entropy decreases faster.

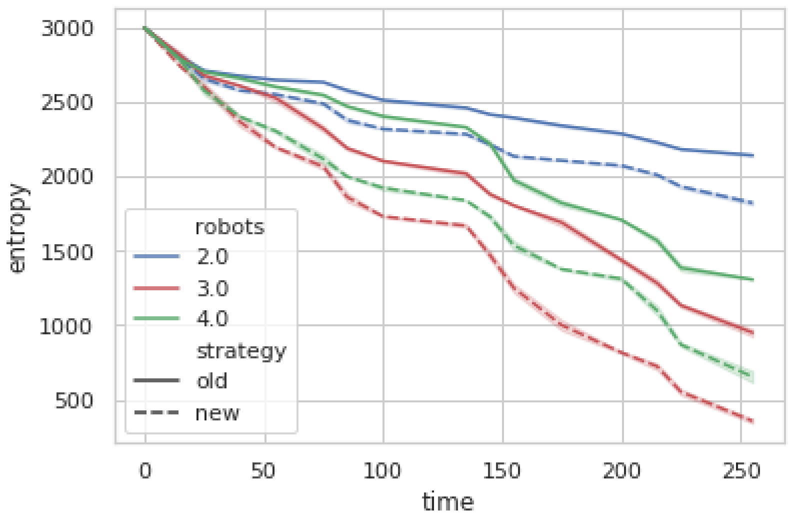


Figure 6.7: The result of running our exploration methods with and without the new coordination strategy module. The entropy decreases faster, because the robots are constantly exploring the environment.

Live Robot Experiments

A series of experiments were performed with a DJI drone which was augmented with a pozyx device and a Zigxbee, and 3 Clearpath Jackals (UGV), which were augmented with a 360 Velodyne sensor, an RGB-d camera, an IMU LORD Microstrain, a Ubiquiti 5ghz bullet antenna, a ZigXbee and a Pozyx device (to test different sensors for seeking RSSI), a GoPro Camera and an additional 2TB SSD for data storage. The UGVs are autonomous in these experiments. However, in the situation when the UGV requires assistance from the UAV, the UGV will stop for setting up the UAV (Figure 6.13).

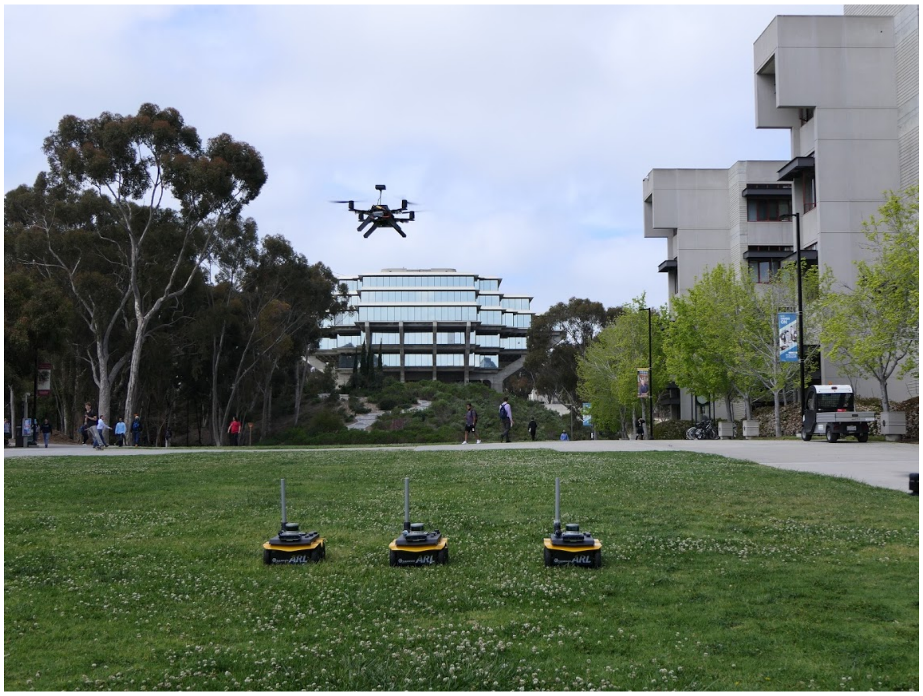


Figure 6.8: Our hetereogeneous UAV/UGV robot team used in these experiments.

System Architecture

Our framework needs to share information between the UAV and the UGV. Unfortunately, our UAV does not have computing power. However, we implemented a streaming node where the UAV transmits its video trough a phone and the phone to an Apple TV. The Apple TV is plugged to an external computer which processes all the visual exploration area and the target recognition. All the actions and the trajectories are generated on the computer. Only the UAV is manually teleoperated, flying to specific waypoints and patterns provided by our system.

Object of Interest - *IED*

In these experiments, we tried to simulate a dangerous situation where IEDs are hidden on the floor of a building. A team of autonomous robots has found, recognized, and defused the device. As for the device, we used a toy from the famous Super Mario Brothers called *Bob-omb*.

The *Bob-omb* has a particular shape as a bomb, this is shown in Figure 6.10 . To make

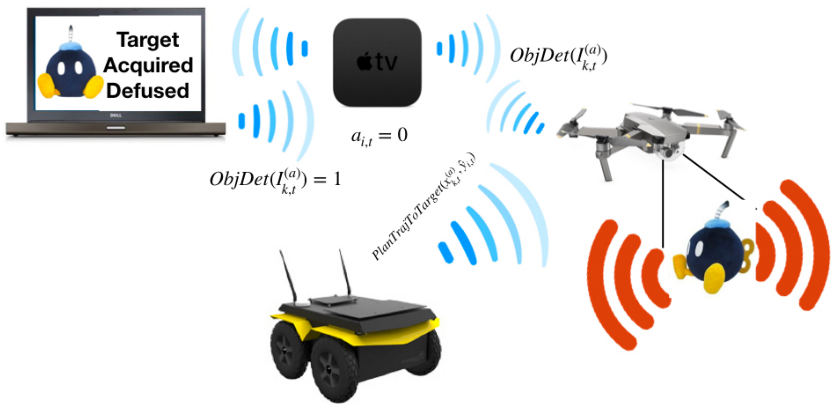

Target
Acquired
Defused
tv
$ObjDet(I^{(a)}_{k,t})$
$a_{i,t} = 0$
$ObjDet(I^{(a)}_{k,t}) = 1$
$PlanTrajToTarget(x^{(a)}_{k,t}, \hat{y}_{i,t})$

Figure 6.9: Simple drawing showing the interaction between the UGV and the UAV doing the source seeking and visual target identification.

visual recognition more complicated, we have to identify it by looking at his face. When the UGV or UAV looks at the front of the Bob-omb, the device is defused.

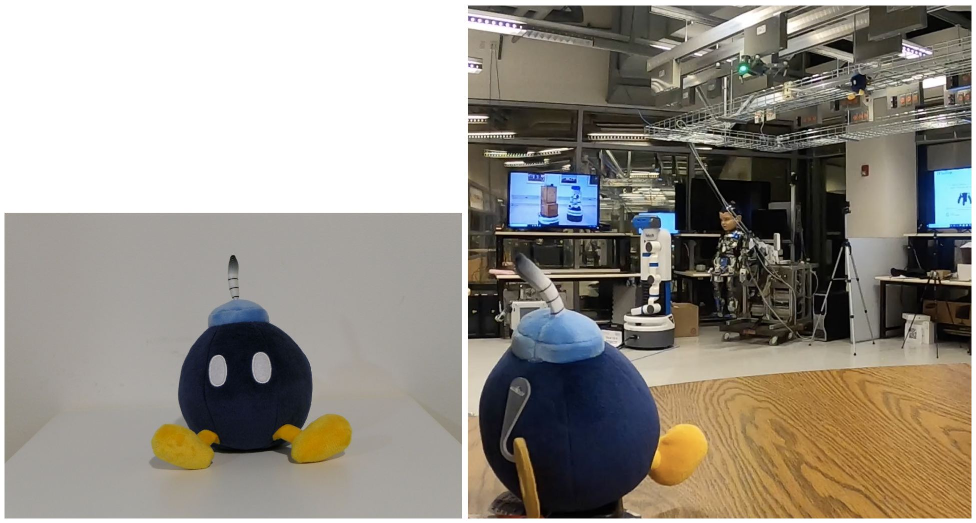


Figure 6.10: Our fake IED *Bob-omb*. The right image shows the location of two of the Bob-ombs at Atkinson Hall 1st Floor. Even that one of the Bob-ombs is hanging from the ceiling, both Bob-ombs can only be reached by an UAV.

The signal that we are simulating as radioactivity or other isotope measurement is provided by the Zixbee (Figure 6.11) and Pozyx. We deployed eleven bob-ombs in different places on the floor. The robots did not know the location of the devices. Therefore, the team had to explore all the buildings.

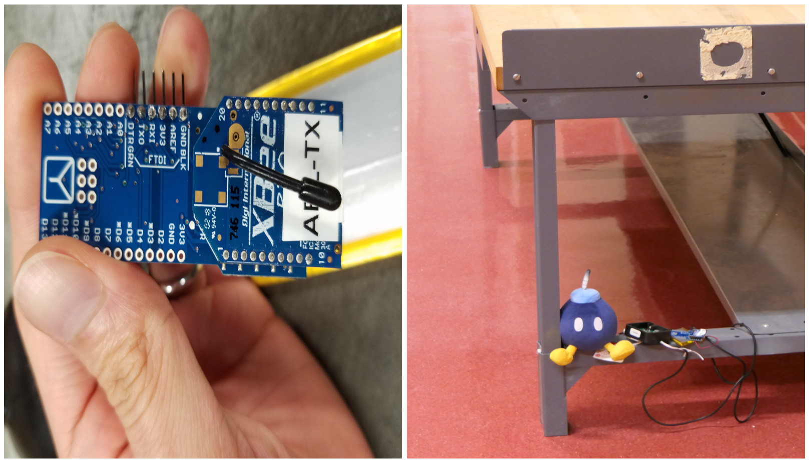


Figure 6.11: Our Arduino with Zigxbee and Psyx devices used for radioactivity emulation from the IED.

6.2.8 Results

The main mission of this work was to locate, identify, neutralize threats on the environment. In this case, we simulated explosives installed and hidden on different floors of our university buildings. We are interested in optimizing the time that the robot team would take to accomplish the task. We ran our exploration and source seeking system in each UGV on board. Each UGV was autonomous and had a UAV that was teleoperated following the path planning computed and visualized by the operator. However, our system is capable sending navigation commands to a UAV with a computer onboard. We deployed the multi-robot UGV/AUV team on the following scenarios:

Atkinson Hall - 1st Floor

This scenario consists of big hallways, two rooms, and a big atrium. Unfortunately, in this scenario, our place categorization only provides two labels. Therefore, it was not relevant to use it to label the exploration regions.

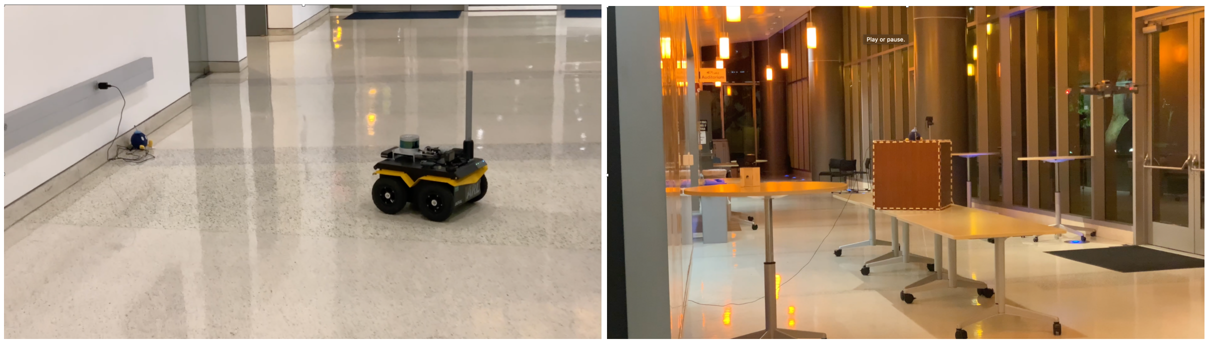
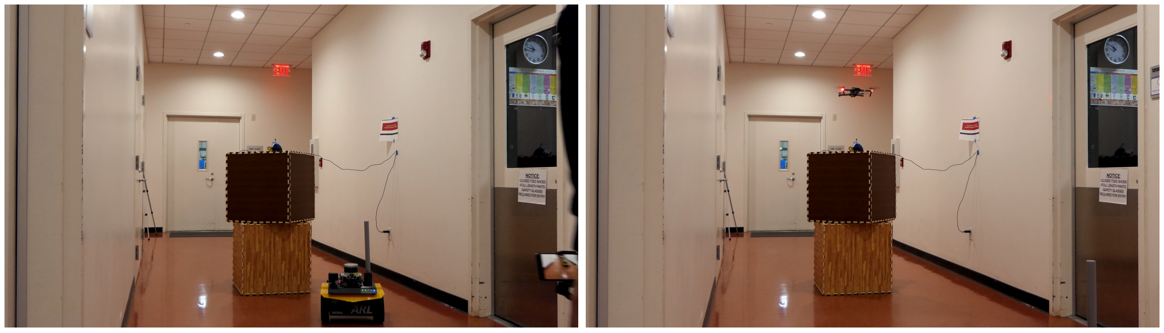


Figure 6.12: Our robot team exploring and defusing IEDs at Atkinson Hall 1st Floor.

Figures 6.12 and 6.13 shows one UGV and UAV exploring Atkinson Hall 1st Floor. We can observe that the UGV finds the IED in the hallway, and the drone flies over the table and the cube to localize the other IED.

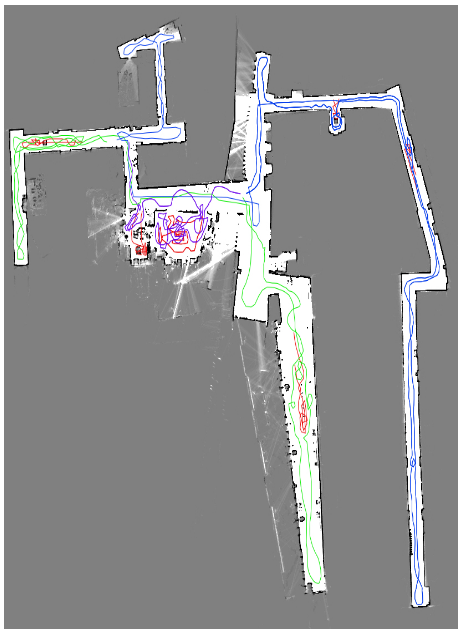

(a) UGV looking for the fake IED which is impossible for it to reach it.

(b) After the UGV failed. The UAV assistance is required and it flies with the trajectory provided by the UGV.

Figure 6.13: Source seeking scene where the UGV is not capable to reach the fake IED and requests the assistance of the UAV.

The visualization of the trajectory of each UGV (blue, green, and purple), and the UAV trajectory in red is shown in Figure 6.14. The IEDs are classified into two groups: only reached by UGV (blue star) and only reached by UAV (red star).

In this experiment, it is clear that the robot with a blue trajectory explored more than the other two. The reason is that the exploration with the UAV was faster than the other two robot teams. We compute the entropy as a metric to guarantee that the robot team explores all the


(a) Full robot trajectories from the 3 teams UAV/UGV. The UAV trajectory is shown in red and the rest of the robots in green, blue and purple.

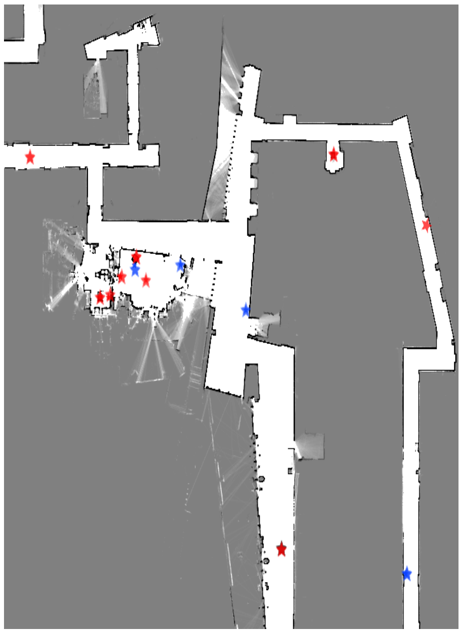

(b) The targets that can only be reached and identified using a UAV are shown with a red star, and the rest are only reached by the UGVs are shown with a blue star.

Figure 6.14: Atkinson Hall 1st Floor explored with drone trajectories and targets.

floor map while at the same time is defusing the IEDs. The comparison of our framework with and without the new coordination strategy is shown in Figure 6.15. As expected, the entropy decreases faster when we use the new coordination strategy.

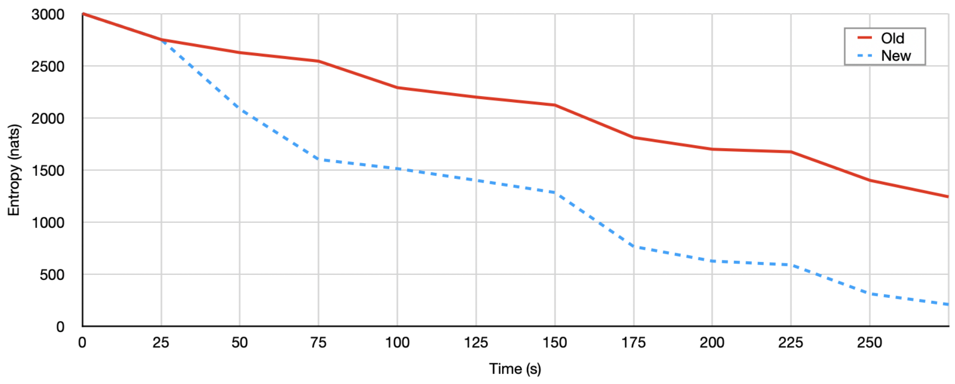


Figure 6.15: A comparison of the entropy over time of the robot team at Atkinson Hall 1st floor.

Atkinson Hall - 6th Floor

This experiment was performed on the 6th Floor of Atkinson Hall. The floor map consists of hallways, and offices with different dimensions. The targets were set up on the top of the tables where the robot has to avoid various obstacles during the exploration.

This scenario is entirely semantically labeled using DEDUCE (explained in section 3.4), as is shown in Figure 3.18. Figure 6.17 shows the trajectory of each UGV (red, blue, and green), and the UAV trajectory (purple).

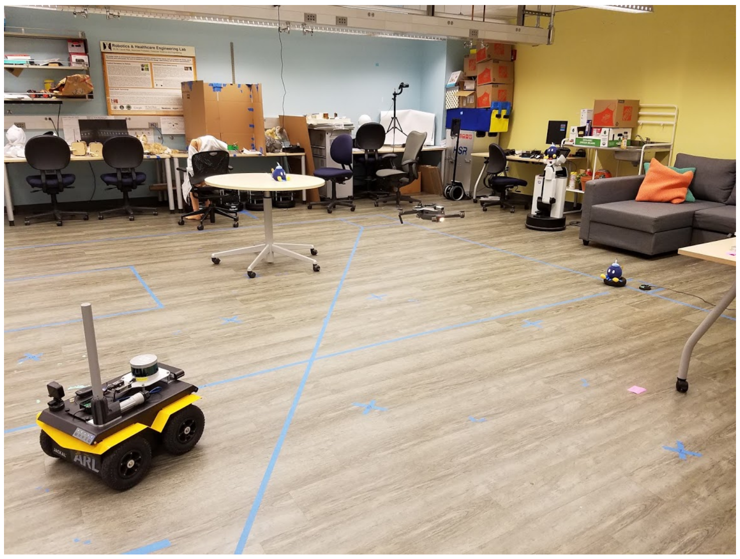
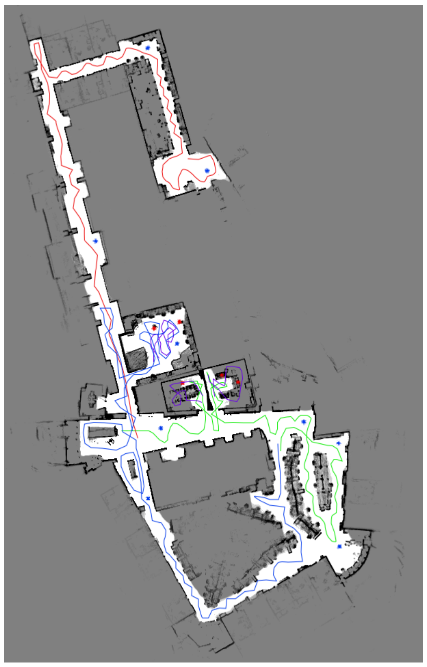


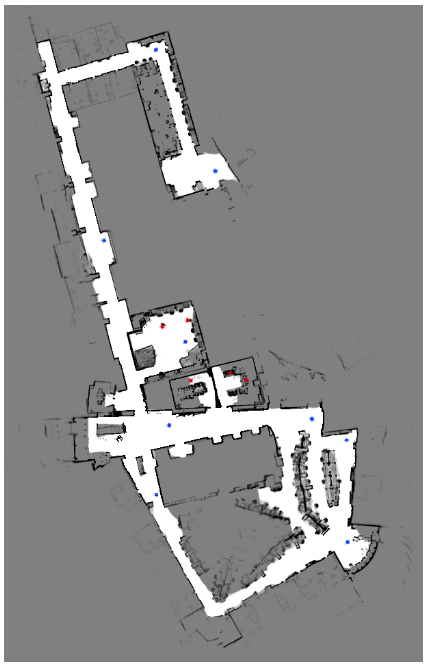
Figure 6.16: Our UGV/UAV exploring and defusing IEDs at Atkinson Hall 6th Floor.

The IEDs are classified into two groups: only reached by UGV (blue star) and only reached by UAV (red star). Besides that, the trajectories in blue and green show that the new coordination strategy re-computed to explore and find the IEDs without waiting for the other robot. The initial and the new segmentation are shown in Figure 6.2.

We compute the entropy as a metric to guarantee that the robot team explores all the floor map while at the same time is defusing the IEDs.



(a) Full robot trajectories from the 3 teams UAV/UGV. The UAV trajectory is shown in red and the rest of the robots in green, blue and purple.



(b) The targets that can only be reached and identified using a UAV are shown with a red star and the rest are only reached by the UGVs are shown with a blue star.

Figure 6.17: Atkinson Hall 6th Floor explored with drone trajectories and targets.

The comparison of our framework with and without the new coordination strategy is shown in Figure 6.18. As expected, the entropy decreases faster when we use the new coordination strategy.

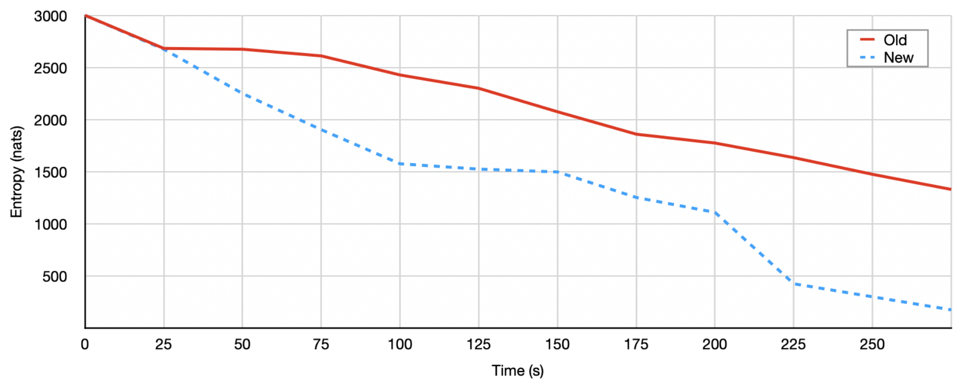


Figure 6.18: A comparison of the entropy over time of the robot team at Atkinson Hall 6th floor.

Computer Science and Engineering Bldg. 2nd Floor

This experiment was performed on the 2nd Floor of the Computer Science and Engineer Building. For safety reasons, we only deployed three UGV robots without UAV. As shown in Figure 6.19, the floor map consists of long hallways, study and social rooms and offices with different dimensions.

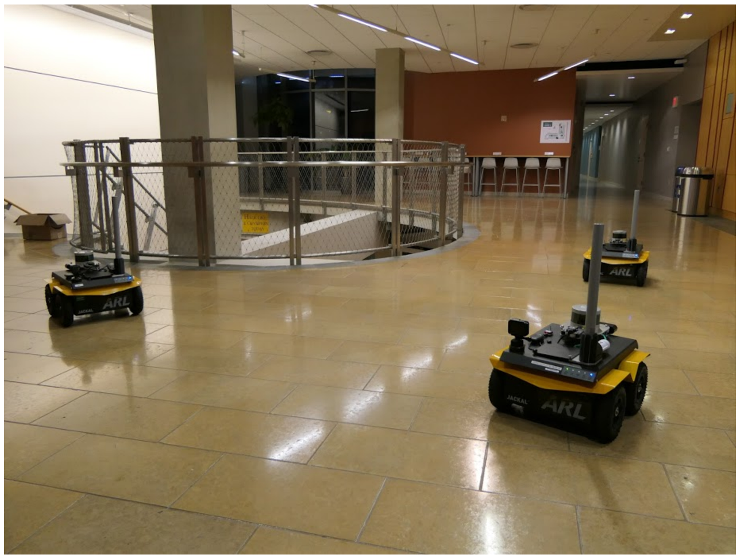
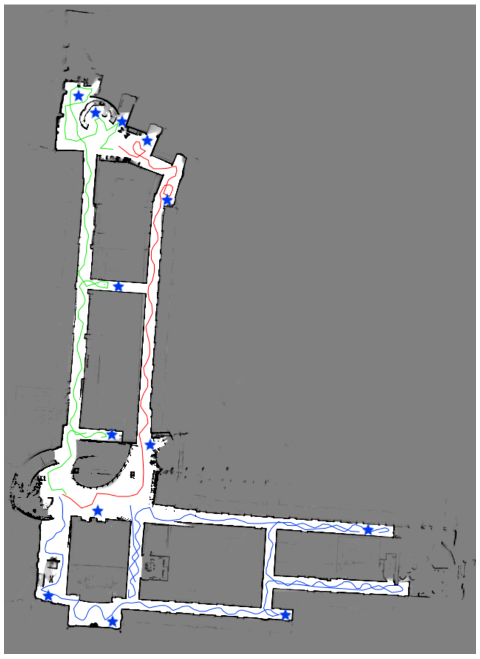


Figure 6.19: UGV team exploring and looking for IEDs at the Computer Science and Engineering Bldg. 2nd Floor.

The targets were set up on the floor. Figure 6.20 shows the trajectory of each robot and the IEDs that they found and defused. In addition, we labeled the floor as is shown in Figure 3.18 using *DEDUCE* which is explained in Section 3.4.6 .

We compute the entropy as a metric to guarantee that the robot team explores all the floor map while at the same time is defusing the IEDs.


(a) Full robot trajectories from the 3 UGVs (green, blue, and red).

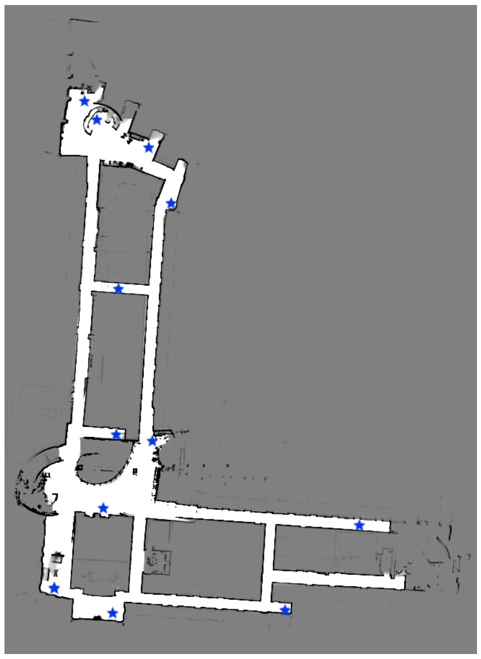

(b) All the targets can be reached and identified by the UGVs are shown with a blue star.

Figure 6.20: CSE Builiding 2nd Floor explored with UGVs' trajectories and the targets.

The comparison of our framework with and without the new coordination strategy is shown in Figure 6.21. As expected, the entropy decreases faster when we use the new coordination strategy

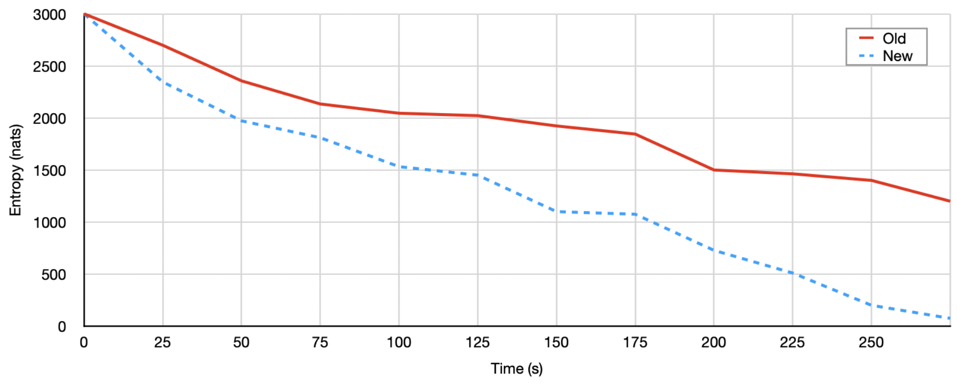


Figure 6.21: A comparison of the entropy over time of the robot team at the Computer Science Building in the 2nd Floor.

6.2.9 Discussion

In this work, we incorporated all the techniques presented in the previous chapters of this thesis. We aim that our framework is robust to find policies that satisfy the exploration and source seeking missions. This capability enables the possibility of performing real scenario missions. For example, the robot is performing a surveillance task when it detects a leak of gas. Then the robot switches to a source seeking task to prevent a possible explosion. Besides that, when the robot has prior knowledge of the environment and its history, the robot can make decisions faster and effectively. Our Risk Maps framework provides this capability. Also, we presented with experimental results that the robots can cooperate and manage their resources. The modularity in which the framework is programmed allows users to extend the system's capabilities and enable scalability on the management of big teams.

Currently, the U.S. Army Combat Capabilities Development Command Army Research Laboratory uses this framework for military and rescue missions.

Acknowledgment

This chapter appears in the following publications:

- Carlos Nieto-Granda, Shengye Wang, Vikas Dhiman, John G. Rogers III and Henrik Christensen. Distributed Heterogeneous Multi-robot source seeking using information based sampling with visual recognition. Submitted to the *2020 International Symposium on Experimental Robotics (ISER)*. Malta 2020.
- Carlos Nieto-Granda, Vikas Dhiman and Henrik Christensen. Multi-robot source seeking in partially observable environment. Submitted to *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Auckland, New Zealand, May 9-13, 2020.

I would like to thank my co-authors Shengye Wang, Vikas Dhiman, John G. Rogers III and Prof. Henrik I. Christensen.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

In this thesis, we provided a complete, robust decision-making framework that allows a team of heterogeneous robots to accomplish different tasks autonomously in a distributed way. We showed that each robot is capable of exploring and navigating in unknown environments while it is performing source seeking tasks. All the experiments performed in each chapter focused on simulating real military and rescue scenarios where the robots have an essential role in accomplishing the troops' mission. Moreover, our approach enables scalability in the number of members of the robot team. Besides that, robots can perform their tasks without interrupting the soldier's mission. This is critical, and it is one of the goals of the current mission of the U.S. Army Combat Capabilities Development Command Army Research Laboratory (ARL). As well, our system could be used not only for military purposes. The same models can be quickly adapted for rescue missions or industry tasks.

Our first contribution is a complete, distributed SLAM solution. Our approach performs in feature-based, object-based, as well as pose-based SLAM. The results in our publications show that we can map large scale areas (more than one kilometer). We developed a Graph SLAM with complete probabilistic data associations and loop closures. Besides that, our mapping system can handle communication constraints, information shared among the significant number of robot teams, and performed in outdoor and indoor environments.

The second contribution is our system framework *Risk Maps*, which provides high and low threat-level information. It provides a detailed action plan for preventing attacks, movement formation, and patrols of possible endangered areas. This framework is already compatible with the software at ARL.

The third contribution is the novel coordination strategies for heterogeneous multi-robot adaptive informative sampling. We show their capabilities in a centralized and distributed system. Our approach provides a policy to solve different tasks, such as source seeking. We demonstrated the effectiveness and robustness of our techniques in extensive simulations and field experiments in different applications (e.g., industrial, military, and service).

The lessons learned during this work presented in this thesis are:

- There is no perfect robotics system that never fails. Therefore, our systems have to be robust enough to predict possible failures and be able to recover from them.
- Our current framework *Risk Maps* has been tested in real scenarios providing action plans that prevent attacks. However, the action plan will not work unless the robot team will perform the task of avoiding switching to other tasks commanded by a human leader. In addition, the action plans can be followed not only by robots. The human platoon could easily use the framework to prevent or be prepared for a probable attack.
- Scalability of a multi-robot system increases the complexity of managing all the resources. It does not guarantee that the time to accomplish a task decreases faster than using a small team of robots.

7.2 Future Work

The capabilities of our *Risk Maps* framework are limited to the use of robots or humans. A simple improvement would be to create a module that allows robot-human teams to share and cooperate to perform some tasks. Besides that, receiving real-time commands from a human operator will enable the framework to learn faster in attack scenarios and consider action plans that human leaders plan on real missions.

Another improvement to our current semantic segmentation of the environment techniques would be to adapt the Gaussian Regions with our Voxel Exploration system. Also, the use of temporal information of the previous exploration will allow the robot to predict the next step to navigate an unknown environment. This helps find loop closures on the map and enables the robot to think in more patterns in different exploration scenarios. Currently, our semantic place categorization does not work in outdoor environments, and it is not capable of making new labels. For example, in the previous training, the neural network learned that a specific room is a garage. However, for task purposes, we would like to label it as a lab.

Signal modeling was not developed for this thesis. However, creating a module that learns transmission models will help to perform source-seeking and to understand the environment where the source is transmitting or emitting. As well, it would provide a solution to find a source when there is a multi-signal situation.

Currently, we are adapting Infosploration and Voxelsploration with the supervisory control of hybrid controllers for UGVs and UAVs. A significant challenge will be an online adaptation of the Motion Grammars. For example, the robot has to defuse a bomb. However, the robot task changes to control a toxic liquid leak. The Motion Grammar does not know what a poisonous liquid leak is and how to manage it.

The improvement of the Coordination Strategies is starting to be adaptable to the current situation of the system. A quick improvement would be to create formations that the robot team can follow depending on the environment's physical conditions. In our previous experiments at the MOUT, the robots had to navigate through a very narrow hallway. The robots try to go through without taking care of the other members of the team. In addition, the robots could learn how to create new formations for the adaptability of the environment and perform specific tasks. Also, we are currently exploring the optimal way to maintain connectivity among the robots. Our preliminary results allow a team of 50 robots to maintain connectivity in a 2D environment. We are extending this capability in 3D scenarios, which will enable the use of UAVs and UGVs.

We are improving our distributed mapping using only visual capabilities, and making the mapper robust using state-of-the-art techniques to decrease outliers on the images. We are also trying to understand the minimum amount of information that we have to share among the robots to create a map that is useful for navigation and exploration of the environment.
Voxelsploration is in current development. We are exploring the problem of map resizing. For example, the robot is mapping the floor plan of a building. After navigating for the tenth time, someone opens a door that increases the map's size, or closes another door to decrease the map. We are interested in finding those changes on the map and adding that information in our graph slam.

Finally, our *Distributed Heteregenous Multi-robot system* will be tested in a real scenario where the robots frequently have to switch tasks. We would like to check the robustness of our framework and find it when it fails. Besides that, we are currently implementing a virtual reality and touchscreen tablet interfaces that enable humans operators to interact with the robots that are members of the team. Also, we are interested in extending the robots' capability to guarantee that they are able to quickly find other robots that will assist them in their missions.

Bibliography

[ACS12] R. Aragues, J. Cortes, and C. Sagues. “Distributed consensus on robot networks for dynamically merging feature-based maps”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 28.Number 4 (2012).

[Aho+07] A. Aho, M. Lam, R. Sethi, and J. Ullman. *Compilers: Principles, Techniques, & Tools*. 2nd. Pearson, 2007.

[ALP14] Nikolay A. Atanasov, Jerome Le Ny, and George J. Pappas. “Distributed Algorithms for Stochastic Source Seeking With Mobile Robot Networks”. In: *Journal of Dynamic Systems, Measurement, and Control* Volume 137.Number 3 (2014).

[Alu+93] R. Alur, C. Courcoubetis, T. Henzinger, and P. Ho. “Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems”. In: *Hybrid systems* (1993).

[AN08] L. Andersson and J. Nygards. “C-SAM : Multi-Robot SLAM using Square Root Information Smoothing”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena CA, USA, 2008.

[Ara+11] R. Aragues, L. Carlone, G. Calafiore, and C. Sagues. “Multi-agent localization from noisy relative pose measurements”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, 2011.

[Ara+12] R. Aragues, L. Carlone, G. Calafiore, and C. Sagues. “Distributed centroid estimation from noisy relative measurements”. In: *Systems & Control Letters* Volume 61.Number 7 (2012).

[Arg+09] B.D. Argall, S. Chernova, M. Veloso, and B. Browning. “A survey of robot learning from demonstration”. In: *the Robotics and Autonomous Systems (RAS).* Volume 57.Number 5 (2009).

[Ark08] R.C. Arkin. “Governing Lethal Behavior: Embedding Ethics in a Hybrid Deliberative/Reactive Robot Architecture”. In: *the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. Amsterdam the Netherlands, 2008.

[Ark87] R.C. Arkin. “Motor Schema-Based Mobile Robot Navigation”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Raleigh NC, USA, 1987.

[Ark99] R.C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1999.

[ARL06] ARL. “Army Research Lab Micro Autonomous Systems and Technology Collaborative Technology Alliance MAST CTA”. In: *http://www.arl.army.mil/www/default.cfm?page=332* (2006).

[Bai+11] Tim Bailey, Mitch Bryson, Hua Mu, John Vial, Lachlan McCalman, and Hugh Durrant-Whyte. “Decentralised Cooperative Localisation for Heterogeneous Teams of Mobile Robots”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai China, 2011.

[Bai03] Tim Bailey. “Constrained Initialisation for Bearing-Only SLAM”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Taipei China, 2003.

[BBS10] S. A. Berrabah, Y. Baudoin, and H. Sahli. “SLAM for robotic assistance to fire-Fighting services”. In: Jinan China, 2010.

[BD06] T. Bailey and H. Durrant-Whyte. “Simultaneous Localisation and Mapping (SLAM): Part II State of the Art”. In: *the IEEE Robotics and Automation Magazine* (2006).

[BDL13] H. Bai, H. David, and W.S. Lee. “Integrated Perception and Planning in the Continuous Space: A POMDP Approach”. In: *the Robotics Science & Systems (RSS)*. Berlin Germany, 2013.

[Bee+07] Patrick Beeson, Matt MacMahon, Joseph Modayil, Aniket Murarka, Benjamin Kuipers, and Brian Stankiewicz. “Integrating multiple representations of spatial knowledge for mapping, navigation, and communication”. In: *Symposium on Interaction Challenges for Intelligent Assistants*. the Proceedings of the AAAI National Conference on Artificial Intelligence Spring Symposium Series. Stanford CA, USA, 2007.

[Ben+10] Michael R. Benjamin, Paul Newman, John J. Leonard, and Henrik Schmidt. *An Overview of MOOS-IvP and a Users Guide to the IvP Helm Autonomy Software*. Tech. rep. MIT-CSAIL-TR-2010-041. MIT, 2010.

[Ber97] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1997.

[BF99] R. I. Hartley B. Triggs P.F. McLauchlan and A. W. Fitzgibbon. “Bundle Adjustment - A Modern Synthesis”. In: *Lecture Notes in Computer Science* Volume 1883 (1999).

[BGG12] D. Benedettelli, A. Garulli, and A. Giannitrapani. “Cooperative SLAM using M-Space representation of linear features”. In: *the Robotics and Autonomous Systems (RAS).* Volume 60.Number 10 (2012).

[BH05a] P. Barooah and J.P. Hespanha. “Semantic Structure from Motion”. In: *the IEEE International Conference on Intelligent Sensing and Information Processing*. Bangalore India, 2005.

[BH05b] Prabir Barooah and João Pedro Hespanha. “Distributed Estimation from Relative Measurements in Sensor Networks”. In: *the International Conference on Intelligent Sensing and Information Processing*. Chennai India, 2005.

[BH07] P. Barooah and J.P. Hespanha. "Estimation on graphs from relative measurements". In: *Control System Magazine* Volume 27.Number 4 (2007).

[Bil+07] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. *Handbook of Robotics Chapter 59: Robot Programming by Demonstration*. Springer, 2007.

[BK+08] C. Baier, J.P. Katoen, et al. *Principles of Model Checking*. MIT Press, Cambridge, MA, 2008.

[BK04] C. Belta and V. Kumar. "Abstraction and control for groups of robots". In: *the IEEE Transactions on Robotics (T-RO)* Volume 20.Number 5 (2004).

[BKV02] O. Brock, O. Khatib, and S. Viji. "Task-consistent obstacle avoidance and motion behavior for mobile manipulation". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Washington DC, USA, 2002.

[Blu] M. Blum. *libgp: a C++ Library for Gaussian Process Regression*. GitHub. `https://github.com/mblum/libgp`, retrieved August, 2017.

[Bor+16] Richard Bormann, Florian Jordan, Wenzhe Li, Joshua Hampp, and Martin Hägele. "Room segmentation: Survey, implementation, and analysis." In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm Sweden, 2016.

[BR00] T. Ball and S. Rajamani. "Bebop: A symbolic model checker for boolean programs". In: *SPIN Model Checking and Software Verification* (2000).

[BR07] C. Bibby and I. Reid. "Simultaneous Localisation and Mapping in Dynamic Environments (SLAMIDE) with Reversible Data Association". In: *the Robotics Science & Systems (RSS)*. Atlanta GA, USA, 2007.

[BR13] Manuel Blum and Martin Riedmiller. "Optimization of Gaussian Process Hyperparameters using Rprop". In: *European Symposium on Artificial Neural Networks. Computational Intelligence and Machine Learning (CIML)*. Bruges Belgium, 2013.

[Bra98] M.S. Branicky. "Multiple Lyapunov functions and other analysis tools for switched and hybrid systems". In: *the IEEE Transactions on Automatic Control* Volume 43.4 (1998), pp. 475–482.

[Bro85] Rodney Brooks. "A robust layered control system for a mobile robot". In: *the IEEE Journal on Robotics and Automation* 2.1 (1985).

[Bro90a] RW Brockett. "Formal Languages for Motion Description and Map Making". In: *Robotics. American Mathematical Society* (1990).

[Bro90b] R.A. Brooks. "Elephants don't play chess". In: *the Robotics and Autonomous Systems (RAS).* Volume 6.Number 1-2 (1990).

[Bro91] W.L. Brogan. *Modern Control Theory*. Prentice-Hall, Upper Saddle River, NJ, 1991.

[Bro93] RW Brockett. "Hybrid Models for Motion Control Systems". In: *Essays on Control: Perspectives in the Theory and its Applications* Birkhauser (1993).

[Brz62] J.A. Brzozowski. “Canonical regular expressions and minimal state graphs for definite events”. In: *Mathematical Theory of Automata* Volume 12 (1962).

[BT10] Efstathios Bakolas and Panagiotis Tsiotras. “The Zermelo–Voronoi diagram: A dynamic partition problem”. In: *Automatica* Volume 46.Number 12 (2010).

[BT89] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[Bur+04] W. Burgard, C. Stachniss, G. Grisetti, B. Steder, R. Kümmerle, C. Dornhege, M. Ruhnke, Alexander Kleiner, and Juan D. Tardós. “A Comparison of SLAM Algorithms Based on a Graph of Relations”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. St. Louis MO, USA, 2004.

[Bur+05] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. “Coordinated multi-robot exploration”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 21.Number 3 (2005).

[Bur+98] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. “Experiences with an Interactive Museum Tour-Guide Robot”. In: *the Artificial Intelligence Journal (AIJ)* Volume 114.Number 1 (1998).

[BWL09] A. Bahr, M.R. Walter, and J.J. Leonard. “Consistent cooperative localization”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe Japan, 2009.

[Byl94] T. Bylander. “The computational complexity of propositional STRIPS planning”. In: *Artificial Intelligence* Volume 69.Number 1-2 (1994).

[CAD11] Gerardo Carrera, Adrien Angeli, and Andrew J. Davison. “SLAM-based automatic extrinsic calibration of a multi-camera rig”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. San Francisco CA, USA, 2011.

[Car+14] L. Carlone, J. Du, M. Kaouk Ng, M. Indri, and B. Bona. “Active SLAM and Exploration with Particle Filters using Kullback-Leibler Divergence”. In: *Journal of Intelligent and Robotic Systems* Volume 75.Number 2 (2014).

[Car+15a] L. Carlone, D.M. Rosen, G.C. Calafiore, J.J. Leonard, and F. Dellaert. “Lagrangian Duality in 3D SLAM: Verification Techniques and Optimal Solutions”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Hamburg Germany, 2015.

[Car+15b] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert. “Initialization Techniques for 3D SLAM: a Survey on Rotation Estimation and its Use in Pose Graph Optimization”. In: (2015).

[CCT11] C. Chao, M. Cakmak, and A.L. Thomaz. “Towards Grounding Concepts for Transfer in Goal Learning from Demonstration”. In: *the IEEE International Conference on Development and Learning*. Frankfurt am Main Germany, 2011.

[Cen07] A. Censi. “An accurate closed-form estimate of ICP’s covariance”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Rome Italy, 2007.

[Cen08] A. Censi. “An ICP variant using a point-to-line metric”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena CA, USA, 2008.

[Cer+09] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D.G. Sorrenti, and P. Taddei. “Rawseeds ground truth collection systems for indoor self-localization and mapping”. In: *Autonomous Robots* Volume 27.Number 4 (2009).

[CGK09] Theofilos Chrysikos, Giannis Georgopoulos, and Stavros Kotsopoulos. “Site-specific validation of ITU indoor path loss model at 2.4 GHz”. In: *the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks & Workshops (WoWMoM)*. Kos Island Greece, 2009.

[cha10] GCC - GNU (GCC 3.4 changes). http://gcc.gnu.org/gcc-3.4/changes.html. July 2010.

[Chi+11] R. Chipalkatty, H. Daepp, M. Egerstedt, and W. Book. “Human-in-the-loop: MPC for shared control of a quadruped rescue robot”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. San Francisco CA, USA, 2011.

[Cho+16a] Siddharth Choudhary, Luca Carlone, Carlos Nieto, John G. Rogers III, Zhen Liu, Henrik I. Christensen, and Frank Dellaert. “Multi Robot Object-Based SLAM”. In: *the International Symposium on Experimental Robotics (ISER)*. Tokyo Japan, 2016.

[Cho+16b] Siddharth Choudhary, Luca Carlone, Carlos Nieto-Granda, John G. Rogers III, Henrik I. Christensen, and Frank Dellaert. “Distributed trajectory estimation with privacy and communication constraints: A two-stage distributed Gauss-Seidel approach”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm Sweden, 2016.

[Cho+17a] Siddharth Choudhary, Luca Carlone, Carlos Nieto, John G. Rogers III, Henrik I Christensen, and Frank Dellaert. “Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models”. In: *the International Journal of Robotics Research (IJRR)*. Volume 36.Number 12 (2017).

[Cho+17b] Sanjiban Choudhury, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. “Adaptive Information Gathering via Imitation Learning”. In: *the Robotics Science & Systems (RSS)*. Cambridge MA, USA, 2017.

[Chr+10] Henrik I Christensen, Changhyun Choi, Victor Emeli, Akansel Cosgun, Jacob Huckaby, Carlos Nieto-Granda, Dev Priya, John G Rogers III, and Alexander J. Trevor. “GT@Home Team Description Paper”. In: *Robocup at Home*. Singapore, 2010.

[CID13] Alexander Cunningham, Vadim Indelman, and Frank Dellaert. “DDF-SAM 2.0: Consistent Distributed Smoothing and Mapping”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Karlsruhe Germany, 2013.

[CKE14] Stephen M. Chaves, Ayoung Kim, and Ryan M. Eustice. “Opportunistic sampling-based planning for active visual SLAM”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Chicago IL, USA, 2014.

[CKP02] J.M. Champarnaud, A. Khorsi, and T. Paranthoën. “Split and join for minimizing: Brzozowski’s algorithm”. In: *the Prague Stringology Conference*. Prague Czech Republic, 2002.

[CL08] C.G. Cassandras and Stéphane Lafortune. *Introduction to Discrete-Event Systems*. 2nd. Springer, 2008.

[CL85] R. Chatila and J.P. Laumond. “Position referencing and consistent world modeling for mobile robots”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. St. Louis MO, USA, 1985.

[CMR04] D. Caltabiano, G. Muscato, and F. Russo. “Localization and self-calibration of a robot for volcano exploration”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Sendai Japan, 2004.

[Cor+02] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. “Coverage control for mobile sensing networks”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Washington DC, USA, 2002.

[CPD10] A. Cunningham, M. Paluri, and F. Dellaert. “DDF-SAM: Fully Distributed SLAM using Constrained Factor Graphs”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[Cra05] J. Craig. *Introduction to Robotics: Mechanics and Control*. 3rd. Pearson, 2005.

[Cro89] J. Crowley. “World modeling and position estimation for a mobile robot using ultra-sonic ranging”. In: (1989).

[CV18] Zhaowei Cai and Nuno Vasconcelos. “Cascade r-cnn: Delving into high quality object detection”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Salt Lake City UT, USA, 2018.

[Dan+12] Neil Dantam, Carlos Nieto-Granda, Henrik I. Christensen, and Mike Stilman. “Linguistic Composition of Semantic Maps and Hybrid Controllers”. In: *the International Symposium on Experimental Robotics (ISER)*. Québec City Canada, 2012.

[Dan+14] Neil Dantam, Heni Ben Amor, Henrik Christensen, and Mike Stilman. “Online Camera Registration for Robot Manipulation”. In: *the International Symposium on Experimental Robotics (ISER)*. Marrakech and Essaouira Morocco, 2014.

[Dav+07] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. “MonoSLAM: Real-Time Single Camera SLAM”. In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Volume 29.Number 6 (2007).

[DB04] Frank Dellaert and David Bruemmer. “Semantic slam for collaborative cognitive workspaces”. In: *the Proceedings of the AAAI National Conference on Artificial Intelligence Fall Symposium Series*. Arlington VA, USA, 2004.

[DB06] H. Durrant-Whyte and T. Bailey. “Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms”. In: *the IEEE Robotics and Automation Magazine* (2006).

[De +06] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger. “Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[De +08] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi. “An atlas of physical human-robot interaction”. In: *Mechanism and Machine Theory* Volume 43.Number 3 (2008).

[Del05a] F. Dellaert. “Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing”. In: *the Robotics Science & Systems (RSS)*. Cambridge MA, USA, 2005.

[Del05b] Frank Dellaert. “Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing”. In: *the Robotics Science & Systems (RSS)*. Cambridge MA, USA, 2005.

[Del12a] Frank Dellaert. “Factor Graphs and GTSAM: A Hands-on Introduction”. In: *the Georgia Tech (GT-RIM-CP&R-2012-002)* (2012).

[Del12b] Frank Dellaert. *Factor Graphs and GTSAM: A Hands-on Introduction*. Tech. rep. GT-RIM-CP&R-2012-002. Georgia Institute of Technology, Sept. 2012.

[Den+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Miami FL, USA, 2009.

[DES11] N. Dantam, M. Egerstedt, and M. Stilman. “Make Your Robot Talk Correctly: Deriving Models of Hybrid System”. In: *the Robotics Science & Systems (RSS)*. Workshop on Grounding Human-Robot Dialog for Spatial Tasks. Los Angeles CA, USA, 2011.

[Die00] T.G. Dietterich. “Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition”. In: *Journal of Artifcial Intelligence Research* Volume 13 (2000).

[DK06a] F. Dellaert and M. Kaess. “Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing”. In: *the International Journal of Robotics Research (IJRR).* Volume 25.Number 12 (2006).

[DK06b] Frank Dellaert and Michael Kaess. “Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing”. In: *the International Journal of Robotics Research (IJRR).* Volume 25.Number 12 (2006).

[DKS11] N. Dantam, P. Kolhe, and M. Stilman. “The Motion Grammar for Physical Human-Robot Games”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai China, 2011.

[Don+15] J. Dong, E. Nelson, V. Indelman, N. Michael, and F. Dellaert. “Distributed Real-time Cooperative Localization and Mapping using an Uncertainty-Aware Expectation Maximization Approach”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Seattle WA, USA, 2015.

[DS10] N. Dantam and M. Stilman. *The Motion Grammar: Linguistic Perception, Planning, and Control*. Tech. rep. GT-GOLEM-2010-001. College of Computing, Georgia Institute of Technology, 2010.

[DS11a] N. Dantam and M. Stilman. “The Motion Grammar: Linguistic Planning and Control”. In: *the Robotics Science & Systems (RSS)*. Los Angeles CA, USA, 2011.

[DS11b] Neil Dantam and Mike Stilman. *Ach: IPC for Real-Time Robot Control*. Tech. rep. GT-GOLEM-2011-003. Georiga Insitute of Technology, 2011.

[DS12] N. Dantam and M. Stilman. “Deriving Models of Context-Free Hybrid Systems”. In: *the American Control Conference (ACC)*. IEEE. 2012.

[Dub15] Aaron Dubrow. “Five lessons from Hurricane Katrina (and 22 subsequent emergency robot deployments)”. In: *Discovery National Science Foundation (NSF) Report* (2015).

[Ear70] J. Earley. “An Efficient Context-Free Parsing Algorithm”. In: *Communications of the ACM* Volume 13.Number 2 (1970).

[Ege02] M. Egerstedt. “Motion Description Languages for Multi-Modal Control in Robotics”. In: *Control Problems in Robotics* Springer (2002).

[EK06] S. Ekvall and D. Kragic. “Learning task models from multiple human demonstrations”. In: *the IEEE International Symposium on Robot and Human Interactive Communication*. Hatfield United Kingdom, 2006.

[EKJ07a] S. Ekvall, D. Kragic, and P. Jensfelt. “Object detection and mapping for service robot tasks”. In: *Robotica: International Journal of Information, Education and Research* Volume 25.Number 2 (2007).

[EKJ07b] Staffan Ekvall, Danica Kragic, and Patric Jensfelt. “Object Detection and Mapping for Service Robot Tasks”. In: *Robotica: International Journal of Information, Education and Research* Volume 25.Number 2 (2007).

[EKS03] J. Esparza, A. Kucera, and S. Schwoon. “Model Checking LTL with Regular Valuations for Pushdown Systems”. In: *Information and Computation* Volume 186.Number 2 (2003).

[EMA] EMAG-Technologies-Inc. *EM Cube Integrated Modular Electromagnetic Modeling Suite*. `https://emagtech.com/em-cube`. Accessed: 2018-02-27.

[EML07] M. Egerstedt, T. Murphey, and J. Ludwig. “Motion Programs for Puppet Choreography and Control”. In: *Hybrid Systems: Computation and Control*. Vol. Springer. 2007.

[Esp+10] P. Espinace, T. Kollar, A. Soto, and N. Roy. "Indoor scene recognition through object detection". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage AK, USA, 2010.

[FB81] M. A. Fischler and R. C. Bolles. "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* Volume 24.Number 6 (1981).

[FC04] J. Folkesson and H. Christensen. "Graphical SLAM - A Self-Correcting Map". In: (2004).

[FDF05] E. Frazzoli, MA Dahleh, and E. Feron. "Maneuver-Based Motion Planning for Nonlinear Systems with Symmetries". In: *the IEEE Transactions on Robotics (T-RO)* Volume 21.Number 6 (2005).

[FG10] M. Franceschelli and A. Gasparri. "On agreement problems with Gossip algorithms in absence of common reference frames". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorange AK, USA, 2010.

[FJC05a] J. Folkesson, P. Jensfelt, and H. Christensen. "Vision SLAM in the measurement subspace". In: (2005).

[FJC05b] J. Folkesson, P. Jensfelt, and H. I. Christensen. "Graphical SLAM using vision and the measurement subspace". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2005).

[FJC07] J. Folkesson, P. Jensfelt, and H. I. Christensen. "The M-Space feature representation for SLAM". In: *the IEEE Transactions on Robotics (T-RO)* Volume 23.Number 5 (2007).

[FKP05] G.E. Fainekos, H. Kress-Gazit, and G.J. Pappas. "Hybrid Controllers for Path Planning: a Temporal Logic Approach". In: *the International Conference on Decision and Control (CDC)*. Seville Spain, 2005.

[Flo62] Robert W Floyd. "Algorithm 97: shortest path". In: *Communications of the ACM* Volume 5.Number 6 (1962).

[Fox+06] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart. "Distributed Multirobot Exploration and Mapping". In: *Proceedings of the IEEE* Volume 94.Number 7 (2006).

[Fox+99] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots". In: *Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence (IAAI)*. AAAI. 1999, pp. 343–349.

[Fox02] Eric M Foxlin. "Generalized architecture for simultaneous localization, auto-calibration, and map-building". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Lausanne Switerland, 2002.

[FS95] Y. Freund and R. E. Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *the European Conference on Computational Learning Theory*. London United Kingdom, 1995.

[FT12] Ehsan Fazl-Ersi and John K Tsotsos. “Histogram of oriented uniform patterns for robust place recognition and categorization”. In: *the International Journal of Robotics Research (IJRR).* Volume 31.Number 4 (2012).

[Fu81] K. Fu. *Syntactic Pattern Recognition and Applications*. Prentice-Hall, 1981.

[Fun+16] Nick Fung, Carlos Nieto-Granda, Jason Gregory, and John G. Rogers III. “Autonomous Exploration Using an Information Gain Metric”. In: *US Army Research Laboratory (ARL-TR-7638).* (2016).

[Fun+19] Nicholas Fung, John G. Rogers III, Carlos Nieto-Granda, Henrik Christensen, Stehanie Kemna, and Gaurav Sukhatme. “Coordinating multi-robot systems through environment partitioning for adaptive informative sampling”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal Canada, 2019.

[Gib66] J. J. Gibson. *The senses considered as perceptual systems*. Boston: Houghton Mifflin, 1966.

[Giu+10] M. Giuliani, C. Lenz, T. Müller, M. Rickert, and A. Knoll. “Design Principles for Safety in Human-Robot Interaction”. In: *International Journal of Social Robotics* Volume 2.Number 3 (2010).

[GKN12] Giorgio Grisetti, Rainer Kümmerle, and Kai Ni. “Robust Optimization of Factor Graphs by using Condensed Measurements”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* Vilamoura Algarve, Portugal, 2012.

[Gol67] E.M. Gold. “Language identification in the limit”. In: *Information and control* Volume 10.Number 5 (1967).

[Gri+07] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. “Efficient estimation of accurate maximum likelihood maps in 3D”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. San Diego CA, USA, 2007.

[GS10] Chao Gao and J.R. Spletzer. “On-line calibration of multiple LIDARs on a mobile vehicle platform”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage AK, USA, 2010.

[GSB07] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. “Improved techniques for grid mapping with rao-blackwellized particle filters”. In: *TRO* Volume 23.Number 1 (2007).

[GVH03] B. Gerkey, R. Vaughan, and A. Howard. “The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems”. In: *the International Conference on Advanced Robotics (ICAR)*. Coimbra Portugal, 2003.

[Häh+03] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. “Map Building with Mobile Robots in Dynamic Environments”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Taipei Taiwan, 2003.

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Las Vegas NV, USA, 2016.

[He+17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. “Mask r-cnn”. In: *the International Conference on Computer Vision (ICCV)*. Venice Italy, 2017.

[Hea07] Department of the Army United States Marine Corps Headquarters. “Improvised Explosive Device Defeat”. In: (2007).

[HEK03] D. Hristu-Varsakelis, M. Egerstedt, and PS Krishnaprasad. “On the Structural Complexity of the Motion Description Language MDLe”. In: *the International Conference on Decision and Control (CDC)*. Maui HI, USA, 2003.

[Hen96] T.A. Henzinger. “The theory of hybrid automata”. In: *Logic in Computer Science*. IEEE. 1996, pp. 278–292.

[Hig10] Colin de la Higuera. *Grammatical Inference*. Cambridge University Press, 2010.

[HL05] D. Hristu-Varsakelis and W.S. Levine, eds. *Handbook of Networked and Embedded Control Systems*. Birkhauser, 2005. ISBN: 0817632395.

[Hol01] G.J. Holzmann. “From code to models”. In: *the IEEE International Conference on Application of Concurrency to System Design*. Tyne United Kingdom, 2001.

[Hol04] G.J. Holzmann. *The Spin Model Checker*. Addison Wesley, Boston, MA, 2004.

[Hop71] John Hopcroft. “An n log n algorithm for minimizing states in a finite automaton”. In: *Theory of Machines and Computations* (1971).

[How06] A. Howard. “Multi-robot Simultaneous Localization and Mapping using Particle Filters”. In: *the International Journal of Robotics Research (IJRR).* Volume 25.Number 12 (2006).

[HP13] Lionel Heng and Marc Pollefeys. “CamOdoCal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Tokyo Japan, 2013.

[HS02] G.J. Holzmann and M.H. Smith. “An automated verification method for distributed systems software based on model extraction”. In: *the IEEE Transactions on Software Engineering* Volume 28.Number 4 (2002).

[HS06] F. Heger and S. Singh. “Sliding Autonomy for Complex Coordinated Multi-Robot Tasks: Analysis and Experiments”. In: *the Robotics Science & Systems (RSS)*. Philadelphia PA, USA, 2006.

[HSB02] D. Hähnel, D. Schulz, and W. Burgard. “Map Building with Mobile Robots in Populated Environments”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Lausanne Switzerland, 2002.

[HSK10] Geoffrey Hollinger, Sanjiv Singh, and Athanasios Kehagias. “Improving the Efficiency of Clearing with Multi-Agent Teams”. In: *the International Journal of Robotics Research (IJRR).* Volume 29.Number 8 (2010).

[HTP05] E. Haghverdi, P. Tabuada, and G.J. Pappas. “Bisimulation relations for dynamical, control, and hybrid systems”. In: *Theoretical Computer Science* Volume 342.Number 2-3 (2005).

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, Reading, MA, 1979.

[Huc+12] Jacob Huckaby, Carlos Nieto-Granda, John G Rogers III, Alexander J. Trevor, Akansel Cosgun, and Henrik I. Christensen. “Mobile Manipulation in Domestic Environments Using A Low Degree of Freedom Manipulator”. In: *Technical Report GT-RIM-CR-2012-002*. Atlanta GA, USA, 2012.

[Hya96] RM Hyatt. “CRAFTY–Chess Program”. In: *ftp://ftp.cis.uab.edu/pub/hyatt* (1996).

[HZ05] F. Han and S.C. Zhu. “Bottom-up/Top-down Image Parsing by Attribute Graph Grammar”. In: *the International Conference on Computer Vision (ICCV)*. Vol. Volume 2. 2005.

[IB00] Y.A. Ivanov and A.F. Bobick. “Recognition of visual activities and interactions by stochastic parsing”. In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Volume 22.Number 8 (2000).

[ICD14] Vadim Indelman, Luca Carlone, and Frank Dellaert. “Planning Under Uncertainty in the Continuous Domain: a Generalized Belief Space Approach”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Ind+12] V. Indelman, P. Gurfil, E. Rivlin, and H. Rotstein. “Graph-Based Distributed Cooperative Navigation for a General Multi-Robot Measurement Model”. In: *the International Journal of Robotics Research (IJRR).* Volume 31.Number p (2012).

[Ind+14] V. Indelman, E. Nelson, N. Michael, and F. Dellaert. “Multi-Robot Pose Graph Localization and Data Association from Unknown Initial Relative Poses via Expectation Maximization”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Jan+13] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. “A category-level 3d object dataset: Putting the kinect to work”. In: (2013).

[Jan87] M. Jantzen. “Language theory of Petri nets”. In: *Petri Nets: Central Models and Their Properties* (1987).

[Jon+83] L.T. Jones, A. Howden, M.S. Knighton, A. Sims, D.L. Kittinger, and R.E. Hollander. *Robot computer chess game*. US Patent 4,398,720. Aug. 1983.

[Joy+09] S. Joyeux, R. Alami, S. Lacroix, and R. Philippsen. "A plan manager for multi-robot systems". In: *the International Journal of Robotics Research (IJRR).* Volume 28.Number 2 (2009).

[JVW91] A.K. Joshi, K. Vijay-Shanker, and D. Weir. "The Convergence of Mildly Context-Sensitive Grammar Formalisms". In: *Foundational Issues in Natural Language Processing* (1991).

[Kae+12] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J. Leonard, and Frank Dellaert. "iSAM2: Incremental smoothing and mapping using the Bayes tree". In: *the International Journal of Robotics Research (IJRR).* Volume 31.Number 2 (2012).

[Kan+02] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, and A.Y. Wu. "An efficient-means clustering algorithm: Analysis and implementation". In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2002).

[KB10] M. Kloetzer and C. Belta. "Automatic Deployment of Distributed Teams of Robots from Temporal Logic Motion Specifications". In: *the IEEE Transactions on Robotics (T-RO)* Volume 26.Number 1 (2010).

[KB13] J. Knuth and P. Barooah. "Collaborative localization with heterogeneous inter-robot measurements by Riemannian optimization". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Karlsruhe Germany, 2013.

[KCB97] J. Košecká, H.I. Christensen, and R. Bajcsy. "Experiments in behavior composition". In: *the Robotics and Autonomous Systems (RAS).* Volume 19.Number 3 (1997).

[KCS16] Stephanie Kemna, David A. Caron, and Sukhatme Gaurav S. "Adaptive Informative Sampling with Autonomous Underwater Vehicles: Acoustic versus Surface Communications". In: *the MTS/IEEE OCEANS*. 2016.

[Kem+17] Stephanie Kemna, John G. Rogers III, Carlos Nieto-Granda, Stuart Young, and Gaurav S. Sukhatme. "Multi-robot coordination through dynamic Voronoi partitioning for informative adaptive sampling in communication-constrained environments". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Singapore, 2017.

[Kem+18] Stephanie Kemna, John G. Rogers III, Carlos Nieto-Granda, Stuart Young, and Gaurav S. Sukhatme. "Multi-robot task allocation for collaborative adaptive informative sampling in structured environments. Workshop on Informative Path Planning and Adaptive Sampling." In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane Australia, 2018.

[KFP09] H. Kress-Gazit, G.E. Fainekos, and G.J. Pappas. "Temporal-Logic-Based Reactive Mission and Motion Planning". In: *the IEEE Transactions on Robotics (T-RO)* Volume 25.Number 6 (2009).

[KG15] Ioannis Kostavelis and Antonios Gasteratos. "Semantic Mapping for Mobile Robotics Tasks". In: *the Robotics and Autonomous Systems (RAS).* Volume 66 (2015).

[KGB11] R. Kummerle, G. Grisetti, and W. Burgard. "Simultaneous Calibration, Localization and Mapping". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. San Francisco CA, USA, 2011.

[KH04] Nathan Koenig and Andrew Howard. "Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sendai Japan, 2004.

[Kie+08] Julie A. Kientz, Shwetak N. Patel, Brian Jones, Ed Price, Elizabeth D. Mynatt, and Gregory D. Abowd. "The Georgia Tech aware home". In: *ACM CHI Conference on Human Factors in Computing Systems*. Florence, Italy, 2008.

[Kim+10] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. "Multiple Relative Pose Graphs for Robust Cooperative Mapping". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage, Alaska, 2010.

[Kla+06] E. Klavins, R. Ghrist, D. Lipsky, E.E. Departjment, and WA Seattle. "A Grammatical Approach to Self-Organizing Robotic Systems". In: *IEEE Transactions on Automatic Control* Volume 51.Number 6 (2006).

[Kla04] E. Klavins. "A language for Modeling and Programming Cooperative Control Systems". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. New Orleans LA, USA, 2004.

[KN94] Dongsung Kim and Ramakant Nevatia. "A Method for Recognition and Localization of Generic Objects for Indoor Navigation". In: *Image and Vision Computing* 16 (1994).

[KNS13] Moritz Knorr, Wolfgang Niehsen, and Christoph Stiller. "Online extrinsic multi-camera calibration using ground plane induced homographies." In: *the IEEE Intelligent Vehicles Symposium*. Gold Coast City, Australia, 2013.

[Kou+09] P. Koutsourakis, L. Simon, O. Teboul, G. Tziritas, and N. Paragios. "Single View Reconstruction Using Shape Grammars for Urban Environments". In: *the International Conference on Computer Vision (ICCV)*. Kyoto Japan, 2009.

[KR09] T. Kollar and N. Roy. "Utilizing object-object and object-scene context when planning to find things". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe Japan, 2009.

[KRD07] M. Kaess, A. Ranganathan, and F. Dellaert. "Fast Incremental Square Root Information Smoothing". In: Hyderabad India, 2007.

[KRD08] M. Kaess, A. Ranganathan, and F. Dellaert. "iSAM: Incremental Smoothing and Mapping". In: *the IEEE Transactions on Robotics (T-RO)* Volume 24.Number 6 (2008).

[KSG08] Andreas Krause, Ajit Singh, and Carlos Guestrin. "Near-Optimal Sensor Placements in Gaussian Processes-Theory, Efficient Algorithms and Empirical Studies". In: *The Journal of Machine Learning Research* Volume 9 (2008).

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *International Conference on Neural Information Processing Systems (NIPS)*. Stateline NV, USA, 2012.

[Kui+00] Benjamin Kuipers, Rob Browning, Bill Gribble, Mike Hewett, and Emilio Remolina. “The Spatial Semantic Hierarchy”. In: *the Artificial Intelligence Journal (AIJ)* Volume 119.Number 1 (2000).

[Kun+10] T. Kunz, U. Reiser, M. Stilman, and A. Verl. “Real-Time Path Planning for a Robot Arm in Changing Environments”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[KZ07] J. Klippenstein and H. Zhang. “Quantitative evaluation of feature extractors for visual SLAM”. In: Montreal Quebec, Canada, 2007.

[LaV06] S.M. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[Laz+11] M.T. Lazaro, L.M Paz, P. Pinies, J.A. Castellanos, and G. Grisetti. “Multi-robot SLAM using condensed measurements”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai China, 2011.

[LB85] T. Lozano-Pérez and R.A. Brooks. *An approach to automatic robot programming*. Tech. rep. A.I. Memo 842. Massachusetts Intitute of Technology, 1985.

[LC18] Ren C. Luo and Michael Chiou. “Hierarchical Semantic Mapping Using Convolutional Neural Networks for Intelligent Service Robotics”. In: *IEEE Access* Volume 6 (2018).

[Lia+16] Yiyi Liao, Sarath Kodagoda, Yue Wang, Lei Shi, and Yong Liu. “Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Orlando FL, USA, 2016.

[Lia+17] Yiyi Liao, Sarath Kodagoda, Yue Wang, Lei Shi, and Yong Liu. “Place classification with a graph regularized deep neural network”. In: *the IEEE Transactions on Cognitive and Developmental Systems* Volume 9.Number 4 (2017).

[Lin+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. “Microsoft coco: Common objects in context”. In: *the European Conference on Computer Vision (ECCV)*. Zürich Switzerland, 2014.

[Lin+17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. “Focal loss for dense object detection”. In: *the International Conference on Computer Vision (ICCV)*. Venice Italy, 2017.

[Lit96] M.L. Littman. “Algorithms for Sequential Decision Making”. PhD thesis. Brown University, 1996.

[Liu+16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. “Ssd: Single shot multibox detector”. In: *the European Conference on Computer Vision (ECCV)*. Amsterdam the Netherlands, 2016.

[LJB07] Magnus Lindhé, Karl Henrik Johansson, and Antonio Bicchi. “An experimental study of exploiting multipath fading for robot communications”. In: *the Robotics Science & Systems (RSS)*. Atlanta GA, USA, 2007.

[LL04] O. Linde and T. Lindeberg. “Object recognition using composed receptive field histograms of higher dimensionality”. In: *the International Conference on Pattern Recognition (ICPR)*. Cambride United Kingdom, 2004.

[LM97] F. Lu and E. Milios. “Globally Consistent Range Scan Alignment for Environment Mapping”. In: *Autonomous Robots* Volume 4 (1997).

[LMK11] Q. Lindsey, D. Mellinger, and V. Kumar. “Construction of cubic structures with quadrotor teams”. In: *the Robotics Science & Systems (RSS)* (2011).

[Low+09] Kian Hsiang Low, John M. Dolan, Jeff Schneider, and Alberto Elfes. “Multi-Robot Adaptive Exploration and Mapping for Environmental Sensing Applications”. PhD thesis. CMU, 2009.

[Low04] D. G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *the International Journal of Computer Vision* Volume 60.Number 2 (2004).

[LSB10] B. Lau, C. Sprunk, and W. Burgard. “Improved updating of Euclidean distance maps and Voronoi diagrams”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[LT13] Jesse Levinson and Sebastian Thrun. “Automatic Online Calibration of Cameras and Lasers”. In: *the Robotics Science & Systems (RSS)*. Berlin Germany, 2013.

[LW77] T. Loiano-Perer and P.H. Winston. “LAMA: a language for automatic mechanical assembly”. In: *the AAAI National Conference on Artificial Intelligence*. Providence RI, USA, 1977.

[Lyg+03] J. Lygeros, K.H. Johansson, S.N. Simic, J. Zhang, and S.S. Sastry. “Dynamical properties of hybrid automata”. In: *the IEEE Transactions on Automatic Control* Volume 48.Number 1 (2003).

[Man+18] Massimiliano Mancini, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. “Robust place categorization with deep domain generalization”. In: *the IEEE Robotics and Automation Letters (RA-L)* Volume 3.Number 3 (2018).

[Mar+15] Alessandro Marino, Gianluca Antonelli, António Pedro Aguiar, António Pascoal, and Stefano Chiaverini. “A Decentralized Strategy for Multirobot Sampling/Patrolling: Theory and Experiments”. In: *the IEEE Transactions on Control Systems Technology* Volume 23.Number 1 (2015).

[May14] J. Maye. “Online Self-Calibration for Robotic Systems”. PhD thesis. ETH Zürich, 2014.

[MB06] O. M. Mozos and W. Burgard. “Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[ME02] D. Moore and I. Essa. “Recognizing multitasked activities from video using stochastic context-free grammar”. In: *the AAAI National Conference on Artificial Intelligence*. Edmonton Alberta, Canada, 2002.

[Mes+05] Elena Messina1, Adam Jacoff, Jean Scholtz, Craig Schlenoff, Hui-Min Huang, Alan Lytle, and John Blitch. “Statement of Requirements for Urban Search and Rescue robot performance standards”. In: *Internal Report NIST* (2005).

[MES03] D. Minnen, I. Essa, and T. Starner. “Expectation grammars: Leveraging high-level expectations for activity recognition”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Vol. Number 2. Madison WI, USA, 2003.

[MHK95] V. Manikonda, J. Hendler, and PS Krishnaprasad. “Formalizing Behavior-Based Planning for Nonholonomic Robots”. In: vol. 14. 1995.

[MHN12] W. Maddern, A. Harrison, and P. Newman. “Lost in translation (and rotation): Rapid extrinsic calibration for 2D and 3D LIDARs”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. St. Paul MN, USA, 2012.

[Mih+03] Lyudmila Mihaylova, Tine Lefebvre, Herman Bruyninckx, Klaas Gadeyne, and Joris De Schutter. “A comparison of decision making criteria and optimization methods for active robotic sensing”. In: *Numerical Methods and Applications*. Springer, 2003.

[MKH98] V. Manikonda, PS Krishnaprasad, and J. Hendler. “Languages, Behaviors, Hybrid Architectures and Motion Control”. In: *Mathematical Control Theory* (1998).

[Mon+03] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. “Fast-SLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges”. In: *the International Conference on Artificial Intelligence (IJCAI)*. Acapulco Guerrero, Mexico, 2003.

[Moz+07a] O. M. Mozos, R. Triebel, P. Jensfelt, A. Rottmann, and W. Burgard. “Supervised Semantic Labeling of Places using Information Extracted from Laser and Vision Sensor Data”. In: *the Robotics and Autonomous Systems (RAS).* Volume 55.Number 5 (2007).

[Moz+07b] Oscar Martinez Mozos, Rudolph Triebel, Patric Jensfelt, Axel Rottmann, and Wolfram Burgard. “Supervised semantic labeling of places using information extracted from sensor data”. In: *the Robotics and Autonomous Systems (RAS).* Volume 55.Number 5 (2007).

[MP07] D. Martinec and T. Pajdla. “Robust Rotation and Translation Estimation in Multi-view Reconstruction”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Minneapolis MN, USA, 2007.

[MR93] Douglas C Montgomery and George C Runger. “Gauge capability and designed experiments. Part I: Basic methods”. In: *Quality Engineering. Taylor & Francis* Volume 6.Number 1. (1993).

[MRD08] David Meger, Ioannis Rekleitis, and Gregory Dudek. “Heuristic search planning to reduce exploration uncertainty”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Nice France, 2008.

[Mül+06] Maurice Müller, Hartmut Surmann, Kai Pervölz, and Stefan May. “The accuracy of 6D SLAM using the AIS 3D laser scanner”. In: *the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. Heidelberg Germany, 2006.

[Mur+16] Varun Murali, Carlos Nieto-Granda, Siddharth Choudhary, and Henrik I. Christensen. “Active planning based extrinsic calibration of exteroceptive sensors in unknown environments”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon South Korea, 2016.

[Ngu+05] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart. “A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2005).

[NH86] Y. Nakamura and H. Hanafusa. “Inverse kinematics solutions with singularity robustness for robot manipulator control”. In: *Journal of Dynamic Systems, Measurement, and Control* Volume 108 (1986).

[NIC13] C. Nieto-Granda, John G. Rogers III, and H. I. Christensen. “Multi-robot exploration strategies for tactical tasks in urban environments”. In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Baltimore MD, USA, 2013.

[NIC14] Carlos Nieto-Granda, John G. Rogers III, and Henrik I. Christensen. “Coordination strategies for multi-robot exploration and mapping”. In: *the International Journal of Robotics Research (IJRR).* Volume 33.4 (2014).

[Nie+10a] C. Nieto-Granda, J. G. Rogers III, A. J. B. Trevor, and H. I. Christensen. “Semantic Map Partitioning in Indoor Environments Using Regional Analysis”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[Nie+10b] Carlos Nieto-Granda, John G. Rogers, Alexander J. B. Trevor, and Henrik Iskov Christensen. “Semantic map partitioning in indoor environments using regional analysis”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[Nie+14] C. Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. “Object guided autonomous exploration for mobile robots in indoor and outdoor environments”. In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Baltimore MD, USA, 2014.

[Nie+15] Carlos Nieto-Granda, S. Choudhary, J. G. Rogers III, J. N. Twigg, V. Murali, and H. I. Christensen. “Towards contextual awareness in robot mapping: extracting semantic hierarchy from point cloud data”. In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Baltimore MD, USA, 2015.

[Nie+18] Carlos Nieto-Granda, John G. Rogers III, Nicolas Fung, Stephanie Kenma, Henrik I. Christensen, and Gaurav Sukhatme. “On-line coordination tasks for multi-robot systems using adaptive informative sampling.” In: *the International Symposium on Experimental Robotics (ISER)*. Buenos Aires Argentina, 2018.

[Nie19] Carlos Nieto-Granda. *Autonomous exploration of unknown environments using Voxel Maps*. Tech. rep. Nov. 2019.

[NRC14] C. Nieto-Granda, J. G. Rogers, and H. I. Christensen. “Coordination strategies for multi-robot exploration and mapping”. In: *the International Journal of Robotics Research (IJRR).* Volume 33.Number 4 (2014).

[NRM09] E.D. Nerurkar, S.I. Roumeliotis, and A. Martinelli. “Distributed maximum a posteriori estimation for multi-robot cooperative localization”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe Japan, 2009.

[NT01a] J. Neira and J.D. Tardós. “Data association in stochastic mapping using the joint compatibility test”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 17.Number 6 (2001).

[NT01b] José Neira and Juan D Tardós. “Data association in stochastic mapping using the joint compatibility test”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 17.Number 6 (2001).

[NWR14] E.D. Nerurkar, K.J. Wu, and S.I. Roumeliotis. “C-KLAM: Constrained Keyframe-Based Localization and Mapping”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Obe+08] Jan Oberländer, Klaus Uhl, Johann Marius Zöllner, and Rüdiger Dillmann. “A region-based SLAM algorithm capturing metric, topological, and semantic properties”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena CA, USA, 2008.

[OF10] J. Owens and M. Fields. “Incremental region segmentation for hybrid map generation”. In: *Army Science Conference*. Orlando FL, USA, 2010.

[OL05] J.M. O’Kane and Steven LaValle. “Almost-sensorless localization”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Barcelona Spain, 2005.

[OL07] J.M. O’Kane and Steven LaValle. “Localization With Limited Sensing”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 23.Number 4 (2007).

[Ols+12] Edwin Olson, Johannes Strom, Ryan Morton, Andrew Richardson, Pradeep Ranganathan, Robert Goeddel, Mihai Bulic, Jacob Crossman, and Bob Marinier. "Progress towards multi-robot reconnaissance and the MAGIC 2010 Competition". In: *the Journal of Field Robotics* Volume 29.Number 5 (2012).

[ORD09] Simon O'Callaghan, Fabio T. Ramos, and Hugh F. Durrant-Whyte. "Contextual occupancy maps using Gaussian processes." In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe Japan, 2009.

[OT01] A. Oliva and A. Torralba. "Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope". In: *the International Journal of Computer Vision* volume 42.Number 3 (2001).

[Ouy+14] Ruofei Ouyang, Kian Hsiang Low, Jie Chen, and Patrick Jaillet. "Multi-Robot Active Sensing of Non-Stationary Gaussian Process-Based Environmental Phenomena". In: *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Paris France, 2014.

[Pac03] R.T. Pack. "IMA: The Intelligent Machine Architecture". PhD thesis. Vanderbilt University, 2003.

[Pag12] Frank Pagel. "Motion Adjustment for Extrinsic Calibration of Cameras with Non-overlapping Views". In: *the Proceedings the Conference on Computer and Robot Vision*. Toronto Ontario, Canada, 2012.

[Pag14] Frank Pagel. "Extrinsic self-calibration of multiple cameras with non-overlapping views in vehicles". In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Ed. by Robert P. Loce and Eli Saber. Baltimore MD, USA, 2014.

[Pan+10] Gaurav Pandey, James McBride, Silvio Savarese, and Ryan Eustice. "Extrinsic calibration of a 3d laser scanner and an omnidirectional camera". In: *the Proceedings on the International Federation of Automatic Control (IFAC) symposium on intelligent autonomous vehicles*. Lecce Italy, 2010.

[Pan+12a] Gaurav Pandey, James R McBride, Silvio Savarese, and Ryan Eustice. "Automatic Targetless Extrinsic Calibration of a 3D Lidar and Camera by Maximizing Mutual Information." In: *the Proceedings of the AAAI National Conference on Artificial Intelligence*. Toronto Canada, 2012.

[Pan+12b] Gaurav Pandey, James R. McBride, Silvio Savarese, and Ryan M. Eustice. "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information". In: *the Proceedings of the AAAI National Conference on Artificial Intelligence*. Toronto Ontario, Canada, 2012.

[Par08] L. Parker. "Handbook of Robotics". In: Springer Verlag, 2008. Chap. Multiple Mobile Robot Systems.

[Pau+15] L. Paull, G. Huang, M. Seto, and J. Leonard. "Communication-Constrained Multi-AUV Cooperative SLAM". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Seattle WA, USA, 2015.

[Pet81] J. L. Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1981.

[PF11] T. Parr and K. Fisher. "LL (*): the foundation of the ANTLR parser generator". In: *ACM SIGPLAN conference on Programming language design and implementation*. Vol. Volume 46. Number 6. 2011.

[PFS13] T. Patten, R. Fitch, and S. Sukkarieh. "Large-Scale Near-Optimal Decentralised Information Gathering with Multiple Mobile Robots". In: *the Australasian Conference on Robotics and Automation*. Sidney Australia, 2013.

[Pil+14] Georgios Piliouras, Carlos Nieto-Granda, Henrik I. Christensen, and Jeff S. Shamma. "Persistent patterns: multi-agent learning beyond equilibrium and utility". In: *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Paris France, 2014.

[Pit+08] James Pita, Manish Jain, Janusz Marecki, Fernando Ordoñez, Christopher Portway, Milind Tambe, Craig Western, Praveen Paruchuri, and Sarit Kraus. "Deployed ARMOR Protection: The Application of a Game Theoretic Model for Security at the Los Angeles International Airport." In: *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*. Estoril Portugal, 2008.

[Ple04] R. Pless. "Extrinsic calibration of a camera and laser range finder (improves camera calibration)". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sendai Japan, 2004.

[PMC08] A. Pronobis, O. M. Mozos, and B. Caputo. "SVM-based discriminative accumulation scheme for place recognition". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena CA, USA, 2008.

[PNC12] Manohar Paluri, Carlos Nieto-Granda, and Henrik I. Christensen. "A Case Study on Effects of Various Parameters in a Localization and Navigation System". In: *the International Symposium on Experimental Robotics (ISER)*. Québec City Canada, 2012.

[PQ95] T.J. Parr and R.W. Quong. "ANTLR: A predicated-LL (k) parser generator". In: *Software: Practice and Experience* Volume 25.Number 7 (1995).

[Pro+06] A. Pronobis, B. Caputo, P. Jensfelt, and H. I. Christensen. "A discriminative approach to robust visual place recognition". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[QT09] Ariadna Quattoni and Antonio B. Torralba. "Recognizing indoor scenes". In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Miami Beach FL, USA, 2009.

[Qui+09a] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. "ROS: an open-source Robot Operating System". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. St. Louis MO, USA, 2009.

[Qui+09b] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully B. Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. “ROS: an open-source Robot Operating System”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Kobe Japan, 2009.

[RB02] S.I. Roumeliotis and G.A. Bekey. “Distributed Multi-Robot Localization”. In: *IEEE Transactions on Robotics and Automation* Volume 18.Number 5 (2002).

[RC11] R. B. Rusu and S. Cousins. “3D is here: Point Cloud Library (PCL)”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai China, 2011.

[Red+16a] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Las Vegas NV, USA, 2016.

[Red+16b] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. “You only look once: Unified, real-time object detection”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Las Vegas NV, USA, 2016.

[Rei79] J.H. Reif. “Complexity of the mover’s problem and generalizations extended abstract”. In: *the IEEE Conference on Foundations of Computer Science*. San Juan Puerto Rico, USA, 1979.

[RF17] Joseph Redmon and Ali Farhadi. “YOLO9000: better, faster, stronger”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Honolulu HI, USA, 2017.

[RF18] J. Redmon and A. Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[RN02] S. Russell and P. Norvig. *Artificial Intelligence: A modern Approach*. 2nd. Prentice-Hall, 2002.

[RNC12] John G. Rogers, Carlos Nieto-Granda, and Henrik I. Christensen. “Coordination Strategies for Multi-robot Exploration and Mapping”. In: *the International Symposium on Experimental Robotics (ISER)*. Québec City Canada, 2012.

[Rog+10a] J. Rogers, A.J.B. Trevor, C. Nieto, A. Cunningham, M. Paluri, N. Michael, F. Dellaert, H.I. Christensen, and V. Kumar. “Effects of sensory perception on mobile robot localization and mapping”. In: *the International Symposium on Experimental Robotics (ISER)*. New Delhi and Agra, India, 2010.

[Rog+10b] John G. Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. “SLAM with Expectation Maximization for moveable object tracking”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Taipei Taiwan, 2010.

[Rog+10c] John G. Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, Alexander Cunningham, Manohar Paluri, Nathan Michael, Frank Dellaert, Henrik I. Christensen, and Vijay Kumar. “Effects of Sensory Precision on Mobile Robot Localization and Mapping”. In: *the International Symposium on Experimental Robotics (ISER)*. New Delhi and Agra India, 2010.

[Rog+11a] John G. Rogers, Alexander J. B. Trevor, Carlos Nieto-Granda, and Henrik Iskov Christensen. “Simultaneous Localization and Mapping with Learned Object Recognition and Semantic Data Association”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. San Francisco CA, USA, 2011.

[Rog+11b] J. G. Rogers III, A. Cunningham, M. Paluri, H. I Christensen, F. Dellaert, N. Michael, V. Kumar, and L. Mathies. “Cooperative Mapping of Indoor Environments”. In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Orando FL, USA, 2011.

[Rog+13] John G. Rogers, Stuart Young, Jason M. Gregory, Carlos Nieto-Granda, and H. I. Christensen. “Robot mapping in large-scale mixed indoor and outdoor environments”. In: *the SPIE Defense + Security conference. Unmanned Systems Technology*. Baltimore, MD., USA, 2013.

[Rot+05] Axel Rottmann, Óscar Martínez Mozos, Cyrill Stachniss, and Wolfram Burgard. “Semantic place classification of indoor environments with mobile robots using boosting”. In: *the AAAI National Conference on Artificial Intelligence*. Pittsburgh PA, USA, 2005.

[Row76] N Rowe. *Grammar as a programming language*. Tech. rep. A.I. Memo 391. MIT Artificial Intelligence Laboratory, Oct. 1976.

[RTH11] Cheta Rawal, Herbert G. Tanner, and Jeffrey Heinz. “(Sub)regular Robotic Languages”. In: *the IEEE Mediterranean Conference. on Control and Automation*. Corfu Greece, 2011.

[Rus+08] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Emanuel Dolha, and Michael Beetz. “Functional Object Mapping of Kitchen Environments”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Nice France, 2008.

[Rus09] Radu Bogdan Rusu. “Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments”. PhD thesis. Technische Universität München, 2009.

[RW06] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning.* MIT press, 2006.

[RW87] P. J. Ramadge and W. M. Wonham. “Supervisory Control of a Class of Discrete Event Processes”. In: *Analysis and Optimization of Systems* Volume 25.Number 1 (1987).

[Şah+07] E. Şahin, M. Çakmak, M.R. Doğar, E. Uğur, and G. üçoluk. “To afford or not to afford: A new formalization of affordances toward affordance-based robot control”. In: *Adaptive Behavior* Volume 15.Number 4 (2007).

[Sal+13a] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. Kelly, and A. J. Davison. “Slam++: Simultaneous localisation and mapping at the level of objects”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Portland OR, USA, 2013.

[Sal+13b] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H. J. Kelly, and Andrew J. Davison. “SLAM++: Simultaneous Localisation and Mapping at the Level of Objects”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Portland OR, USA, 2013.

[Sän+18] Niko Sänderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. “The limits and potentials of deep learning for robotics”. In: *the International Journal of Robotics Research (IJRR).* Volume 37.Number 4-5 (2018).

[SB05] C. Stachniss and W. Burgard. “Mobile Robot Mapping and Localization in Non-Static Environments”. In: *the Proceedings of the National Conference on Artificial Intelligence*. 3. Pittsburgh PA, USA, 2005.

[SC87] R. Smith and P. Cheeseman. “On the representation and estimation of spatial uncertainty”. In: *the International Journal of Robotics Research (IJRR).* Volume 5.Number 4 (1987).

[Sch+14] Mac Schwager, Michael P. Vitus, Samantha Powers, Daniela Rus, and Claire J. Tomlin. “Robust Adaptive Coverage Control for Robotic Sensor Networks”. In: *the IEEE Transactions on Control of Network Systems* Volume 4.Number 3 (2014).

[Sch14] T. Schneider. *Goby Underwater Autonomy Project - User Manual*. 2014. URL: `http://gobysoft.org/doc/2.0/`.

[SGB05] C. Stachniss, G. Grisetti, and W. Burgard. “Information Gain-based Exploration Using Rao-Blackwellized Particle Filters”. In: *the Robotics Science & Systems (RSS)*. Cambridge MA, USA, 2005.

[SHB04] C. Stachniss, D. Hahnel, and W. Burgard. “Exploration with active loop-closing for FastSLAM”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sendai Japan, 2004.

[Shr+19] R. Shrestha, F. Tian, W. Feng, P. Tan, and R. Vaughan. “Learned Map Prediction for Enhanced Mobile Robot Exploration”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Montreal Canada, 2019.

[SHT05] A. Segal, D. Haehnel, and S. Thrun. “Generalized-ICP”. In: *the Robotics Science & Systems (RSS)*. Seattle, USA, 2005.

[SI07] C. Siagian and L. Itti. “Rapid Biologically-Inspired Scene Classification Using Features Shared with Visual Attention”. In: vol. Volume 29. Number 2. 2007.

[SIB03] S. Schaal, A. Ijspeert, and A. Billard. “Computational approaches to motor learning by imitation”. In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 358.1431 (2003).

[Sil+12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. “Indoor segmentation and support inference from rgbd images”. In: *the European Conference on Computer Vision (ECCV)*. Florence Italy, 2012.

[Sim+00a] Reid Simmons, David Apfelbaum, Wolfram Burgard, and Dieter Fox. “Coordination for multi-robot exploration and mapping”. In: Austin TX, USA, 2000.

[Sim+00b] Reid Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Håkan Younes. “Coordination for multi-robot exploration and mapping”. In: *the AAAI National Conference on Artificial Intelligence*. Austin TX, USA, 2000.

[Sin+07] Amarjeet Singh, Andreas Krause, Carlos Guestrin, WJ Kaiser, and Maxim Batalin. “Efficient Planning of Informative Paths for Multiple Robots”. In: *the International Conference on Artificial Intelligence (IJCAI)*. Hyderabad India, 2007.

[Sin+14] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel. “BigBIRD: A large-scale 3D database of object instances”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Sip96] M. Sipser. *Introduction to the Theory of Computation*. Thomson Publishing, 1996.

[SL+91] J.J.E. Slotine, W. Li, et al. *Applied nonlinear control*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1991.

[SLX15] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. “Sun rgb-d: A rgb-d scene understanding benchmark suite”. In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. Boston MA, USA, 2015.

[SM00] Jianbo Shi and Jitendra Malik. “Normalized cuts and image segmentation”. In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Volume 22.Number 8 (2000).

[Spe+06] Thorsten Spexard, Shuyin Li, Britta Wrede, Jannik Fritsch, Gerhard Sagerer, Olaf Booij, Zoran Zivkovic, Bas Terwijn, and Ben Kröse. “BIRON, where are you? - Enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[Spl10] John R Spletzer. “On-line calibration of multiple LIDARs on a mobile vehicle platform”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage AK, USA, 2010.

[SSR12] Daniel E. Soltero, Mac Schwager, and Daniela Rus. “Generating informative paths for persistent sensing in unknown environments”. In: Vilamoura Algarve, Portugal, 2012.

[SSR14] Daniel E. Soltero, Mac Schwager, and Daniela Rus. “Decentralized path planning for coverage tasks using gradient descent adaptive control”. In: *the International Journal of Robotics Research (IJRR).* Volume 33.Number 3 (2014).

[Sti00] B. Stilman. *Linguistic Geometry: From Search to Construction*. Kluwer Academic Publishers, 2000.

[Stu+12] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. "A Benchmark for the Evaluation of RGB-D SLAM Systems". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vilamoura Algarve, Portugal, 2012.

[Sug+14] B. Suger, G.D. Tipaldi, L. Spinello, and W. Burgard. "An Approach to Solving Large-Scale SLAM Problems with a Small Memory Footprint". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Sün+16] Niko Sünderhauf, Feras Dayoub, Sean McMahon, Ben Talbot, Ruth Schulz, Peter Corke, Gordon Wyeth, Ben Upcroft, and Michael Milford. "Place categorization and semantic mapping on a mobile robot". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm Sweden, 2016.

[Sun+18] Hao Sun, Zehui Meng, Pey Yuen Tao, and Marcelo H Ang. "Scene Recognition and Object Detection in a Unified Convolutional Neural Network on a Mobile Manipulator". In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane Australia, 2018.

[SX04] Weihua Sheng and Ning Xi. "Multi-robot area exploration with limited-range communications". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sendai Japan, 2004.

[TC06] Elin A. Topp and Henrik I. Christensen. "Topological Modelling for Human Augmented Mapping". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[TC10] E. A. Topp and H. I. Christensen. "Detecting Region Transitions for Human-Augmented Mapping". In: *the IEEE Transactions on Robotics (T-RO)* Volume 26.Number 4 (2010).

[Thr98] S. Thrun. "Learning Metric-Topological Maps for Indoor Mobile Robot Navigation". In: *the Artificial Intelligence Journal (AIJ)* Volume 99.Number 1 (1998).

[TL03] S. Thrun and Y. Liu. "Multi-Robot SLAM With Sparse Extended Information Filters". In: *the International Symposium of Robotics Research*. Siena Italy, 2003.

[TMT10] A. Toshev, P. Mordohai, and B. Taskar. "Detecting and Parsing Architecture at City Scale from Range Data". In: *the Conference on Computer Vision and Pattern Recognition(CVPR)*. San Francisco CA, USA, 2010.

[TNC10] Alex Trevor, John G. Rogersand Carlos Nieto-Granda, and Henrik Iskov Christensen. "Tables, Counters, and Shelves: Semantic Mapping of Surfaces in 3D". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Workshop: Semantic Mapping and Autonomous Knowledge Acquisition.* Taipei Taiwan, 2010.

[TRC12] A. J. B. Trevor, J. G. Rogers III, and H.I. Christensen. “Planar Surface SLAM with 3D and 2D Sensors”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Minneapolis MN, USA, 2012.

[TRC14a] Alexander JB Trevor, John G Rogers, and Henrik I Christensen. “Omnimapper: A modular multimodal mapping framework”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[TRC14b] Alexander JB Trevor, John G Rogers III, and Henrik I Christensen. “Omnimapper: A modular multimodal mapping framework”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong China, 2014.

[Tre+10a] A. J. B. Trevor, J. G. Rogers III, C. Nieto-Granda, and H.I. Christensen. “Tables, Counters, and Shelves: Semantic Mapping of Surfaces in 3D”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Workshop on Semantic Mapping and Autonomous Knowledge Acquisition. Taipei Taiwan, 2010.

[Tre+10b] Alex Trevor, John G. Rogers III, Carlos Nieto, and Henrik Christensen. “Virtual Measurements for Robotic Mapping”. In: *GTRC ID 5160 Invention Disclosure Serial number 61/328, 818.* (2010).

[Tre+10c] Alexander J. B. Trevor, John G. Rogers, Carlos Nieto-Granda, and Henrik Iskov Christensen. “Applying domain knowledge to SLAM using virtual measurements”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Anchorage AK, USA, 2010.

[Tre+11] A. Trevor, J. G. Rogers III, C. Nieto, and H. Christensen. “Feature-based Mapping with Grounded Landmark and Place Labels”. In: *the Robotics Science & Systems (RSS) Workshop on Grounding Human-Robot Dialog for Spatial Tasks*. Los Angeles CA, USA, 2011.

[Tre+13] Alexander J. B. Trevor, Suat Gedikli, Radu B. Rusu, and Henrik I. Christensen. “Efficient Organized Point Cloud Segmentation with Connected Components”. In: *Semantic Perception Mapping and Exploration (SPME)*. Karlsruhe Germany, 2013.

[TWP11] David R. Thompson, David S. Wettergreen, and Francisco J. Calderon Peralta. “Autonomous Science during Large-Scale Robotic Survey”. In: *the Journal of Field Robotics* Volume 28.Number 4 (2011).

[UB03] D. Urting and Y. Berbers. “Marineblue: A Low-cost Chess Robot”. In: *Robotics and Applications* (2003).

[UH05] Ranjith Unnikrishnan and Martial Hebert. “Fast extrinsic calibration of a laser rangefinder to a camera”. In: *Carnegie Mellon University (CMU-RI-TR-05-09)* (2005).

[Ull+08] M. M. Ullah, A. Pronobis, B. Caputo, J. Luo, P. Jensfelt, and H. I. Christensen. “Towards Robust Place Recognition for Robot Localization”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena CA, USA, 2008.

[Val+13] R. Valencia, M. Morta, J. Andrade-Cetto, and J.M. Porta. “Planning Reliable Paths With Pose SLAM”. In: *the IEEE Transactions on Robotics (T-RO)* Volume 29.Number 4 (2013).

[Vin+08] R. Vincent, D. Fox, J. Ko, K. Konolige, B. Limketkai, B. Morisset, C. Ortiz, D. Schulz, and B. Stewart. “Distributed multirobot exploration, mapping, and task allocation”. In: *Annals of Mathematics and Artificial Intelligence* Volume 52.Number 2 (2008).

[VPA12] J. Van den Berg, S. Patil, and R. Alterovitz. “Motion planning under uncertainty using iterative local optimization in belief space”. In: *the International Journal of Robotics Research (IJRR).* Volume 31.Number 11 (2012).

[WBF11] René Wagner, Oliver Birbach, and Udo Frese. “Rapid development of manifold-based graph optimization systems for multi-sensor calibration and SLAM”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* San Francisco CA, USA, 2011.

[WCR09] J. Wu, H. I. Christensen, and J. M. Rehg. “Visual Place Categorization: Problem, Dataset, and Algorithm”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* St. Louis MO, USA, 2009.

[Wil92] R.H. Wilson. “On Geometric Assembly Planning.” PhD thesis. Stanford University, 1992.

[WMU13] Michael Warren, David McKinnon, and Ben Upcroft. “Online calibration of stereo rigs for long-term autonomy”. In: *the IEEE International Conference on Robotics and Automation (ICRA).* Tokyo Japan, 2013.

[WR11] J. Wu and J. M. Rehg. “CENTRIST: A Visual Descriptor for Scene Categorization”. In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Volume 33.Number 8 (2011).

[WS03] D. Wolf and G. S. Sukhatme. “Towards Mapping Dynamic Environments”. In: *the International Conference on Advanced Robotics (ICAR).* Coimbra Portugal, 2003.

[WS04] D. Wolf and G. S. Sukhatme. “Online Simultaneous Localization and Mapping in Dynamic Environments”. In: *the IEEE International Conference on Robotics and Automation (ICRA).* New Orleans LA, USA, 2004.

[WS05] D. Wolf and G. S. Sukhatme. “Mobile Robot Simultaneous Localization and Mapping in Dynamic Environments”. In: *Autonomous Robots* Volume 19.Issue 1 (2005).

[WS07] D. Wagner and D. Schmalstieg. “ARToolKitPlus for Pose Tracking on Mobile Devices”. In: *the Proceedings of the Computer Vision Winter Workshop.* St. Lambrecht Austria, 2007.

[WT02] C. Wang and C. Thorpe. “Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects”. In: *the IEEE International Conference on Robotics and Automation (ICRA).* Washington DC, USA, 2002.

[WTT03] C. Wang, C. Thorpe, and S. Thrun. “Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas”. In: *the IEEE International Conference on Robotics and Automation (ICRA)*. Taipei Taiwan, 2003.

[XOT13] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. “Sun3d: A database of big spaces reconstructed using SFM and object labels”. In: *the International Conference on Computer Vision (ICCV)*. Sidney Australia, 2013.

[Xu+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. “Show, attend and tell: Neural image caption generation with visual attention”. In: *the International Conference on Machine Learning (ICML)*. Lille France, 2015.

[Yam98] Brian Yamauchi. “Frontier-based exploration using multiple robots”. In: *the International Conference on Autonomous agents*. Minneapoli MN, USA, 1998.

[YMH02] H. Ye, A.N. Michel, and L. Hou. “Stability theory for hybrid dynamical systems”. In: *the IEEE Transactions on Automatic Control* Volume 43.Number 4 (2002).

[YW12] Hao Yang and Jianxin Wu. “Object templates for visual place categorization”. In: *Asian Conference on Computer Vision (ACCV)*. Daejeon Korea, 2012.

[ZCS07] Huijing Zhao, Yuzhong Chen, and Ryosuke Shibasaki. “An efficient extrinsic calibration of a multiple laser scanners and cameras’ sensor system on a mobile platform”. In: *the Proceedings on the IEEE Intelligent Vehicles Symposium*. Istanbul Turkey, 2007.

[ZHD13] L. Zhao, S. Huang, and G. Dissanayake. “Linear SLAM: A linear solution to the feature-based and pose graph SLAM based on submap joining”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Tokyo Japan, 2013.

[Zhe+11] K. Zheng, D.F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. “How many social robots can one operator control?” In: *the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. Lausanne Switzerland, 2011.

[Zho+17] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. “Places: A 10 million Image Database for Scene Recognition”. In: *the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Volume 40 (2017).

[ZK16] Sergey Zagoruyko and Nikos Komodakis. “Wide residual networks”. In: *arXiv preprint arXiv:1605.07146* (2016).

[Zlo+02] R. Zlot, A. Stenz, M. B. Dias, and S. Thayer. “Multi-Robot Exploration Controller by a Market Economy”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Lausanne Switzerland, 2002.

[ZP04] Q. Zhang and R. Pless. “Extrinsic calibration of a camera and laser range finder (improves camera calibration)”. In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Chicago IL, USA, 2004.

[ZR06] X. Zhou and S.I. Roumeliotis. "Multi-robot SLAM with unknown initial correspondence: The robot rendezvous case". In: *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Beijing China, 2006.

[ZST15] Chao Zhang, Arunesh Sinha, and Milind Tambe. "Keeping pace with criminals: Designing patrol allocation against adaptive opportunistic criminals." In: *the International conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Istanbul Turkey, 2015.