Applications of Magnetic Particle Imaging to Brain Imaging

by

Ryan Daniel Orendorff

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Joint Doctor of Philosophy with University of California, San Francisco

in

Bioengineering

in the

Graduate Division

of the

University of California, Berkeley

Summer 2017

Applications of Magnetic Particle Imaging to Brain Imaging

Abstract

Applications of Magnetic Particle Imaging to Brain Imaging

by

Ryan Daniel Orendorff

Joint Doctor of Philosophy with University of California, San Francisco in Bioengineering

University of California, Berkeley

Professor Steven M. Conolly, Chair

Magnetic Particle Imaging is a novel imaging modality with many applications in the preclinical, and soon, clinical space. In particular, MPI shows promise for imaging of various brain related pathologies, such as stroke, hemmorhage, and traumatic brain injury. In this thesis, we demonstrate a series of first-in-animal proof-of-concept experiments that MPI could soon be superior to our conventional imaging technologies, including X-ray CT, MRI, ultrasounds and nuclear medicine for particular neuroimaging applications. In the process, we will develop algorithmic enhancements to the image reconstruction of MPI signals in order to achieve real time interventional imaging, much like X-ray fluoroscopic imaging, but without ionizing radiation and significant risks of catheter and iodinated contrast media.

To my parents and friends

Contents

List of Figures

Acknowledgments

The acquisition of a doctoral degree is not just the accomplishments of a singular person, but a triumph for the community they surround themselves with.

I would like to acknowledge my advisor, Professor Steven Conolly. Originally I thought he was too distant from the work at hand; it was only partway through the doctoral process that I understood his light touch was just the right amount of encouragement and assistance to allow me to grow as an independent researcher. Perhaps more importantly, Steve cares about his graduate students as if we were related: he cares about the events in our lives, our well beings, and our futures. We were never just researchers to him, and he cultivated a quirky family in the lab. I am humbly grateful for his guidance, patience, passion, goofiness, and kindness. I matured not only as a researcher with his help, but as a person as well.

Dr. Patrick Goodwill is *the* driving force behind the method of MPI performed in the United States, and a leader of the field. He has defined the field for over a decade. It is without a doubt that we are all here because of his work and guidance. He has been an excellent mentor and a friend.

My labmates made the graduate school experience feel more like good times with the family, even as we pushed hard on our work. Xinyi Zhao (周新懿) brought more than just cookies to the lab: she kept us grounded and provided insights that kept us moving forward, giving the lab a vision. I am also grateful for the discussions we had on LGBT culture and the support she gives graduate LGBT students by hosting and organizing events. Zhi Wei Tay (智伟) pushed the limits of MPI in unprecedented ways; his knack for ignoring commonly held beliefs about the abilities of MPI opened up completely new fields of research. Dr. Paul Keselman taught me much about MRI and about how to love living on a really steep hill. Dr. Kuan Lu (路宽) brought a certain light and happiness to the lab that would raise everyone's spirits, and then drop some incredible theory into our laps that had our jaws hitting the floor. Dr. Justin Konkle set the (very) high bar on image reconstruction methods and theory, and was an insightful and kind mentor to me. Dr. Prashant Chandrasekharan has been immensely helpful in applying MPI to the preclinical imaging space, proving the viability of MPI. Our visiting scholars like Nicole Hinterberger, Friso Heslinga, and Yannick Bliesener have all been great friends and given us fresh perspectives on our work.

The undergraduate researchers I have had the pleasure of working with have provided me with an opportunity to grow as a mentor, and always obliterated any challenge I placed in front of them. Evan Frenklak is an incredibly skilled computer scientist that can pick up anything thrown at him. Mindy Bishop doesn't know the definition of failure—her only mode is kicking butt. She is a wonderful friend, and I have cherished the times I have had with her, her daughter Sophia, and her husband.

I would like to thank my friend Dr. Claus Danielson for not only teaching an excellent course on controls but for also bringing me on as an intern at MERL. I have learned an incredible amount from him, and his math chops are something to behold. My time at MERL was filled with intense discussions, great friendships, and an unfathomable amount of ice cream from a local parlor.

Dr. Daniel Hensley and I formed a near instant friendship at the start of the PhD program. Our mutual interest in many topics led to a plethora of deep discussions that were enlightening for each of us. Our friendship helped me grow personally and intellectually, and I appreciate the time he has spent with me on this journey.

Dr. Elaine Yu (游于藝) is the living incarnation of the word effervescent. Her kindness, patience, and goofiness brought up the spirits of everyone in the lab, and in many ways she acted as the glue in our lab family. Her engineering chops are truly incredible. It has been an honor to learn and grow alongside her.

I must thank Dr. Bo Zheng (郑波). He is an incredibly talented researcher; his work and method of approaching problems amazes me every time we interact, even after five years. Beyond the lab, he exemplifies the person I strive to be: kind, patient, humble, and selfless. He is one of my personal heroes, a person I deeply admire and respect—I am lucky to have had so much time to learn from him.

This would not have been possible without my friends, housemates, and the Berkeley Bioengineering graduate student group. I have no idea how or why you guys put up with me, but you did. I have had some amazing adventures and made lifelong friends during my time at Berkeley.

Finally, I would like to thank my parents. I am surprised that they kept me around after I vomited on their heads as a baby, or when I decided to do a middle school project on an ancient compass at the last possible second and dragged them into it. But they persisted! They have shown me what unconditional love is, and have been my personal cheer squad through it all.

We did it! Congratulations everyone.

Chapter 1

Introduction to Magnetic Particle Imaging

Magnetic particle imaging (MPI) is a novel tracer-based medical imaging modality that shows great promise for imaging biologically important phenomenon with good resolution, high signal to noise ratio (SNR), linearity and shift invariance, and fast acquisition capabilities. Here we will introduce the basics of magnetic particle imaging (MPI), and compare MPI with various other imaging technologies.

1.1 Basics of Mpi

The formation of images with MPI are described in detail here: [1, 2, 3, 4, 5, 6, 7]. We briefly summarize the basic theory behind MPI.

Super paramagnetic iron oxide nanoparticles (SPIOs) are nanoscale iron particles with a diameter of approximately 30 nm or less that have a single magnetic domain that, in aggregate, align to an applied magnetic field almost instantaneously and without any coercivity. The magnetic field of a population of particles in response to an applied magnetic field can be modeled by the Langevin function [2], as shown in Figure 1.1. For small amplitude applied magnetic fields, the particle population creates a net magnetization with an approximately linear dependence on the applied field. However, at larger applied field strengths, the particles "saturate", meaning that nearly all of the particles have aligned with the applied field and therefore higher field strengths have diminishing effects on the resulting magnetization response from the particles.

Magnetic particle imaging exploits this saturation phenomenon to image only a small number of particles within a volume. Magnets in MPI are designed to oppose each other and create a magnetic field gradient, with a field free region (FFR) in the center that experiences zero applied field (to within some tolerance). This arrangement allows particles to linearly respond to an applied magnetic field inside the FFR while eliminating signal outside of the FFR due to the saturation of the particles. To produce an image, the FFR is rastered over

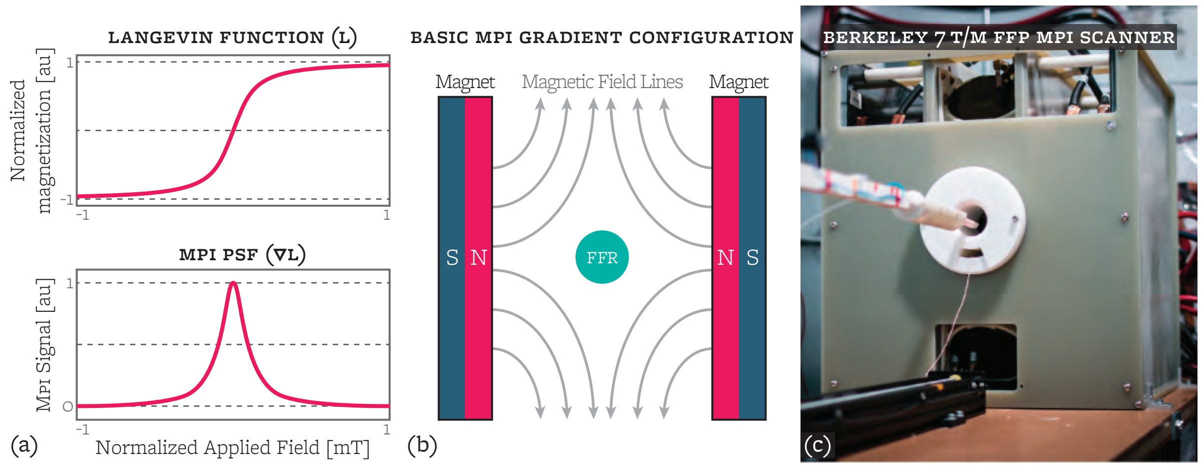

LANGEVIN FUNCTION (L)
Normalized magnetization [au]
1
-1
-1
1
MPI PSF (∇L)
MPI Signal [au]
1
0
-1
1
(a)
Normalized Applied Field [mT]
BASIC MPI GRADIENT CONFIGURATION
Magnet
Magnetic Field Lines
Magnet
S N
FFR
N S
(b)
BERKELEY 7 T/M FFP MPI SCANNER
(c)

Figure 1.1: Introduction on magnetic particle imaging (MPI). (a top) MPI abides by the physics of the SPIO nanoparticles that it images. The SPIO particles have a nonlinear magnetization response to an applied field, characterized by the Langevin function. When the applied magnetic field is low, the particles have an approximately linear response to the applied field; however, when the applied magnetic field is strengthened, the response magnetization in the SPIO particles saturates. (a bottom) MPI detects the derivative of the SPIO magnetic response, which forms MPI's PSF. (b) All MPI scanners must make a gradient magnetic field to exploit the Langevin physics. In this case, two permanent magnets are placed opposite each other to create a magnetic field gradient. In the middle of the magnets is a field free region (FFR) where the particles have a near linear dependence on the applied field: any particles not located in the FFR experience a saturating applied field, and therefore produce no signal. The MPI scanner creates an image by rastering the FFR over space to acquire signal from the SPIO particle distribution at each sample point. (c) An image of the Berkeley FFP scanner used in this report.

space to acquire the magnetization response of the particles as the FFR passes over them. As MPI detection is done through inductive means, the derivative of the magnetization response is captured. As described in [2, 7], MPI is a linear and shift invariant (LSI) system with a spatially-invariant 1D point-spread function (PSF)—the derivative of the Langevin function—as shown in Figure 1.1.

1.2 Comparison of MPI to other imaging modalities

The MPI modality offers some excellent properties over its main competitors, positron emission tomography (PET), single photon emission computed tomography (SPECT), and fluoroscopic imaging. All of these modalities are known as tracer based imaging systems, which are generally characterized by the response of some injected contrast agent to some perturbation to produce a signal that is detected. In addition, MPI competes with imaging modalities that images all the anatomy inside the imaging volume, such as computed tomography (CT) or magnetic resonance imaging (MRI). These imaging methods can often have enhanced contrast by injections of a contrast agent, such as iodine for CT or gadolinium in the case of MRI.

MPI compares with those modalities in the following manner.

- *Non-toxic tracer without ionizing radiation*: in PET and SPECT imaging, a radioactive tracer is injected into the patient, which decays to produce light that is picked up by a receiver outside of the patient. The tracer is generated by attaching a positron-emitting radionuclide onto a host molecule, often glucose in the form of fludeoxyglucose. As this molecule decays, it produces gamma rays that are detected by a ring of detectors outside of the patient. This process exposes the patient to the ionizing radiation of the gamma rays, limiting the dose that can be given to a patient.

  In contrast, MPI uses an iron-oxide tracer that does not produce any ionizing radiation in the process of acquiring an MPI signal. While making tracers in MPI is still an emerging field, some common tracers such as feramoxitol are common iron supplements that are given to patients suffering from anemia. Additionally, the safe dosage range with such iron supplements is large and the MPI signal can be detected from a very minute amount of iron, allowing MPI to operate within already defined safety limits for iron supplement exposure.

- *Long duration studies*: the iron tracers used in magnetic particle imaging are non-toxic, and the signal from the iron particles does not degrade with time (unlike CT or SPECT tracers). This allows for long term studies to be performed. This manuscript describes one several-day study for traumatic brain injury, and there have been publications showing long term tracking of different nanoparticle types over weeks, tracking of nanoparticle uptake in cancerous tissues, and others.

- *No attenuation or scattering*: MPI signals do not change with regards to the depth of the iron tracer within a sample. This property is not shared with many of its brethren: PET and SPECT can both experience scattering and absorption events that cause artifacts within an image, and fluorescence based imaging techniques experience heavy scattering that leads to depth dependent resolution and signal strength. CT (including contrast enhanced variants) also experiences scattering and absorption; while the scattering is malevolent in terms of producing artifacts, the absorption is the quantity measured and does not pose a problem. MRI is similar to MPI in that it also experiences no attenuation with depth.

- *Quantitative (linear and shift invariant)*: iron signals gathered in MPI have a property known as linear and shift invariant (LSI). The linear component means that if twice as much iron is placed at a point, twice the signal is produced. The shift invariance means that a point of iron produces the same signal at any location in space. This allows MPI signals to be quantitative, in that MPI images give an exact quantity of iron in a given voxel. This allows researchers to determine exactly how much iron is in a given vessel or absorbed by a given organ.

- *Resolution*: The resolution in MPI is currently comparable to (or a tad better than) PET and SPECT imaging at approximately a millimeter on newer MPI architectures. This resolution is poorer than what can be achieved in MRI and CT at the moment, although new developments in particle signal responses and nanoparticle design are rapidly changing the MPI field.
- *Temporal resolution*: The temporal resolution of current MPI systems can be as fast as 30 frames per second, although this speed is achieved at lower resolutions. At millimeter resolutions, the current speed achievable is approximately half a second per frame, as shown in a later chapter. In comparison, PET and SPECT scans can take on the order of a few minutes per acquisitions, while CT and MRI can image fast enough for interventional methods.
- *Small molecule dynamics*: PET and SPECT have the leg up on MPI in terms of tracking small molecule dynamics. Since the tracers used in these modalities can be attached to biologically active molecules such as glucose, these modalities can track the uptake and metabolism of small molecules. Additionally, these small molecules allow for better imaging inside the brain tissue itself, as the tracers are small enough to pass through the blood brain barrier.
- *Contrast or attenuation from tissues*: One of the advantages of MPI is that it does not image anything other than the injected iron. This means that, if the iron can be directed towards an area of interest in the body, the contrast in the image is equivalent to the signal in the image. However, the downside of this ability is that there is no reference for where a particular signal comes from. MPI signals share this property in common with PET and SPECT, where the image after acquisition is some amorphous blob shape in a black background. In order to overcome these limitations, tracer based imaging modalities are often paired with CT or MRI in order to co-register the tracer images with an anatomical reference.

1.3 Current challenges in magnetic particle imaging

The age of MPI (approximately 12 years as of this writing) means that there are still many open challenges in the field to solve. These fall into a few categories.

1.3.1 Nanoparticle physics

Designing SPIO particles suitable for MPI is a rich and active area of research [5, 8]. In current MPI imaging setups, SPIO particles are ideally monodisperse, and each iron core is surrounded by an organic coating. The physics of the iron core and the chemistry of the coating allows for tailoring the particles to many applications, allowing researchers to alter attributes including blood circulation times, particle clearance routes, ability to permeate tissues, adsorption, cellular uptake, and ability to act as a biomarker. As an extra degree of freedom, polydisperse particle distributions have been shown to produce MPI signals.

The design of iron core has implications for the imaging physics as well; in particular, image resolution and signal intensity per gram of iron can be improved by choosing the appropriate iron crystal formulation. Some other properties must be carefully tuned to the application at hand. A particle with a significant blood half-life may be unsuitable for imaging transient phenomenon such as blood volume or flow changes, while a particle with a short half-life may not experience significant uptake in some biological models. Tailoring the blood half-life is quite important if multiple imaging sessions of a dynamic phenomenon are to be performed serially; if the particles are too slow to clear before a new injection is given then particles from prior scans may introduce an undesirable baseline signal and disrupt attempts to analyze the system of interest.

1.3.2 Hardware

While great strides have been made in detecting small concentrations of iron nanoparticles in MPI, there are still many open hardware challenges to be tackled. The current designs are coil-noise or amplifier noise dominated, meaning that the noise from the detection system dominate noise from the patient. New imaging hardware designs are devised and developed on a yearly cycle, leading to a rich literature of hardware devices that each have their own benefits and drawbacks. The hardware has also yet to scale to a size beyond imaging a rabbit, meaning that attempting to build human-sized scanners is only just beginning. However, there is absolutely no fundamental physics barrier to human MPI; after MPI-tailored SPIOs are advanced significantly, then the cost of a human scanner with roughly 0.7T/m gradients should be safe, 1 mm resolution, highly sensitive (1 micromolar) and cost effective.

1.3.3 Imaging theory and reconstruction

The method by which MPI acquires and converts a signal into an image is unlike the processes done in other modalities. In other medical imaging techniques, the method by which to reconstruct images is well known, and algorithms that are developed introduce subtle changes in image quality or other metrics. The age of MPI means that it does not yet have this wealth of image reconstruction techniques, and many avenues of image reconstruction have yet to be explored.

In this thesis, we will investigate novel image reconstruction methods that enable online (or interventional) imaging. These methods are critical to enabling clinical use of MPI; many dynamic imaging methods require an interventional response or real time analysis from the physician performing the imaging.

1.4 Thesis Order

The manuscript is divided into the following sections.

1. Traumatic Brain Injury (TBI) imaging, a new approach to quantifying the results of a traumatic impact on the skull.
2. Brain perfusion imaging, demonstrating MPIs potential as a technology in diagnosing types of strokes, brain hemorrhaging, aneurysms, and more.
3. A matrix-free optimization framework called PyOp, which allows for optimization problems to be solved in Python using far fewer resources that with standard matrix methods.
4. Sparse reconstruction in field free line (FFL) MPI, which allows for radially undersampled data to be reconstructed. This allows for the acquisition time to be decreased by approximately 4 fold.

The thesis is roughly divided into two major components. The first section describes novel applications of MPI, showing the flexibility and versatility of the technique. The second section focuses on the algorithmic advances in image reconstruction that assist in enabling faster acquisition speeds, which in turn enables a larger application space for MPI.

Part I

Brain Imaging Applications using Magnetic Particle Imaging

Chapter 2

First *in-vivo* Traumatic Brain Injury Imaging via Magnetic Particle Imaging

2.1 Introduction to TBI

The Centers for Disease Control and Prevention estimates 1.6–2.1 million emergency visits for Traumatic Brain Injury (TBI) annually in the US alone, with additional cases unaccounted for or untreated [9, 10]. Despite the prevalence of TBI in humans and various animal models of TBI [11], classifying the severity of injury remains an open challenge.

In clinical settings, diagnoses and recovery are assessed using neuropsychological batteries and the Glasgow Coma Scale (GCTS), with efforts to develop a more robust classification system under way [12, 13]. Depending on initial clinical presentation, physicians may order X-ray computed tomography (CT) or diffusion tensor magnetic resonance imaging (DTI) to visualize potential hemorrhages, lesions, or contusions caused by severe impacts [14, 15]. While these techniques are suitable for severe TBI injuries, many mild to medium grade impacts go undiagnosed, untreated, or are deemed too mild to necessitate imaging [14]. Severity is determined by observing motor, reflex, cognitive, and histological changes using the modified Neurological Severity Score (MNSS), cognitive tasks, and post mortem analysis, respectively [16, 17].

Magnetic particle imaging (MPI) is a promising new modality that images a distribution of superparamagnetic iron oxide (SPIO) nanoparticles inside the body [1]. MPI has many advantageous properties, including a safe tracer [18], zero ionizing radiation, minuscule changes in magnetic reporter signal over time, 200-nM sensitivity [19, 20], fast image acquisition and reconstruction [4, 21], robust linear and shift invariant (LSI) properties [7], and sub-millimeter resolution [22]. In addition, particles can be tracked for long periods of time in both the blood (with a half-life of approximately 3 hours) and inside cells (with a half-life on the order of months [23]). Due to their size, the iron oxide tracers are excreted through the liver and spleen, instead of the kidneys [23]. This has prompted interest in using one SPIO agent (Ferumoxytol) as a kidney-safe MRI T1 contrast agent for patients with chronic kidney disease [24], who are at risk for contrast induced nephropathy from iodine contrast agents [25].

Compared to contrast-enhanced CT and MRI, MPI has favorable contrast since there is no signal from obscuring tissue; MPI produces images with excellent contrast to noise ratio (CNR) as only the iron particles produce a signal. This property makes MPI well suited for angiography and blood pool imaging, as only the blood appears in the resulting image due to the particles staying inside the vasculature. The only limitation to imaging capillary-level perfusion or blood volume is signal to noise ratio (SNR). Typically, in CT angiography (CTA) and MR angiography (MRA), our ability to discern a 5% blood volume in voxels with capillaries (without larger-diameter blood vessels) is obscured by so-called "partial volume" effects; essentially the background signal variations are indistinguishable from the minute variations of contrast agent in the capillaries. Since MPI does not detect the tissue surrounding a capillary—only the particles in the capillaries themselves—it is mostly robust to this partial volume effect.

In this report, we present preliminary data and methods showing that MPI can provide high contrast to noise ratio (CNR) images of internal bleeding, and in particular cerebral bleeding caused by TBI. This application requires all the fundamental advantages of MPI, including excellent contrast, sensitivity, spatial resolution and long circulation time.

2.2 Materials and Methods

To assess the ability of MPI to detect internal hemorrhaging, a closed head TBI model was used with two Fisher-344 female rats. We briefly describe the associated induction procedure; see Figure 2.1 for a graphical representation of the technique.

2.2.1 Closed Skull Tbi Induction

All TBI animal experiments were approved by the UC Berkeley Animal Care and Use Committee (ACUC).

Both animals were female Fisher-344 rats. The TBI impact animal weighted 207 g while the control weighed 136 g.

Each rat was anesthetized using 3% isoflurane and a bolus of 2.4 mg/kg LS-13 SPIOs was administered via tail vein injection. One animal was then placed in a rectangular animal bed filled with a plush bedding material underneath a custom designed closed impact TBI device (Figure 2.1). The animal's head was placed under a bolt with a hex nut attached to the threaded end, with the hex nut gently resting on the animal's skull near the lambda skull suture (see FIgure 2.2). A 454 g weight—consisting of a Falcon tube filled with lead shot pellets—was then calibrated to drop from one meter above the top of the bolt. A forepaw pinch was performed to confirm that the animal was still under the effects of the isoflurane,

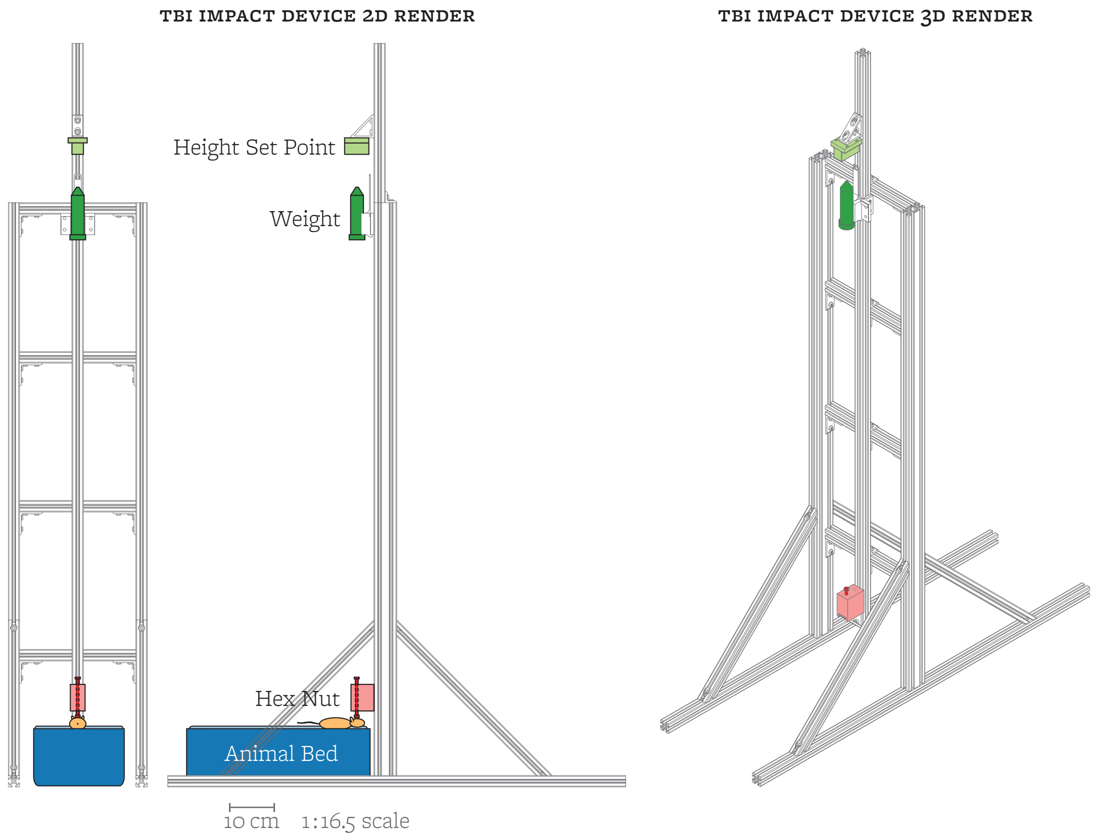


Figure 2.1: Diagram of the TBI impact device, which imparts a closed skull TBI injury onto an animal. An impact procedure is briefly described as follows. A 50 mL Falcon tube (dark green) is filled with lead pellets to the desired weight and the height set point block (light green) is set to the desired drop height. The animal (light orange) is placed in the animal bed (dark blue) on top of a soft bedding material (light blue). The weight is then dropped onto a bolt resting on the animal's skull (dark red). The bolt is held in place by a 3D printed part (light red). Impact device is drawn to a 1:16.5 scale with respect to the page.

after which the weight was released. The second animal went through the same procedure as a sham control without the weight drop.

After being in the TBI impact device, both animals were allowed to wake up to check for signs of lethal trauma such as erratic movements, high frequency whisker twitches, and (in the worst case) mortality.

2.2.2 Mpi Imaging Protocol

A detailed design of the Berkeley field free point (FFP) scanner can be found in the literature [3]; here we describe only the imaging parameters.

Following the injury and post-operative checks, both animals were imaged in the Berkeley FFP MPI scanner with gradient strengths of 7 T/m by 3.5 T/m by 3.5 T/m in x, y, and z, respectively. The drive field amplitude was set to 40 mTpp at a frequency of 20.225 kHz along the axis of the bore (z). Selection field electromagnets moved the FFP in the x and y directions to acquire samples on a rectilinear grid for one XY slab through the imaging volume. The width of each slab was 0.91 cm in z. 40 XY slabs were acquired at increments of 0.2 cm in z.

Scans were performed with the following parameters: 4.0 cm by 3.75 cm by 8.5 cm field of view (FOV) (x, y, z) using LS-13 SPIO particles. The acquisition time was measured to be approximately ten minutes.

X-Space image reconstruction—including DC recovery and slab stitching—was performed as described by [7]. No deconvolution was applied to the image.

The longitudinal analysis was performed by imaging the animals over successive days using the same scan parameters. Before each scan, the animals were anesthetized with isoflurane and kept under anesthesia for the duration of the scan. The images were then registered with the initial image using a cross-correlation method [26]. Regions of interest were then selected within these images and the maximum signal in each region over time was plotted; see Figure 2.5.

Anatomical x-ray images were taken using a Kubtec XPERT 40 cabinet, with an energy range of 10 kV to 50 kV, a tube current up to 1.0 mA, and a resolution of 10 lp/mm. The animals were placed in contact-mode into the cabinet and imaged using the auto-calibration setting.

2.3 Results

Signal in the region affected by the TBI impact can be clearly seen in the MPI scans, both immediately after and a few days following the impact (see Figure 2.5). The acquired MPI images indicate that SPIO particles accumulated quickly after impact at some location around the skull and persisted for a two week period. Additionally, X-ray scans of the TBI animal indicated a small skull fracture in a $\in$-shape near lambda in the XZ X-ray projection.

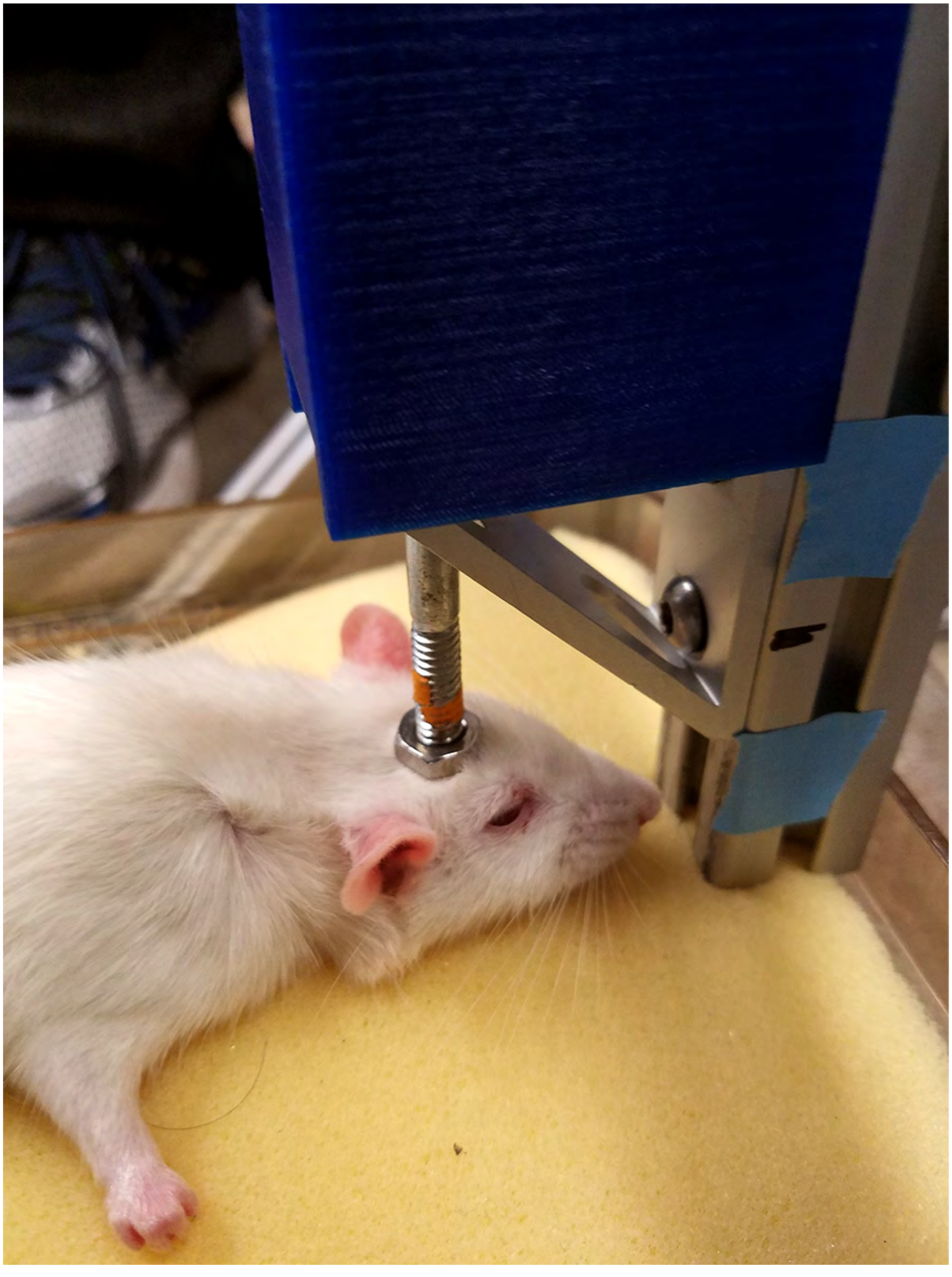


Figure 2.2: Animal in the TBI impactor. A 450 gram weight is dropped onto the screw/hex nut from 1.1 meters, causing a head injury approximately at the blue circle.

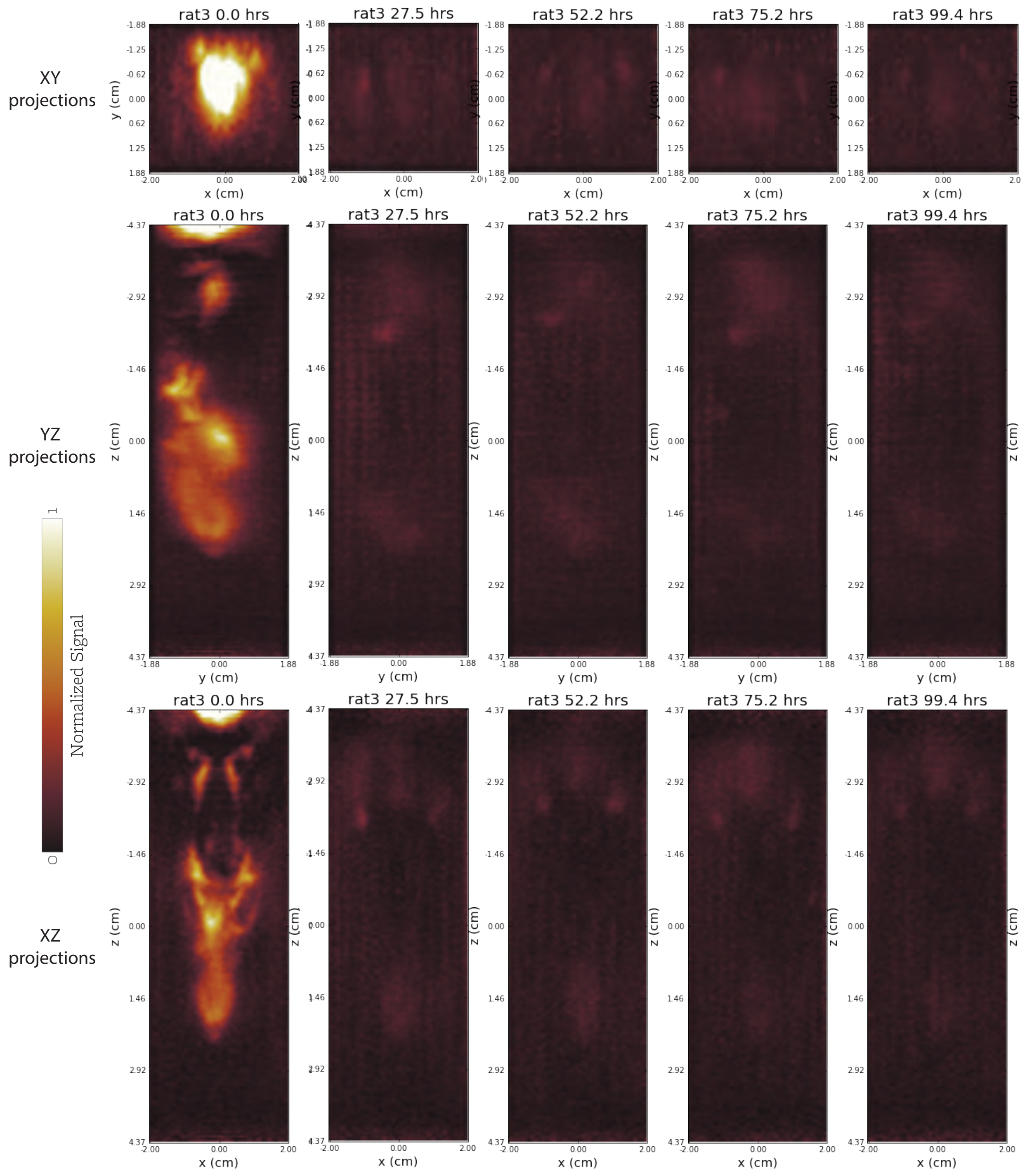


Figure 2.3: Maximum intensity projection images of the image acquisition sequence over time for the control animal. Each time point is approximately 24 hours apart. The images show a quick decrease in the signal over the imaging days. All images are plotted with the same intensity

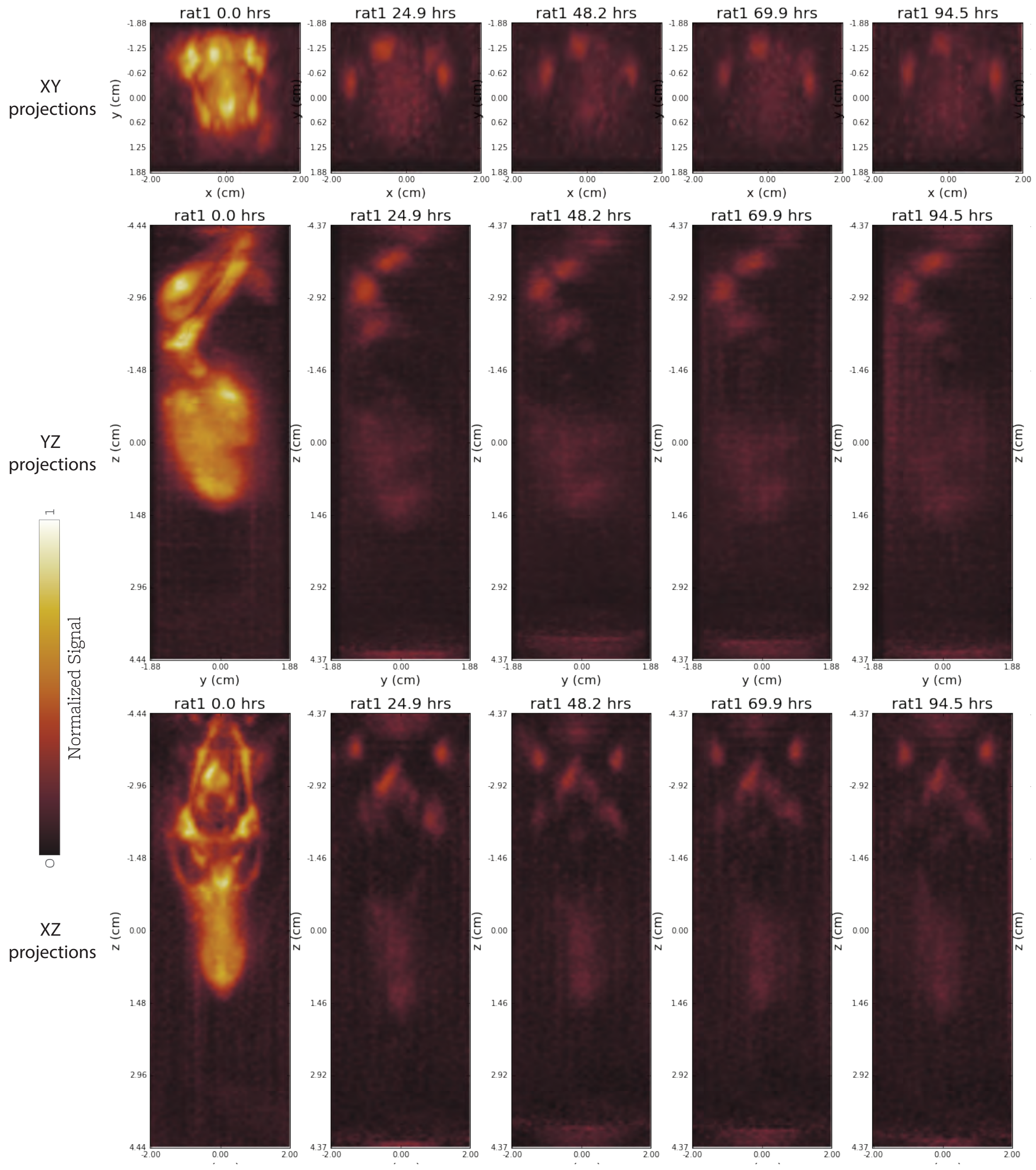


Figure 2.4: Maximum intensity projection images of the image acquisition sequence over time for the TBI animal. Each time point is approximately 24 hours apart. The images show a slower decrease in iron signal in the brain of the animal over time in comparison to the control. All images are plotted with the same intensity

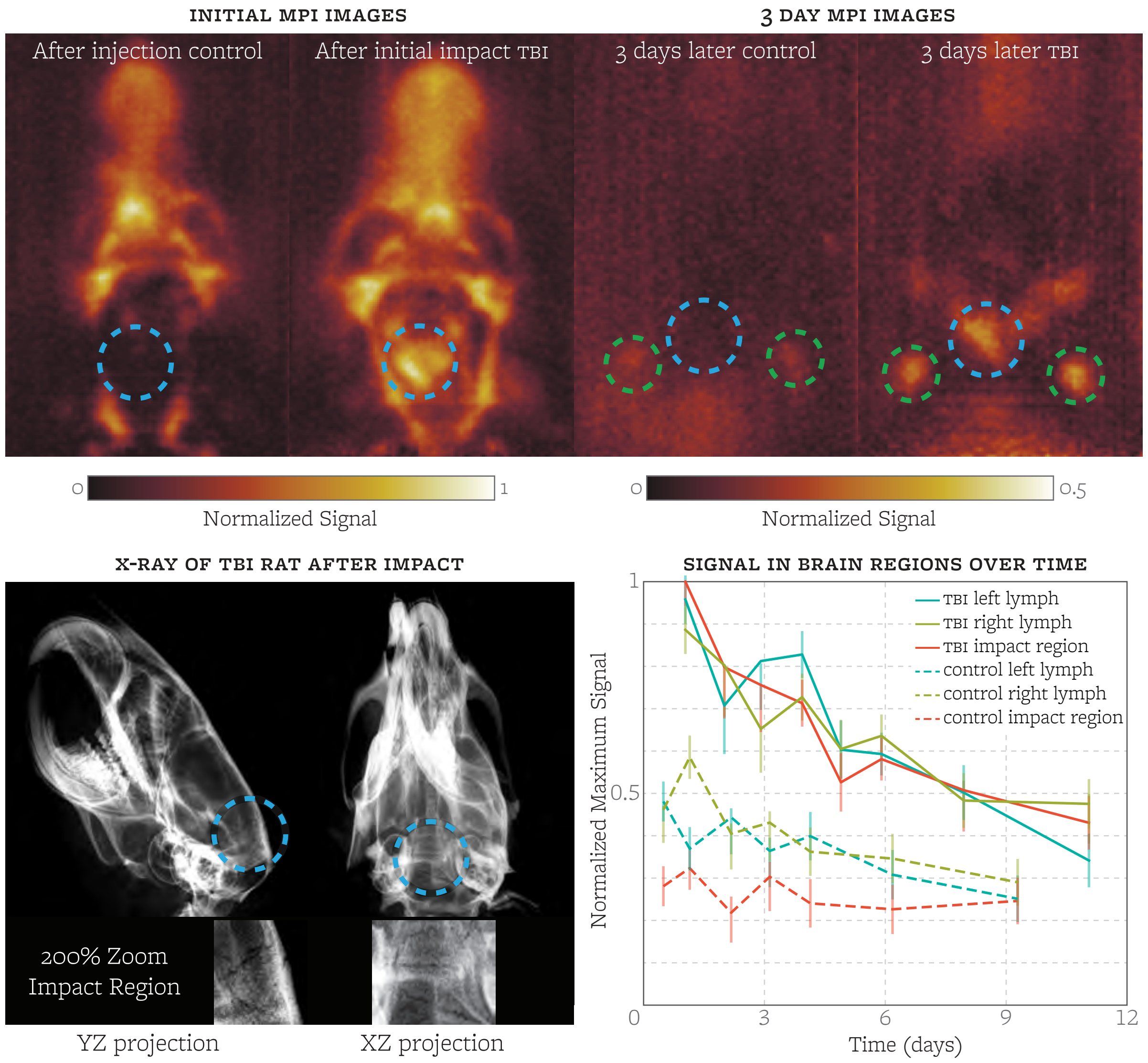


Figure 2.5: (Top left) XZ maximum intensity projection (MIP) of both animals after injection/impact. Note the large hemorrhage in the TBI animal. (Top right) MIP of the animals after 3 days; blue circle represents the impact site, green circles indicate lymph nodes. The TBI rat continues to have significant signal from the hemorrhage and inside the lymph nodes, unlike the control. The field of view was 4.00 cm by 3.75 cm by 8.5 cm (x, y, z). (Bottom left) X-ray images taken of the rat that underwent the TBI impact. A faint ϵ-shaped fracture can be seen at the back of the head in the XZ projection. (Bottom right) Signal from the SPIO nanoparticles over time inside the skull. The clearance rate at the point of injury and the lymph nodes in the TBI rat is significantly slower than in the control animal. Error bars represent two standard deviations of the background noise at each time point.

The clearance rate of particles at the impact site and locations within certain regions of the head were significantly retarded from the estimated six hour blood half-life of LS-13 particles (Figure 2.5). The half-life of the particle signal in the impact region was approximately four days for the TBI animal, while no signal above the noise was detected in the equivalent area of the sham animal.

In both animals, signal accumulated in small regions on both sides in the neck. Due to the location of the signal and the clearance pathway, these regions are suspected to be lymph nodes. In both animals, the lymph nodes started to appear in the MPI images after approximately half a day. The clearance half-life rate for the control animal was three days, while the TBI animal experienced a slightly delayed clearance of four days. The lymph node signal was still visible in the TBI animal at the last time point of eleven days, while the lymph signal in the control animal was still visible at the last time point of nine days. The bottom right pane of Figure 2.5 provides a more detailed illustration of the particle signal over time in both the impact region and lymph region.

2.4 Discussion

The acquired MPI images indicate that the internal bleeding was significant enough to allow for SPIO particles to infiltrate into blood pools in the interstitial space. Signal decay of the control is consistent with typical clearance of SPIOs through the circulatory system and liver [27, 28]. Furthermore, appearance of signal lateral to the cervical spine with unique decay rates suggests SPIO clearance mechanisms from cerebral interstitium are facilitated through the brain's lymphatic system which have been shown to drain to submandibular and cervical lymph nodes [29, 30]. Recent reports have also shown meningeal localization of these vessels [31, 32]. While further experiments must be performed for validation, the clearance through lymphatic nodes provides a method by which to observe and analyze the lymphatic pathways in the skull.

The results show potential for MPI in clinical settings. The bleeding caused by the TBI was visible both immediately after the injury as well as many days post injury. The ability to monitor the change in the impact zone provides a metric for measuring the healing process as blood is cleared from the interstitial space.

In this study particles were injected prior to injury: this scenario is unlikely to occur in clinical practice. As such, further studies should be performed to determine the uptake of particles after an injury occurred. Of particular interest would be the uptake over time, as it would indicate whether the particles would need to be injected soon after the injury was suspected or if this injection could be delayed until arriving at a clinic.

The ability to image hemorrhages in MPI makes it a potential augmentation or replacement for ruling out hemmorhhages following brain trauma or following stroke, which is currently done by CT. While CT is able to image severe hemorrhages quite well without a contrast agent, it remains an open challenge to image mild to medium severity injuries [14]. Part of the challenge lies in the contrast to noise ratio in imaging modalities like CT, where signal of interest must be compared to a similarly strong signal from background tissues. MPI does not detect background tissues and the MPI signal is completely unattenuated with depth in tissue, allowing the hemorrhage to show up with unprecedented contrast and sensitivity.

2.5 Conclusion

Our work represents the first visualization of primary brain injury and associated hematoma in any TBI model using MPI. Currently investigators and clinicians rely on changes in the Neurological Severity Score (NSS) and GCTS, which primarily evaluate cognition, motor function, and loss of reflexes to evaluate injury severity. We demonstrate that MPI can be used to determine site, severity, and depth of bleeding from a closed skull TBI model regardless of brain regions targeted and behavioral manifestation of injury.

Furthermore, by demonstrating the ability to detect extravasation of blood associated SPIO into extravascular space we propose that MPI is a modality well suited to localizing and quantifying internal bleeding regardless of pathophysiological cause. Additional experiments will be performed to analyze how the iron particles migrate in the brain after a TBI event. These results demonstrate a potential future for noninvasive diagnosis of internal bleeding for patients suffering from trauma in the emergency setting using MPI.

Acknowledgements

We are grateful for funding support from the Keck Foundation Grant 009323, NIH 1R01EB019458, NIH 1R24MH106053, and a UC Discovery Grant. LodeSpin Labs is also grateful for NIH 2R42EB013520-02A1. Ryan Orendorff would like to thank the NSF GRFP for funding support.

Chapter 3

Brain Perfusion Imaging in Magnetic Particle Imaging

3.1 Stroke Prevalence, Presentation, and Diagnosis

Strokes represent the fifth leading cause of the deaths in the United States, with over 795,000 cases per year and 130,000 deaths [33, 34]; this represents a stroke every 40 seconds and a death every 4 minutes in the US [35]. Stroke is *the leading preventable cause of disability* in the US [35]. When a stroke occurs, the patient demonstrates symptoms that are often described by the acronym FAST[36]: Facial drooping (potentially one sided), weakness in the Arms or other extremities, slurred or incoherent Speech, and finally Time to call emergency services. Persons experiencing these symptoms are rushed to medical services, as time is a significant factor in the clinical outcome for the patient. The guideline is that the patient should get to the clinic within six hours, as after this point brain tissue can become necrotic due to a lack of oxygen and nutrients.

Strokes are generally classified into two different causes: ischemic stroke and hemorrhagic stroke. In ischemic stroke (approximately 87% of cases [33]), an artery within the brain has experienced a blockage within the vessel that prevents blood flow to a section of the brain, while a hemorrhagic stroke (13% of cases) is when a vessel bleeds into the intracranial space. It is possible to treat both, although in slightly different manners. Ischemic stroke is often treated with some form of a “clot-buster” such as a tissue plasminogen activator (tPA) [34, 37], which attempts to dissolve the clot and thereby restore blood flow to the affected parts of the brain. Treating hemorrhagic stroke is done by a very different method [34]: the patient may be given drugs to thicken the blood or lower blood pressure, or simple supportive medical care may be performed.

The clinical need to diagnose the cause of stroke is due to the opposing natures of the two types of stroke [38]. Using a clot-busting drug like tPA will greatly increase the survivability and clinical outcome of patients with ischemic stroke; in cases where patients are suffering a hemorrhagic stroke, pushing drugs such as tPA will only cause them to bleed out more, leading to worse clinical outcomes and potentially death. Similarly, if patients with ischemic stroke are given the treatment for hemorrhagic stroke then the arterial blockage will not be addressed and the affected brain tissue may die, leading to potentially poor clinical outcomes for the patient. In general the medical doctors (MDs) at the emergency doctor (ER) must make a decision within 3 hours in order to save the most amount of brain tissue [33].

3.2 Current methods for diagnosis

In order to diagnose stroke and other related diseases, doctors use CT [39, 40], CT perfusion (CTP) [41, 42, 43], and CTA [39] to determine blood flow rate and volume through the cerebral vasculature. While these techniques work well for determining large-scale deficits like hemorrhagic stroke, they are fundamentally ill-equipped to track blood changes over time due to high background signal from the tissue surrounding the vasculature [44]. Additionally, CTP and CTA have high radiation dose and use an iodinated contrast agent that is toxic for patients with chronic kidney disease (CKD); the majority of those who have a stroke have some form of CKD [45].

Magnetic particle imaging (MPI) is a promising new modality that images only a safe magnetic tracer, commonly SPIO nanoparticles. MPI contains no signal from tissue, and thus can see the cerebral vascular clearly as iron nanoparticles flow through the brain. In this section we shall show some of the first results demonstrating the potential to use MPI in diagnosing stroke.

3.3 Methods to calculate brain perfusion metrics in dynamic imaging modalities

The methods for diagnosing strokes operate in one of two ways: either a static image is captured in the attempt to identify a hemorrhagic stroke, or dynamic (time series) imaging is performed to determine areas of the brain that are receiving less blood flow or volume, or that experience a delayed delivery of a bolus of blood compared to neighboring tissues [46]. Here, we will be discussing methods that focus on identifying ischemic stroke from dynamic images.

Dynamic imaging allows clinicians to identify two pieces of tissue within the brain that are affected by an ischemic stroke: an ischemic core of dead tissue and a section of nearby tissue that is at risk of becoming necrotic, known as the penumbra [47]. Identification of the core indicates how much of the brain is permanently damaged, while the penumbra is an marker of what tissue could possibly be saved by clinical intervention.

There are generally four metrics, visualized as maps of the relevant values over a brain slice, that are used to determine the core and penumbra [48, 49, 43, 50, 51].

- Mean transit time (MTT): a measure of how long it takes for a bolus of blood to pass through a location, in units of seconds.
- Cerebral blood flow (CBF) and Relative cerebral blood flow (rCBF): a metric of the blood flow at a location, in units of mL/100 mg brain tissue per minute.
- Cerebral blood volume (CBV) and Relative cerebral blood volume (rCBV): a measure of the blood volume at a particular location, in units of mL/100 mg brain tissue.
- Time to peak (TTP): a measure of how long it takes for a bolus of blood to arrive at a location, in units of seconds.

3.3.1 Indicator Dilution Theory

The way in which these metrics are measured is commonly done by using indicator dilution theory. This theory assumes that the brain vasculature is a single input, single output stationary flow system with an instantaneous bolus injection of an amount of a tracer with mass m. This allows for the steady state flow to be measured by the simple relation [48, 52]

$$\int_{t_a}^{t_d} c(t)\,dt = \frac{m}{CBF} \tag{3.1}$$

where $c(t)$ is the concentration of the tracer at time t in units of mol/L, CBF is the stationary flow rate, m is the mass of the injected tracer at $t = 0$, and t_a and t_d are the times when the tracer appears and disappears at the outlet, respectively. Intuitively, this equation is simply another statement of the conservation of mass; the change in mass can be expressed as the how much material flows through a volume over time ($dm = c(t)Q\,dt$), and all the mass must that entered the system must leave the system.

From here, it is possible to calculate the time it takes for the bolus to pass through the system, known as the mean transit time.

$$MTT = \frac{\int_{t_a}^{t_d} c(t)t\,dt}{\int_{t_a}^{t_d} c(t)\,dt} \tag{3.2}$$

The volume of fluid that passes during this time can then be simply calculated from the flow rate and the amount of time.

$$CBV = CBF \cdot MTT \tag{3.3}$$

This model allows for the calculation of the relevant parameters (with TTP coming from the position of the peak of $c(t)$) in the case of an instantaneous bolus. However, it is not possible to administer an instantaneous bolus due to the time of the injection and the broadening that the bolus experiences due to friction. As such, this model is often augmented with an additional measurement of the arterial input function ($a(t)$), which can capture the distortion of the bolus from an impulse input [51]. The measured concentration signal is then modeled as a convolution.

$$c_m(t) = c(t) * a(t) \tag{3.4}$$

One can then use deconvolution to determine the concentration response as if the input were an impulse response and thereby use the equations from the indicator dilution theory.

There are approximations for calculating some of these parameters. rCBF can be estimated by calculating the maximum slope of the bolus injection under the original impulse assumption ($\max(dc_m(t)/dt),\ \forall t \in [t_a, t_d]$). Additionally, rCBV can be calculated as the total material to pass through a specific location during the bolus ($\int_{t_a}^{t_d} c_m(t)\,dt$) [48, 53].

This theory is used for most of the dynamic imaging studies, whether in CT or in MRI perfusion studies. MPI can use the same theory when tracking the passage of iron nanoparticles through the brain vasculature. In this manuscript, we measure the relative metrics as they are simpler to obtain without knowledge of the arterial input function (AIF), and because medical metric of importance is the differences between the CBV, CBF, and MTT maps, not the quantitative value. However, MPI can measure the AIF if the artery supplying the brain is measured during the scan, directly providing the AIF.

3.4 Verification of iron nanoparticle tracer using MRI

We have attempted to capture perfusion metrics in three different ways to verify the potential for MPI to be used in perfusion imaging studies.

First, we wanted to verify that the iron nanoparticles used in MPI (specifically Resovist [6]) can be used to image perfusion through the brain vasculature by capturing the passage of Resovist using MRI. In this type of imaging, the iron nanoparticles act as a negative contrast agent by causing the signal where the Resovist passes through to drop due to T_2^*-dephasing effects [54]. When performing the perfusion study, we are looking for when the signal within the brain *drops*, which is the opposite effect that is observed in CT and MPI perfusion studies.

We can use the change in intensity in the MRI image to calculate the relative concentration curves over time through the brain, in order to get back similar curves to what we expect in MPI and CT [54]

$$c(t) \propto \Delta R_2^*(t) = \frac{-\ln(I(t)/I(0))}{TE} \tag{3.5}$$

where contrast media concentration $c(t)$ is proportional to the relaxation rate $\Delta R_2^*(t) = 1/\Delta T_2^*(t)$; $I(0)$ is MRI signal intensity before contrast arrival (usually from an average of 4-8 pre-contrast images) and $I(t)$ is signal intensity after contrast arrival. Relatively concentration profiles can be compared with this relation, and one can measure the proportionality constant for a particular system to get the real $c(t)$ should that be desired. The relative blood flow and volume can be used in the clinic, so the lack of this proportionality constant does not write off this technique from practical use. Additionally, time based metrics such as mean transit time and time to peak are unaffected by this transformation.

With this understanding, we performed MRI experiments to track the flow of an iron tracer through the brain vasculature.

3.4.1 Materials and Methods

MRI was performed on a Fisher-344 rat over a period of a few minutes; a brief description of this process follows. The rat was first anesthetized using 3% isoflurane. A catheter was then inserted into the tail vein of the rat. Then the rat was placed in the scanner with access to the catheter from outside the imaging bore. An injection of 0.2 mL of 25 mg Fe/mL Resovist iron nanoparticle solution was performed as a bolus into the animal over a period of six seconds.

MRI was used to monitor the iron over time, using an echo planar imaging (EPI) sequence with a TE of 14 ms and a TR of 500 ms. The total acquisition time was 20 seconds with a rate of 2 frames per second. The FOV was 2.4 cm by 4 cm with a 2 mm slice thickness.

3.4.2 Results

We were able to successfully image the transit of Resovist within a rat model. As figure 3.1 demonstrates, the passing of the Resovist bolus can be seen as a darkening of the image (panel b), and then a restoration of the brain tissue signal after the iron passes through the plane. By analyzing the time course curves over each pixel in the image, we are able to derive maps of time for the peak to appear in a given pixel (panel e). The maximum slope of the concentration curve is used to estimate the relative blood flow, as show in panel f.

These results confirm that MRI T_2^* agents can provide quantitative estimates of brain perfusion metrics like TTP and rCBF. What is likely the most important metric out of this data set is an estimate of the mean transit time of a bolus through the animal's brain. We can see from panel d that the time it takes for the signal to decrease as the iron passes through the slice, and then return to baseline, is approximately four seconds (with some error due to the noisiness of the signal). This indicates that for MPI perfusion studies to accurately capture the dynamics of Resovist within this particular animal model, a minimum temporal resolution of at least 0.5 FPS is required in order to capture at least some of the peak dynamics.

3.5 Acquisition of transverse plane using MPI

Given the experiment performed imaging the passing of a resovist bolus of a rat in MRI, we then attempted to perform the same imaging study using MPI. The animal protocol is similar to the MRI study, imaged using the Berkeley FFP MPI scanner instead.

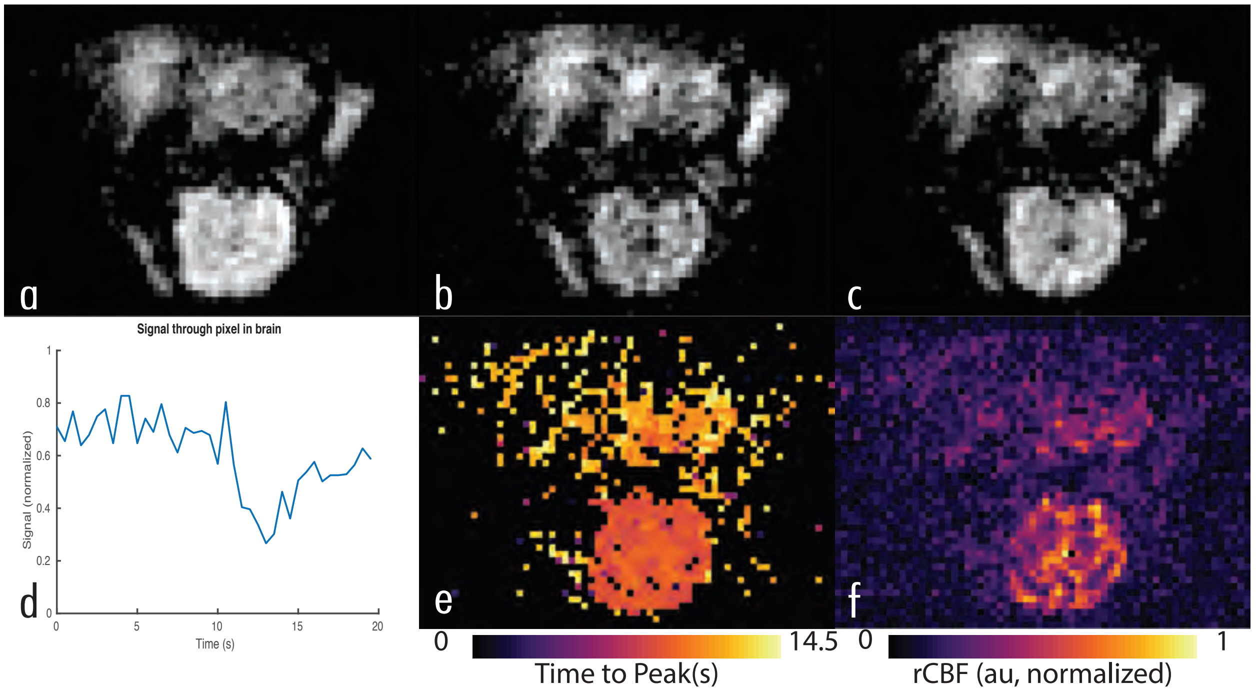
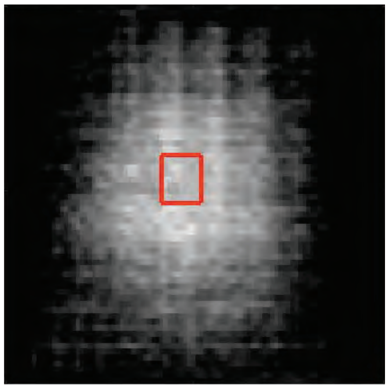
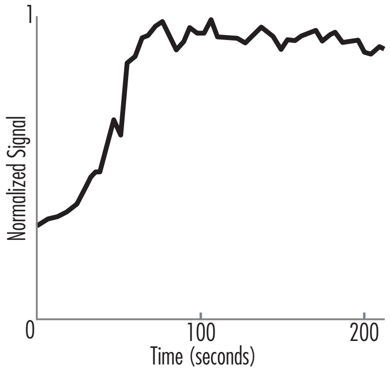
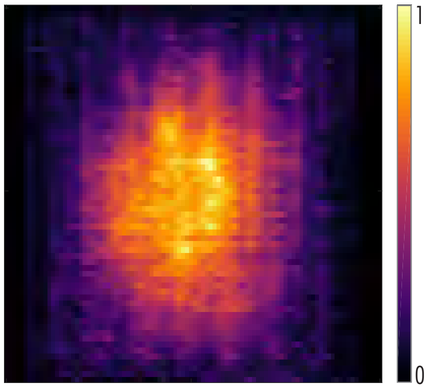


Figure 3.1: Transverse slice of a rat skull as a bolus of Resovist passes. 7 T MRI EPI sequence (TE=14 ms, TR=500 ms). Acquisition time was 20 seconds at a rate of 2 FPS, 2.4 cm by 4 cm FOV, with a 2 mm slice thickness. Tracer was Resovist at a concentration of 5.7 mg/ml at a volume of 0.2 mL per injection. a) Image of the rat skull immediately after the injection, before the bolus reaches the brain. The brain is seen as the brain white structure in the bottom of the image, while the other structures are from the jaw. Significant blowout is seen due to the air in the ears, which limited which brain slice could be acquired. b) The Resovist bolus passes through the rat brain, decreasing the signal temporarily as shown by the lower signal inside the brain. This occurs at the 12.5 second mark. c) The rat skull at the end of the 20 second scan after the bolus has passed. The brain signal has largely returned to normal. d) The signal of a single pixel in the brain over time, showing the effect of the bolus as it passes through the brain. e) A map showing the time taken for bolus to cause the largest decrease in signal from the initial time point, from the . f) The relative cerebral blood flow (rCBF) through the brain.


(a) MPI image at 210 seconds



(b) Time course through red region in (a)



(c) Relative Cerebral Blood Flow (rCBF)

Figure 3.2: (a) An MPI image of a resovist bolus 210 seconds after the start of the acquisition. We can see signal from the brain, although substructures are not apparent in this image due to the PSF of resovist. (b) An time course of the signal through the red region in part (a), averaged at each time point to increase the SNR of the signal. We can see a rise as the resovist passes through the brain, but no peak. The non-zero y-crossing of the time course is due to iron circulating from a prior injection. (c) A map of the relative cerebral blood flow, measured using the maximum derivative of the signal time course at each pixel in the image.

3.5.1 Materials and Methods

MPI was performed on a Fisher-344 rat over a period of a few minutes; a brief description of this process follows. The rat was first anesthetized using 3% isoflurane. A catheter was then inserted into the tail vein of the rat. Then the rat was placed in the scanner with access to the catheter from outside the imaging bore. 0.2 mL of 25 mg Fe/mL Resovist iron nanoparticle solution was injected as a bolus into the animal over a period of six seconds. MPI was used to monitor the iron over time.

Imaging was performed using the Berkeley field free point (FFP) scanner. This MPI scanner has a gradient strength of 7 T/m by 3.5 T/m by 3.5 T/m in the x, y, and z directions, respectively. A single transverse, xy plane through the rat's brain was captured at a rate of one frame every 2.16 seconds, for a period of three and a half minutes. The field of view was 3.25 cm by 3.25 cm. The time series images were then processed to produce a map of the relative cerebral blood flow (rCBF) through each pixel in the image. This calculation was done by finding the maximum rate of change of the iron signal through each pixel in the image, using a Savitzky-Golay filter to calculate the derivative of the signal. The maximum slope at each pixel was then plotted to produce an rCBF map.

3.5.2 Results

This experiment represents some of the first *in-vivo* perfusion results in MPI. For this imaging study, we were able to show that we were able to capture the MPI time course passing of the resovist through the transverse plane of the rat. However, we note that we were not able to detect a peak in the MPI signal. We believe this is due to the poor temporal resolution, which could have skipped over the peak signal.

We also note a potential problem for the MPI version of perfusion imaging; after the bolus passes, a large residual signal exists. We shall see a similar trend in later experiments, and we hypothesize that this is due to the relatively long blood half-life of Resovist, which is approximately 15 minutes. MRI imaging of the resovist would also have this problem, of which there are indications of this in the time course series shown in Figure 3.1. However, the MRI signal change is unlikely to be sensitive to smaller concentrations of iron than MPI, as MPI is incredibly sensitive. Most variants or calculations using indicator dilution theory assume a return to baseline signal after the bolus passes, and this high signal after the bolus decreases the conditioning of certain analysis tools such as gamma-variant fitting.

These results indicate a promise of using MPI for perfusion, but by missing the peak signal we cannot say that these results are definitive proof of the potential for MPI. The calculation of CBF, CBV, and MTT all require an accurate assessment of the bolus shape, and unfortunately this experiment missed that particular signal. However, the acquired signal does show the flushing of the MPI signal through the brain tissue, which represents a first step in getting to perfusion imaging.

3.6 Acquisition of horizontal plane

We would have liked to continue scanning the transverse plane, as it both offers potentially greater acquisition speed while being more relevant to researchers analysing rat brain perfusion studies, but alas the scanner broke due to an accident relating to cooling. As such, we tried something new! We decided to alter the current FFP MPI scanner to enable the fast image acquisition in a horizontal plane, which enabled us to acquire more perfusion data (albeit in a different plane).

3.6.1 Alterations to the Berkeley FFP scanner to enable horizontal plane imaging

In order to continue acquiring dynamic images, we altered the way in which the FFP electromagnets generate their shift field. The Berkeley FFP scanner (see Figure 1.1) has electromagnets used to move the magnetic field slowly the y axis of the scanner, as show in part a of Figure 3.3. In this configuration, the scanner cannot normally shift the field in the z direction, as there are no electromagnets currently on the system to perform this movement (although they could be added). Instead of adding new electromagnets, we altered the di-

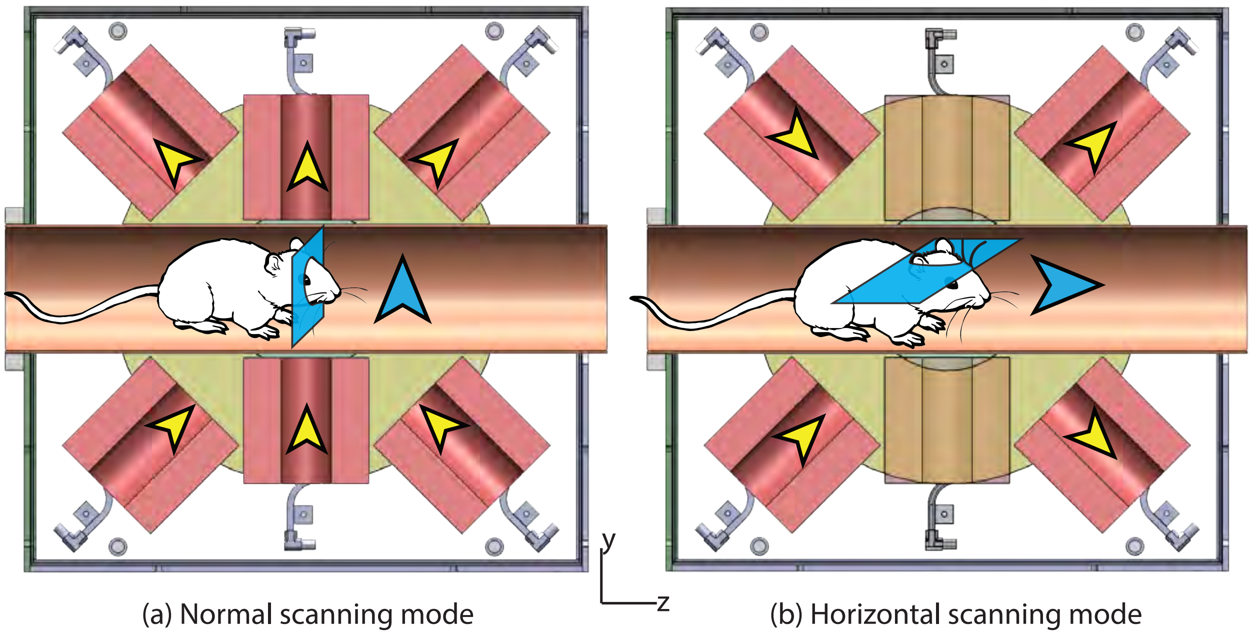


Figure 3.3: Alterations to the Berkeley 7 T/m FFP scanner to enable horizontal plane imaging. (a) an image of the Berkeley FFP from an yz cross-section of the scanner. The red cylinders are electromagnets used to move the magnetic field slowly inside the bore (copper cylinder), with the yellow arrows indicating the direction of the field produced by the magnets. The blue arrow shows the summation of the contribution of all of the electromagnets, leading to a shift in the y direction. (b) By altering the direction of the currents through the electromagnets (and disabling the two middle magnets), it is possible to get the magnetic field to shift in the z direction instead of the y direction.

rection of the current going through some of the electromagnets, allowing the scanner to acquire a horizontal plane through the animal.

3.6.2 Alterations to the image reconstruction algorithm

After these hardware modifications, slight alterations were performed to the software in order to put the correct current waveforms through the coils during acquisition. This process mirrors the normal scanning sequence performed with the FFP point scanner exactly, and hence the only reconstruction changes that needed were to reorient some intermediate data structures in the code to the correct format to be called by the reconstruction software.

However, in order to enable interventional style imaging, slight changes were made to several components of the reconstruction pipeline. From a high level, the reconstruction down into the following components [2].

1. Filtering of the time domain signal using a high pass filter. This both removes the

feedthrough experienced in MPI signals as well as undoes a nonlinear phase alteration performed by an analog filter applied before the digital acquisition.

2. Gridding of the time domain signal to a spatial signal. Each point in the time series signal corresponds to a certain position of the FFR in the imaging volume, meaning that the time domain signal and the spatial location are bijective via the scanning sequence.

3. constant (DC) reconstruction of the image, as the DC component of the signal in the spatial signal is lost due to corruption from the feedthrough.

The components that take the most amount of time are the filtering step and the DC recovery. The filtering was originally performed using the default Matlab implementation, which was not particularly fast. This was replaced with a faster, fast fourier transform (FFT) based implementation of filtering. In addition, the DC recovery was performed using the optimization techniques discussed in the next section, with a warm start for each acquired frame.

With these alterations, the rate limiting step was still the filtering. However, the DC recovery could be done at approximately 1/30th of a second, and the filtering could be done around 1/3th of a second, meaning that reconstruction can occur at approximately 3 frames per second (FPS). For an acquisition that lasts 61 seconds (see below), the reconstruction time was 21.48 seconds, meaning that image reconstruction can complete within the acquisition of the next frame, as desired.

3.6.3 Materials and Methods

MPI was performed on a Fisher-344 rat; a brief description follows. The rat was anesthetized using 3% isoflurane and a catheter was inserted into a tail vein. The rat was then placed in a MPI scanner with access to the catheter from outside the scanner. 0.2 mL of 5.7 mg Fe/mL Resovist iron nanoparticle solution was injected into tail vein over a period of six seconds.

The result was imaged using the Berkeley 7 T/m FFP scanner at 1.28 FPS with a 2.6 cm by 2.6 cm field of view. The imaging was performed continuously over a period of 160 seconds (Figure 3.5). The time series images were processed to produce a map of the relative cerebral blood flow (rCBF) in the image. This calculation was done by finding the maximum rate of change of the iron signal through each pixel in the image, which was plotted to produce an rCBF map (see Figure 3.4).

3.6.4 Results

With this experiment, we have managed to acquire similar data as before but in a different anatomical plane (see Figure 3.4 and Figure 3.5. This leads credence to a consistent phenomenon matching our imaging expectations. This method shows that our acquisition speed is still a challenge, as we are only able to see the peak of iron flow through the brain for

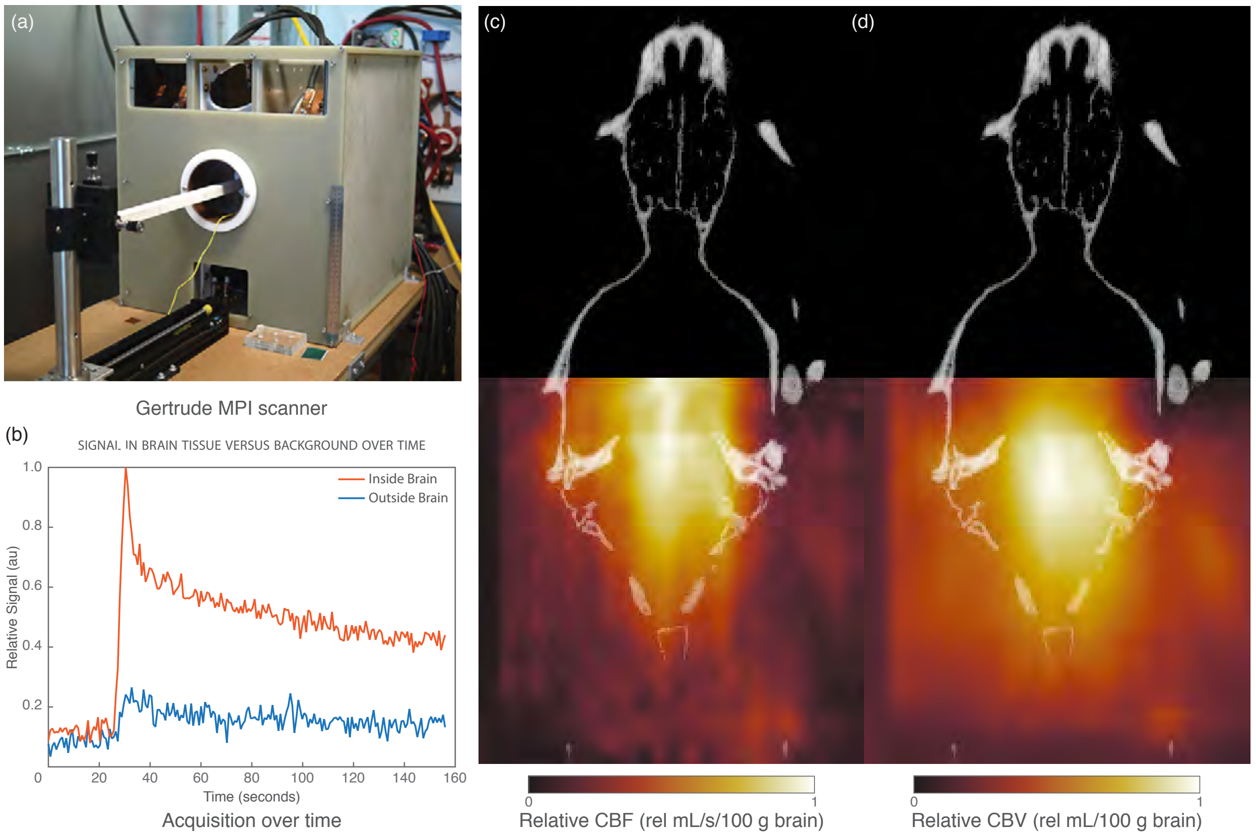

(a)
Gertrude MPI scanner
(b)
SIGNAL IN BRAIN TISSUE VERSUS BACKGROUND OVER TIME
Inside Brain
Outside Brain
Relative Signal (au)
Time (seconds)
Acquisition over time
(c)
(d)
Relative CBF (rel mL/s/100 g brain)
Relative CBV (rel mL/100 g brain)

Figure 3.4: Here we demonstrate in-vivo perfusion maps of the horizontal plane created by MPI. (a) Image of the Berkeley field free point (FFP) MPI scanner, named Gertrude. This scanner has a gradient field of 7 T/m by 3.5 T/m by 3.5 T/m respectively. For this experiment, we imaged anesthetized Fisher-344 rats that were catheterized via their tail vein. A bolus of Resovist iron nanoparticles were injected into the animal over a period of six seconds, and the result imaged at a 1.28 FPS with a 2.6 cm by 2.6 cm field of view. (b) The time course plot shows a marked peak increase in signal as the particles first pass through the brain, followed by a slow decay as the particles recirculate in the bloodstream and are filtered out by the liver. (c) A measure of the relative Cerebral Blood Flow (rCBF) for each pixel in the image, clearly showing high perfusion rates inside the brain. (d) A measure of the relative Cerebral Blood Volume (rCBV) for each pixel in the image, which is a measure of the amount of iron/blood flowing through the brain as the first bolus passes.

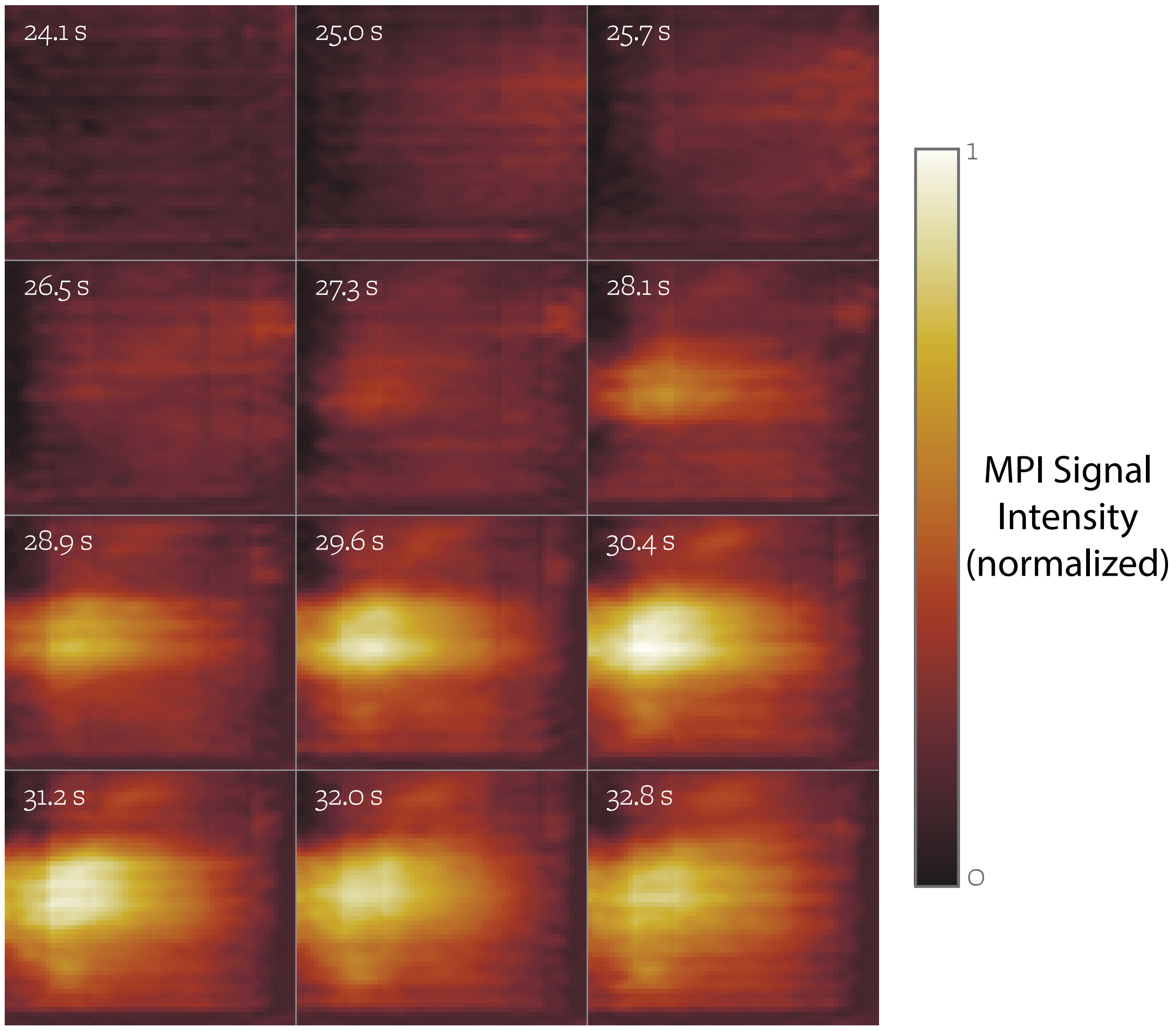


Figure 3.5: Time series of the horizontal plane perfusion images shown in Figure 3.4. These images (with timestamps in the upper left) show how the MPI signal evolved in the rat's brain over time. We can see no signal in the initial images, followed by a flushing as resovist flows into the vessels of the brain. This signal peaks at approximately 30 seconds after the start of imaging, after which a steady state of iron in the blood is reached.

one or two frames at most. This data set also further demonstrates the slow removal of iron from the vasculature over time, in a similar manner to the transverse plane images.

We believe that this data indicates that MPI does show promise but requires even faster acquisition rates. In addition, the high signal after the peak could be mitigated by particles that clear from the blood quickly, which would allow for a faster return to baseline.

3.7 Extensions of perfusion studies for stroke imaging

The MPI perfusion studies showed promise, although were plagued with some unfortunate losses in hardware due to heating and other problems over time.

We had an approved animal protocol to perform the same experiment, but in animals where a stroke was induced unilaterally in one hemisphere to demonstrate the ability of MPI to image some physiological phenomenon of note. In order to perform these studies, one can induce strokes in an animal using a photothrombotic method [55], where a dye is injected into the animal and then coaxed to coagulate in the presence of a strong light applied to the brain. This procedure can even be done without any type of surgery whatsoever, using only an injection of a dye (in our case, rose bengal) and using a strong enough light source placed directly on the skull.

We perform some initial testing of this method (along with MRI imaging) to determine the efficacy of this method. It appeared that we were able to induce strokes in a targeted area in Fisher-344 female rats without performing any sort of surgery. This allows for less pain for the animals and simpler animal protocols. Alas, we were not able to execute this protocol due to a lack of certification to administer pain medication, as well as some final major hardware bugs that limited the ability to acquire the desired images.

In addition to stroke based models, it should be possible perform functional imaging using MPI and this acquisition technique, as some preliminary (unpublished) data from colleagues indicates that the MPI signal changes when the animal is exposed to a hypoxic environment for a brief period of time [56].

3.8 Conclusion

MPI shows promise for perfusion imaging while having many benefits; MPI produces zero radiation while proceeding unprecedented contrast, lack of attenuation from tissues, and quantitative measurements. Akin to nuclear medicine without radiation or cost. There are currently some limitations due to the resolution of the images and the acquisition time is a tad short of what would be sufficient for imaging the bolus as it passes through the brain. However, these challenges are not insurmountable; a focus on hardware designed for faster acquisition times would alleviate many of the challenges, while new particles with lower blood half-life and higher resolution will help with the calculations of MTT and CBV. When these problems are solved, MPI is poised to be able to give images of the brain vasculature with impeccable clarity, and offer clinicians insights into the types of vascular problems that a patient is suffering from in the process to diagnosing and treating stroke.

Part II

Optimization Image Reconstruction Techniques for Magnetic Particle Imaging

Chapter 4

Matrix-Free Optimization Frameworks and Their Application to Magnetic Particle Imaging

Linear systems abound in engineering disciplines such as imaging. Indeed, most if not all of the major imaging systems used in the world are linear and shift invariant (LSI). This is not just happenstance—LSI is a crucial feature for efficiency and quantitation of an imaging system and, for example, its noise properties.

There are major ramifications for image reconstruction given LSI properties. For example, one popular method for image reconstruction is to solve a so-called inverse problem. If a system is LSI, it can always be described with the notation $y = Ax$ where y is the output of the system and A characterizes all of the linear transformations associated with the system. These could include the fundamental physics of the modality, signal processing steps, and the particulars of the scanning sequence. To recover the image x, we must invert this system.

In practice, A is the composition of many linear transformations or constructed from block components, and it is often these different sub-components that one can formulate. For example, we often have

$$A = A_1 A_2 \dots A_N \tag{4.1}$$

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \tag{4.2}$$

or a combination these. As long as we can define each of these sub-components, composition or blocking of linear operators is straightforward. In our image reconstruction problem, we can then find the solution to the inverse problem to reconstruct an image:

$$\min_x \quad \|Ax - y\|_2 \tag{4.3}$$

In general, this problem has an analytical solution given by:

$$\hat{x} = A^{\dagger} y \tag{4.4}$$

where $A^\dagger$ is the Moore-Penrose inverse of A. For ill-conditioned problems, we may add Tikhonov regularization:

$$\min_x \quad \|Ax - y\|_2 + \sqrt{\lambda}\,\|x\|_2 \tag{4.5}$$

which has the analytical solution:

$$\hat{x} = (A^\top A + \lambda I)^{-1} A^\top y \tag{4.6}$$

We may also want to incorporate *a priori* information such as positivity of the image ‘x‘ leading to a convex optimization formulation:

$$\min_x \quad \|Ax - y\|_2^2 \tag{4.7}$$

$$\text{s.t.} \quad x \succeq 0 \tag{4.8}$$

which has no analytical solution.

Often, A and the subcomponents are defined in terms of matrices. When this is possible and easy, then formulating and solving the reconstruction problem is very straightforward. However, in some applications, including image reconstruction, the matrix form (even when sparse and stored accordingly) can be too big to fit in memory and/or cause cache thrashing issues. In these cases, “matrix-free” methods become much more attractive, if not necessary.

A matrix is only one way of representing a linear transformation. In the most general case, a linear transformation need only have some functional representation for its forward operation. However the adjoint operation is often just as necessary. When we encapsulate these operations in the form of a function, we can store this information on the order of kB of memory as opposed to the GBs needed for large matrices.

Considering the image reconstruction problems described above, when couched as a convex optimization problem, gradient descent or other methods can be used to solve the reconstruction problem using only the forward and adjoint operations (compared to the analytical solutions which require an inverse).

In general, when using linear operators, we need to be able to properly formulate the forward and adjoint operations and accommodate various forms of the operator depending on the specific context (e.g., matrix vs. functional). PyOp is designed to give users the flexibility to seamlessly create, compose, and use linear operators defined as dense matrices, sparse matrices, in functional form, or any arbitrary mix.

In this chapter, we develop a python package called PyOp that implements linear operators in a functional, matrix free form. This enables MPI reconstruction techniques to be described as convex optimization problems and solved efficiently.

4.1 Comparison to other matrix-free frameworks

PyOp is but one of many similar frameworks for creating matrix-free operators in a simple fashion in Python. For example, the SciPy project contains a `LinearOperator` class [57], as well as methods on these linear operators such as composition and eigenvalue decomposition. There are other packages such as pyoperators [58] and linop [59] that accomplish similar goals as well. In addition, the optimization package CVX recently starting to implement certain matrix free operations [60].

All of these packages are complete and can be used to perform matrix-free operations. We believe that the simplicity of PyOp's design (based on the fundamentals of lambda calculus and functional programming) enable a robust formulation of the matrix-free operators that has been well tested and documented. In addition, PyOp provides operators and convenient higher order functions specifically for imaging applications, allowing functions on multidimensional arrays to be lifted into the framework operating on matrices with ease.

In the following sections, we will describe how to use PyOp and some results obtained from using PyOp in image reconstruction methods.

4.2 PyOp Design for implementing linear operators

PyOp is based around the simple concept of linear operators as higher order functions. The basic definition of the `LinearOperator` class is not much deeper than its declaration.

```
class LinearOperator(object):

    def __init__(shape, forward, adjoint):
      # Store the input parameters
```

This is to say that `LinearOperators` can be described by the triple (shape, f, f^T), where shape is the shape of the linear operator if it were in matrix form, f is the forward function representing Ax, and f^T is the adjoint function representing A^Tx

The real design of PyOp is not this container (which is shared with many of its brethren), but the simple way in which the operators are composed. For example, the way on which addition of two linear operators A and B can be written as follows, where the first element of the pair is the forward and the second element is the adjoint function.

$$A + B = (x \mapsto Ax + Bx, x \mapsto A^Tx + B^Tx) \tag{4.9}$$

In PyOp, the addition function is written as follows.

```
class LinearOperator(object):

  # ...

  def __add__(self, other):

    return LinearOperator(self.shape,
                    lambda x: self.forward(x) + other.forward(x),
                    lambda x: self.adjoint(x) + other.adjoint(x))
```

This is a simplified form of what is written in PyOp, as there are additional techs to make sure that the sizes of the operators matches and other properties are met.

PyOp has several combinators built in.

- Negation
- Scaling
- Addition
- Subtraction (through negation and addition)
- Multiplication
- Power
- Vertical block operators
- Horizontal block operators
- Diagonal block operators
- Arbitrary block operators

All of these combinators return a new `LinearOperator` whenever they are called. While this can technically take up more space, the result means that the operators can be treated as mathematical objects without worrying about conflicts arising between operators or mismanaged pointers.

4.2.1 Drawbacks to PyOp's design

PyOp is simple in design, but in some regards this is to PyOp's detriment. The way in which PyOp is coded does not allow for introspection of the inner design of any particular operator; there is no available abstract syntax tree (AST) to manipulate in PyOp. In a related point, PyOp does not attempt to include a description of what properties any particular operator has, such as idempotence or symmetry. These problems mean that optimizations like those performed by the package Theano and similar packages cannot be applied to this framework. This drawback is shared with all other implementations of matrix free linear operators as of the time of this writing, although it is not an unsolvable problem by any means.

Additionally, Python has some quirks in its multiprocessing library that make parallelizing PyOp challenging. This is because the default serialization package known as pickle cannot "pickle" lambda expressions. There are other serialization libraries such as dill that do have this ability, but they are not standard and do not always play well with the current python concurrent programming frameworks. Agsin, this is not a fundamental problem for linear operator packages, just a hurdle caused by the architecture of both linear operator packages and python's concurrency framework.

4.3 Basic PyOp Use

We shall now go over some basic usage of PyOp. PyOp is designed to generalize the concept of a linear operator in the context of scientific computing and software development.

`LinearOperator` instances are quite simple to create. Below we discuss different ways of defining operators. For more information look at the source for those already defined in PyOp itself.

4.3.1 Unbound Functions

The basic way to create a LinearOperator instance is to just pass forward and adjoint functions to the `LinearOperator` constructor. The following example implements a square identity in this manner.

```
def id(x):
  return x

I = LinearOperator((4, 4), id, id)
```

This is a great way to test new forward and adjoint functions alone during development.

4.3.2 Nested Functions/Closures

Often it is helpful to create a `LinearOperator` in a nested function.

```
def squareIdentity(shape):

    # Do some pre-processing here.

    def id(x):
        return x

    return LinearOperator((shape, shape), id, id)
```

Note that in the above example, since `squareIdentity` is a nested function, the inner `id` function can operate on any data created before it in the `squareIdentity` lexical environment. This can very useful when realizing complicated operators or, for example, to create many similar operators that define the same basic morphism over different dimensions, as shown here.

4.4 Operator Rules

The most basic form of a `LinearOperator` takes in a shape defining the operation and a function (called the forward function) that operators on only one input. However, certain operators are practically useless.

```
A = LinearOperator((4, 4), lambda x: array([0, 2, 0, 0])
A(array([1, 1, 1, 1])) # The result is probably not what you wanted.
```

While the `LinearOperator` cannot automatically determine non-linear or bogus forward functions (or at least not easily), it does perform the following checks when the call method is used.

- both the operator and the input have a shape attribute,
- the inner dimensions of the operator and the input match, and
- the return of the forward function matches the expected dimensions (the outer dimensions).

Therefore, it is impossible to use the following operators due to the checks.

```
LinearOperator((4, 4), lambda x: "Hinc lucem et pocula sacra.")
LinearOperator((4, 4), lambda x: "Pax et Lux.")
LinearOperator((4, 4), lambda x, y: x + y)
```

Similarly, the following will be rejected because only positive pair shapes make sense, as they define a mapping from one finite dimension (the domain) to another (the codomain).

```
LinearOperator((4, 0), lambda x: x)
LinearOperator((4, -1), lambda x: x)
LinearOperator((4, ), lambda x: x)
```

Even though there is no automatic way to determine if an operator provides a correct (or linear) function–at least without intense lexical analyzing that could solve only the most basic of cases–adhering to the following rules will help ensure that an operator is functioning properly.

4.4.1 Guidelines for creating LinearOperators

There are a few guidelines for creating linear operators.

- `LinearOperator` instances are nearly useless without an adjoint function. While it is not required, it becomes impossible to perform calculations such as an eigenvalue decomposition without an adjoint, as well as any first order convex optimization solver.
- For `LinearOperator` instances that have an adjoint function defined, they should pass the `adjointTest`. This is a basic check that the forward and adjoint are indeed are each other's adjoints, although it does not guarantee that the operator is defined correctly.
- If there is a reference operation (such as a matrix), check that the function output matches the reference. While an operator that passes the adjoint test is usually doing what you expect, this is not always the case (defining the convolution operator is an example where an adjoint test can easily pass without defining the correct operator).
- Forward and adjoint functions that are defined on NumPy arrays should respect the dimensionality of their input. This means that 1D inputs should result in 1D outputs, and 2D inputs should lead to 2D outputs. This follows the way that many NumPy functions are styled.
- Forward and adjoint functions should be defined to work on 2D array inputs when the inputs are NumPy arrays. This is to say that the functions should be defined as matrix-matrix multiplications instead of matrix-vector multiplications, for example. This allows for the `toMatrix` function to work, which is helpful for turning an operator into its matrix form for further manipulation (for example, factorization or low rank approximations). `toMatrix` even works on operators that are created out of any of the composition or combining tools, as long as all the functions are defined as matrix-matrix multiplies.
- Do not make operator functions that hold state. Since the forward and adjoint functions can be anything that implements `call`, they can be functions, closures, or classes. All of these can contain state, but doing so breaks the immutable flavour of the operator and makes understanding the code more difficult. For example

```
res = [1, 1]
def forward(x):
    res *= x
    return res

A = LinearOperator(..., forward, ...)

## These two sequences lead to different results for the
## second statement.
A(array([1, 0]))
A(array([2, 1]))

A(array([0, 0]))
A(array([2, 1]))
```

In other words, the forward and adjoint functions, if they are to match those of standard linear operator theory, must be pure functions.

4.4.2 Tips for crafting LinearOperators

Here are some tips for developing operators that come from sometimes painful experience.

- Define the forward and adjoint functions in a non-nested namespace before putting them inside either a nested function or a class. They are simpler to iterate through in this form, while the nested/class form is really just a nice (simple) packaging of the result.
- Use the utilities decorators when possible to ease the creation of matrix-matrix functions. These decorators convert vector or vectorized (a nD array vectorized) functions to matrix-matrix functions and take care of concatenating the results together, flattening along the correct dimensions, etc.

One thing to note is that the checks do *not* specify the input data type. While the input to a `LinearOperator` is often a NumPy ndarray, it is entirely possible to use any input type that defines a shape pair. This could be useful, for example, if the input was a graph that was simpler to express and operate on as a “Graph“ type instead of an adjacency matrix. This is an (accidental) result of Python’s duck typing.

`LinearOperator` instances are designed to behave in an *immutable* manner, although they are strictly not immutable as a determined programmer can always redefine the functions held by a `LinearOperator` or modify the instance (or class) at runtime. However, in the course of normal programming, they can be treated as a non-hashable immutable as all of the composition rules create new operators containing the old ones. This means that if an operator is given to a function, it will not change after the function executes.

4.5 Convenient decorators

Often, certain forward and adjoint functions are simpler to define as a matrix-matrix multiplication or to operate on a reshape version of the input. The decorators defined here allow for functions of these forms to be modified to matrix-matrix functions easily. The examples for each decorator give a good sample of how to use such functions.

4.5.1 matmat

Convert 1D array to 2D column and back to 1D after calculation.

This function is meant to work only with functions following the `LinearOperator` calling declaration, which is a function taking in only one argument, the data to work on.

If the function receives a sparse input, then this wrapper does nothing since a sparse matrix cannot be squeeze or reshaped without significant alterations in its structure.

```
>>> import numpy as np
>>> from pyop import LinearOperator
>>>
>>> def squareEye(shape):
...     #
...     @matmat
...     def id(x):
...         return x
...     #
...     return LinearOperator((shape, shape), id, id)
>>>
>>> I = squareEye(4)
>>> I(np.arange(4))
array([0, 1, 2, 3])
```

4.5.2 matvec

Takes some function and Operates on an input matrix one column at a time. A function that implements a matrix-matrix function from a matrix-vector function. This decorator takes an input matrix, feeds each column as a 1D array to the function f, and then combines the results.

Often it is more natural to define matrix-vector function, since the matrix-matrix function is just the collection of the matrix-vector function applied to each column of the input. For these cases, use this decorator.

```
>>> import numpy as np
>>> from pyop import matvec
>>>
>>> @matvec
... def multFirstColumn(column):
...     img = column.reshape((2, 2), order = 'C')
...     img[:, 0] *= 2
...     return img.flatten(0)
>>>
>>> multFirstColumn(np.array([[1, 1, 1, 1], [2, 1, 2, 1]]).T)
```

```
array([[2, 4],
       [1, 1],
       [2, 4],
       [1, 1]])
```

4.5.3 matvectorized

A decorator to turn a function on a matrix into matrix-matrix product. This decorator is an extension of matvec. It operates on each column of a matrix at a time, but before passing the column to the decorated function, matvectorized reshapes the column based on the dimensions and order supplied into a nD array.

```
>>> import numpy as np
>>> from pyop import matvectorized
>>>
>>> @matvectorized((2, 2))
... def multFirstColumn(img):
...     img[:, 0] *= 2
...     return img
>>>
>>> multFirstColumn(np.array([[1, 1, 1, 1], [2, 1, 2, 1]]).T)
array([[2, 4],
       [1, 1],
       [2, 4],
       [1, 1]])
```

4.6 Currently Implemented Operators

PyOp has some operators defined in the package that assist with image reconstruction algorithms. These include

- `zeros`: The zeros matrix,
- `ones`: The ones matrix,
- `select`: a permutation matrix,
- `diag`: diagonal element matrix,
- `fftn`: an n-dimensional DFT,
- `ifftn`: an n-dimensional IDFT,
- `convolve`: an n-dimensional convolution with a kernel,

- `gradient`: an n-dimensional gradient based on the gradient operator.

4.6.1 Magnetic Particle Imaging Specific Operators

We have implemented operators that represent the forward model for image reconstruction from acquired but DC lacking image data [61]. There are two operators that are part of this forward model: an operator describing splitting the image into acquired chunks based on the scanning sequence, and a DC removal operator that models the loss of the DC component of the signal in each patch. A graphical representation of these operators can be seen in Figure 4.1.

In order to represent these operators, we first define some helper functions to massage input data into the correct format that the operators expect.

```python
## Tuples to describe the imaging sizes and the size of each patch
__ImageShape = namedtuple('ImageShape', 'height width')
__Blocks = namedtuple('Blocks', 'len height num shift')

## Takes the image shape and patch information, performs some checks,
## and returns the named tuples above.
def __toImageShapeAndBlocks(image_shape, blocks):

    h, w = image_shape
    length, shift = blocks

    image_shape = __ImageShape(*image_shape)
    blocks = __Blocks(length, h, int((w - length) / shift) + 1, shift)

    if any(x <= 0 for x in image_shape):
        raise ValueError("Image shape contains a non-positive. {}".
                         format(image_shape))

    if any(x <= 0 for x in blocks):
        raise ValueError("blocks contains a non-positive. {}".
                         format(blocks))

    return image_shape, blocks


## Take a stream ov values and return all adjacent pairs.
def __pairwise(x):
    ''' s --> (s0, s1), (s1, s2), (s2, s3), ...  '''
    a, b = tee(x)
    next(b, None)
    return zip(a, b)
```

Using these helper functions we can now implement the splitting operator quite easily. This operator models the splitting of the image into the acquired patches/blocks.

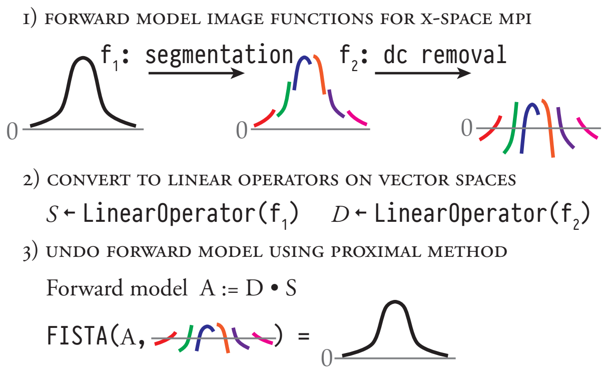
```
def split(image_shape, blocks):

    image_shape, blocks = __toImageShapeAndBlocks(image_shape, blocks)

    h, w = image_shape
    length, _, num, shift = blocks

    op_shape = (length * h * num, h * w)

    def stride(x, shift, length):
        i = 0

        ## Run until we can't take any more lengths out
        while i + length <= len(x):
            yield x[i: i + length]
            i += shift


    @matvectorized((h, w), order = 'F')
    def splitting(img):
        return hstack(z.T for z in stride(img.T, shift, length))


    @matvectorized((h, -1), order='F')
    def splittingAdjoint(block_set):
        block_list = hsplit(block_set, num)

        combined = zeros((h, w))
        for step, b in zip(count(step = shift), block_list):
            ## Add in each block in the right spot
            combined[:, step : step + length] += b

        return combined.flatten(1)


    return LinearOperator(op_shape, splitting, splittingAdjoint)
```

The DC removal operator is even simpler to define, as it is self-adjoint. While we do not note this property in PyOp itself, the self-adjoint nature does make specifying the operator simpler as the forward and ajoint functions are the same.

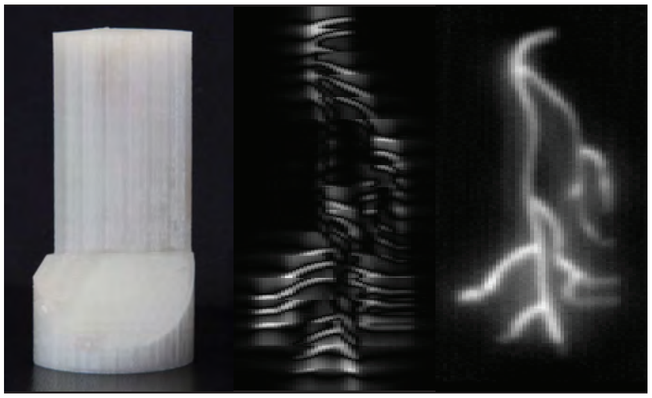
```python
def dcRemove(image_shape, blocks):

    image_shape, blocks = __toImageShapeAndBlocks(image_shape, blocks)

    h, _ = image_shape
    length, _, num, _ = blocks

    op_size = length * h * num
    op_shape = (op_size, op_size)

    @matvectorized((h, -1), order = 'F')
    def dcRem(block_set):
        ## Partial field of views, one per row
        pfovs = block_set.reshape((-1, length))

        ## Sum across rows to get the average of each pfov.
        dc_values = pfovs.sum(1) / length

        ## Tiling to apply DC removal to each point in each pfov.
        ## Transpose due to tile treating 1D as a row vector.
        dc_values_rep = tile(dc_values, (length, 1)).T

        ## reshape into the blocks_set image. Turn to column format
        return (pfovs - dc_values_rep).reshape((h, -1))


    return LinearOperator(op_shape, dcRem, dcRem)
```

4.7 Using PyOp for DC recovery in Magnetic Particle Imaging

In MPI, we have to perform a process known as DC recovery [2, 7]. The basic problem is that while the response from the particles is easily picked up, the way in which the particles are caused to flip means that the transduction signal is picked up in the receive signal. This feed through is removed by filtering, but this filtering removes the particle signal at that frequency. From the MPI imaging theory, this feedthrough means that the DC component of the signal is lost, and must be recovered to regain the LSI property of MPI.


1) forward model image functions for x-space mpi
f1: segmentation
f2: dc removal
0
0
0
2) convert to linear operators on vector spaces
S ← LinearOperator(f1)
D ← LinearOperator(f2)
3) undo forward model using proximal method
Forward model A := D • S
FISTA(A,) =
0

(a) PyOp reconstruction technique using the FISTA proximal operator method.



(b) 3D printed model of coronary arteries (c) image prior to stitching (d) image after stitching

Figure 4.1: A 3D coronary artery phantom. (a) PyOp is used to first convert image segmentation and DC removal functions into linear transformations, which are then used in a gradient descent algorithm (FISTA) to reconstruct the image. (b) A 3D printed coronary artery phantom used to test the PyOp based reconstruction. (c) 2D MIP of the data after the gridding step, prior to the matrix free stitching. (d) 2D MIP after stitching. The total imaging time was 10 min with a field of view of 4.5 cm by 3.5 cm by 9.5 cm (x,y,z).

One method of recovery uses convex optimization techniques to determine the DC offsets that match the *a priori* knowledge [61]. This method produces excellent images but requires significant time and memory resources to perform.

Here we demonstrate the benefits of PyOp as applied to x-space MPI DC reconstruction, allowing the DC reconstruction to both run faster and in less space; this enables the DC reconstruction to run on modest hardware.

4.7.1 Methods

For MPI we have defined two operators, DC removal (D) and image segmentation (S) that combined represent a forward model of the x-space MPI DC recovery process (see Figure 4.1. In finite form, these matrices (especially D) are not sparse and take up approximately n^2 amount of data, where n is the total number of pixels in the input data. The description of these operators are found in a prior section.

With the MPI functions defined as linear transformations through PyOp, we were able to use a variety of proximal operator based optimization methods (such as the Fast Iterative Soft-Thresholding Algorithm (FISTA) [62]) to reconstruct an MPI image. The solver methods are implemented in a general manner, allowing for more *a priori* information to be easily added. See Figure 4.1 for more details.

4.7.2 Results

Prior DC recovery on 3D x-space MPI datasets could take over an hour and 150 GBs of memory; with PyOp the same recovery takes under ten minutes and *bytes* of space. This represents a 10^9 times improvement in space and a 6.625 times improvement in speed. The PyOp method will always guarantee that the operators will take *bytes* of memory. We note that this method does not alter the results of the image reconstruction; PyOp merely allows us to perform the computation in a more efficient manner.

Method	Memory Used	Compute Time (Minutes)
Matrix	150 GB ($\mathcal{O}(n^2)$)	53
PyOp	Bytes ($\mathcal{O}(1)$)	8
Improvement	10^9	6.625

Additionally, for a 100 by 100 pixel 2D final image, the combined FISTA and PyOp approach allows for the DC reconstruction to occur in as fast as 37 ms (27 FPS). This speed is required to enable real time, interventional MPI in the future.

4.8 Conclusion

PyOp defines an interface to convert functions on images into linear transformations easily, provides a robust testing framework that enables mathematical robustness, and implements a complete set of composition rules for ease of use. The PyOp technique is efficient and completely generalizable to other reconstruction problems in MPI and linear algebra in general. In addition, this software framework can be used to easily augment the image reconstruction with additions such as filtering in the Fourier domain or alternate optimization formulations.

Acknowledgements

We would like to thank Justin Konkle for the optimization formulation, as well as the design and printing of the coronary phantom. We would thank Bo Zheng, Daniel Hensley, and Patrick Goodwill for insightful commentary on the work.

Chapter 5

Sparse Image Reconstruction for Field Free Line Magnetic Particle Imaging by Angular Subsampling

5.1 Introduction to Field Free Line Magnetic Particle Imaging

Magnetic Particle Imaging (MPI) is a promising new modality that images only a magnetic tracer, commonly super-paramagnetic iron oxide (SPIO) nanoparticles. This technique requires scanning a field free region (FFR) over space, taking a significant amount of time to acquire an image. Reduction of imaging time is achievable via new scanner configurations, such as field free line (FFL) scanners that take projections of an object (similar to CT), as well as optimal spatial scanning trajectories.

This chapter discusses an efficient implementation of FFL MPI 3D image reconstruction using proximal optimization methods that allow for significant undersampling of the required number of projections, while including a novel and generalized forward model that allows for arbitrary scanning trajectories to be taken into account.

Notation: The real and natural numbers are denoted by $\mathbb{R}$ and $\mathbb{N}$, respectively. The vector of ones is denoted by $\mathbf{1}$. The convex-hull of a set $\mathcal{X} \subseteq \mathbb{R}^n$ is the set $\operatorname{conv}(\mathcal{X}) = \left\{\sum_{i=0}^{n} \lambda_i x_i : x_i \in \mathcal{X}, \lambda \geq 0, \mathbf{1}^T\lambda = 1\right\}$. We define a predicate set intersection to determine if two sets intersect each other $a \sqcap b \equiv a \cap b \neq \emptyset$, and a shorthand $\bigsqcap_a B = \{b : b \in B, a \sqcap b\}$

5.2 Introduction to the Radon Transform

The radon transform describes this process mathematically [63]. We will discuss the derivation in two dimensions, although the formulations are simple to extend to higher dimensions.

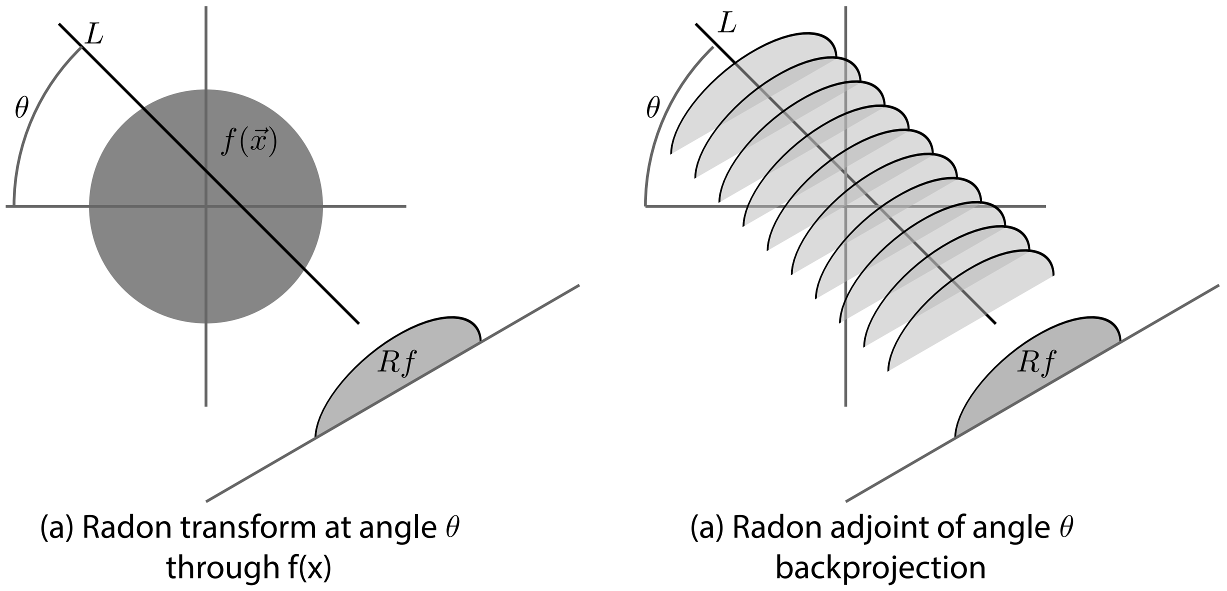

L
θ
$f(\vec{x})$
Rf
(a) Radon transform at angle θ through f(x)
L
θ
Rf
(a) Radon adjoint of angle θ backprojection

Figure 5.1: A graphical representation of the radon transform and its adjoint. (a) The radon transform takes some function f and performs an integral projection along a line L. Many lines at the same angle θ can be grouped together to show the "shadow" of f at that angle. (b) In the adjoint of the radon transform, we take the shadow and "smear" it back across the original domain. This process is often called backprojection.

The common form of the radon transform is the line integral through some function f.

$$R_f(L) = \int_L f(\vec{x})\, d\vec{x} \tag{5.1}$$

In this equation, L is a line passing through $\mathbb{R}^2$ and $f \in \mathbb{R}^2 \to \mathbb{R}$ is a function of finite support of which is to be projected (in FFL MPI, this is the nanoparticle distribution). We can cast the definition of L as the set of all points some distance s from the y axis rotated by some angle θ

$$L(\theta, s) = \left\{ R_\theta \vec{x} : \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \vec{x} = \begin{bmatrix} s \\ 0 \end{bmatrix},\ \vec{x} \in \mathbb{R}^n, s \in \mathbb{R}, \theta \in [0, \pi) \right\} \tag{5.2}$$

where R_θ is the rotation matrix. Intuitively, equation (5.2) says that we first select a vertical line by choosing an fixed horizontal offset for that line (s), and then rotate that line by θ around the origin to get our desired target line through a two dimensional space.

Often we are interested in all of the lines at a given fixed angle θ projected onto a line.

$$\text{Proj}_\theta(f) = \{R_f(L(\theta, r)) : r \in \mathbb{R}\} \tag{5.3}$$

With this in place, it is common to create what is called a sinogram, which is a set of projections over a finite set of (usually unique) angles Θ.

$$\mathrm{S}(f, \Theta) = \bigcup_{\theta \in \Theta} \mathrm{Proj}_\theta(f) \tag{5.4}$$

With the sinogram in hand, it is possible to reconstruct many projection based imaging systems such as MPI or CT. The radon transforms describes the forward model of these imaging systems, and therefore the transform can can be used to either analytically solve the for the original function f from the sinogram data, or an iterative method can be used.

In FFL MPI, the FFR is a line that passes through a 3D volume (Figure 5.2) [4]. The FFL is then moved around on a plane using electromagnets to cause the iron particles that are in (or near) the FFR to align with the changing magnetic field, inducing a signal in a pickup coil. Since an entire line of particles responds in FFL MPI, the sum of the signal from all of the nanoparticle near the FFR is detected.

5.2.1 Generalization of radon transform to projections over sets

While the standard definition of the radon transform is useful, it is somewhat restricted geometrically by requiring the projection to occur over lines (in (5.1)). In MPI, there are cases where it can be beneficial to take projections over more complicated shapes. For example, some current FFL MPI systems project three dimensional data down to two dimensions in a lissajous pattern, where the weighting of each node trajectory point is given by the relative volume of the surrounding voronoi cell (see Figure 5.3) [64]. Other FFL scanners project two dimensional data down to one dimension but with continuous rotation instead of discrete steps in θ [65].

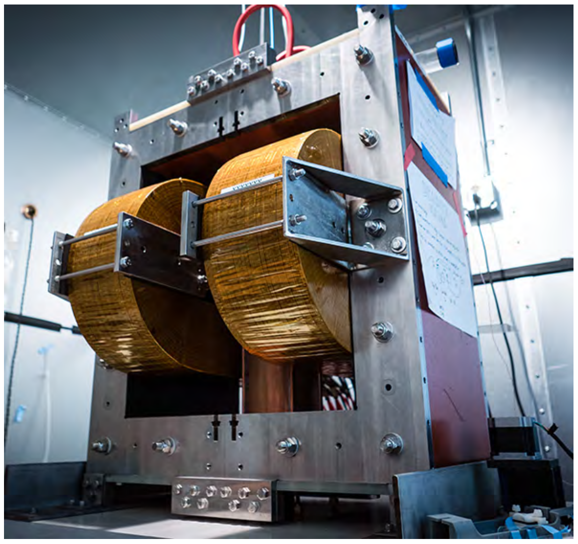
To capture these different potentially scanning modalities, we simply define the radon transform to be over sets instead of over a line

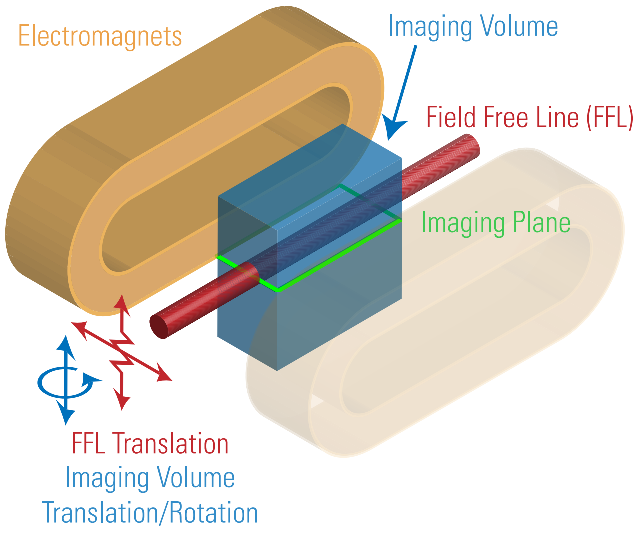
$$R_f(s) = \int_s f(\vec{x})\, d\vec{x} \tag{5.5}$$

where s is some arbitrary subset of the space that f is defined over (in any dimension). This "modification" is not actually a change at all; we defined the line L as a set that we can replace with any subset of the support of f that we desire. While this is not a drastic change, it presents computational challenges both in terms of efficiency and in terms of numerical accuracy.

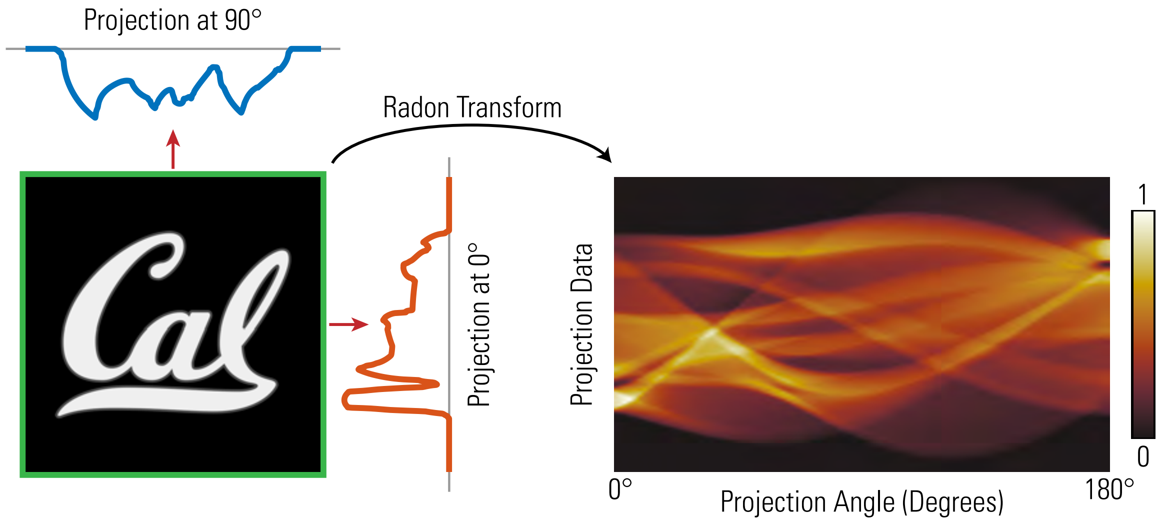
5.2.2 Algorithm for computing sinograms of discrete functions

The more generalized definition of the sinogram allows for one to mathematically encapsulate more types of imaging designs, but can be tricky to implement due to numerical error. It


(a) Berkeley 6.3 T/m FFL MPI Scanner



(b) FFL MPI Magnets and FFL Movement



(c) Projections of the imaging plane at two angles (d) Sinogram: projection of the imaging plane at many angles

Figure 5.2: A diagrammatic overview of FFL MPI. (a) An image of the Berkeley 6.3 T/m FFL MPI scanner. The large copper coils seen in the image generate opposing magnetic fields, generating a FFL in the center. The center copper shield contains another coil that enables fast movement of the FFL. (b) A simplified rendering of the Berkeley scanner. The coils (orange) generate the FFL (red cylinder) that passes through the imaging volume (blue box). The electromagnets enable the FFL to move across a 2D cross section (red arrow) of the imaging volume (the imaging plane, green). An additional coil inside the copper shield in panel a allows for quick movement of the FFL in the vertical direction (red zig-zag arrow). The electromagnets only allow for one projection angle and a small amount of the imaging volume to be captured, so translational and rotational stages move the imaging volume to get fill 2D projections from many angles (blue arrows) (c) An example of the projections acquired at two different angles through an imaging plane. Many such planes are acquired in order to reconstruct a 3D volume. (d) The many projections of a single imaging slice at many angles are collected into an image known as a sinogram. Panel c and d are related by the radon transform.

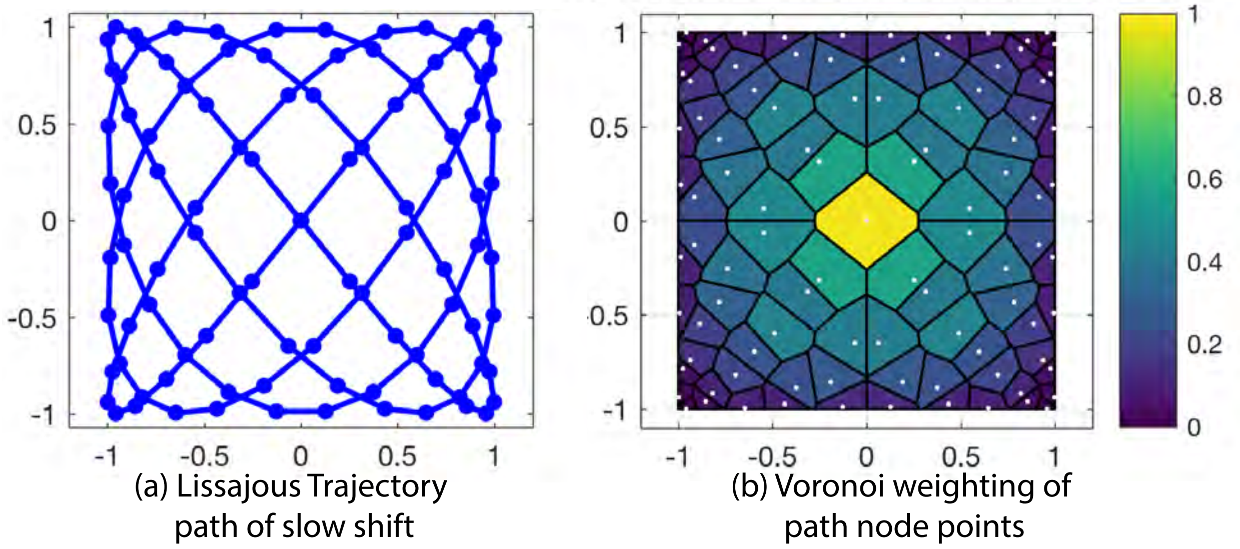


Figure 5.3: Lissajous sampling trajectory, along with a weighting of the sample points. (Top) The Lissajous pattern can be defined by a set of coprime frequencies a and b, their amplitudes A and B, and an offset parameter δ: $x = A\sin(at + \delta), y = B\sin(bt)$, $t \in [0, 2\pi)$. 99 points were selected from this set. (Bottom) A Voronoi tesselation of the space (bounded by a box) allows for the Lissajous sample points (white dots) to be weighted by the area based on the density of nearby points. This discretized sampling of some two dimensional function is often used as a metric to compensate for oversampling certain regions while undersampling others.

is a well known problem in computational geometry that algorithms often fail due to the lack of numerical precision. A straightfoward implementation of equation (5.5) can lead to cases where calculating $R_f(s)$ produces wildly wrong results. For example, the calculation of the voronoi regions in Figure 5.4 was originally performed using Qhull [66], but this resulted in voronoi regions where the boundary intersected its associated site, or where the voronoi region did not contain its associated site, of which both cases are impossible given the definition of a voronoi region.

To mitigate this problem, we implement (5.5) using CGAL, a computational geometry framework designed to accurately answer geometric predicates without the problems of intermediate round off errors [67]. We start by sampling f on some grid $\mathcal{G}$, where $\mathcal{G}$ is some decomposition of space into sets. For example, $\mathcal{G}$ can be the voronoi tesselation of the space, which for equispaced sampling points corresponds to a rectilinear grid. We denote the sampled version of f on $\mathcal{G}$ as $\hat{f}$. In addition, we denote the set of "sensing functions" as $\mathcal{S}$, which is the set of functions that select some subset of the domain of $\hat{f}$ ($s(\text{dom}(f)) \subseteq \text{dom}(f)$, $\forall s \in \mathcal{S}$).

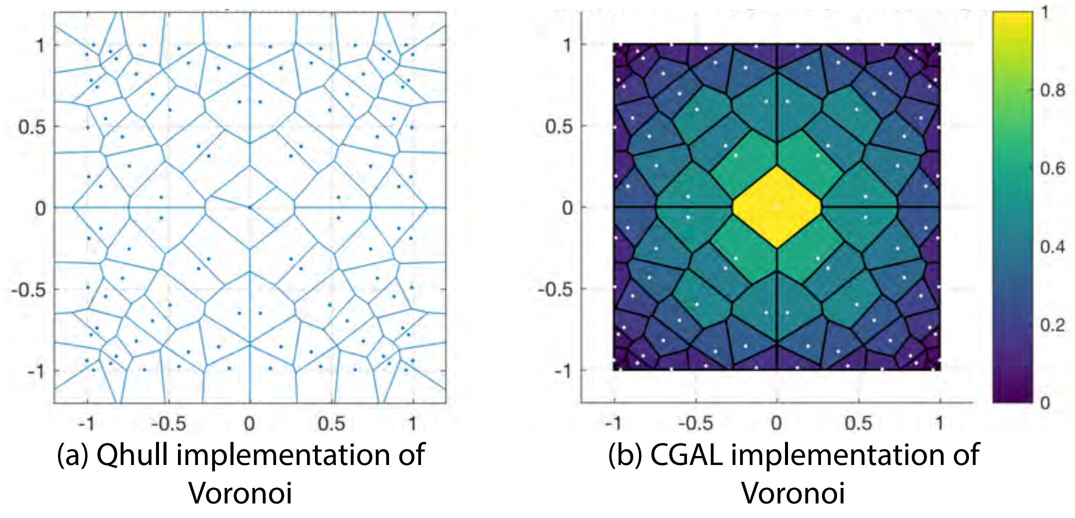


Figure 5.4: A comparison of the performance of various computational geometry libraries when attempting to perform the Voronoi tesselation of a lissajous pattern. (a) The common package to perform the voronoi calculation is known as QHull. This program is used by Matlab to implement the voronoi tesselation. The results on the lissajous pattern show an error in the center cell, where the center cell is divided incorrectly into three pieces. (b) The CGAL implementation of the same algorithm does not experience this error.

The radon transform can then be defined by the following linear operator

$$R = \Sigma W P = \operatorname{diag}(\mathbf{1}^T, \ldots, \mathbf{1}^T) \operatorname{diag}(\operatorname{diag}(w_{s_1}), \ldots, \operatorname{diag}(w_{s_n})) \begin{bmatrix} \prod_{s_1} \mathcal{G} \\ \vdots \\ \prod_{s_n} \mathcal{G} \end{bmatrix} \tag{5.6}$$

where P is a permutation operator selecting the intersection of some $s \in \mathcal{S}$ with the grid $\mathcal{G}$, $\operatorname{diag}(w)$ is a diagonal square weighting operator that weights how much any much a particular s intersects a particular g (with values between 0 and 1), and Σ sums the result. Algorithmically, we can write this out as follows.

The adjoint of the radon transform (also known as backprojection) can then be derived by simply taking the transpose.

$$R^T = P^T W \Sigma^T = \left[\prod_{s_1} \mathcal{G}, \ldots, \prod_{s_n} \mathcal{G}\right] \operatorname{diag}(\operatorname{diag}(w_{s_1}), \ldots, \operatorname{diag}(w_{s_n})) \operatorname{diag}(\mathbf{1}, \ldots, \mathbf{1}) \tag{5.7}$$

The linear algebra form provides some intuitions to why backprojection is the adjoint operator; any particular point in the original grid g is defined by the sum of all the projections through g, weighted by how much any particular sensing function intersected g. We can write this algorithmically as well. We have not implemented this part algorithmically, as we merely store the algorithm as a sparse matrix and transpose that matrix.

Algorithm 1 discrete radon transform (DRT)

```
Input: $\hat{f}$, the sampled version of $f$ on the grid $\mathcal{G}$
  $\mathcal{G}$, the sampling grid,
  $\mathcal{S}$, the set of “sensing functions”
Output: S, the projected data
  for all $s \in \mathcal{S}$ do
    for all $g \in \mathcal{G}$ do
      $v(s, g) = \frac{\text{vol}(s \cap g)}{\text{vol}(g)} \hat{f}(g)$
    end for
    $S(s) = \sum_{g \in \mathcal{G}} v(s, g)$
  end for
```

Most radon transform implementations follow this model, but the individual operations such as set intersection, volume calculation, or grid definition are specialized to handle the specific case of that particular projection model [68]. For example, code to perform the radon transform for cone beam projections in CT exists and run on graphical processing units (GPUs), but may only work over functions discretized on a rectilinear grid.

The value of our generalization is that it allows for flexibility to create image reconstruction methods without needing to come up with a specific projection implementation, allowing for faster testing of new acquisition sequences. Additionally, this algorithm is relatively simple to make more performant: simply replace any slow calculation with a faster one.

For example, the initial version of this algorithm was written using half-plane descriptions of the sets and then used a linear program to calculate the volumes. This process took approximately 6 hours to complete for image sets with approximately 20 projections of a 100 by 100 pixel grid; by parallelizing the for loops and performing smarter set intersection, we were able to get the runtime down to approximately a minute (a 360 fold speed improvement).

5.2.3 Forward model for FFL MPI

The discrete radon transform (DRT) represents the forward model of FFL MPI reconstruction of a 3D dataset from 2D projections. In the current CGAL implementation of the radon transform, the output of the algorithm is v formatted into a sparse matrix (the DRT). This allows us to trade the computation time of the DRT for storage space, which is minimal due to the small cardinality of $\mathcal{S}$ for sparse acquisitions in practice and that few grid cells intersect any particular sensing function.

As input to our algorithm, we have defined the following sets in two dimensions.

- $\mathcal{S} = \left\{ R_\theta \left(\begin{bmatrix} 1, 0 \\ -1, 0 \end{bmatrix} x \leq \begin{bmatrix} c + w/2 \\ -c + w/2 \end{bmatrix} \right) : x \in \mathbb{R}^2, c \in C, \theta \in \Theta \right\}$ where C is the set of center points of the sensing function, w is the width of each sensing function, and Θ

is the set of projection angles taken in a scan. This represents vertical slabs through space. We note that the algorithm presented can handle unbounded sets without problem.

- $\mathcal{G} = \{\text{conv}(z) : z = (x \pm w_x, y \pm w_y), \forall x \in X, y \in Y\}$ where X and Y are the locations of the pixel centers (for example, given a min $x_{\min}$ and max $x_{\max}$ point, $X = x_{\min}, x_{\min} + w_x, x_{\min} + 2w_x, \ldots, x_{\max}$) and w_x, w_y are the width of each pixel. This defines a 2D rectilinear grid.

We then plug the result into our algorithm 1 as $\text{DRT}(\mathcal{S}, \mathcal{G})$. We have not provided $\hat{f}$ in this case, as we can simply store the result of the intersection and volume calculations in a matrix and supply $\hat{f}$ later as a vector.

We can take the 2D result from this and convert it into the 3D implementation by simply repeating the DRT matrix for the number of rows in the projection images: $\text{DRT}_{3D} = \text{diag}(\text{DRT}(\mathcal{S}, \mathcal{G}), \ldots, \text{DRT}(\mathcal{S}, \mathcal{G}))$. This ability is due to the locality of the radon transform in any dimension higher than 2D [63].

We have also implemented a modified version of the radon transform and adjoint from Gao [68], which combined with the python linear operator package PyOp allows us to create a matrix-free linear operator reconstruction. This variant performs the same operations as defined by algorithm 1, but in a more performant form. The implemented algorithm can be seen in algorithm 2 and algorithm 3.

5.2.4 Ways to reconstruct images modeled by the radon transform

From $S(f)$, there are a variety of ways to reconstruct the original f. The most simple is filtered backprojection [63], which uses the dual Radon transform (often called backprojection) to first undo the projection and then applies a k-space filter (often a ramp) to account for oversampling the center of f when acquiring the projections. Simultaneous Algebraic Reconstruction Technique (SART) is another method by which to perform the reconstruction of f, and is often preferred in CT imaging for its robustness when given few projection angles [69]. A final method involves solving a simple forward model l_2 norm convex optimization with any desired constraints (often none), where the forward model is the radon transform [70, 71]. This can be solved either through direct inversion such as the penrose inverse or via some iterative method such as gradient descent or second order methods like Newton's method.

Here we solve the following optimization problem,

$$\min ||R\rho - y||_2^2 + g(\rho) \tag{5.8}$$

where ρ is the concentration of iron particles over space, R is the DRT, y is the projection data from the FFL MPI scanner, and g is any regularization function. In MPI, we often use a null function for g, although one can choose some *a priori* information such as smoothness

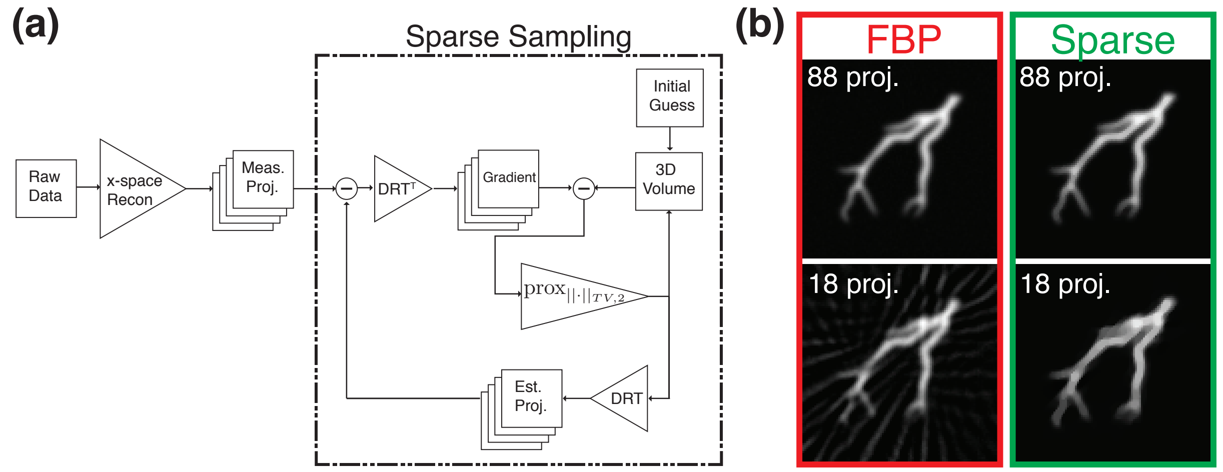


Figure 5.5: A diagrammatic overview of the sparse reconstruction algorithm. (a) The sparse reconstruction follows the standard FFL MPI reconstruction procedure up to the point of generating the projection data. After this point, instead of performing a radon transform, we implement a proximal update algorithm in order to iteratively find the filtered backprojection image. (b) A result of reconstructing a coronary phantom image using the proposed algorithm, both on data that has been sampled angularly at Nyquist and data that is subsampled by a factor of 4.8x. The results clearly show the sparse reconstruction algorithm performing better on the sparsely sampled data, with no streak artifacts. The sparse algorithm also correctly reconstructs the fully sampled data.

of the image or reducing the DC component. In our case, we set the regularization function $g(x) = ||x||_{TV,n} = |Dx|_n$, which is the TV-norm and D is the first order finite difference. A diagrammatic depiction of this process can be seen in Figure 5.5

This optimization can be solved with any proximal algorithm. We tried Nesterov's algorithm (NESTA) [72] and FISTA [62] algorithms to solve this problem, with equivalent results. In this case, we have chosen FISTA for pragmatic reasons. For this algorithm, we provided the objective function $F(x) \equiv f(x)+g(x) = ||R\rho-y||_2^2+||x||_{TV,2}$. The proximal TV operator was provided by the proxTV python package [73, 74].

5.3 Simulation Methods

A simulation of the Berkeley 6.3 T/m FFL MPI scanner was used to create sample images of a coronary phantom. The required number of projections to satisfy the Nyquist sampling theorem was calculated and used to simulate a image acquisition with an SNR of 50 using 25nm iron particles. Image acquisition with 1/4th fewer projections was also simulated. The acquired data was then reconstructed using the filtered back-projection (FBP) and using the proposed sparse sampling algorithm.

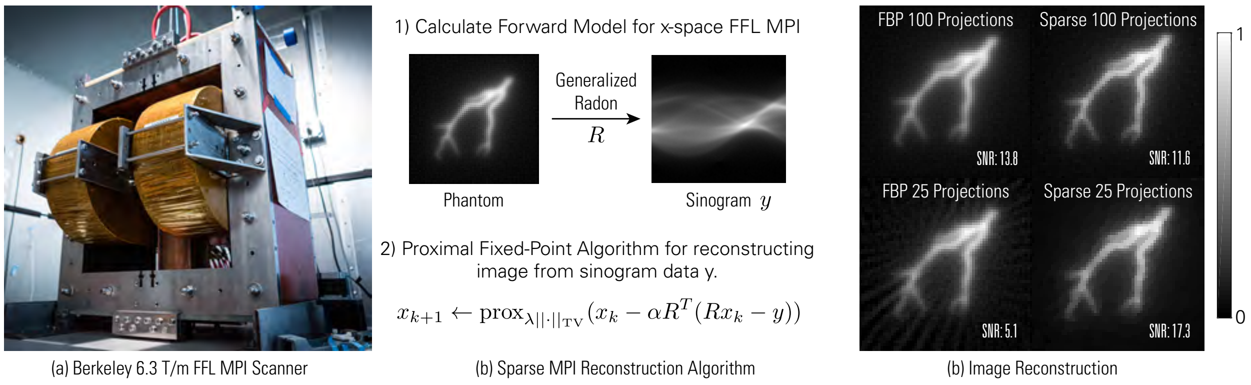

1) Calculate Forward Model for x-space FFL MPI
Generalized Radon
R
Phantom
Sinogram y
2) Proximal Fixed-Point Algorithm for reconstructing image from sinogram data y.
$x_{k+1} \leftarrow \mathrm{prox}_{\lambda||\cdot||_{\mathrm{TV}}}(x_k - \alpha R^T(Rx_k - y))$
FBP 100 Projections
Sparse 100 Projections
SNR: 13.8
SNR: 11.6
FBP 25 Projections
Sparse 25 Projections
SNR: 5.1
SNR: 17.3
1
0
(a) Berkeley 6.3 T/m FFL MPI Scanner
(b) Sparse MPI Reconstruction Algorithm
(b) Image Reconstruction

Figure 5.6: Sparse sampling reconstruction algorithm for FFL MPI. (a) Image of the Berkeley FFL MPI scanner simulated in this study. (b) The FFL MPI forward model is calculated from a generalized Radon transform that allows for arbitrary trajectories of space. The model is used in a proximal fixed-point algorithm to reconstruct the image given some penalty on the TV norm. (c) The results of the reconstruction on a simulated phantom image. 100 projections was calculated as the minimum number of projections satisfying the Nyquist sampling theorem. The CS reconstruction achieves similar results as the fully sampled image without streak artifacts in a fourth of the number of projections. Sinogram data was simulated with a 25nm iron particle (0.95mm FWHM in a 6.3 T/m FFL MPI scanner) with a 4cm by 4cm, 64x64 pixel FOV. 2D reconstruction time is 1 second; a tilted 3D version of the coronary phantom takes 10 seconds for a 64x64x64 pixel image.

The novel sparse sampling algorithm solves a convex optimization that attempts to minimize both a data consistency term (between the simulated scan and the acquired data) and a sparse domain (total variation). The algorithm was designed matrix-free using PyOp. In addition, the novel forward model allows for reconstructing data taken using arbitrary scanning trajectories, opening up new possibilities for image acquisition sequences.

5.4 Simulation Results

The coronary phantom was reconstructed using FBP and the proposed sparse sampling MPI method (Figure 5.6; the results show that while FBP for images sampled with 1/4 the number of projections leads to significant artifacts, the proposed method correctly reconstructs the coronary phantom. This means that the amount of time required to acquire the image is reduced by a factor of 4. The sparse method took 10 seconds to reconstruct a 3D version of the phantom, demonstrating that this algorithm is fast in practice.

5.5 Conclusion

The sparse sampling MPI algorithm enables fast and robust reconstruction of undersampled, noisy FFL MPI data in a manner that reconstructs the underlying data more accurately than the state of the art reconstruction method. In addition, the algorithm and code developed here is general enough to allow for many extensions in the future. For example, reconstructing a sequence where the imaging volume is continually rotating and translating (often known as a spiral trajectory in CT) can be reconstructed easily with this method by changing the input sensing functions.

Acknowledgements

We would like to thank Evan Frenklak for programming the Gao variant in C. We would like to thank Elaine Yu for the helpful discussions and ideas on phantoms. Additionally, we would like to thank Yueting Shi for insightful comments on compressed sensing algorithms.

Bibliography

[1] Bernhard Gleich and Jürgen Weizenecker. "Tomographic imaging using the nonlinear response of magnetic particles". In: *Nature* 435.7046 (2005), pp. 1214–1217.

[2] Patrick W Goodwill and Steven M Conolly. "The X-space formulation of the magnetic particle imaging process: 1-D signal, resolution, bandwidth, SNR, SAR, and magnetostimulation". In: *IEEE transactions on medical imaging* 29.11 (2010), pp. 1851–1859.

[3] Emine U Saritas et al. "Magnetic particle imaging (MPI) for NMR and MRI researchers". In: *Journal of Magnetic Resonance* 229 (2013), pp. 116–126.

[4] Justin J Konkle et al. "Projection reconstruction magnetic particle imaging". In: *IEEE transactions on medical imaging* 32.2 (2013), pp. 338–347.

[5] R Matthew Ferguson et al. "Magnetic particle imaging with tailored iron oxide nanoparticle tracers". In: *IEEE transactions on medical imaging* 34.5 (2015), pp. 1077–1084.

[6] Patrick W Goodwill and Steven M Conolly. "Multidimensional x-space magnetic particle imaging". In: *IEEE transactions on medical imaging* 30.9 (2011), pp. 1581–1590.

[7] Kuan Lu et al. "Linearity and shift invariance for quantitative magnetic particle imaging". In: *IEEE transactions on medical imaging* 32.9 (2013), pp. 1565–1575.

[8] Scott J Kemp et al. "Monodisperse magnetite nanoparticles with nearly ideal saturation magnetization". In: *RSC Advances* 6.81 (2016), pp. 77452–77464.

[9] Jean A Langlois, Wesley Rutland-Brown, and Karen E Thomas. *Traumatic brain injury in the United States: emergency department visits, hospitalizations, and deaths.* Department of Health et al., 2004.

[10] Jean A Langlois, Wesley Rutland-Brown, and Marlena M Wald. "The epidemiology and impact of traumatic brain injury: a brief overview". In: *The Journal of head trauma rehabilitation* 21.5 (2006), pp. 375–378.

[11] Christiane Albert-Weissenberger and Anna-Leena Sirén. "Experimental traumatic brain injury". In: *Experimental & translational stroke medicine* 2.1 (2010), p. 1.

[12] Graham Teasdale and Bryan Jennett. "Assessment of coma and impaired consciousness: a practical scale". In: *The Lancet* 304.7872 (1974), pp. 81–84.

[13] Kathryn E Saatman et al. "Classification of traumatic brain injury for targeted therapies". In: *Journal of neurotrauma* 25.7 (2008), pp. 719–738.

[14] Bruce Lee and Andrew Newberg. “Neuroimaging in traumatic brain imaging”. In: *NeuroRx* 2.2 (2005), pp. 372–383.

[15] Amon Y Liu et al. “Traumatic brain injury: diffusion-weighted MR imaging findings”. In: *American Journal of Neuroradiology* 20.9 (1999), pp. 1636–1641.

[16] Michael A Flierl et al. “Mouse closed head injury model induced by a weight-drop device”. In: *Nature protocols* 4.9 (2009), pp. 1328–1337.

[17] Kathryn E Saatman et al. “Differential behavioral and histopathological responses to graded cortical impact injury in mice”. In: *Journal of neurotrauma* 23.8 (2006), pp. 1241–1253.

[18] Peter Reimer and Thomas Balzer. “Ferucarbotran (Resovist): a new clinically approved RES-specific contrast agent for contrast-enhanced MRI of the liver: properties, clinical development, and applications”. In: *European radiology* 13.6 (2003), pp. 1266–1276.

[19] Bo Zheng et al. “Magnetic Particle Imaging tracks the long-term fate of in vivo neural cell implants with high image contrast”. In: *Scientific reports* 5 (2015).

[20] Bo Zheng et al. “Quantitative magnetic particle imaging monitors the transplantation, biodistribution, and clearance of stem cells in vivo”. In: *Theranostics* 6.3 (2016), p. 291.

[21] Justin J Konkle et al. “Twenty-fold acceleration of 3D projection reconstruction MPI”. In: *Biomedizinische Technik/Biomedical Engineering* 58.6 (2013), pp. 565–576.

[22] Elaine Yu et al. “First demonstration of in vivo Cancer Magnetic Particle Imaging with IV-administered Passive Long-circulating SPIOs”. In: *World Molecular Imaging Congress*. 2016.

[23] Paul Keselman et al. “Magnetic Particle Imaging as a Tool for Tracking in vivo Biodistribution and Long-term Tracking of Iron Oxide Nanoparticle Tracers and Therapeutics”. In: *World Molecular Imaging Congress*. 2016.

[24] Min Lu et al. “FDA report: ferumoxytol for intravenous iron therapy in adult patients with chronic kidney disease”. In: *American journal of hematology* 85.5 (2010), pp. 315–319.

[25] Fulvio Stacul et al. “Contrast induced nephropathy: updated ESUR contrast media safety committee guidelines”. In: *European radiology* 21.12 (2011), pp. 2527–2541.

[26] Manuel Guizar-Sicairos, Samuel T Thurman, and James R Fienup. “Efficient subpixel image registration algorithms”. In: *Optics letters* 33.2 (2008), pp. 156–158.

[27] Tapan K Jain et al. “Biodistribution, clearance, and biocompatibility of iron oxide magnetic nanoparticles in rats”. In: *Molecular pharmaceutics* 5.2 (2008), pp. 316–327.

[28] Michelle Longmire, Peter L Choyke, and Hisataka Kobayashi. “Clearance properties of nano-sized particles and molecules as imaging agents: considerations and caveats”. In: *Future Medicine* (2008).

[29] MW Bradbury, HF Cserr, and RJ Westrop. "Drainage of cerebral interstitial fluid into deep cervical lymph of the rabbit". In: *American Journal of Physiology-Renal Physiology* 240.4 (1981), F329–F336.

[30] Emily Mathieu et al. "In vivo imaging of lymphatic drainage of cerebrospinal fluid in mouse". In: *Fluids and Barriers of the CNS* 10.1 (2013), p. 1.

[31] Antoine Louveau et al. "Structural and functional features of central nervous system lymphatic vessels". In: *Nature* (2015).

[32] Aleksanteri Aspelund et al. "A dural lymphatic vascular system that drains brain interstitial fluid and macromolecules". In: *The Journal of experimental medicine* 212.7 (2015), pp. 991–999.

[33] Center for Disease Control. *Stroke Facts*. 2017. URL: `https://www.cdc.gov/stroke/facts.htm` (visited on 05/09/2017).

[34] American Stroke Association. *Stroke Treatment*. 2017. URL: `http://www.strokeassociation.org/STROKEORG/AboutStroke/Treatment/Stroke-Treatment_UCM_492017_SubHomePage.jsp` (visited on 07/25/2017).

[35] Emelia J. Benjamin et al. "Heart Disease and Stroke Statistics—2017 Update: A Report From the American Heart Association". In: *Circulation* (2017). ISSN: 0009-7322. DOI: `10.1161/CIR.0000000000000485`. eprint: `http://circ.ahajournals.org/content/early/2017/01/25/CIR.0000000000000485.full.pdf`. URL: `http://circ.ahajournals.org/content/early/2017/01/25/CIR.0000000000000485`.

[36] Wikipedia. *FAST (stroke)*. 2017. URL: `https://en.wikipedia.org/wiki/FAST_(stroke)` (visited on 07/25/2017).

[37] Gerard DeMers et al. "Tissue plasminogen activator and stroke: review of the literature for the clinician". In: *The Journal of emergency medicine* 43.6 (2012), pp. 1149–1154.

[38] Dale Birenbaum. "Emergency neurological care of strokes and bleeds". In: *Journal of emergencies, trauma and shock* 3.1 (2010), p. 52.

[39] Richard E Latchaw et al. "Recommendations for Imaging of Acute Ischemic Stroke". In: *Stroke* 40.11 (Oct. 2009), pp. 3646–3678.

[40] Kenneth A Miles. "Perfusion imaging with computed tomography: brain and beyond". In: *European Radiology Supplements* 16.S7 (Nov. 2006), pp. M37–M43.

[41] A M Laslo et al. "CT Perfusion-Derived Mean Transit Time Predicts Early Mortality and Delayed Vasospasm after Experimental Subarachnoid Hemorrhage". In: *American Journal of Neuroradiology* 29.1 (Jan. 2008), pp. 79–85.

[42] J M Biesbroek et al. "Diagnostic Accuracy of CT Perfusion Imaging for Detecting Acute Ischemic Stroke: A Systematic Review and Meta-Analysis". In: *Cerebrovascular Diseases* 35.6 (2013), pp. 493–501.

[43] Max Wintermark et al. "Simultaneous Measurement of Regional Cerebral Blood Flow by Perfusion CT and Stable Xenon CT: A Validation Study". In: *American Journal of Neuroradiology* 22.5 (May 2001), pp. 905–914.

[44] Kohsuke Kudo et al. "Differences in CT Perfusion Maps Generated by Different Commercial Software: Quantitative Analysis by Using Identical Source Data of Acute Stroke Patients". In: *Radiology* 254.1 (2010), pp. 200–209. DOI: `10.1148/radiol.254082000`.

[45] P Rama Krishna et al. "Stroke in chronic kidney disease". In: *Indian journal of nephrology* 19.1 (2009), p. 5.

[46] Paul Meier and Kenneth L Zierler. "On the theory of the indicator-dilution method for measurement of blood flow and volume". In: *Journal of applied physiology* 6.12 (1954), pp. 731–744.

[47] S K Shetty and M H Lev. *CT perfusion.* Vol. 21. Acute ischemic stroke. Heidelberg, 2006.

[48] L Axel. "Cerebral blood flow determination by rapid-sequence computed tomography: theoretical analysis." In: *Radiology* (1980).

[49] Charlotte H P Cremers et al. "Different CT perfusion algorithms in the detection of delayed cerebral ischemia after aneurysmal subarachnoid hemorrhage". In: *Neuroradiology* 57.5 (Jan. 2015), pp. 469–474.

[50] M Sasaki. *Procedure Guidelines for CT/MR Perfusion Imaging 2006, Joint Committee for the Procedure Guidelines for CT/MR Perfusion Imaging.*

[51] A M Smith et al. "Whole brain quantitative CBF, CBV, and MTT measurements using MRI bolus tracking: implementation and application to data acquired from hyperacute stroke patients." In: *Journal of Magnetic Resonance Imaging* 12.3 (Sept. 2000), pp. 400–410.

[52] P.-A. Doriot et al. "Is the indicator dilution theory really the adequate base of many blood flow measurement techniques?" In: *Medical Physics* 24.12 (1997), pp. 1889–1898. ISSN: 2473-4209. DOI: `10.1118/1.598102`. URL: `http://dx.doi.org/10.1118/1.598102`.

[53] L Axel. "Tissue mean transit time from dynamic computed tomography by a simple deconvolution technique." In: *Investigative Radiology* 18.1 (Jan. 1983), pp. 94–99.

[54] John C Wood et al. "MRI R2 and R2* mapping accurately estimates hepatic iron concentration in transfusion-dependent thalassemia and sickle cell disease patients". In: *Blood* 106.4 (2005), pp. 1460–1465.

[55] Antje Schmidt et al. "Photochemically induced ischemic stroke in rats". In: *Experimental & Translational Stroke Medicine* 4.1 (Aug. 2012), pp. 1–1.

[56] Andreas Kastrup et al. “Assessment of cerebrovascular reactivity with functional magnetic resonance imaging: comparison of CO 2 and breath holding”. In: *Magnetic resonance imaging* 19.1 (2001), pp. 13–20.

[57] The Scipy Community. *scipy.sparse.linalg.LinearOperator—SciPy v0.19.0 Reference Guide*. 2017. URL: `https://docs.scipy.org/doc/scipy-0.19.0/reference/generated/scipy.sparse.linalg.LinearOperator.html` (visited on 07/25/2017).

[58] P Chanial and N Barbey. “PyOperators: Operators and solvers for high-performance computing”. In: *SF2A-2012: Proceedings of the Annual meeting of the French Society of Astronomy and Astrophysics*. 2012, pp. 513–517.

[59] Ghislain Vaillant. *linop documentation*. 2017. URL: `http://pythonhosted.org/linop/` (visited on 07/25/2017).

[60] Steven Diamond and Stephen Boyd. “Matrix-free convex optimization modeling”. In: *Optimization and Its Applications in Control and Data Sciences*. Springer, 2016, pp. 221–264.

[61] Justin J Konkle et al. “A convex formulation for magnetic particle imaging x-space reconstruction”. In: *PloS one* 10.10 (2015), e0140137.

[62] Amir Beck and Marc Teboulle. “A fast iterative shrinkage-thresholding algorithm for linear inverse problems”. In: *SIAM journal on imaging sciences* 2.1 (2009), pp. 183–202.

[63] Stanley R Deans. *The Radon transform and some of its applications*. Courier Corporation, 2007.

[64] Christian Kaethner et al. “Non-equispaced system matrix acquisition for magnetic particle imaging based on Lissajous node points”. In: *IEEE transactions on medical imaging* 35.11 (2016), pp. 2476–2485.

[65] Klaas Bente et al. “Electronic field free line rotation and relaxation deconvolution in magnetic particle imaging”. In: *IEEE transactions on medical imaging* 34.2 (2015), pp. 644–651.

[66] Brad Barber and H Huhdanpaa. “Qhull”. In: *The Geometry Center, University of Minnesota, http://www. geom. umn. edu/software/qhull* (1995).

[67] Andreas Fabri and Sylvain Pion. “CGAL: The computational geometry algorithms library”. In: *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. ACM. 2009, pp. 538–539.

[68] Hao Gao. “Fast parallel algorithms for the x-ray transform and its adjoint”. In: *Medical physics* 39.11 (2012), pp. 7110–7120.

[69] Anders H Andersen and Avinash C Kak. “Simultaneous algebraic reconstruction technique (SART): a superior implementation of the ART algorithm”. In: *Ultrasonic imaging* 6.1 (1984), pp. 81–94.

[70] Kihwan Choi et al. “Compressed sensing based cone-beam computed tomography reconstruction with a first-order method”. In: *Med. Phys* 37.9 (2010), pp. 5113–13.

[71] Xueli Li and Shuqian Luo. “A compressed sensing-based iterative algorithm for CT reconstruction and its possible application to phase contrast imaging”. In: *BioMedical Engineering OnLine* 10.1 (Aug. 2011), p. 73.

[72] Stephen Becker, Jérôme Bobin, and Emmanuel J Candès. “NESTA: A fast and accurate first-order method for sparse recovery”. In: *SIAM Journal on Imaging Sciences* 4.1 (2011), pp. 1–39.

[73] Álvaro Barbero and Suvrit Sra. “Fast Newton-type Methods for Total Variation Regularization.” In: *ICML*. Ed. by Lise Getoor and Tobias Scheffer. Omnipress, 2011, pp. 313–320. URL: `http://dblp.uni-trier.de/db/conf/icml/icml2011.html#JimenezS11`.

[74] Álvaro Barbero and Suvrit Sra. “Modular proximal optimization for multidimensional total-variation regularization”. In: (2014). URL: `http://arxiv.org/abs/1411.0589`.

Appendix A

Code for Radon Transform in Sparse Sampling Magnetic Particle Imaging

In the sparse reconstruction chapter, we discussed an algorithm from Gao [68] that calculates the radon transform quickly. The algorithm achieves faster speeds largely by reducing the number of set intersections performed through determining which sets cannot possibly intersect, and by utilitizing the high degree of parallelizability inherent in the radon transform. The intersection method for a finite parallel beam with a unit square of known center can also be made highly efficient, for example using the 2D intersecting area formula from Gao's Appendix B.

The Gao variant requires the restriction that the image grid $\mathcal{G}$ is rectilinear grid with square cells of unit dimension, and is approximately two times slower in practice than the sparse matrix multiply. However, the only optimizations in this formulation is to parallelize the algorithm using OpenMP; a version of this algorithm running on a GPU is likely to be quite fast in comparison.

The following code shows the modifications made to Gao to work for parallel beam radon projections.

Algorithm 2 Gao's Forward DRT for Regular Square Regions, Adapted for Finite-Width Beam

```
Input: $n_x \in \mathbb{Z}_+$, number of pixels along the x-axis, centered at 0;
  $n_y \in \mathbb{Z}_+$, number of pixels along the y-axis, centered at 0;
  List $\mathcal{P}$ of triples $P_j = (j, \tau_{j-}\tau_{j+})$ for vertical projection beam bounded by $[\tau_{j-}, \tau_{j+}]$ along the x-axis.
  List $\mathcal{F}$ of triples $F_{xy} = (x, y, v[x, y])$ for unit square pixel region $r_{xy}$ with value $v[x, y]$ and center located at $(x, y)$;
Output: The sinogram output S($f$) after applying the discrete radon transform (DRT)

  for all $(j, \tau_{j-}, \tau_{j+}) \in \mathcal{P}, \theta_k \in \Theta$ do
    $S(j, k) = 0$
    $m = -1/\tan(\theta_k)$
    $a = \cos(\theta_k)$
    $b = \sin(\theta_k)$

    if $|m| < 1$ then
      $\tau_-, \tau_+ = \begin{cases} \tau_{j-}, \tau_{j+} & \text{if } b > 0 \\ \tau_{j+}, \tau_{j-} & \text{if } b \leq 0 \end{cases}$
      for $x \in \mathbb{Z} : 0.5 - n_x/2 \leq x \leq n_x/2 - 0.5$ do
        $q_- = \lfloor \tau_-/b + mx \rfloor - 0.5 - \text{mod}(n_y, 2)/2$
        $y_{min} = \max(0.5 - n_y/2, q_-)$
        $q_+ = \lceil \tau_+/b + mx \rceil + 0.5 + \text{mod}(n_y, 2)/2$
        $y_{max} = \min(n_y/2 - 0.5, q_+)$
        for $y \in \mathbb{Z} : y_{min} \leq y \leq y_{max}$ do
          $S(j, k) = S(j, k) + \text{vol}(r_{xy} \cap p_j) v[x, y]$
        end for
      end for

    else
      $\tau_-, \tau_+ = \begin{cases} \tau_{j-}, \tau_{j+} & \text{if } a > 0 \\ \tau_{j+}, \tau_{j-} & \text{if } a \leq 0 \end{cases}$
      for $y \in \mathbb{Z} : 0.5 - n_y/2 \leq y \leq n_y/2 - 0.5$ do
        $q_- = \lfloor \tau_-/a + y/m \rfloor - 0.5 - \text{mod}(n_y, 2)/2$
        $x_{min} = \max(0.5 - n_x/2, q_-)$
        $q_+ = \lceil \tau_+/a + y/m \rceil + 0.5 + \text{mod}(n_y, 2)/2$
        $x_{max} = \min(n_x/2 - 0.5, q_+)$
        for $x \in \mathbb{Z} : x_{min} \leq x \leq x_{max}$ do
          $S(j, k) = S(j, k) + \text{vol}(r_{xy} \cap p_j) v[x, y]$
        end for
      end for

    end if
  end for
```

Algorithm 3 Gao's Adjoint DRT for Regular Square Regions, Adapted for Finite-Width Beam

Input: $n_x \in \mathbb{Z}_+$, number of pixels along the x-axis, centered at 0;
$n_y \in \mathbb{Z}_+$, number of pixels along the y-axis, centered at 0;
List $\mathcal{P}$ of triples $P_j = (j, \tau_{j-}\tau_{j+})$ for vertical projection beam bounded by $[\tau_{j-}, \tau_{j+}]$ along the x-axis;
The sinogram input in the form of triples $(p_j, \theta_k, S(j,k))$ representing the sample value S(j,k) of slab p_j rotated by θ_k.

Output: Cartesian image, formed after applying the adjoint DRT, in the form of triples $F_{xy} = (x, y, v[x,y])$ for unit square pixel region r_{xy} with value $v[x,y]$ and center located at (x,y).

```
for all $(j, \tau_{j-}, \tau_{j+}) \in \mathcal{P}, \theta_k \in \Theta$ do
    $S(j,k) = 0$
    $m = -1/\tan(\theta_k)$
    $a = \cos(\theta_k)$
    $b = \sin(\theta_k)$

    if $|m| < 1$ then
        $\tau_-, \tau_+ = \begin{cases} \tau_{j-}, \tau_{j+} & \text{if } b > 0 \\ \tau_{j+}, \tau_{j-} & \text{if } b \le 0 \end{cases}$
        for $x \in \mathbb{Z} : 0.5 - n_x/2 \le x \le n_x/2 - 0.5$ do
            $q_- = \lfloor \tau_-/b + mx \rfloor - 0.5 - \mathrm{mod}(n_y, 2)/2$
            $y_{min} = \max(0.5 - n_y/2, q_-)$
            $q_+ = \lceil \tau_+/b + mx \rceil + 0.5 + \mathrm{mod}(n_y, 2)/2$
            $y_{max} = \min(n_y/2 - 0.5, q_+)$
            for $y \in \mathbb{Z} : y_{min} \le y \le y_{max}$ do
                $v[x,y] = v[x,y] + \mathrm{vol}(r_{xy} \cap p_j) S(j,k)$
            end for
        end for

    else
        $\tau_-, \tau_+ = \begin{cases} \tau_{j-}, \tau_{j+} & \text{if } a > 0 \\ \tau_{j+}, \tau_{j-} & \text{if } a \le 0 \end{cases}$
        for $y \in \mathbb{Z} : 0.5 - n_y/2 \le y \le n_y/2 - 0.5$ do
            $q_- = \lfloor \tau_-/a + y/m \rfloor - 0.5 - \mathrm{mod}(n_y, 2)/2$
            $x_{min} = \max(0.5 - n_x/2, q_-)$
            $q_+ = \lceil \tau_+/a + y/m \rceil + 0.5 + \mathrm{mod}(n_y, 2)/2$
            $x_{max} = \min(n_x/2 - 0.5, q_+)$
            for $x \in \mathbb{Z} : x_{min} \le x \le x_{max}$ do
                $v[x,y] = v[x,y] + \mathrm{vol}(r_{xy} \cap p_j) S(j,k)$
            end for
        end for

    end if
end for
```