Title
Comprehensive Serial Manipulator Forward Kinematics Calibration Scheme

Author
Wu, Wen-Cheng

Publication Date
2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Comprehensive Serial Manipulator Forward Kinematics Calibration Scheme

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Mechanical Engineering

by

Wen-Cheng Wu

2023

ABSTRACT OF THE THESIS

Comprehensive Serial Manipulator Forward Kinematics Calibration Scheme

by

Wen-Cheng Wu
Master of Science in Mechanical Engineering
University of California, Los Angeles, 2023
Professor Tsu-Chin Tsao, Chair

The misalignment error during assembly and/or manufacturing error of robot linkages may compromise the accuracy of a serial manipulator, causing the actual forward kinematics model parameter to differ from the designed value. This demands the need for forward kinematics calibration.

To identify the actual model parameter, the calibration process involves recording data from a set of robot poses as input to a parameter identification algorithm. In order to save time on the data acquisition, optimal design of experiment is employed to use the least amount of data points. Also, to avoid unnecessary calculations and numerical issues during parameter identification, identifiability analysis is performed to determine and eliminate unidentifiable parameters. Lastly, this work proposes novel nested algorithm to improve computation efficiency and robustness for parameter identification. This approach exploits the fact that part of the model parameter can be obtained explicitly given the rest of the parameters. This effectively reduces the parameters to find during the iterative identification process and is shown to be more efficient and robust than the commonly used generic method in the literature.

The calibration method is applied to a common 6-DOF industrial robot and a customarily designed and fabricated 4-DOF surgical robot. For the industrial robot, the nested algorithm demonstrates better robustness, faster error convergence over iteration, and less calculation time than the commonly used generic algorithm. For the surgical robot, the results for optimally designed data points demonstrate a faster convergence of residual error over the number of data points compared to randomly designed data points.

The thesis of Wen-Cheng Wu is approved.

Robert Thomas M'Closkey

Dennis W. Hong

Tsu-Chin Tsao, Committee Chair

University of California, Los Angeles

2023

To my family, your guidance and support make me who I am today.

To my friends at UCLA, the sweetest nuisance parameters to my optimization problem.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my gratitude to Professor Tsu-Chin Tsao, my advisor, whose exceptional engineering mind and guidance have been invaluable throughout my research. Discussions with him are always a pleasure. When conveying ideas, his choice of words is precise and insightful, which always rewards me with more insight than I initially had. His research directions are both focused and inspiring without limiting my potential. On the other hand, when trying to understand my ideas, his ability to quickly grasp complex and abstract ideas always amazes me. This greatly facilitates a smooth discussion, allowing us to move on to even more complex topics swiftly. I am deeply grateful for his guidance and support.

To Yuting Lai, Kevin, your efficiency and dedication are something to look up to. I am especially grateful that you helped me a lot in setting up the Labview environment. Every time When I reached out to you for help, it would be done within the next two days. I am grateful, and I believe that the whole lab shares the same gratefulness toward you. In addition to being a good colleague, you are also a good friend, who helped me a lot when I initially get here. I will never forget your kindness and support.

For the completion of Ch. 6 in this work, Industrial Technology Research Institute (ITRI) provided equipment for the experiment. Special thanks to Tien-Yun Chi, who taught me how to use the equipment and provided assistance when needed. It helped me a lot when I first get started.

Finally, to my family, your support throughout my academic life means a lot to me. To my friends at UCLA, love you as always.

VITA

2019	B.S. in Mechanical Engineering, National Taiwan University
2020-present	Research Assistant, Mechatronics and Control Lab, Mechanical Engineering Department, UCLA

CHAPTER 1

Introduction

The demand for the accuracy of robotic serial manipulators is increasingly high with more demanding tasks. Here, the accuracy of a serial manipulator is defined as the difference between the actual and estimated position/orientation of the robot end-effector. The estimation is done using prior knowledge of the robot model, known as the Forward Kinematics (FK), which includes its designed dimensions and other mechanical properties. However, due to misalignment errors and/or manufacturing errors of the linkage, the accuracy of the FK estimation may be compromised. Thus, FK calibration is needed to identify the said misalignment and manufacturing error, and then compensate for them.

Chapter 2 introduces the mathematical tool needed to perform FK calibration. The FK calibration involves using a number of experimental data to fit the FK model parameters via an optimization procedure, which minimizes the error between the measured and model-estimated endpoint coordinates. The formulation of the problem along with the algorithms for identifying the model parameters will be introduced. In addition to commonly known methods, such as Gauss-Newton method and Levenberg-Marquardt method, this work proposes a nested optimization algorithm for the purpose of making a more efficient numerical optimization procedure and computation. The proposed method solves a smaller optimization problem, which eliminates part of the parameters and reduces the computation burden. The condition is that the optimal solution of the eliminated parameters can be found explicitly when other parameters are known. It is related to the concept of profile likelihood.

Chapter 3 discusses the identifiability analysis of the model parameters. The concept of

identifiability of model parameters and its association with the precision of model parameter estimation will be introduced. Also, analytical and numerical methods for determining identifiability of model parameters shall be explained. One of those methods involves the Singular Value Decomposition (SVD) analysis of a scaled Jacobian, which gives a more physical sense and fewer numerical issues compared to previously known approach. Additionally, since most literature customarily proposes to fix those unidentifiable parameters in solving the optimization problem, we will analyze the effect of fixing the unidentifiable parameters on the accuracy and precision of the model parameter estimation. A sufficient condition to lower the mean squared error of parameter estimation is given, which justifies the fixing of parameters. This is also known as variance-bias trade-off.

The calibration also involves the selection of the experimental data. Chapter 4 considers the optimal design of the experiment, sometimes simply called optimal design. To calibrate the FK parameter, an experiment using some sort of coordinate measurement machine (CMM) has to be performed to measure the position/orientation of the robot end-effector. This measurement acts as the true value for the end-effector pose. In order to achieve better identifiability of the parameters, multiple measurements are measured with different poses of the robot. The goal of optimal design is to maximize the information (or equivalently, minimizing precision error) extracted from the measurement by finding the optimal set of poses for the robot, within certain pose constraints. Optimal design will allow the estimation of FK parameter to be more precise with fewer poses. This will be beneficial if online calibration is needed and minimal data acquisition time is sought after.

In practice, the FK calibration also involves the identification of other parameters, such as joint stiffness and transformation between the measurement and robot coordinates. Chapter 5 proposes FK calibration steps, including robot modeling, experiment design, identifiability analysis, measuring robot poses, and the FK model parameter identification. FK model uses the following model parameter, including DH parameters, joint stiffness constant (also called joint compliance coefficient in this work), and Coordinate Transformation (CT) parameter.

CT parameter is used to represent the Coordinate Transformation between the CMM frame to robot base frame. The estimation of CT is also called CMM registration problem in this literature. Experiment design discusses some general constraints for the data points and the optimal design problem. For the section about identifiability analysis, the method for detecting and countering non-identifiability is discussed. Also, a few prominent examples of non-identifiability in FK calibration are listed and explained. For the measuring step, the CMM used in this literature will also be introduced. This includes Optical coherence tomography (OCT), commonly used in eye surgery. A method to locate the tool held by the robot from OCT V-scan is also introduced. For the identification step, a generic scheme is introduced which will collectively optimize all FK parameters all at once. A novel scheme using nested optimization will also be discussed. CT parameter is usually not of interest after calibration; however, it is still needed to be included in the optimization. Fortunately, the optimal CT parameter can be explicitly found if other FK parameters are known. Therefore, nested optimization algorithm can be performed to eliminate CT parameters from calibration to reduce the number of parameters to be optimized.

Chapter 6 presents the result of applying the proposed method to an industrial robot. The FK model of that robot is discussed, which assumes a noticeable joint compliance effect. For comparison purposes, the results with and without considering joint compliance are listed. It is shown that calibration with the joint compliance model demonstrates consistent accuracy against varying loads, while calibration without the joint compliance model does not. Unidentifiable parameters are determined. The comparison of generic and proposed nested optimization is listed. It is observed that the estimated parameter values are the same for both algorithms. The result also shows that the calculation time for each iteration is significantly less for nested optimization (18% less for each iteration), and that nested schemes require less iteration to achieve the same error convergence rate (33% less). In addition to the superior efficiency, the nested scheme can be shown to be robust against bad initial guesses of certain model parameters compared to the generic scheme. The residual

error is 0.15 mm for calibration points and 0.49 mm for validation points.

Chapter 7 presents the results of applying the proposed method to a customarily designed and fabricated surgical robot. The joint stiffness effect is not modeled and neglected because of small joint torque levels. The robotic surgical workflow calls for two calibration scenarios. The first is the full calibration scenario, where the whole kinematic chain of the robot is calibrated. The second is the in situ partial calibration scenario, where only the last 2 joints of the robot are calibrated. The motivation behind the second scenario comes from the special mechanism of the surgical robot, which can exchange tools. Also, Since the CMM used for the surgical robot is an OCT (with $25\mu m$ measurement accuracy), there are certain robot poses that can produce poor-quality OCT measurements. Certain Cartesian space constraints have to be applied to the data points. Furthermore, it is desired to use fewer data points to reduce data acquisition time. Therefore, optimal design with the said constraint is performed for both calibration scenarios. The result for the optimally designed points shows a faster convergence rate in residual error over the number of data points, compared to randomly designed data points. The residual error of validation points is around $45\mu m$ after full calibration and around $60\mu m$ with only the partial calibration of the last two links, while the parameters of the first two links from the full calibration are fixed.

CHAPTER 2

Model regression: Optimization Algorithm and Analysis

The calibration problem is actually a model regression problem. In a model regression problem, there is a mathematical model estimating the behavior of a real-world system. The goal of model regression is to find the best set of model parameters to make this estimation as accurate as possible. In the case of Forward Kinematics (FK) calibration, the model is the FK model of the robot, denoted $f(x, \beta)$, predicting the position/orientation of the robot end-effector, denoted y.

$$\hat{y} = f(x, \beta) \tag{2.1}$$

, where the overhead symbol $\wedge$ means prediction. The formulation of $f(\cdot)$ is derived using the knowledge of Kinematics. The model function accepts certain variables x as inputs. This variable is varying and can be measured or controlled. In FK calibration, x would be joint configuration and sometimes includes joint torque. The model function also accepts model parameters β as input. Unlike controlled input x, the model parameter β is unknown (can not be directly measured) and is a fixed constant. For FK calibration, the most important model parameter would be the Denavit–Hartenberg parameters (DH parameters) [DH55]. Other related parameters would be introduced in 5. The goal of FK calibration is to find the optimal set of model parameters β^* to best fit/predict the position/orientation of the end effector y, more specifically, minimizing $\|y - \hat{y}\|$.

2.1 Problem formulation

Given a set of data points (x, y), where $x \in R^p$ is the input variable (also known as independent variable, controlled variable), and $y \in R^m$ is the output variable (else called measured variable, dependent variable),

$$x \coloneqq [x_1^T \ x_2^T \ ... \ x_p^T]^T, \quad y \coloneqq [y_1^T \ y_2^T \ ... \ y_m^T]^T \tag{2.2}$$

Usually, controlled input x is actively changed, and then measurement of output y is performed. A model function $f : (R^p, R^n) \to R^m$ is also given as

$$\hat{y} = f(\hat{x}, \beta) \tag{2.3}$$

, which estimates output variable $\hat{y}$ with respect to a certain input variable, x, and the model parameter $\beta \in R^n$. The residual error is then given by

$$e(\beta) = [e_1^T \ e_2^T \ ... \ e_m^T]^T = y - f(x, \beta) \tag{2.4}$$

Nonlinear Least Square (NLS) optimization is a model fitting problem, aiming to find the optimal set of model parameter, β^*, which minimizes the 2-norm of residual error. More specifically, the objective function, $S(\beta)$, and the optimization problem can be defined as followed

$$\beta^* = arg \min_{\beta} S(\beta), \quad S(\beta) \coloneqq e^T W e = \sum_{i=1}^{m} w_i e_i^T e_i \tag{2.5}$$

, where $W = diag(w_1, w_2, \cdots, w_m)$ is the weighting matrix, which weighs each element in the residual error. Weighting matrix doesn't play an important role until there are measurements of mixed units or mixed measuring methods inside y. In such case, $w_i = 1/\sigma_i^2$ is normally

used, where $\sigma_i^2 = Var(y_i)$. As for the reason for choosing such weight and choosing the form of least square (2.5) as the objective is explained in the next section, Section 2.2.

2.2 Maximum Likelihood Estimation

Choosing the form of least square as given in (2.5) is to maximize the log likelihood function. This is also known as Maximum Likelihood Estimation (MLE). To see this, the following statistical model is assumed

$$\begin{aligned} y &= f(x_t, \beta_t) + e_y \\ x &= x_t + e_x \end{aligned} \tag{2.6}$$

(x_t, β_t) is the true measurement and true parameter, and $\beta \in R^n$. There is random error e_y and e_x on the measurement $y \in R^m$ $(m > n)$ and controlled input $x \in R^p$. e_y and e_x are assumed to be normal random vectors with zero mean, which means

$$\begin{aligned} e_x &\sim N(0, V_x), \quad V_x = diag(\sigma_{x,1}^2, \sigma_{x,2}^2, \cdots, \sigma_{x,p}^2) \\ e_y &\sim N(0, V_y), \quad V_y = diag(\sigma_{y,1}^2, \sigma_{y,2}^2, \cdots, \sigma_{y,m}^2) \end{aligned} \tag{2.7}$$

, where $\sigma_{x,i}$ and $\sigma_{y,i}$ are standard deviation of the normal distribution for i-th element of e_x and e_y.

The log likelihood function is

$$L(\theta; Z) = -\frac{1}{2}(e_x^T V_x^{-1} e_x + e_y^T V_y^{-1} e_y) + constant \tag{2.8}$$

Unknown parameter θ actually includes both β_t and x_t. The observation is $Z = [y; x]$. Assume e_x is small and the following approximation can be made

$$f(x, \beta) \cong f(x_t, \beta) + J_x e_x, \quad J_x = \left.\frac{\partial f(x, \beta)}{\partial x}\right|_{x=x_t,\ \beta=\beta_t} \tag{2.9}$$

To maximize log likelihood function, it is the same to minimize $S'(\theta; Z) = e_x^T V_x^{-1} e_x + e_y^T V_y^{-1} e_y$.

$$\begin{aligned} S'(\theta; Z) &= e_x^T V_x^{-1} e_x + [y - f(x_t, \beta_t)]^T V_y^{-1} [y - f(x_t, \beta_t)] \\ &= e_x^T V_x^{-1} e_x + [y - f(x, \beta_t) + J_x e_x]^T V_y^{-1} [y - f(x, \beta_t) + J_x e_x] \\ &= e_x^T V_x^{-1} e_x + [e(\beta_t) + J_x e_x]^T V_y^{-1} [e(\beta_t) + J_x e_x] \\ &= [e_x + C_x J_x^T V_y^{-1} e(\beta_t)]^T C_x^{-1} [e_x + C_x J_x^T V_y^{-1} e(\beta_t)] + e^T(\beta_t) V_e^{-1} e(\beta_t) \end{aligned} \tag{2.10}$$

, where $C_x = (V_x^{-1} + J_x^T V_y^{-1} J_x)^{-1}$ and $V_e = V_y + J_x V_x J_x^T$. The last equality of the above equation uses Matrix Inversion Lemma. Note that the second term is purely the function of β_t not x_t, if J_x is stays relatively the same with small variation of x, meaning$J_x(x_t, \beta_t) \cong J_x(x, \beta_t)$. the problem of maximizing log likelihood function then becomes two independent problems.

$$\begin{aligned} &\min_{\beta_t} e^T(\beta_t) V_e^{-1} e(\beta_t) \\ &x_t = x + C_x J_x^T V_y^{-1} e(\beta_t) \end{aligned} \tag{2.11}$$

In practice, x_t is usually not of interest so the second equation listed in (2.11) is rarely used. Also for the first optimization problem listed in (2.11), since $e(\beta_t)$ uses the measurement x instead of the unknown true value x_t, the problem can be solved without explicitly derive x_t. Lastly, the first optimization problem listed in (2.11) is of least square form. If the prior knowledge of V_y and V_x is obtained, the weighting matrix in (2.5) can be set to $W = V_e^{-1}$. This effectively makes the minimizing least square problem into MLE. Although, in practice, simplification and approximation of $W = V_e^{-1}$ are usually performed for convenience. Such simplification usually assumes controlled input can be measured precisely, and $V_x \approx 0$. Therefore, $V_e \approx V_y$. This is why the weight matrix is usually chosen as the inverse of V_y (mentioned at the end of Section 2.1).

2.3 optimization algorithm

There are several methods for solving the NLS optimization problem given in (2.5), some of which, including the Gauss-Newton (GN) method [GG09] and the Levenburg-Marquardt (LM) method [LEV44, Mar63] will be introduced in the following sections.

2.3.1 Gauss-Newton method

With a small deviation of model parameter $\Delta\beta \in R^n$, assume that the model function can be approximated as

$$f(x, \beta + \Delta\beta) \cong f(x, \beta) + J\Delta\beta \tag{2.12}$$

where J is the multi-variable derivative of f by β, which is else called the Jacobian matrix and is defined as

$$J(\beta) = \frac{\partial f(x, \beta)}{\partial \beta} \tag{2.13}$$

The Jacobian matrix can be calculated by explicit function if such function is known. However, in most cases, such function may be too complicated to calculate or not known at all. Then, Jacobian matrix has to be calculated numerically by approximation, such as forward finite difference as followed

$$J_j(\beta) \cong \frac{f(x, \beta + \delta_j u_j) - f(x, \beta)}{\delta_j} \tag{2.14}$$

, where $J_j \in R^m$ is the j-th column of J. The finite difference step, $\delta_j \in R$, for j-th parameter is chosen as a small value. $u_j \in R^n$ is defined as,

$$[u_j]_k = \begin{cases} 1 & k = j \\ 0 & k \neq j \end{cases} \tag{2.15}$$

The difference between the approximation made by (2.14) and the true Jacobian matrix is referred to as truncation error, ϵ_t, which is proportional to the step size δ_j [Mat12, Olv14] (see (2.16)).

$$\epsilon_t = C_t|\delta_j| \tag{2.16}$$

Also, since every finite precision computing machine has a floating point error, ϵ, when computing the (2.14), there will be roundoff error, ϵ_r [Mat12]. In the most simple case, ϵ_r takes the following form

$$\epsilon_r = C_r \frac{\epsilon}{|\delta_j|} \tag{2.17}$$

The dilemma in choosing the right sizing for δ_j is that, if δ_j is too large there will be large truncation error. However, if δ_j is too small, roundoff error would be large. To choose the right sizing, both error should be the same size, such that

$$\epsilon_t = \epsilon_r \Rightarrow C_t|\delta_j| = C_r \frac{\epsilon}{|\delta_j|} \Rightarrow |\delta_j| = C\sqrt{\epsilon} \tag{2.18}$$

As shown in the above equation, the optimal finite step δ_j should be proportional to the square root of ϵ. However, since calculating C_t and C_r involves additional computation whether by analytical or numerical method, they are not found in practice. Rather, the following step size has been used and proved satisfactory.

$$\delta_j = sign(\beta_j) max(1, |\beta_j|)\sqrt{\epsilon} \tag{2.19}$$

With the assumption (2.12), then the following approximation of residual error (2.4) and objective function (2.5) can be written

$$e(\beta + \Delta\beta) \cong e(\beta) - J\Delta\beta \tag{2.20}$$

$$\begin{aligned} S(\beta + \Delta\beta) &= e^T(\beta + \Delta\beta)We(\beta + \Delta\beta) \\ &\cong [e(\beta) - J\Delta\beta]^T W[e(\beta) - J\Delta\beta] \\ &= e^T(\beta)We(\beta) - 2e^T(\beta)WJ\Delta\beta + \Delta\beta^T J^T WJ\Delta\beta \\ &= S(\beta) - 2e^T(\beta)WJ\Delta\beta + \Delta\beta^T J^T WJ\Delta\beta \end{aligned} \tag{2.21}$$

When $S(\beta + \Delta\beta)$ reaches minimum, its derivative by $\Delta\beta$ should be 0 vector, such that

$$\frac{\partial S(\beta + \Delta\beta)}{\partial \Delta\beta} = -2e(\beta)^T WJ + 2\Delta\beta^T J^T WJ = 0$$

$$\Rightarrow \ \Delta\beta^* = (J^T WJ)^{-1} J^T We(\beta) \tag{2.22}$$

Note that the inversion in (2.22) implies that $J^T WJ$ is non-singular, which is assumed true for now. The singularity of $J^T WJ$ is associated with the identifiability of model parameters. The identifiability of a certain parameter can be linked to the precision of the estimation for that parameter. The identified value of unidentifiable parameters would vary a lot if the data point (y, x) is collected again and the model regression is conducted again. Mathematically speaking, This corresponds to the situation that, when $J^T WJ$ is near singular, any small deviation/noise of $e(\beta)$ would be amplified, and $\Delta\beta^*$ would vary greatly. There are ways to detect such non-identifiability. After detection, to counter the non-identifiability, either remodeling of the model can be performed or the unidentifiable parameters can be fixed at a nominal value as a constant. All these are to prevent the inverse of a singular matrix and its accompanying numerical issue. These will be further discussed in Chapter 3.

The solution of (2.22) will then be applied iteratively to solve the NLS problem (2.5) as shown in Figure 2.1. The implementation of this method is shown in Algorithm 1. The stopping criteria ("check convergence" function) for the algorithm returns true if any of the following is true.

- First-Order Optimality: if the first order gradient of S(β) is smaller than a gradient tolerance, ϵ_g

$$\frac{\partial S(\beta)}{\partial \beta} = \frac{\partial e^T(\beta) W e(\beta)}{\partial \beta} = 2e^T(\beta) W \frac{\partial e(\beta)}{\partial \beta} = -2e^T(\beta) W J(\beta) \tag{2.23}$$

$$\left\| \frac{\partial S(\beta)}{\partial \beta} \right\|_{\beta=\beta_k} = \left\| -2J_k^T W e_k \right\| < \epsilon_g \tag{2.24}$$

- step size: if the maximum parameter step size, $\Delta\beta_{k,i}$ (i means i-th element in $\Delta\beta_k$) is smaller than a step size tolerance, ϵ_β, in the following sense

$$\max_{i \in [1,n]} \frac{\|\Delta\beta_{k,i}\|}{\|\beta_{k,i}\|} < \epsilon_\beta \tag{2.25}$$

- error change: if the residual error change of e_k is smaller than an error change tolerance, ϵ_e, in the following sense

$$|\|e_k\| - \|e_{k-1}\|| < \epsilon_f \|e_{k-1}\| \tag{2.26}$$

The success of GN method depends on the assumption (2.12), which is approximately correct when the step size, $\Delta\beta$, is small. However, this is not often true. The calculated $\Delta\beta^*$ from (2.22) is often too large for (2.12) to hold true. This implies that this method heavily depends on the correctness of the initial guess, β_0; otherwise, the calculated step will jump too far. This is why Levenburg-Marquardt method is employed instead since LM method can limit the step size.

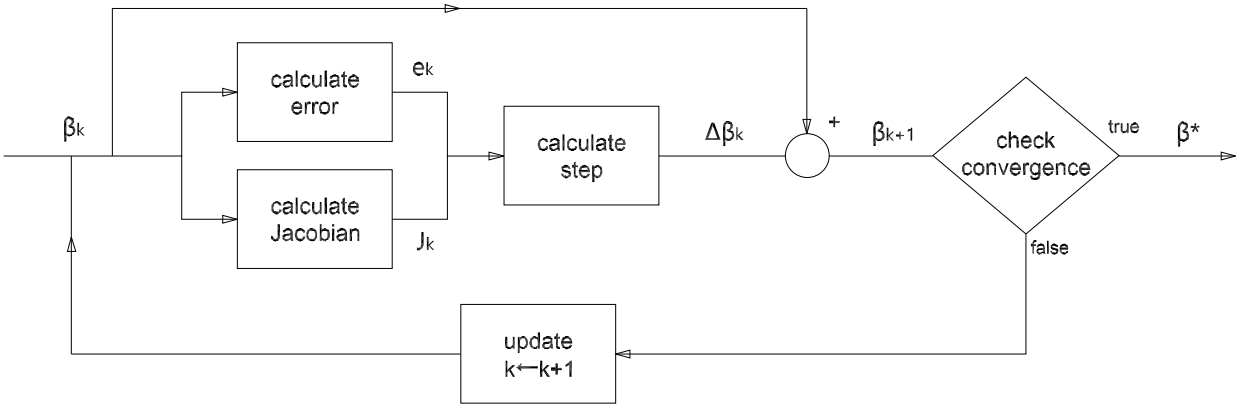


Figure 2.1: NLS solving algorithm diagram

Algorithm 1 algorithm for Guass-Newton method

Input: β_0, ϵ_g, ϵ_e, ϵ_f, W

Output: β_k

$k \leftarrow 0$

while $\sim$ converged **do**

    $J_k = J(\beta_k), \quad e_k = e(\beta_k)$

    $\Delta\beta_k = (J_k^T W J_k)^{-1} J_k^T W e_k$

    $\beta_{k+1} \leftarrow \beta_k + \Delta\beta_k$

    $converged \leftarrow check\_convergence(\epsilon_g, \epsilon_f, \epsilon_\beta, J_k, e_k, \Delta\beta_k, \beta_k)$

    $k \leftarrow k + 1$

end while

2.3.2 Levenburg-Marquardt method

Levenburg-Marquardt (LM) method ([LEV44, Mar63]) changes the parameter updating step (2.22) of GN method to the following form, which introduces in a damping factor λ

$$\Delta\beta^* = (J^T W J + \lambda I)^{-1} J^T W e(\beta) \tag{2.27}$$

Three important theorems that support this method are addressed in [Mar63].

- THEOREM1: The updating step, $\Delta\beta^*$, given in (2.27) minimize the following optimization problem

$$\begin{aligned} &\min_{\Delta\beta} S(\beta + \Delta\beta) \\ &s.t.\ \Delta\beta^T \Delta\beta = (\Delta\beta^*)^T (\Delta\beta^*) \end{aligned} \tag{2.28}$$

  where the objective function is the approximation given in (2.21)

- THEOREM2: $\|\Delta\beta^*\|^2$ is continuous decreasing function of λ, with

$$\lim_{\lambda\to\infty} \|\Delta\beta^*\|^2 = 0 \tag{2.29}$$

- THEOREM3: The angle, γ, between $\Delta\beta^*$ and steepest descent direction of $S(\beta+\Delta\beta)$ given in (2.21) is a continuous decreasing function of λ

$$\lim_{\lambda\to\infty} \gamma = 0 \tag{2.30}$$

Theorem 2 means increasing λ will limit the updating step size. Theorem 1 proves that this modified method could still minimize the approximated objected function (2.21) within this limited step size. Theorem 3 means that this method would transform into steepest descend method with increasing λ.

LM method can be viewed from another aspect. Given a damping factor λ, LM method is actually solving the modified problem below.

$$\min_{\Delta\beta} \tilde{e}^T(\beta+\Delta\beta)\tilde{W}\tilde{e}(\beta+\Delta\beta) \tag{2.31}$$

$$\tilde{e}(\beta+\Delta\beta) \coloneqq \begin{bmatrix} e(\beta+\Delta\beta) \\ \Delta\beta \end{bmatrix}, \quad \tilde{W} \coloneqq \begin{bmatrix} W & 0 \\ 0 & \lambda \end{bmatrix} \tag{2.32}$$

The modified problem (2.31) tries to minimize the norm of the modified error, $\tilde{e}$, which includes both the original error, e, and the updating step, $\Delta\beta$, with λ acting as its weight. This is how LM can minimize the objective function while limiting the step size at the same time. Following the steps given in section 2.3.1, the updating step for the problem (2.31) is

$$\begin{aligned} \Delta\tilde{\beta} &= (\tilde{J}^T\tilde{W}\tilde{J})^{-1}\tilde{J}^T\tilde{W}\tilde{e}(\beta) \\ &= (J^TWJ+\lambda I)^{-1}\begin{bmatrix} J^T(\beta) & I \end{bmatrix}\begin{bmatrix} W & 0 \\ 0 & \lambda \end{bmatrix}\begin{bmatrix} e(\beta) \\ 0 \end{bmatrix} \\ &= (J^TWJ+\lambda I)^{-1}J^TWe(\beta) \end{aligned} \tag{2.33}$$

, which is exactly like (2.27).

There is still one issue remaining for LM method. As shown in (2.31), the optimization problem tries to minimize the 2-norm of $\Delta\beta$. This would become an issue if $\Delta\beta$ has different units for its elements (parameters). Then, the 2-norm $\Delta\beta$ doesn't have a physical meaning. An unfortunate choice of unit may result in drastic differences among the value of different parameters. Therefore, an improved scaled optimization problem is solved instead

$$\min_{\Delta\beta} \tilde{e}^T(\beta+\Delta\beta)\tilde{W}\tilde{e}(\beta+\Delta\beta) \tag{2.34}$$

$$\tilde{e}(\beta+\Delta\beta) \coloneqq \begin{bmatrix} e(\beta+\Delta\beta) \\ diag(J^TWJ)^{\frac{1}{2}}\Delta\beta \end{bmatrix}, \quad \tilde{W} \coloneqq \begin{bmatrix} W & 0 \\ 0 & \lambda \end{bmatrix} \tag{2.35}$$

, where $diag(J^TWJ)$ is a diagonal matrix, whose diagonal elements are those of the J^TWJ. The absolute value of i-th element in $diag(J^TWJ)^{\frac{1}{2}}\Delta\beta$ represents the weighted error change $W^{\frac{1}{2}}e(\beta+\Delta\beta)$ incurred by $\Delta\beta_i$ (step change of i-th parameter) alone. For example, if β_1 is changed by $\Delta\beta_1$ while all other parameters remain the same, the incurred change on weighted error is

$$\begin{aligned} \left\| W^{\frac{1}{2}}[e(\beta+\Delta\beta)-e(\beta)] \right\| &= \left\| -W^{\frac{1}{2}}J(\beta)\Delta\beta \right\| \\ &= \left\| W^{\frac{1}{2}}\left[J_1 \ J_2 \ \cdots \ J_n\right]\left[\Delta\beta_1 \ 0 \ \cdots \ 0\right]^T \right\| \\ &= \left\| W^{\frac{1}{2}}J_1\Delta\beta_1 \right\| \\ &= (J_1^TWJ_1)^{\frac{1}{2}}|\Delta\beta_1| \\ &= \left| [diag(J^TWJ)^{\frac{1}{2}}\Delta\beta]_1 \right| \end{aligned} \tag{2.36}$$

The term $diag(J^TWJ)$ acts as a weight for each parameter in $\Delta\beta$, making them comparable to each other in terms of their individual effect on the weighted residual error, $W^{\frac{1}{2}}e(\beta)$. This scaled optimization problem therefore aims to minimize the original error while limiting the norm of maximum weighted error changes incurred by the individual change of each parameter.

Following previous procedure, the optimal updating step for this scaled problem is

$$\begin{aligned} \Delta\tilde{\beta} &= (\tilde{J}^T\tilde{W}\tilde{J})^{-1}\tilde{J}^T\tilde{W}\tilde{e}(\beta) \\ &= (J^T\tilde{W}J+\lambda diag(J^TWJ))^{-1}\begin{bmatrix} J^T(\beta) & diag(J^TWJ)^{\frac{1}{2}} \end{bmatrix}\begin{bmatrix} W & 0 \\ 0 & \lambda \end{bmatrix}\begin{bmatrix} e(\beta) \\ 0 \end{bmatrix} \\ &= (J^TWJ+\lambda diag(J^TWJ))^{-1}J^TWe(\beta) \end{aligned} \tag{2.37}$$

According to the above text, tuning λ would have an effect on the updating step size and the method for updating. When (2.12) and (2.21) fail to approximate the model function and objective function, and the objective increases, increasing λ will force the step size to decrease (hence a better approximation for (2.21)) and make the updating step walk along steepest descend direction. The implemented LM method is reported in Algorithm 2

Algorithm 2 algorithm for Levenburg-Marquardt method

Input: β_0, ϵ_g, ϵ_e, ϵ_f, λ, W
Output: β_k
$k \leftarrow 0$
while $\sim$ converged **do**
    $J_k = J(\beta_k), \quad e_k = e(\beta_k)$
    $\Delta\beta_k = (J_k^T W J_k + \lambda I)^{-1} J_k^T W e_k$ or $\Delta\beta_k = (J_k^T W J_k + \lambda diag(J_k^T W J_k))^{-1} J_k^T W e_k$
    $\beta_{k+1} \leftarrow \beta_k + \Delta\beta_k$
    $converged \leftarrow check\_convergence(\epsilon_g, \epsilon_f, \epsilon_\beta, J_k, e_k, \Delta\beta_k, \beta_k)$
    if $(k \geq 1) \cap (\|e_k\| < \|e_{k-1}\|)$ **then**
        $\lambda \leftarrow \lambda/2$
    else
        $\lambda \leftarrow \lambda * 2$
    end if
    $k \leftarrow k + 1$
end while

2.4 Nested optimization scheme

In section 2.3, algorithm for solving NLS problem is introduced. However, those algorithm belongs to low-level schemes. In this section, high-level optimization scheme will be introduced. This scheme will seek to take advantage of the nature of the optimization problem to better solve the problem.

As discussed in section 2.3, GN method depends heavily on the initial guess of the model parameters. LM method tries to remedy this disadvantage by tuning the optimization step. However, this will slow down the process. Depending on the nature of the NLS problem, sometimes a subset of the model parameter can be solved in a closed-form solution if given the rest of the model parameters. Nested optimization scheme takes advantage of this and breaks up the model parameters into two groups. One group is solved using LM method, and the other group (called nuisance parameters) can be solved using closed-form solution. This nested scheme can achieve the same result as the generic one with faster convergence speed, since fewer parameters are involved in each iterative, and the accuracy of the initial guess for nested parameters becomes irrelevant.

2.4.1 formulation of nested scheme

Suppose that model parameter β in problem (2.5) can be separated into 2 groups of parameters, $\beta = [\beta_1^T \ \beta_2^T]^T$, $\beta_1 \in R^{n_1}$, $\beta_2 \in R^{n_2}$. With a slight twist of problem (2.5), the NLS problem can be rewritten in the following sense.

$$\begin{aligned} (\beta_1^*, \beta_2^*) &= arg \min_{\beta_1, \beta_2^*} S(\beta_1, \beta_2^*) \\ \beta_2^*(\beta_1) &= arg \min_{\beta_2} S(\beta_1, \beta_2) \end{aligned} \tag{2.38}$$

Now, suppose that, given β_1, the optimal β_2^* (nested parameters) can be calculated explicitly with the optimal solution function $g : R^{n_1} \to R^{n_2}$. Then the problem becomes

$$\begin{aligned}\beta_1^* &= arg\min_{\beta_1} S(\beta_1, g(\beta_1)) \\ \beta_2^* &= g(\beta_1) = arg\min_{\beta_2} S(\beta_1, \beta_2)\end{aligned} \tag{2.39}$$

The equivalence of solving the optimal parameter using generic scheme and nested scheme is proven in [GM95]. The proof given there is to maximize the log likelihood function, which is equivalent to minimizing the least square error function given in (2.5) since normal distribution is assumed. This is closely related to concentrated or profile likelihood, where the optimal value of a subset of parameter β_2 (called nuisance parameter) can be found explicitly by other parameter of interest β_1. Then, the likelihood function is expressed solely by parameter of interest, thus concentrated. The optimal parameter β_1 can be found by maximizing the concentrated likelihood function. This is illustrated in Figure 2.2. The blue line is the contour line of log likelihood function and the red line is $g(\beta_1)$. The red curve on the right is the concentrated likelihood function (or profile).

As shown in (2.39), the nested objective function, $S(\beta_1, g(\beta_1))$, is now only dependent on β_1. See Figure 2.4, LM method can be applied to this nested objective function to solve for optimal β_1^* first. Then the optimal β_2 can be calculated from function $g(\cdot)$. Compared to the generic LM method (Figure 2.3), the nested scheme effectively reduces the parameter number in the iterative LM process. Moreover, since β_2 is calculated explicitly, its initial guess is no longer a problem.

2.4.2 comparison of nested method to generic method

The difference between a generic method and a nested method can be seen in their updating steps. Weighted Jacobian can be partitioned into two groups which correspond to β_1 and β_2.

$$W^{\frac{1}{2}}J = \begin{bmatrix} J_1 & J_2 \end{bmatrix} \tag{2.40}$$

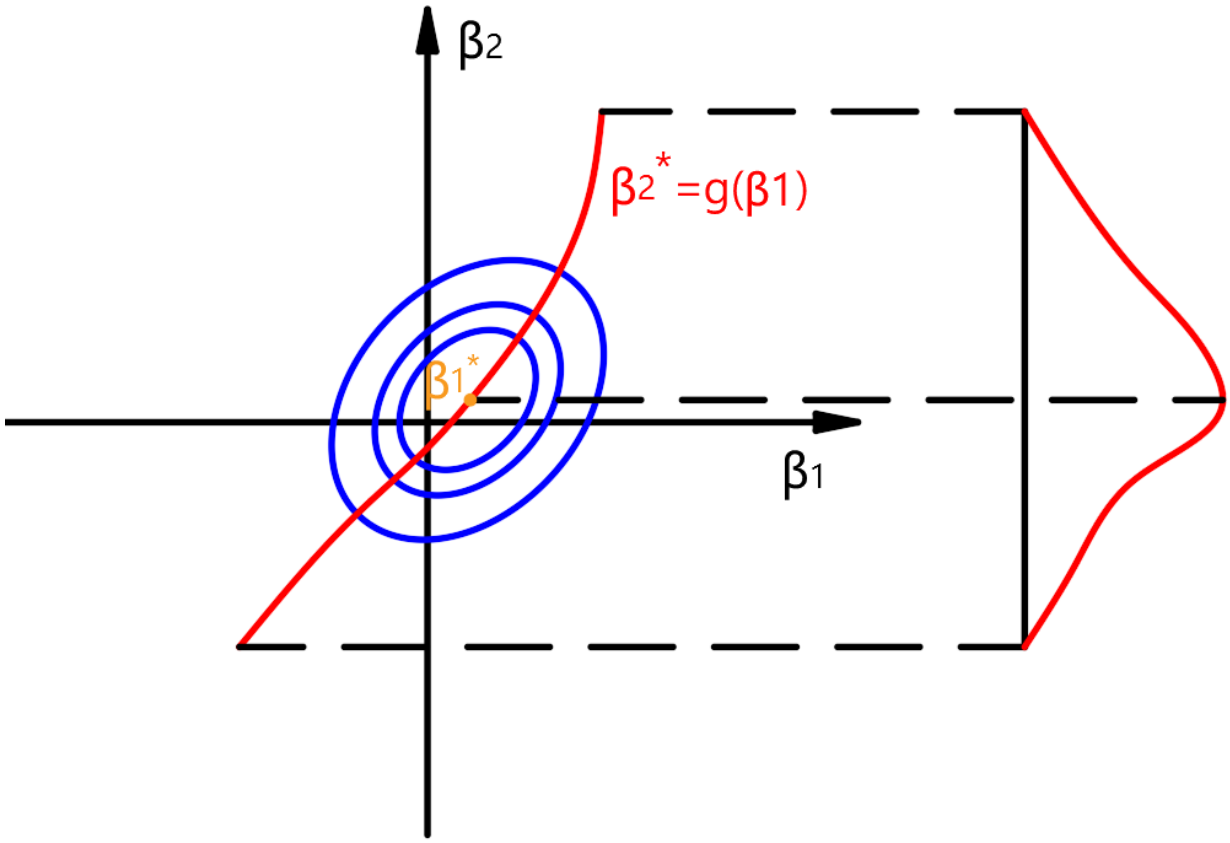


Figure 2.2: concentrated likelihood illustration

Figure 2.3: generic LM method

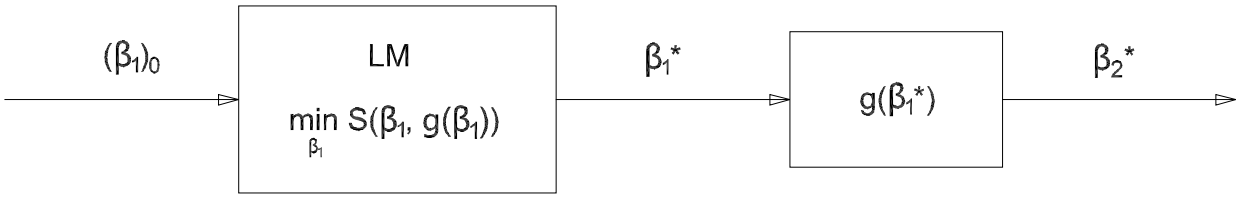


Figure 2.4: nested LM method

For simplicity, here GN method is used instead of LM method for this demonstration. The updating step as described in (2.22) can be written into the following form using block matrix inversion lemma 3.1.2 and matrix inversion lemma 3.1.3

$$\Delta\beta^* = (J^T W J)^{-1} J^T W e$$

$$\Delta\beta^* = \begin{bmatrix} \Delta\beta_1^* \\ \Delta\beta_2^* \end{bmatrix} = \begin{bmatrix} (J_{1\perp}^T J_{1\perp})^{-1} J_{1\perp}^T \\ (J_{2\perp}^T J_{2\perp})^{-1} J_{2\perp}^T \end{bmatrix} W^{\frac{1}{2}} e \tag{2.41}$$

, where

$$\begin{aligned} J_{1\perp} &= [I - J_2 (J_2^T J_2)^{-1} J_2^T] J_1 \\ J_{2\perp} &= [I - J_1 (J_1^T J_1)^{-1} J_1^T] J_2 \end{aligned} \tag{2.42}$$

In comparison, for the nested method, suppose it is possible to have a closed-form estimator of β_2 if given β_1 and (y, x)

$$\beta_2^* = g(\beta_1, y, x) \tag{2.43}$$

The weighted Jacobian matrix by β_1 for the nested scheme is

$$
\begin{aligned}
J_{1g} &= \frac{dW^{\frac{1}{2}} f(\beta_1, g(\beta_1))}{d\beta_1} \\
&= \frac{\partial W^{\frac{1}{2}} f(\beta_1, \beta_2)}{\partial \beta_1} + \frac{\partial W^{\frac{1}{2}} f(\beta_1, \beta_2)}{\partial \beta_2} \frac{\partial g(\beta_1, y, x)}{\partial \beta_2} \\
&= J_1 + J_2 J_g(\beta_1, y, x)
\end{aligned} \tag{2.44}
$$

Then, the updating step for the nested scheme is

$$
\Delta\beta_1^* = (J_{1g}^T J_{1g})^{-1} J_{1g}^T W^{\frac{1}{2}} e \tag{2.45}
$$

A comparison between the updating step of the generic method (2.41) and nested method (2.45) shows the difference comes from $J_{1\perp}$ and J_{1g}. Further inspection between the two in (2.42) and (2.44) shows that the difference comes from J_g and $A = -(J_2^T J_2)^{-1} J_2^T J_1$. Since J_1 and J_2 is not a function of y, $A = -(J_2^T J_2)^{-1} J_2^T J_1$ is not either. However, J_g is a function of y and will be modified by the change of y. It is suspected that this is the reason why the nested scheme has a faster convergence rate than the generic one. More specifically, since the Jacobian matrix will be modified according to the measured data y instead of just the prediction by model alone, it is suspected that the updating step for the nested scheme will be more accurate in predicting the optimal step.

2.4.3 time efficiency of nested scheme

As will be demonstrated in Chapter 7, the LM method takes most of the time ($\sim 90\%$) to calculate Jacobian matrix for each iteration. Therefore, whether nested scheme can reduce the calculation of each iteration depends heavily on the calculation of Jacobian matrix. Using symbol $Time(\cdot)$ to denote the calculation time of its input process, and supposing that $J \in R^{m\times(n_1+n_2)}$, then the calculation time of Jacobian matrix for generic LM method is

$$
Time(J_{generic}) = m(n_1 + n_2) Time(f(\beta_1, \beta_2)) \tag{2.46}
$$

Since Jacobian is calculated using the forward finite difference method (2.14), the calculation of model function f (2.3) is repeated for each element inside Jacobian matrix J. Similarly, for the calculation time of the Jacobian in nested scheme

$$Time(J_{nested}) = mn_1 Time(f(\beta_1, g(\beta_1))) \tag{2.47}$$

Suppose the additional calculation introduced by the calculation of $g(\beta_1)$ is denoted ΔT_g

$$\Delta T_g = Time(f(\beta_1, g(\beta_1))) - Time(f(\beta_1, \beta_2)) \tag{2.48}$$

Then, the time saved by nested scheme is

$$\begin{aligned} \Delta T &= Time(J_{generic}) - Time(J_{nested}) \\ &= m[n_2 Time(f(\beta_1, \beta_2)) - n_1 \Delta T_g] \end{aligned} \tag{2.49}$$

Thus, to maximize ΔT, it is better to employ nested scheme when the additional calculation incurred by $g(\cdot)$ is small, or the number of the parameters saved from the LM method, n_2, is large. Also, it shall be demonstrated in Chapter 7 and Chapter 6 that the nested scheme could reduced total iteration number required to reach the same error convergence stopping criteria. Therefore, even if $\Delta T < 0$, meaning nested scheme takes more time to complete an iteration than generic scheme, the total calculation time could still be reduced by the nested scheme with its reduced iteration number.

CHAPTER 3

Identifiability analysis

In model regression, identifiability refers to the ability to precisely estimate the unknown model parameters from the measured data, y, and controlled input, x. Identifiability is associated with model formulation. The definition of identifiability is

Definition 3.0.1. *A model $f(x, \beta)$ is said to be locally identifiable at $\beta = a$ if there is a open set B, s.t. $a \in B$, and the following is true*

$$f(x, \beta_1) = f(x, \beta_2) \Rightarrow \beta_1 = \beta_2, \quad \forall \beta_1, \beta_2 \in B \tag{3.1}$$

Global identifiability can also be defined if the injective (one-to-one) property of $f(x, \beta)$ is true for all $\beta \in R^n$. The definition comes from definition 5.2 of [LC98], except it uses probability density function instead of model function $f(x, \beta)$. For nonlinear model regression, most of the time, global identifiability is neither achieved nor of interest; Thus, local identifiability is referred to as simply "identifiability" unless clearly specified otherwise.

If a model is unidentifiable, there may be multiple possible β that has the same model prediction $f(x, \beta)$. In another words, the optimal estimation of model parameter may not be unique and have large variance. However, it is not necessarily true that all parameters from an unidentifiable model are unidentifiable. There may still be a subset of identifiable parameters from an unidentifiable model. In light of this, a group of methods referred to as identifiability analysis is then used to determine the identifiability of each parameter in a given model.

As the corner stone of the identifiability analysis, the following theorem relates the singularity of Jacobian, defined in 2.13, with the identifiability of a model

Theorem 3.0.1. *A model $f(x, \beta)$ is locally identifiable at $\beta = a$, if $J_a = J(x, a) \in R^{m \times n}$ has full column rank, assuming $m > n$.*

Proof. Supposing J_a has full column rank, $rank(J_a) = n$, to prove local identifiability, it is equivalent to show

$$\exists B,\ s.t.\ a \in B, and\ \forall \beta_1, \beta_2 \in B,\ \beta_1 \neq \beta_2 \Rightarrow f(x, \beta_1) \neq f(x, \beta_1)$$

First, define a function $g : R^n \to R^n$

$$g(\beta) = J_a^T f(x, \beta)$$

Take the derivative of g

$$\nabla g = J_a^T J_a$$

Since $g(\beta)$ is a function satisfying $g : R^n \to R^n$, and also its derivative, $\nabla g(a) = J_a^T J_a \in R^{n \times n}$, has full rank. Invoking Inverse function theorem, $g(\beta)$ is locally invertible, which implies it is injective as well

$$\begin{aligned} \exists B,\ s.t.\ a \in B, and\ \forall \beta_1, \beta_2 \in B,\ \beta_1 \neq \beta_2 &\Rightarrow J_a^T f(x, \beta_1) \neq J_a^T f(x, \beta_1) \\ &\Rightarrow f(x, \beta_1) \neq f(x, \beta_1) \end{aligned}$$

□

An estimator $\beta^* = T(Z)$ is a method to calculate an estimation of the true model parameter β_t given a measurement Z. As shall be seen in the following section, there is a theoretical lower bound on the variance of any unbiased estimator β^*. The mentioned lower

bound is called Cramér–Rao bound (CRB) [Cra46, Rao92], which gives the theoretically best precision one can inferred from the measured data. When model is close to non-identifiability, it shall be shown that CRB would have large value. This means non-identifiability implies bad precision for parameter estimation. Also, any estimator variance that can reach the CRB is called efficient. It shall be shown in Section 3.1 that GN method is efficient, supposing that certain conditions are met.

To achieve identifiability, there are two possible solutions. One solution is to change model parameterization so that the new model is identifiable. As shall be explained in later sections, this might be unwanted in some application such as the case in this literature. The other solution is to fix some of the unidentifiable parameters. This is equivalent to introducing bias in parameter estimation in exchange for lower variance. If prior information on the upper bound of certain parameter bias is known, and the CRB for that parameter is larger, fixing the parameter wouldn't be a bad trade. This is known as variance-bias trade-off.

Sometimes, non-identifiability is caused by a poor choice of controlled input, making the corresponding measurement unable to provide sufficient information to identify certain parameters. In this case, CRB can provide a measure of how good a set of controlled input is. This is further used to provide optimality criterion in optimal design of experiment [KB19, ZWR94, ZWH96].

3.1 Cramér–Rao bound

Cramér–Rao bound (CRB) is the theoretical lower bound on the variance of an estimator. The method used to estimate an unknown parameter using measurment data is called estimator. The mentioned GN method and LM method are both examples of estimator. In this section, the following proposition shall be proved.

Proposition 3.1.1. *For the statistical model given in (2.6), the CRB is given by the following equation*

$$Cov[\beta^*] \geq (J^T V_e^{-1} J)^{-1} \tag{3.2}$$

where

$$V_e \coloneqq V_y + J_x V_x J_x^T, \quad J_x = \left.\frac{\partial f(x,\beta)}{\partial x}\right|_{x=x_t,\ \beta=\beta_t}, \quad J = \left.\frac{\partial f(x,\beta)}{\partial \beta}\right|_{x=x_t,\ \beta=\beta_t}, \tag{3.3}$$

The implication of Proposition 3.1.1 is to link CRB with the Jacobian matrix J. This further links identifiability with CRB by Theorem 3.0.1. More specifically,

$$\begin{aligned} unidentifiable\ model &\Rightarrow rank(J) < n \\ &\Rightarrow (J^T V_e^{-1} J)^{-1} undefined \end{aligned}$$

This means the closer the model to non-identifiability, the larger the CRB and variance on model parameter estimator.

There is another angle to view the relation between CRB and Jacobian. Suppose the parameter can be partitioned into two groups $\beta_1 \in R^{n_1}$ and $\beta_2 \in R^{n_2}$. Also, the weighted Jacobian is partitioned into $J_1 \in R^{m \times n_1}$ and $J_2 \in R^{m \times n_2}$.

$$V_e^{-\frac{1}{2}} J = [J_1 \quad J_2], \quad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}$$

Decompose J_2 into two components, one component $J_{2\|} \in Col(J_1)$ and another component $J_{2\perp} \in Col^{\perp}(J_1)$.

$$J_2 = J_{2\|} + J_{2\perp}, \quad \begin{cases} \exists A \in R^{n_1 \times n_2},\ s.t.\ J_{2\|} = J_1 A \\ J_1^T J_{2\perp} = 0 \end{cases}$$

Find CRB using block matrix inversion lemma and matrix inversion lemma (see following),

$$CRB(\beta^*) = (J^T V_e^{-1} J)^{-1} = \begin{bmatrix} J_1^T J_1 & J_1^T J_2 \\ J_2^T J_1 & J_2^T J_2 \end{bmatrix}^{-1}$$

Lemma 3.1.2. *block-wise matrix inversion*

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1} \end{bmatrix} \tag{3.4}$$

Lemma 3.1.3 (matrix inversion lemma)**.**

$$(A - BD^{-1}C)^{-1} = A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} \tag{3.5}$$

Then examine the (2,2) block of $Cov(\beta^*)$

$$\begin{aligned} CRB(\beta_2^*) &= [J_2^T J_2 - J_2^T J_1 (J_1^T J_1)^{-1} J_1^T J_2]^{-1} \\ &= [J_{2\perp}^T J_{2\perp} + J_{2\|}^T J_{2\|} - J_{2\|}^T J_1 (J_1^T J_1)^{-1} J_1^T J_1 A]^{-1} \\ &= (J_{2\perp}^T J_{2\perp})^{-1} \end{aligned} \tag{3.6}$$

The above equation shows that, if $\|J_{2\perp}\|$ is small, $CRB(\beta_2^*)$ would be large. Examine a special case where $n_2 = 1$, which means β_2 is a scalar. It can be seen that, if $J_2 \in Col(J_1)$, $J_{2\perp} = 0$ and the variance of β_2^* would be infinite. This shows that, if any specific column of J is linear dependent with other columns, the corresponding parameter would have infinite CRB and zero precision for estimation.

Before diving into the proof of Proposition 3.1.1, a few components have to be introduced. First, CRB is found to be the inverse of Fisher information matrix (FIM) [Cra46, Rao92]. Fisher information can be perceived as a quantification of information drawn from measurable random variable/vector, denoted Z, on some unknown deterministic parameter (or vector), denoted θ. With the definition of bias and FIM given below, CRB is given in Theorem 3.1.4

Definition 3.1.1. *Suppose $\theta^* = T(Z)$ is an estimator of unknown deterministic parameter θ, with its expectation being $E[\theta^*] = \phi(\theta)$. Then, its bias is defined as*

$$b(\theta) \coloneqq E[\theta^*] - \theta = \phi(\theta) - \theta \tag{3.7}$$

Definition 3.1.2. *Denote the conditional probability density function of Z given $\theta \in R^n$ to be $f_{Z|\theta}$. The (i,j) element of Fisher information matrix, $F(\theta) \in R^{n\times n}$, is*

$$[F(\theta)]_{i,j} \coloneqq E\left[\left(\frac{\partial log\, f_{Z|\theta}(z,\theta)}{\partial\theta_i}\right)\left(\frac{\partial log\, f_{Z|\theta}(z,\theta)}{\partial\theta_j}\right)\middle|\,\theta\right] \tag{3.8}$$

Theorem 3.1.4. *The covariance of estimator $\theta^* = T(Z)$ has a lower bound*

$$Cov[\theta^*] = E[(\theta^* - \phi(\theta))(\theta^* - \phi(\theta))^T] \geq (I + \nabla_\theta b)F(\theta)^{-1}(I + \nabla_\theta b)^T \tag{3.9}$$

The proof of theorem 3.1.4 is given in [GH90]. Note that the above theorem does not specifically specify the estimator to be unbiased. When the estimator is unbiased ($b(\theta) = 0$), the lower bound degenerate back to the inverse of FIM. For the special case of normal distribution, FIM is derived as followed [Kay93].

Lemma 3.1.5. *Suppose normal distribution for $Z \sim N(\mu(\theta), \Sigma(\theta))$, then the (i,j) entry of FIM is*

$$F_{ij} = \frac{\partial\mu^T}{\partial\theta_i}\Sigma^{-1}\frac{\partial\mu}{\partial\theta_j} + \frac{1}{2}tr\left(\Sigma^{-1}\frac{\partial\Sigma}{\partial\theta_i}\Sigma^{-1}\frac{\partial\Sigma}{\partial\theta_j}\right) \tag{3.10}$$

With all the above Theorem and Lemma, Proposition 3.1.1 can be proved.

Proof. **Proposition 3.1.1**

The random noise model in 2.6 can be rearranged as followed

$$\begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} e_y \\ e_x \end{bmatrix} + \begin{bmatrix} f(x_t, \beta_t) \\ x_t \end{bmatrix} \tag{3.11}$$

Define measurable random vector Z and unknown deterministic parameter θ as

$$Z \coloneqq \begin{bmatrix} y \\ x \end{bmatrix}, \quad \theta \coloneqq \begin{bmatrix} \beta_t \\ x_t \end{bmatrix} \tag{3.12}$$

The expectation and covariance matrix of Z is

$$\begin{aligned} \mu(\theta) = E[Z] &= \begin{bmatrix} f(x_t, \beta_t) \\ x_t \end{bmatrix} \\ \Sigma(\theta) = E[(Z-\mu)(Z-\mu)^T] &= \begin{bmatrix} V_y & 0 \\ 0 & V_x \end{bmatrix} \end{aligned} \tag{3.13}$$

Since e_y and e_x are both normal random variable, Lemma 3.1.5 can be used.

$$\frac{\partial \mu}{\partial \theta} = \begin{bmatrix} J & J_x \\ 0 & I \end{bmatrix}, \quad \frac{\partial \Sigma}{\partial \theta} = 0 \tag{3.14}$$

$$\Rightarrow F = \frac{\partial \mu^T}{\partial \theta} \Sigma^{-1} \frac{\partial \mu}{\partial \theta} = \begin{bmatrix} J^T V_y^{-1} J & J^T V_y^{-1} J_x \\ J_x^T V_y^{-1} J & J_x^T V_y^{-1} J_x + V_x^{-1} \end{bmatrix} \tag{3.15}$$

CRB of β_t can be obtained by examining the (1,1) entry of the block-wise matrix inversion of F

$$\begin{aligned} CRB(\beta_t) &= (F)_{11}^{-1} \\ &= [J^T V_y^{-1} J - J^T V_y^{-1} J_x (J_x^T V_y^{-1} J_x + V_x^{-1})^{-1} J_x^T V_y^{-1} J]^{-1} \\ &= \{J^T [V_y^{-1} - V_y^{-1} J_x (J_x^T V_y^{-1} J_x + V_x^{-1})^{-1} J_x^T V_y^{-1}] J\}^{-1} \\ &= [J^T (V_y + J_x V_x J_x^T)^{-1} J]^{-1} \end{aligned} \tag{3.16}$$

where the second equality comes from Lemma 3.1.2, and the third equality uses Lemma 3.1.3.

□

3.2 Expectation and Covariance matrix of Guass-Newton estimator

In this section, as an example, the expectation $E[\Delta\beta^*]$ and covariance matrix $Cov[\Delta\beta^*]$ of GN estimator deviation $\Delta\beta^*$ from true value will be derived. $\Delta\beta^*$ is defined as

$$\Delta\beta^* \coloneqq \beta^* - \beta_t \tag{3.17}$$

, where β^* represents the optimal parameter minimizing the objective function, and β_t represents model parameter estimated by the GN estimator. Due to the presence of random noise, $\Delta\beta^*$ is a random vector. The relationship between the random noise and $Cov(\Delta\beta^*)$ shall be presented.

residual error at true parameter

For the residual error definition given in (2.4) and the statistical model given in (2.6),

$$\begin{aligned} e(\beta_t) &= y - f(x, \beta_t) \\ &= f(x_t, \beta_t) + e_y - f(x_t + e_x, \beta_t) \\ &= (e_y - J_x e_x) \end{aligned} \tag{3.18}$$

, where the third approximation is made using (3.19) with the assumption that e_x is very small, and J_x is the derivative of the model function by x (3.20).

$$f(x_t + e_x, \beta_t) \cong f(x_t, \beta_t) + J_x e_x \tag{3.19}$$

$$J_x = J_x(x_t, \beta_t) = \left.\frac{\partial f(x, \beta)}{\partial x}\right|_{x=x_t,\ \beta=\beta_t} \tag{3.20}$$

In (3.18), it can be seen that, with the presence of random noise, even if β_t is known and used, the residual error is still not 0. Its expectation and covariance can be estimated as

followed

$$\begin{aligned} E[e(\beta_t)] &= 0 \\ Cov[e(\beta_t)] &= Cov[(e_y - J_x e_x)] \\ &= V_y + J_x V_x J_x^T \\ &= V_e \end{aligned} \tag{3.21}$$

, where $V_e \coloneqq V_y + J_x V_x J_x^T$.

optimal parameter deviation

When β reaches optimal value β^* minimizing the objective function $S(\beta)$, derivative of $S(\beta)$ (2.23) should be 0, then

$$\left.\frac{\partial S(\beta)}{\partial \beta}\right|_{\beta=\beta^*} = 0 \;\Rightarrow\; J^T(\beta^*) W e(\beta^*) = 0 \tag{3.22}$$

The residual error when reaching the optimal point $e(\beta^*)$ can be approximated by (2.20)

$$\begin{aligned} e(\beta^*) &\cong e(\beta_t) - J(\beta^*)(\beta^* - \beta_t) \\ &\cong (e_y - J_x e_x) - J\Delta\beta^* \end{aligned} \tag{3.23}$$

The above equation uses the assumption that $\Delta\beta^* << 1$. Also, denote $J = J(\beta^*)$. Then, combining (3.23) and (3.22)

$$J^T W J \Delta\beta^* = J^T W[(e_y - J_x e_x)] \tag{3.24}$$

If the J has full column rank, $Null(J) = 0$, and the solution is

$$\Delta\beta^* = (J^T W J)^{-1} J^T W[(e_y - J_x e_x)] \tag{3.25}$$

The method for picking out unidentifiable parameters to make J has full column rank, and

the physical interpretation of the singularity of Jacobian matrix will be presented in section 3.3

The expectation and covariance of $\Delta\beta^*$ is

$$\begin{aligned} E[\Delta\beta^*] &= 0 \\ Cov[\Delta\beta^*] &= (J^TWJ)^{-1}J^TWCov[(e_y - J_x e_x)]WJ(J^TWJ)^{-1} \\ &= (J^TWJ)^{-1}J^TWV_eWJ(J^TWJ)^{-1} \end{aligned} \tag{3.26}$$

,and since

$$X = \begin{bmatrix} J^TWV_eWJ & J^TWJ \\ J^TWJ & J^TV_e^{-1}J \end{bmatrix} = \begin{bmatrix} V_e^{\frac{1}{2}}WJ & V_e^{-\frac{1}{2}}J \end{bmatrix}^T \begin{bmatrix} V_e^{\frac{1}{2}}WJ & V_e^{-\frac{1}{2}}J \end{bmatrix} \geq 0 \tag{3.27}$$

$$J^TV_e^{-1}J > 0 \tag{3.28}$$

Then, the Schur complement $X/(J^TV_e^{-1}J) \geq 0$

$$\begin{aligned} & J^TWV_eWJ - (J^TWJ)J^TV_e^{-1}J(J^TWJ) \geq 0 \\ & \Rightarrow (J^TWJ)^{-1}J^TWV_eWJ(J^TWJ)^{-1} - J^TV_e^{-1}J \geq 0 \\ & \Rightarrow Cov[\Delta\beta^*] \geq (J^TV_e^{-1}J)^{-1} \end{aligned} \tag{3.29}$$

(3.29) means that the $diag(Cov[\Delta\beta^*])$ has a lower bound $diag((J^TV_e^{-1}J)^{-1})$, and the equality holds when $W = V_e^{-1}$. This lower bound is the same as CRB presented in Section 3.1. This suggests that, if V_e is known, $W = V_e^{-1}$ should be used to calculate objective function (2.5) to achieve theoretically most precise optimal parameter identification. This makes sense in that the measurement with less variance (more precise) should be considered with more weight.

3.3 determine unidentifiable parameters

From previous sections, it is shown that, when model is unidentifiable model, the CRB of model parameters will become undefined. The concept of identifiability is examined from the perspective of the whole model. However, it is also stated that even if the model is unidentifiable, some subset of model parameter can still be identified. In this section, the concept of identifiability will be examined from the perspective of individual parameter. More specifically, various methods for determining which parameter is unidentifiable will be given.

Note that all methods given in this section are a prior method, meaning no measurement have to be done beforehand to perform them (y and x is not observed). Since the true parameter β_t is not known apriori, the calculation of Jacobian and model function uses the initial guess or nominal value of β instead. For the controlled input x, the designed value is used instead. If good initial guess nor nominal value is available, an initial optimization can be done ahead to serve as starting seed. Also, the variance of measurement noise V_e is assumed known or measured by prior experiments.

3.3.1 identifiability by model function

Suppose that the model function is simple enough. Then, the simplest way to determine if a specific parameter is unidentifiable is by checking if that parameter can be changed without changing the model function prediction $f(x, \beta)$. This is a direct interpretation of the definition given in Definition 3.0.1. The non-identifiability of model parameters can be further categorized into two different situations. If the change of one single parameter results in no change of the model function prediction $f(x, \beta)$, then that parameter is called **redundant**. If the change of multiple parameters results in no change of the model function prediction $f(x, \beta)$, then those parameters are called **dependent** parameters, or **correlated** parameters. Usually, it is pretty easy to spot any redundant parameters by any of the

methods given in this section. In contrast, dependency among parameters is less apparent to find.

For example, for the following model function

$$f(a, b, d, x) = (a + b)x + 0 \times d \tag{3.30}$$

and the true value for the model parameter is $[a\ b\ d]_{true} = [2\ 0\ 0]$. It is clear that with the following model parameter deviation for (a, b), f is unchanged

$$\begin{matrix} \Delta a = -u \\ \Delta b = u \end{matrix} \Rightarrow f(a + \Delta a, b + \Delta b, d, x) = f(a, b, d, x) \tag{3.31}$$

Then, it means that (a, b) are dependent parameters. Also, since any change on d would result in no change on f, d is redundant.

3.3.2 identifiability by Jacobian column

Identifiability can also be determined by checking the linear dependency of the Jacobian columns. More, specifically, check if it is possible to find a vector $\Delta\beta_{null} \in R^n/\{0\}$, s.t.

$$J\Delta\beta_{null} = 0 \Rightarrow rank(J) < n$$

, which further implies the Jacobian does not have full column rank; thus, by Theorem 3.0.1, the model is unidentifiable. The parameter corresponding to the linear dependent columns is unidentifiable. If a minimal linear dependent set includes only one Jacobian column, then that column is 0 vector, and the parameter corresponding to the column is redundant. If there are multiple Jacobian columns inside one minimal linear dependent set, then the parameters corresponding to those columns are dependent. A less formal but helpful interpretation of the above statement is by checking if there is any subset of parameter (including single

parameter) deviation that would cancel each other out in terms of their respective effect on the model function $f(x, \beta)$.

3.3.3 identifiability by SVD

The methods mentioned above depends either on the simplicity of the model or on the non-intuitive inspection on linear dependency. Oftentimes, correlation between parameters is not apparent by performing previously mentioned methods. The method to be introduced below uses CRB as a way to quantify identifiability, and uses singular value decomposition (SVD) to determine identifiability of the parameters. Compared to previous methods, it is more straightforward and intuitive, and can provide physical meaning to identifiability in terms of the precision of the model parameter estimation. It is the aim of this method to provide a numerical way to examine and quantify identifiability as opposed to treating the identifiability as a binary property associated with model parameter. However, before diving into the numerical aspect, the analytical aspect will be discussed first. To fully explain identifiability by SVD, a few concepts have to be introduced first.

scaled parameter

The following method will use the formulation in Proposition 3.1.1 to calculate CRB, but with a slight twist. Define parameter scaling matrix D (a diagonal matrix), and scaled parameter β_s as

$$D = diag((J^T V_e^{-1} J))^{\frac{1}{2}} \in R^{n \times n}, \quad \beta_s = D\beta \tag{3.32}$$

One consequence of scaling the parameters is to change the dimension of the parameter from their respective unit (e.g. mm, rad, etc.) to be dimensionless (the unit of weighted error). Furthermore, as mentioned in Section 2.3.2, what is of interest is not the absolute value of the scaled parameter but its deviation. If i-th parameter changes by a small deviation $\delta\beta_i$, the scaled parameter deviation $\delta\beta_{s,i} = D\delta\beta_i$ represents the incurred change on the norm

of residual error. This allows the comparison among parameters in terms of how much each parameter deviation would effect the weighted residual error. Lastly, there is another insight into the scaling matrix D. Inspecting its i-th diagonal element D_i, its inverse is actually the square root of $CRB_{ind}(\beta_i)$, where subscript "ind" means β_i is estimated individually while assuming other parameter is known.

$$D_i = (J_i^T V_e^{-1} J_i)^{\frac{1}{2}} = \frac{1}{\sqrt{CRB_{ind}(\beta_i)}} \tag{3.33}$$

Therefore, the small deviation of scaled parameter $\delta\beta_{s,i}$ also means the ratio between the actual parameter deviation $\delta\beta_i$ and $\sqrt{CRB_{ind}(\beta_i)}$. This concept will become useful when inspecting $CRB(\beta_s)$, which will be introduced shortly.

scaled Jacobian

Define the scaled Jacobian as the derivative of weighted residual error (2.4) by scaled parameter,

$$J_s \coloneqq \frac{\partial(-V_e^{-\frac{1}{2}} e)}{\partial \beta_s} = V_e^{-\frac{1}{2}} J D^{-1} \tag{3.34}$$

, where the weighting matrix is selected to be $W = V_e^{-\frac{1}{2}} > 0$.

One assumption here is there are no redundant parameters in the model, or otherwise, they are taken out of the model beforehand. This is justified because the deviation of the redundant parameter would have no effect on the model function; thus, its disappearance would cause no consequences. The purpose of this assumption is to make J_s always defined. More specifically, since there is no redundant parameters, i-th column of Jacobian J_i would have non-zero norm, which means $D > 0$

$$\|J_i\| > 0 \Rightarrow D_i = diag_i(D) = \left\| V_e^{-\frac{1}{2}} J_i \right\| > 0$$

Following from the definition given in (3.34), it can be seen that the scaled Jacobian

matrix is dimensionless, which makes the singular value decomposition (SVD) of J_s physically meaningful instead of just being a numerical operation. The significance of this will be given shortly. First, the notation for SVD has to be introduced

$$J_s = USV^T \tag{3.35}$$

, where $U \in R^{m\times n}$ and $V \in R^{n\times n}$ have orthonormal columns. U and V are called left and right singular matrix respectively. $S = diag(\sigma_1, \sigma_2, ..., \sigma_n) \in R^{n\times n}$, with $\sigma_1 \geq \sigma_2 ... \geq \sigma_n$. Another worth-noting property is that the scaled Jacobian is the Jacobian matrix with normalized columns. By inspecting its matrix norm and invoking the equivalence of norm inequality, the following property for its singular value could be derived

$$\begin{aligned} & \|A\|_{max} \leq \|A\| \leq \sqrt{mn}\|A\|_{max} \\ \Rightarrow \quad & 1 \leq \sigma_1 = \|J_s\| \leq \sqrt{mn} \\ & 0 \leq \sigma_k \leq \sqrt{mn}, \quad k = 2 \sim n \end{aligned} \tag{3.36}$$

CRB of scaled parameter

Following similar derivation as in Proposition 3.1.1, the CRB for scaled parameter is

$$CRB(\beta_s) = (J_s^T J_s)^{-1} \tag{3.37}$$

The scaled Jacobian can be used to represent the CRB of model parameter β

$$CRB(\beta) = (J^T V_e^{-1} J)^{-1} = D^{-1}(J_s^T J_s)^{-1} D^{-1} = D^{-1} CRB(\beta_s) D^{-1} \tag{3.38}$$

The derivation of the above equation uses the definition of scaled Jacobian given in (3.34). Recall (3.33), the CRB of i-th scaled parameter can then be written as

$$CRB(\beta_{s,i}) = D_i \; CRB(\beta) \; D_i = \frac{CRB(\beta_i)}{CRB_{ind}(\beta_i)} \tag{3.39}$$

The above equation shows yet another benefit for using scaled parameter. The CRB of scaled parameter gauges how precise each parameter estimation is when that parameter is estimated collectively with other parameters versus individually. Without scaled parameter, it is not clear as to how large the variance or CRB of a specific parameter is too large. It can be shown that $CRB(\beta_{s,i}) \geq 1$ with block matrix inversion lemma presented in Lemma 3.1.2, which is trivial and not proven here. This shows the involvement of other parameter will always deteriorate the precision of a specific parameter estimation.

It is mentioned in Section 3.1 that the CRB of unidentifiable parameters would be infinitely large or, more formally, undefined. That was done using the normal component $J_{2\perp}$ of one Jacobian column to the others. Alternatively, SVD can be used to represent the CRB of scaled parameter

$$CRB(\beta_s) = (J^T V_e^{-1} J)^{-1} = V S^{-2} V^T = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} V_i V_i^T \tag{3.40}$$

, where V_i is the i-th column of right singular matrix V. From above equation, it can be seen that CRB can be decomposed into n different component with $1/\sigma_i^2$ being its weight. If, for example, $\sigma_n = 0$, then $V_n \in Null(J_s)$. The non-zero entries in V_n would be unidentifiable, and the component, $V_n V_n^T$ would have infinite (undefined) weighting, which means the CRB of the corresponding unidentifiable parameters would be infinite (undefined).

identifiability by SVD

From previous argument about $CRB(\beta_s)$, the closer σ_i is to 0, the larger the CRB of corresponding unidentifiable parameter is. However, CRB is not a good way to determine non-identifiability mainly due to its inversion calculation. When encountering strictly 0 singular values, calculation of CRB would involve inversion of a singular matrix. Therefore, $CRB(\beta_s)$ is introduced here for explanation purpose. It will later come into use when dealing with variance-bias trade off issues. As a numerically stable alternative, SVD can be used to find out the basis for $Null(J_s)$ and subsequently the dependent parameters. V_i which

corresponds to zero or close to zero σ_i is the basis for null space of J_s. Then, the dependent parameters are those parameters with non-zero entries in V_i.

The immediate question is perhaps that, to find the unidentifiable parameters, why not use SVD of J instead? Why bother using J_s? After all, using J to determine identifiability is what Theorem 3.0.1 originally suggested. Indeed, when it comes to robot FK calibration, that is what most of the literature suggested as well [LLW21, MD00, KJB16, ZRH90b]. However, [Sch93] suggested using scaled Jacobian with column-wise normalization by infinity norm of the Jacobian column. [DMB93] suggested normalization by 2-norm of the Jacobian column or 1 depending on whether $D_i = 0$. However, both state that the scaling is for comparing singular values when the magnitude among parameters is greatly different, without getting into too much detail as to why that is the case.

When the singular value is exactly at 0, it is completely equivalent to finding dependent parameters using SVD of J and J_s. Since $D > 0$ and $V_e > 0$, $Null(J_s) = \{0\}$ if an only if $Null(J) = \{0\}$. Also, since D is a diagonal matrix with positive diagonal values, the dependent parameter predicted by SVD of J and J_s will be the same. However, when the Jacobian does not have strictly 0 but close to 0 singular value. In this case, the last singular value σ_n and its corresponding right singular vector, V_n, are considered instead, since σ_n is the least of all. It can be seen that the σ_n of J_s would contribute most to the summation of the CRB of all scaled parameters

$$\sum_{i=1}^{n} CRB(\beta_{s,i}) = Tr(CRB(\beta_s)) = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} \tag{3.41}$$

Using SVD of J_s for the prediction of unidentifiable parameters thus has a physical meaning. Also, this determination is invariant with the selection of units for the parameters since they are scaled and become dimensionless. However, the same can not be said for the SVD of J or $V_e^{-\frac{1}{2}}J$. In fact, denote the singular value of weighted Jacobian $V_e^{-\frac{1}{2}}J$ to be γ_i, then

$$\sum_{i=1}^{n} CRB(\beta_i) = Tr((J^T V_e^{-1} J)^{-1}) = \sum_{i=1}^{n} \frac{1}{\gamma_i^2} \tag{3.42}$$

It can be seen that γ_n would contribute most to the summation of the CRB of all originally "unscaled" parameters. The units and magnitude among parameters may differ. This summation is a purely numerical operation and does not possess physical meaning. If the unit of one parameter changes (e.g. mm to m), the determination of unidentifiable parameters would have very different results.

In practice, when using finite precision machine to calculate singular values, it is unlikely that a singular value would have a strict 0 value, even if it is 0 analytically. Similarly, it is also necessary to have a method to determine the non-zero entries inside V_i. In this case, a threshold on singular value and right singular vector is performed. The u-th parameter, β_u, is considered unidentified if

$$\exists v \in \{1, 2, ..., n\},\ s.t.\ \sigma_v < \epsilon\sigma_1,\ and\ V_{uv}^2 > \epsilon max\{\max_i V_{iv}^2,\ \max_j V_{uj}^2\} \tag{3.43}$$

, where ϵ is the floating point error.

3.4 Variance-bias trade off

The performance of an estimator can be quantified by its variance and bias. Variance shows how precise the estimation is, whereas bias shows accuracy. An index combining both quantity is mean square error (MSE). It can be shown that, for the estimation of i-th parameter inside $\beta \in R^n$

$$MSE(\beta_i^*) = Var(\beta_i^*) + b_i^2 \tag{3.44}$$

, where β_i^* is the i-th element inside estimator $\beta^* = T(Z)$. This can be easily proven by the following

$$
\begin{aligned}
E[(\beta^* - \beta_t)(\beta^* - \beta_t)^T] &= E[(\beta^* - \overline{\beta^*} + \overline{\beta^*} - \beta_t)(\beta^* - \overline{\beta^*} + \overline{\beta^*} - \beta_t)^T] \\
&= Cov(\beta^*) + bb^T - bE[(\beta^* - \overline{\beta^*})^T] - E[(\beta^* - \overline{\beta^*})]b^T \qquad (3.45) \\
&= Cov(\beta^*) + bb^T
\end{aligned}
$$

Then, by examining the i-th diagonal element, (3.44) is shown. Equation (3.45) will come in handy in later proof.

It is mentioned that unidentified parameter will have large CRB. In this case, even if the estimator is unbiased, MSE will still be very large. Suppose there is a good guess β_g on the unidentified parameter, and the difference between the guess and the true value β_t is guaranteed to be smaller than a bias bound B. Then, a fixed-parameter estimator which estimates the unidentified parameter to be $\beta^* = \beta_g$ regardless of the measurement (observation) $Z = (y, x)$ will have an MSE smaller than B^2. This is one way to implement variance-bias trade off. By deliberately increasing the bias and decreasing the variance of the estimator, MSE is lowered and a better estimator is obtained.

3.4.1 sufficient condition for lowering MSE for all parameters

For an unbiased estimator, after fixing a subset of parameters, the variance of those fixed parameter is lowered to 0, and their bias would most likely increase, unless a perfect guess of the fixed parameter is made, which is unlikely. For "unfixed" parameter, it can be shown that the variance would also decrease and the bias would increase from 0. Nevertheless, it is very inconvenient to calculate and track down the difference of bias and variance for both fixed and unfixed parameter before determining if a certain parameter should be fixed. The following theorem solves this problem by giving the sufficient condition, on which the MSE of all parameter (fixed and unfixed alike) will be smaller after fixing a subset of parameter.

Theorem 3.4.1. *Suppose the model parameter $\beta \in R^n$ can be partitioned into two groups*

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}, \quad \beta_1 \in R^{n_1},\ \beta_2 \in R^{n_2}$$

For β_2, there exits a guess $\beta_{2g} \in R^{n_2}$ and a bias bound $B_2 \in R^{n_2}_{>0}$, such that

$$|(\beta_{2g} - \beta_{2t})_i| < (B_2)_i, \quad i \in \{1, 2, ..., n_2\}$$

Denote an unbiased estimator $\beta^* = T(Z)$ to estimate β, where Z is one observation. Suppose β^* is in multivariate normal distribution and has a covariance matrix

$$Cov(\beta^*) = \begin{bmatrix} Cov(\beta_1^*) & Cov(\beta_1^*, \beta_2^*) \\ Cov(\beta_2^*, \beta_1^*) & Cov(\beta_2^*) \end{bmatrix}$$

Denote another estimator, $\beta' = T(Z|\ \beta_2 = \beta_{2g})$, which estimates β under the condition that $\beta_2 = \beta_{2g}$

When the following condition is satisfied

$$\sigma_{min}(Cov(\beta_2^*)) \geq \sum_{i=1}^{n_2} (B_2)_i^2 \tag{3.46}$$

Then, $\forall i \in \{1, 2, ..., n\}$

$$MSE((\beta^*)_i) \geq MSE((\beta')_i)$$

The proof of Theorem 3.4.1 requires the following Lemma [Mis64]

Lemma 3.4.2. *A multivariate normal random vector $Z = [X; Y]$ has covariance matrix*

$$Cov(Z) = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$$

Then, the conditional variance and expectation is

$$E[X|Y] = E[X] + BC^{-1}(Y - E[Y])$$
$$Cov(X|Y) = A - BC^{-1}B^T$$

Proof. of Theorem 3.4.1

Denote the bias to be b_1^* and b_2^* for β^*, and $b_1^{'}$ and $b_2^{'}$ for $\beta^{'}$

$$b^* = E[\beta^*] - \beta_t = \begin{bmatrix} b_1^* \\ b_2^* \end{bmatrix}, \quad b^{'} = E[\beta^{'}] - \beta_t = \begin{bmatrix} b_1^{'} \\ b_2^{'} \end{bmatrix} \tag{3.47}$$

Since β^* is unbiased, $b_1^* = 0$ and $b_2^* = 0$. Also, since $\beta_2^{'} = \beta_{2g}$ is given, $b_2^{'} = \beta_{2g} - \beta_{2t}$ and $Cov(\beta_2^{'}) = 0$.

Suppose condition (3.46) is met, then

$$\begin{aligned}
&\sigma_{min}(Cov(\beta_2^*)) \geq \sum_{i=1}^{n_2} (B_2)_i^2 \\
&\Rightarrow x^T Cov(\beta_2^*) x \geq \sum_{i=1}^{n_2} (B_2)_i^2 \geq x^T b_2^{'} b_2^{'T} x, \quad \forall x \in R^{n_2}/\{0\} \\
&\Rightarrow Cov(\beta_2^*) \geq b_2^{'} b_2^{'T}
\end{aligned}$$

For fixed parameter β_2, from the above inequality, the following can be derived

$$Cov(\beta_2^*) \geq b_2^{'} b_2^{'T} \Rightarrow Cov(\beta_2^*) + b_2^* b_2^{*T} \geq Cov(\beta_2^{'}) + b_2^{'} b_2^{'T}$$

Examine the diagonal entry of the above inequality, it can be concluded that, for fixed parameter, the mean squared error is smaller

$$MSE((\beta^*)_i) > MSE((\beta')_i), \quad \forall i \in fixed\ parameter$$

For unfixed parameter β_1, apply Lemma 3.4.2 to β^*,

$$E[\beta_1^{'}] = E[\beta_1^*] + K(\beta_{2g} - E[\beta_2^*])$$
$$Cov(\beta_1^{'}) = Cov(\beta_1^*) - KCov(\beta_2^*)K^T$$

,where $K = Cov(\beta_1^*, \beta_2^*)Cov(\beta_2^*)^{-1}$. Since $E[\beta_1^*] = \beta_{1t}$ and $E[\beta_2^*] = \beta_{2t}$

$$b_1^{'} = Kb_2^{'}$$

$$\Rightarrow Cov(\beta_1^*) + b_1^* b_1^{*T} - [Cov(\beta_1^{'}) + b_1^{'} b_1^{'T}] = K[Cov(\beta_2^*) - b_2^{'} b_2^{'T}]K^T \geq 0$$
$$\Rightarrow Cov(\beta_1^*) + b_1^* b_1^{*T} \geq Cov(\beta_1^{'}) + b_1^{'} b_1^{'T}$$

Similarly, by examining the diagonal entry for the above inequality and combining the case for fixed parameters, the following can be obtained

$$MSE((\beta^*)_i) > MSE((\beta')_i), \quad \forall i \in \{1, 2, ..., n\}$$

□

3.4.2 further interpretation

The condition (3.46) is given in unscaled parameter. To avoid numerical issue from badly-scaled parameter, previously mentioned in Section 3.3.3, scaled parameter should be used instead.

$$\sigma_{min}(Cov(\beta_{2s}^*)) \geq \sum_{i=1}^{n_2} (B_{2s})_i^2 \tag{3.48}$$

,where β_{2s}^* and B_{2s} is the scaled β_2^* and B_2. Furthermore, since $CRB(\beta_{2s}^*) \leq Cov(\beta_{2s}^*)$, a stricter condition for lowering MSE can be written using CRB as the covariance matrix

$$\sigma_{min}(CRB(\beta_{2s}^*)) \geq \sum_{i=1}^{n_2}(B_{2s})_i^2 \tag{3.49}$$

This condition is still not useful enough since it involves calculation of a possibly singular matrix $CRB(\beta_{2s}^*)$. From (3.6), the above condition can be further simplified as

$$\sigma_{min}(CRB(\beta_{2s}^*)) = \frac{1}{\sigma_{max}(J_{2s\perp}^T J_{2s\perp})} \geq \sum_{i=1}^{n_2}(B_{2s})_i^2$$

$$\Rightarrow \left\| J_{2s\perp}^T J_{2s\perp} \right\| \sum_{i=1}^{n_2}(B_{2s})_i^2 \leq 1 \tag{3.50}$$

, where $J_{2s\perp}$ is the component of J_{2s} normal to $Col(J_{1s})$. The simplified condition can be checked without worries of singularity issue. As can be seen from the above inequality, to lower MSE, it is better to fix parameter that has smaller bias bound B_{2s} (more accurate guess) and small $J_{2s\perp}$ (unidentifiable parameters and poor estimator precision). If there is no good guess for a specific parameter, then it means $B_s \to \infty$. An extreme cases arises when there is no good guess $B_s \to \infty$ while the parameter is unidentifiable $\left\| J_{2s\perp}^T J_{2s\perp} \right\| = 0$. In this case, either fixing it or leaving it is acceptable.

CHAPTER 4

Optimal Design of Experiment

As mentioned in previous chapter, CRB is a quantification of precision of an estimator. It is also closely related to identifiability of parameters and linear dependency of the Jacobian matrix. Since Jacobian matrix is a function of both the parameters β and the controlled input x, one way to decrease the CRB is to select suitable x to disrupt the linear dependency of the Jacobian matrix. Thus, optimal design of experiment is needed. The goal of that is to maximize the Fisher information matrix (in terms of some concept of matrix norm) or minimize the CRB over the controlled input x. In FK calibration, this is equivalent to find the optimal poses of the robot for the CMM to measure.

The mathematical formulation of the optimal design problem is as followed

$$\begin{aligned} &\min_{x} \Phi(F(x)) \\ &s.t.\ C(x) \leq 0 \end{aligned} \tag{4.1}$$

, where $C(x)$ is some sort of constraint function. Since Fisher information matrix $F(x) = J^T V_e^{-1} J$ is a matrix, scalarization function $\Phi(\cdot) : R^{n \times n} \rightarrow R$ has to be used to quantify $F(x)$. Of course, there are many scalarization functions to choose from as optimality criteria. Famous optimality criteria include

- A-optimality: $\Phi(F) = Tr(F^{-1})$
- D-optimality: $\Phi(F) = -log(det(F))$
- E-optimality: $\Phi(F) = -\sigma_{min}(F)$

The calculation of the objective and constraints $F(x)$ and $C(x)$ often need prior knowledge on β since $J = J(\beta, x)$. The nominal value of β has to be used instead, or some sort of simple experiment has to be done ahead to infer the rough value of the parameters. This problem will occur later in this literature as well and shall be discussed in more details in Ch. 7

The FIM can be decomposed into the addition of FIM of individual poses. To see this, it is necessary to extend the definition of input x and output y defined in (2.2). Suppose there are N data points (x_i, y_i) being collected, and $x_i \in R^{p'}$, $y_i \in R^{m'}$. The input vector and output vector then become

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \in R^p, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in R^m \tag{4.2}$$

Obviously, $p = Np'$ and $m = Nm'$.

Recall from Chapter 3.1 that the FIM can be formulated as

$$F(x) = J^T V_e^{-1} J \tag{4.3}$$

Partition the weighted Jacobian as

$$V_e^{-\frac{1}{2}} J = \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_N \end{bmatrix}, \quad J_i \in R^{m' \times n} \tag{4.4}$$

With the restructured definition of x and y, $V_e = V_y + J_x V_x J_x^T$ becomes a diagonal block matrix of the following form.

$$V_e(x) = \begin{bmatrix} V_1(x_1) & 0 & \cdots & 0 \\ 0 & V_2(x_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & V_N(x_N) \end{bmatrix} \tag{4.5}$$

This implies that $J_i = J_i(x_i)$. Then the FIM becomes

$$F(x) = \sum_{i=1}^{N} F_i, \quad F_i = F_i(x_i) = J_i^T J_i \tag{4.6}$$

The above equation shows that the FIM $F(x)$ can be decomposed into $F_i(x_i)$, which is a function of individual input x_i. This formulation allows convenient mathematical manipulation for the following.

4.1 optimal design formulation

In general, there are two groups of methods to further formulate the optimization problem.

- point by point search method: find N input x_i from all feasible poses $X_F = \{x : C(x) \geq 0\}$ to minimize $F(x)$ in a point-by-point fashion
- search from pool method: find N input x_i from a finite set (the pool) $X_P \subset X_F$, to minimize $F(x)$

4.1.1 search from pool method

As the name suggests, this method pick the inputs from a pool of finite candidates. This setup can turn the optimal design problem into convex optimization. Suppose that the finite set X_P (candidate pool) is given, and number of element inside X_P is $M = |X_P| >> N$. The individual FIM F_i corresponding to the i-th candidate input x_i ($i \in [M]$) can be calculated. Then, the formulation described in (4.1) can be rearranged as

$$\begin{aligned} &\min_{\rho_i} \ \Phi(\sum_{i=1}^{M} \rho_i F_i) \\ &s.t. \ \sum_{i=1}^{M} \rho_i = N, \quad \rho_i \in \{0,1\} \end{aligned} \tag{4.7}$$

, where the input weight $\rho_i \in \{0,1\}$ is a binary number and indicate whether the i-th candidate input x_i is selected or not. However this problem is NP hard. This problem can be relaxed as followed

$$\begin{aligned} &\min_{\rho_i} \ \Phi(\sum_{i=1}^{M} \rho_i F_i) \\ &s.t. \ \sum_{i=1}^{M} \rho_i = N, \quad \rho_i \geq 0 \end{aligned} \tag{4.8}$$

Note that if the scalarization function $\Phi(\cdot)$ is convex, then this relaxed problem becomes a convex programming problem. In fact, all the optimality criteria mentioned earlier in this chapter is convex. Furthermore, A-optimality and E-optimality can be further transformed into Semidefinite programming (SDP) problem [BV04]. It is also reported that D-optimality can be transformed into Second Order Cone programming (SOCP) problem [Sag09]. Both SDP and SOCP can be solved efficiently by interior point method (IPM) [Jor06]. After the relaxed optimal design problem is solved, and the relaxed optimal input weight $\hat{\rho}$ is found, the sub-optimal solution to the original problem (4.7) can be found by thresholding.

Despite the many benefits presented above, the downside of this problem is that it requires pre-calculation of large amount of Jacobian matrix J_i. Ideally, the candidate pool $x_i \in X_P$ has to have a wide yet dense distribution. This could pose quite a problem for time-pressing online calibration purposes. It can be argued that the calculation of J_i could be done prior and stored for later use. However, since J_i is a function of not only x but also β as well, if it is wished to use a different set of parameter for calculation of J_i, the previous calculation has to be done again. For instance, suppose that originally nominal value of parameter β_0

is used for optimal design of experiment. Then, experiment is done, and measurement of y and x is obtained. Later, estimation of parameter is performed, and it is found out that the estimated parameter β^* has noticeable difference from β_0. It is then wished to use β^* instead of β_0 for future optimal experiment design. This happens when there is a constant need for estimation of a subset of changing parameters while other parameters are estimated before. For example, a surgical robot with exchangeable tool constantly changes its tool. Each time tool exchange happens, FK calibration has to be performed to estimate the FK parameter representing the changed mechanism, while the remaining mechanism is still best described by the previously calibrated value. This will be discussed with more details in Chapter 7.

4.1.2 point by point search method

Due to the downside mentioned earlier, an alternative method to solve optimal design problem is proposed here. This method is called point-by-point search method. Just as the name suggested, this method iteratively solves for the next optimal data point, $x_{k+1} \in R^{p'}$, with k being the number of current data points, until $k = N$ is reached. Suppose that k input, x_i, is already given, so $F_i, i \in [k]$ can be calculated. Then, the next optimal input can be found by solving

$$\begin{aligned} &\min_{x_{k+1}} \Phi(\sum_{i=1}^{k} F_i + F_{k+1}(x_{k+1})) \\ &s.t.\ C(x_{k+1}) \geq 0 \end{aligned} \tag{4.9}$$

Note that, unlike search from pool method, there is no way to guarantee the above problem to be convex optimization problem. However, it can still be solved by IPM, and the calculation required is not so big if N is small.

This method should not start from $k = 1$ since $F(x)$ would probably not have full rank, and the optimality criterion is likely to be undefined or 0, which will cause further issue when performing IPM to solve the problem. Instead, a minimum amount, denoted N_{min}, of

data points should be selected to ensure that $F(x)$ has full rank. This selection of initial seed input $x_{min} \in R^{p'N_{min}}$ could be done either manually or randomly. Also, Suppose that $rank(J_i) \geq r$. To ensure full rank for F, the necessary condition is

$$\begin{aligned} rank(F) = n &\Rightarrow rank(J) = n \\ &\Rightarrow N_{min} r \geq n \end{aligned} \tag{4.10}$$

Note that this is only necessary condition. Whether this is sufficient condition depends on x_{min}. If it is not sufficient, additional input has to be included into x_{min}. However, this necessary condition is generally sufficient as well if the selected inputs are not close to each other.

CHAPTER 5

Forward Kinematic calibration

The accuracy of a serial manipulator is defined by the difference of actual and estimated end-effector pose, where pose means both position and orientation. The estimated pose is calculated by the FK of the robot. FK model describes how the pose of the end-effector would move with respect to the joint variable and applied force/torque. Like any model, the FK model can be parameterized. Unfortunately, due to misalignment error during assembly, manufacturing error of linkage, and/or simply poor knowledge on the model, the true FK parameters often deviates from nominal value (designed value). This would compromise the accuracy of FK. To improve accuracy, FK calibration is performed. Its purpose is to identify (estimate) the true value of the FK parameters.

FK calibration setup

The setup for the FK calibration is illustrated in Figure 5.1. The whole setup consists of a robot and a Coordinate Measuring Machine (CMM). Both are fixed on the ground. CMM can measure the whole or partial pose (i.e. position and orientation) of a special tool mounted onto the flange. There are four major frames for this setup. The robot base frame $\{b\}$ is attached to the base of the robot. Flange frame $\{f\}$ is attached to the robot flange. Tool frame $\{t\}$ is attached to the end of the tool. The origin of $\{t\}$ is called the tool center point (TCP). Note that the tool frame is considered the same as the end-effector frame in this literature. CMM frame $\{m\}$ is the reference frame for all pose measurements by CMM.

There are three measurable quantities. The tool pose ${}^{m}_{t}T$ can be fully or partially mea-

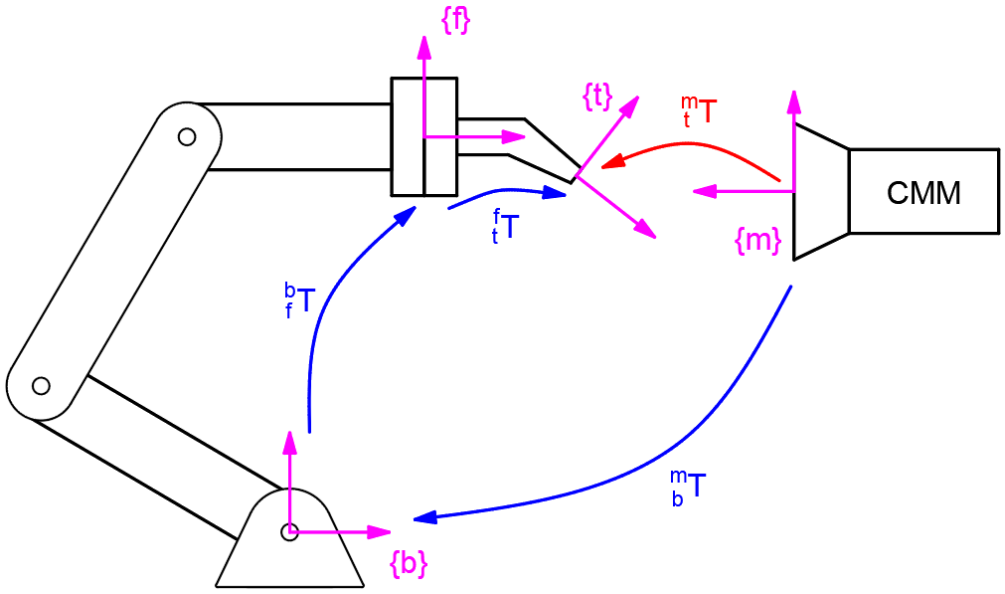


Figure 5.1: frames and transformations used in FK calibration

sured by CMM. An example of "partial" measurement of ${}^{m}_{t}T$ is that CMM may only measure position of the tooltip and/or direction of the tool centerline. Joint angle/translation (collectively called joint variable) q can be measured by encoder. Joint torque/force τ can be measured by motor current and/or force sensor.

There are four major transformations in this setup. Denote the transformation from frame $\{i\}$ to frame $\{j\}$ to be ${}^{i}_{j}T$

- ${}^{m}_{b}T$: It is parameterized by Coordinate Transform (CT) parameter $t_{mb} \in R^6$, ${}^{m}_{b}T = {}^{m}_{b}T(t_{mb})$. t_{mb} is an unknown parameter and needs to be inferred during FK calibration.
- ${}^{b}_{f}T$: It is parameterized by DH parameter η and is a function of q.

$$ {}^{b}_{f}T = {}^{b}_{f}T(q, \eta) \tag{5.1} $$

This function is known as FK of the robot. η represents the geometry of the robot linkage and how they are connected. The above assumes rigid joint model. In case

where joint compliance has perceivable effect, flexible joint model is assumed. The above becomes

$$ {}^{b}_{f}T = {}^{b}_{f}T(q, \eta, \tau, C) \tag{5.2} $$

, where C is joint compliance coefficient. It describes how much the joint q would move when sustaining joint force/torque τ. The true value of both parameters, η and C, are unknown and needs to be inferred during FK calibration.

- ${}^{f}_{t}T$: It is called tool transformation matrix in this literature. The dimension of the tool is assumed known so the transformation ${}^{f}_{t}T$ is known as well.

- ${}^{m}_{t}T$: This transformation can be (partially) measured by CMM (red arrow in Figure 5.1), or estimated using FK (blue arrow in Figure 5.1).

$$ {}^{m}_{t}T = {}^{m}_{b}T(t_{mb})\, {}^{b}_{f}T(q, \eta, \tau, C)\, {}^{f}_{t}T \tag{5.3} $$

In summary, FK calibration is to identify the model parameter ($\beta = [\eta; C; t_{mb}]$) of the model function $\hat{y} = f(x, \beta)$ given in (5.3), which estimates the output ($y = {}^{m}_{t}T$) given an independent input ($x = [q; \tau]$). The true intention for FK calibration is to obtain an accurate FK function, ${}^{b}_{f}T(q, \eta, \tau, C)$, for further application of the robot. Therefore, finding η and C is of interest. As for CT parameter, it depends on whether further application of the robot still needs CMM. In most cases, CMM is not used after FK calibration is done; thus, the CT parameter is not important. However, some application, like the surgical robot mentioned in Chapter 7, would still require the use of CMM (an Optical Coherence Tomography, OCT) after FK calibration. In this case, CT parameters are of interest.

FK calibration procedure

See Figure 5.2. FK calibration proposed in this literature includes 5 steps: modeling FK, experiment design, identifiability analysis, measuring using coordinate measuring machine (CMM), and finally identify the parameters.

1. Modeling: model the output y (i.e. ${}^{m}_{b}T$) as a function of independent input x (i.e. q and τ) and model parameter β (i.e. η, C, and t_{mb}). Or more simply, derive the analytical expression of equation (5.3).

2. Experiment design: design the model input x_{des} for the measuring step. It needs to satisfy certain constraints (e.g. end-effector has to be within CMM working range), while trying to increase FIM (improving precision of parameter estimation).

3. Identifiability analysis: check if there are any unidentifiable parameters either by examining the model function (Section 3.3.1) or by SVD (Section 3.3.3). Note that identifiability analysis by SVD needs to use the designed input x_{des} from the last step. If there are unidentifiable parameters, try to eliminate them by remodeling or re-design the experiment. If both are unsuitable, use Theorem 3.4.1 to check if it is suitable to fix them as constant in the estimator.

4. Measuring: Command robot to the designed input x_{des}. Use CMM to measure y, and simultaneously record x

5. Identifying: Use the measurement y and x obtained from last step as the input to certain estimator algorithm. The estimator would then identify the model parameters β^*.

In this chapter, all of the above steps will be addressed.

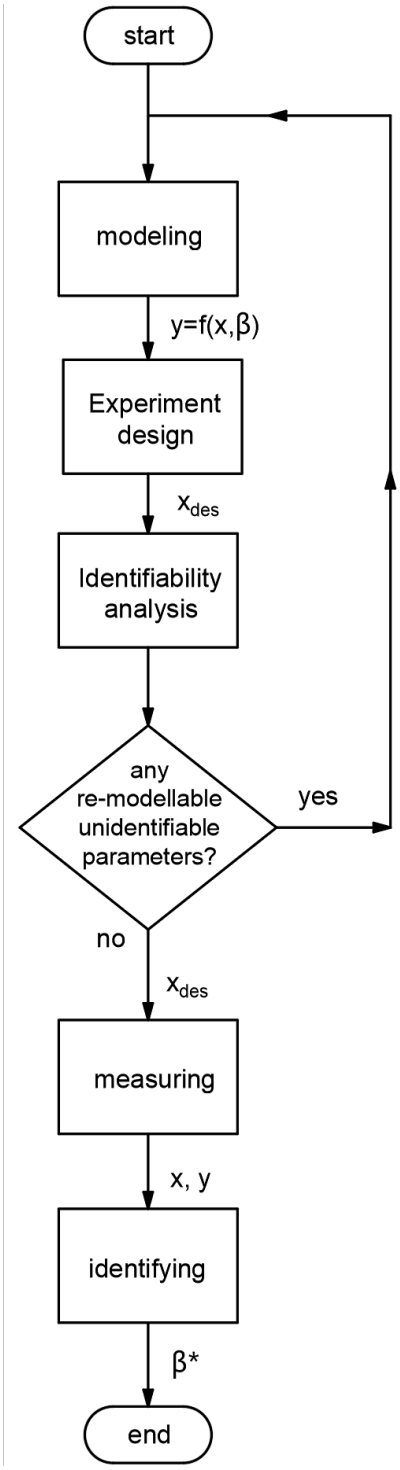


Figure 5.2: calibration flowchart

5.1 Forward Kinematics model

The purpose of modeling is to derive and/or guess the analytical expression of a model function which best predicts the behavior of ${}^{m}_{t}T$ w.r.t. input (q and τ) and model parameter (η, t_{mb} and C).

$$\begin{aligned} {}^{m}_{t}\hat{T} &= {}^{m}_{t}\hat{T}(q, \tau, \eta, C, t_{mb}) \\ &= {}^{m}_{b}\hat{T}(t_{mb})\, {}^{b}_{f}\hat{T}(q, \eta, \tau, C)\, {}^{f}_{t}T \end{aligned} \tag{5.4}$$

, where the hat overhead sign, $\hat{A}$, over some physical quantity A means that A is estimated by the FK model. As demonstrated above, ${}^{m}_{t}\hat{T}$ is the multiplication of three transformations. The model for ${}^{m}_{b}\hat{T}(t_{mb})$ and ${}^{b}_{f}\hat{T}(q, \eta, \tau, C)$ will be discussed in this section.

For explanation purposes in the later text, ${}^{m}_{t}T$ and ${}^{b}_{t}T$ are partitioned as the following.

$${}^{m}_{t}T = \begin{bmatrix} R_m & p_m \\ 0 & 1 \end{bmatrix}, \quad R_m = \begin{bmatrix} x_m & y_m & z_m \end{bmatrix} \tag{5.5}$$

$${}^{b}_{t}T = \begin{bmatrix} R_b & p_b \\ 0 & 1 \end{bmatrix}, \quad R_b = \begin{bmatrix} x_b & y_b & z_b \end{bmatrix} \tag{5.6}$$

, where $p_m, x_m, y_m, z_m \in R^3$ and $p_b, x_b, y_b, z_b \in R^3$. p_m and p_b mean the TCP position relative to CMM frame $\{m\}$ and base frame $\{b\}$. z_m and z_b mean z direction of the tool frame $\{t\}$, relative to CMM frame $\{m\}$ and base frame $\{b\}$. z-direction of the tool frame $\{t\}$ is usually selected to be the tool centerline as well. The meaning of the x_m, x_b, y_m, y_b are similar to z_m and z_b, which altogether form rotation matrix R_m and R_b.

5.1.1 Denavit–Hartenberg convention

The most widely used convention for representing the kinematics of a serial manipulator is Denavit–Hartenberg convention [DH55]. See Figure 5.3, suppose that there are h joints.

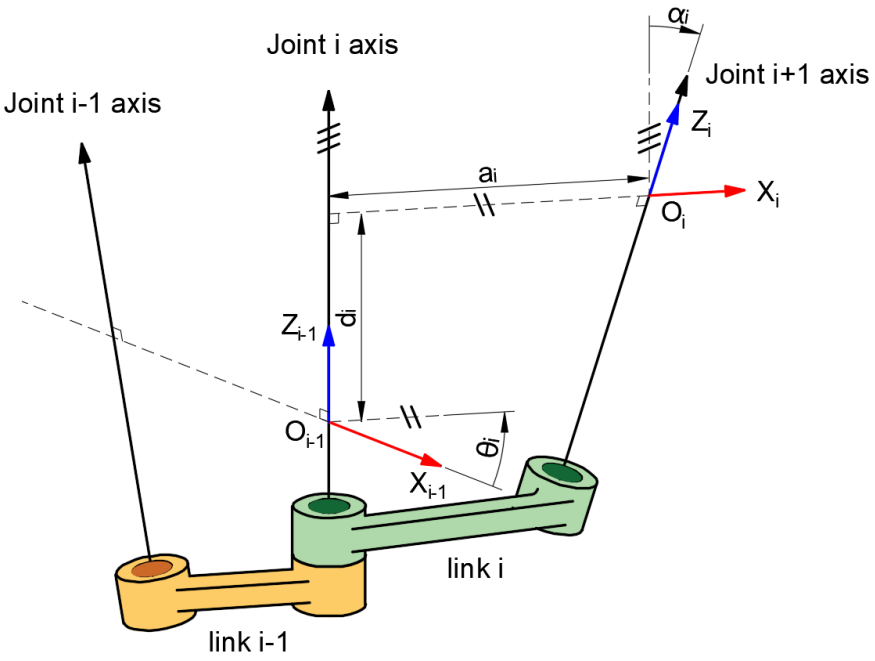


Figure 5.3: DH convention

This convention attaches a series of coordinate frames $\{i\}$, called link frames hereafter, to the joint of the robot. Then, a group of 4 parameters $(d_i, \theta_i, a_i, \alpha_i)$, called DH parameters η_i, are defined to represent the transformation between successive link frames ${}_{i+1}^{i}T$. The flange frame of the robot can then be described by DH parameters η_i and joint variables q_i from each joint. For now, rigid joint model is assumed, meaning applied joint torque/force would not affect joint variable. The effect of joint compliance or flexible joint model will be discussed in Section 5.1.3.

Define link frame

The guideline for defining link frame $\{i\}$ axis is as followed

- Z_i is the same as the axis of joint $i+1$ for revolute joint. For prismatic joint, the direction of Z_i is the same as the direction of joint $i+1$ translation, but its location is arbitrary.

- X_i is defined as the common normal of Z_i and Z_{i-1}, and its direction is $X_i = Z_{i-1} \times Z_i$. In case $Z_i \parallel Z_{i-1}$, the direction of X_i points away from Z_{i-1}, and its location can be chosen arbitrarily to be on any common normal.
- Y_i is chosen so that it completes the right-hand coordinate.

A few things to note for the first and last link of the Kinematic chain

- Link 0 is base, and link h is robot flange.
- Frame $\{0\}$ can be defined using the above guideline by arbitrarily place X_0 to intersect with Z_0 perpendicularly. Base frame $\{b\}$ is defined to be frame $\{0\}$.
- $\{f\}$ is defined using flange geometry. For example, flange geometry is usually cylindrical, and Z_f can be chosen to be the axis of the cylindrical flange.
- After defining $\{f\}$, frame $\{h\}$ is defined by the above guideline using Z_f as joint axis $h+1$.

With the above definition, frame $\{i\}$ is attached (fixed) to the link i on joint axis $i+1$. This is true for all frames and links $\forall i \in [h]$.

Define DH parameter

The guideline for defining DH parameter for joint i, $\eta_i = [d_i; \theta_i; a_i; \alpha_i] \in R^4$, is

- d_i: called link offset, the distance between origin O_i and O_{i-1} along Z_{i-1}
- θ_i: called joint angle, the angle between X_i and X_{i-1} about Z_{i-1}
- a_i: called link length, the length of the common normal between Z_i and Z_{i-1} along X_i
- α_i: called joint twist, the angle between between Z_i and Z_{i-1} about X_i

DH matrix representation

the transformation from $\{i-1\}$ to $\{i\}$ is

$$
\begin{aligned}
{}^{i-1}_{i}T(q_i,\eta_i) &= Q_i(q_i)Tran_z(d_i)Rot_z(\theta_i)Tran_x(a_i)Rot_x(\alpha_i) \\
&= Q_i(q_i)\begin{bmatrix} cos(\theta_i) & -sin(\theta_i)cos(\alpha_i) & sin(\theta_i)sin(\alpha_i) & a_icos(\theta_i) \\ sin(\theta_i) & cos(\theta_i)cos(\alpha_i) & -cos(\theta_i)sin(\alpha_i) & a_isin(\theta_i) \\ 0 & sin(\alpha_i) & cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{5.7}
$$

, where $Tran_z(d_i)$, $Tran_x(a_i)$, $Rot_z(\theta_i)$, $Rot_x(\alpha_i) \in R^{4\times4}$ are defined as

$$
Tran_z(d_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Tran_x(a_i) = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.8}
$$

$$
Rot_z(\theta_i) = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & 0 \\ sin(\theta_i) & cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Rot_x(\alpha_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_i) & -sin(\alpha_i) & 0 \\ 0 & sin(\alpha_i) & cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.9}
$$

, and $Q_i(q_i)$ is defined as

$$
Q_i(q_i) = \begin{cases} Tran_z(q_i), & joint\ i\ is\ prismatic\ joint \\ Rot_z(q_i) \ , & joint\ i\ is\ revolute\ joint \end{cases} \tag{5.10}
$$

For purpose of clarity, this kind of notation deliberately separates the joint variable q_i from the FK parameter θ_i and d_i. This concept comes from [ZRH90a, ZWR93]. In this way,

when referring to θ_i and d_i, it means model parameter, which remains constant during FK calibration. When referring to joint variable q_i, it means model input, which is not a constant and may vary with different data points.

Finally, the transformation from base frame $\{b\}$ to flange frame $\{f\}$ is

$$ {}^{b}_{f}\hat{T}(q,\eta) = {}^{0}_{1}T\,{}^{1}_{2}T \ldots {}^{h-1}_{h}T \tag{5.11} $$

, where $q = [q_1; q_2; ...; q_h] \in R^h$ and $\eta = [\eta_1; \eta_2; ...; \eta_h] \in R^{4h}$.

In keeping with the 4-parameter style of parameterization for each link, one major shortcoming of the DH convention is that it implicitly assumes $\{h\}$ to be the same as $\{f\}$, and uses $\{h\}$ to represent $\{f\}$. Although it is true that their Z axis is the same by definition, $Z_h \coloneqq Z_f$, their X axis is not necessarily the same, $X_h \neq X_f$. More specifically, X_f is defined based on flange geometry, and X_h is defined by the common normal of $Z_h = Z_f$ and Z_{h-1}. Their definitions are not the same. An example of this is shown in Figure 5.4. This makes the FK model represented by the DH convention incomplete in the sense that it fails to capture all the behavior of the studied system. This calls for a remodeled version of the DH parameters below.

Extended DH parameter

The original DH convention with only 4 parameters for each joint is not enough to represent the arbitrary placement of the last link frame $\{h\}$, which should take 6 parameters to express. This issue is also addressed in [ZRH90a, ZWR93], which suggested using a 6-parameter model to represent the last link transformation. Here, with the addition of 2 more parameters β_h and b_h, the Extended DH (EDH) parameter, $\eta_h = [d_h; \theta_h; a_h; \alpha_h; b_h; \beta_h] \in R^6$, is then proposed to replace the DH parameter on the last link transformation. The EDH representation below is designed to resemble the original DH transformation for program writing purposes.

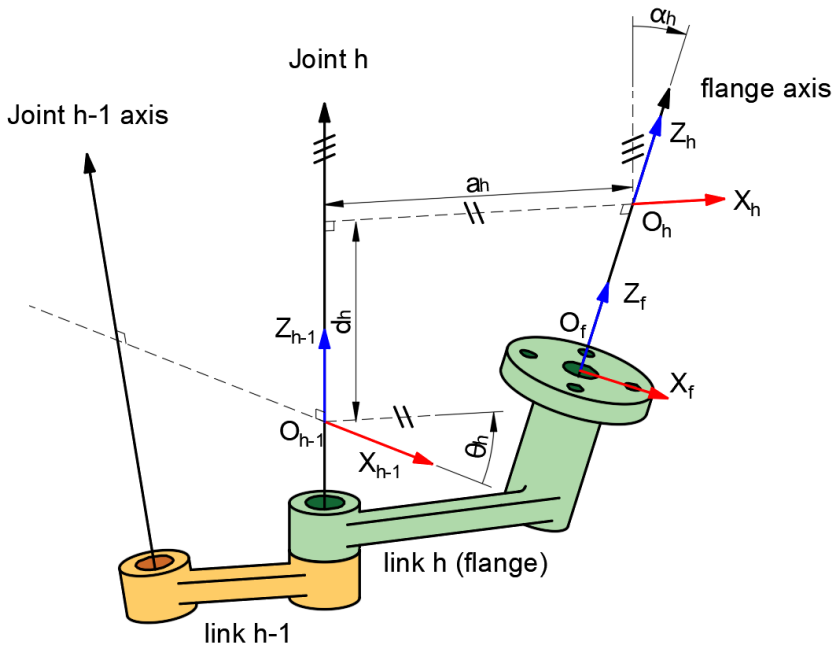


Figure 5.4: An example of $X_f \neq X_h$

$$
\begin{aligned}
&{}^{h-1}_{h}T(q_h, \eta_h) \\
&= Q_h(q_h)Tran_z(d_h)Rot_z(\theta_h)Tran_x(a_h)Rot_x(\alpha_h)Tran_y(b_h)Rot_y(\beta_h) \\
&= Q_h(q_h) \begin{bmatrix} c_{bh}c_{th} - s_{ah}s_{bh}s_{th} & -s_{th}c_{ah} & s_{bh}c_{th} + c_{bh}s_{th}s_{ah} & a_hc_{th} - bc_{ah}s_{th} \\ c_{bh}s_{th} + s_{ah}s_{bh}c_{th} & c_{th}c_{ah} & s_{bh}s_{th} - c_{bh}c_{th}s_{ah} & a_hs_{th} + b_hc_{ah}c_{th} \\ -c_{ah}s_{bh} & s_{ah} & c_{ah}c_{bh} & d_h + b_hs_{ah} \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{5.12}
$$

, where the symbol c and s means $cos(\cdot)$ and $sin(\cdot)$, and the subscript ah, bh, th means $\alpha_h, \beta_h, \theta_h$. $Tran_y$ and Rot_y is defined as

$$
Tran_z(b_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad Rot_y(\beta_i) = \begin{bmatrix} cos(\beta_i) & 0 & sin(\beta_i) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\beta_i) & 0 & cos(\beta_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.13}
$$

5.1.2 Coordinate Transform

To parameterize ${}^{m}_{b}T$, define a function $H_S(\cdot) : R^6 \to R^{4\times 4}$

$$H_S(t) = \begin{bmatrix} & & t_1 \\ & R_S(t_4, t_5, t_6) & t_2 \\ & & t_3 \\ 0 & & 1 \end{bmatrix} \tag{5.14}$$

, where S is a sequence representing the Euler angle system used for rotation matrix parameterization R_S (e.g. $S = ZYX$). As demonstrated above, the first three elements in t represent the translation vector, and the last three elements represent the Euler angle representation for the rotation matrix.

Partition the CT parameters $t_{mb} \in R^6$ into two groups. One group p_{mb} is for representing the translation vector. The other is Euler angle r_{mb} for representing the rotation matrix

$$t_{mb} = \begin{bmatrix} p_{mb} \\ r_{mb} \end{bmatrix}, \quad p_{mb} \in R^3,\ r_{mb} \in R^3 \tag{5.15}$$

Then the ${}^{m}_{b}T$ is represented using $H_S(\cdot)$

$${}^{m}_{b}\hat{T} = H_S(t_{mb}) \tag{5.16}$$

$$\Rightarrow \begin{cases} {}^{m}_{b}R = R_S(r_{mb}) \\ {}^{m}p_b = p_{mb} \end{cases}$$

Ideally, the sequence S is chosen so that no gimbal lock happens. Gimbal lock will be detected after performing identifiability analysis. When it happens, two of the Euler angles will be dependent on each other, rendering those two parameters unidentifiable. Normally,

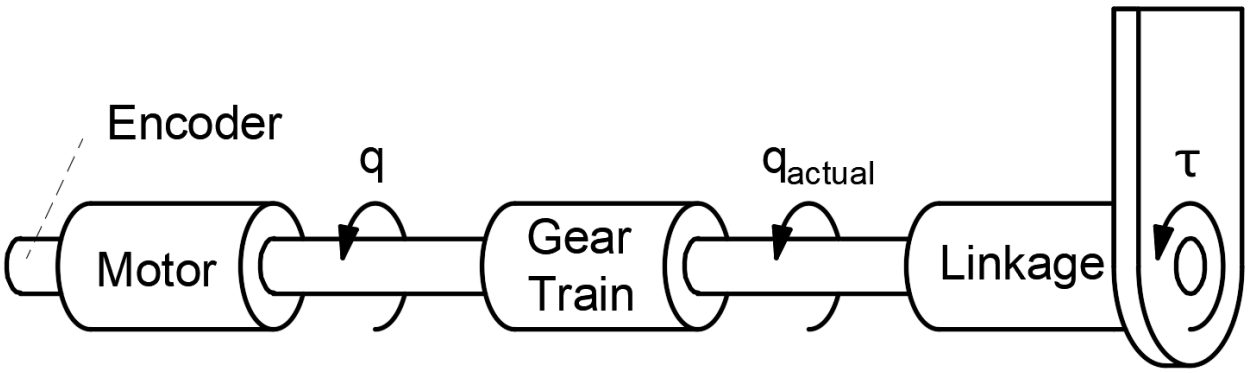


Figure 5.5: power transmission setup for revolute joint

an arbitrary choice of S will do. In the unfortunate case where gimbal lock happens to occur, re-parameterization is needed and different S should be used.

5.1.3 Joint compliance model

Joint compliance, or joint stiffness, refers to the phenomenon where joint undergoes elastic deformation when applied outside force/torque. Many literatures [LHS08, WZF09, KB19, ODB12, GYN00] reported that joint compliance has noticeable effect on the robot TCP position and that incorporating the joint compliance model would increase the accuracy of FK estimation.

See Figure 5.5. For the revolute joint, it has been suggested [ODB12, AWR07] that the main source of compliance comes from the torsion-spring-like behaviour of the gear train between the motor and driven linkage. Since most industrial robot designs mount the encoder on the motor side, the actual joint value q_{actual} on the linkage side is not directly measured. Therefore, the torsional deformation Δq that happens at the gear train is not visible to the motor-side encoder. Although there may be other sources of compliance (e.g. linkage deformation), the gear train deformation has the dominant effect. For the prismatic joint, it most likely involves some screw design to convert the rotary movement of the motor into linear translation. Also, there must be a gear train to amplify motor torque as well. It

is unknown how the applied force would deform the combined mechanism of the gear train and screw. For simplicity, rigid joint is assumed for the prismatic joint.

To model the joint compliance deflection for revolute joint, the torsion spring model is used. The joint deflection is assumed to be linear to applied force/torque.

$$\Delta q = q_{actual} - q = C\tau \tag{5.17}$$

, where q is the joint value measured by encoder (already scaled by gear ratio). This is the most used model due to its simplicity. To apply this, modify the $Q_i(q_i)$ matrix in (5.10) to include i-th joint compliance coefficient C_i and joint torque/force measurement τ_i

$$Q_i(q_i, \tau_i, C_i) = \begin{cases} Tran_z(q_i), & joint\ i\ is\ prismatic\ joint \\ Rot_z(q_i + C_i\tau_i) \quad , & joint\ i\ is\ revolute\ joint \end{cases} \tag{5.18}$$

5.2 Experiment design

The experiment design is to design a set of joint variables and joint torque (if calibration of the flexible joint model is desired), which will be used for the measuring step. To improve the precision of parameter estimation, a good general rule is to spread the range of the designed joint variable and joint torque as much as permitted. If it is desired, an optimal experiment design could be performed. Recall (4.1)

$$\begin{aligned} &\min_{q,\tau} \Phi(F(q,\tau)) \\ &s.t.\ C(q,\tau) \leq 0 \end{aligned} \tag{5.19}$$

Note that optimal design over joint torque/force is generally not possible since it requires applying precise force/torque to the joints. Some sort of equipment is needed to achieve this requirement. However, this has been done before [KB19]. In most cases, however, payloads with different weights are mounted on the tool to increase the range for joint torque/force.

There are some constraints on the designed joint variable and joint torque $C(q, \tau) \leq 0$. Some of the common constraints include but are not limited to the following.

- joint limit constraint: joint variable and joint torque have to be within a certain range for safe operation

$$\begin{aligned} q_{lb} \leq q \leq q_{ub} \\ \tau_{lb} \leq \tau \leq \tau_{ub} \end{aligned} \tag{5.20}$$

- robot working range constraint: The FK model might behave differently on different points and/or on different orientations; thus it is desired to calibrate the robot within its normal working range (defined relative to $\{b\}$).

$$C_w({}^b_tT) \leq 0 \tag{5.21}$$

- CMM sensing constraint: TCP and tool orientation has to be within a certain set (defined relative to $\{m\}$), where CMM can obtain adequate measurements.

$$C_s({}^m_tT) \leq 0 \tag{5.22}$$

- collision constraint: the robot must not collide with itself.

Using the FK model function described in Section 5.1, all constraint functions above (except joint torque constraint and collision constraint) would eventually turn into constraints on q. Depending on the type of CMM and robot used, one constraint might be more strict than the other. To simplify the problem, it is possible to satisfy multiple constraints by just meeting the most demanding one.

Checking the constraints $C(q, \tau) \leq 0$ and calculating objective function $F(q)$ with a given q would require the knowledge of η, C, and t_{mb}. However, their true value is unknown at

this stage, and initial guesses have to be used instead. For DH parameters, a good initial guess η_0 can be obtained from the designed nominal value or from the manufacturer manual. For the joint compliance coefficient, the rigid joint model is assumed, and its initial guess is $C_0 = 0$. For CT parameters, there is usually no good initial guess, and a small experiment has to be performed to find them.

The task of finding t_{mb} or ${}^m_bT$ alone is called a registration problem. Registering the robot with CMM means locating the robot base frame $\{b\}$ relative to the CMM frame $\{m\}$. A generic method to do this is by point cloud registration. First, command the robot to N different joint variables and record their value $q_i \in R^h, i \in [N]$ using encoder, and measure their TCP position $p_{mi} \in R^3$ using CMM. Calculate TCP position relative to $\{b\}$, $\hat{p}_{bi} = \hat{p}_{bi}(q_i, \eta_0) \in R^3$, using FK model function. Now, there are N pairs of $(p_{m,i}, \hat{p}_{bi})$. Then, ${}^m_bT$ can be found using point cloud registration (PCR) algorithm. The details of the PCR algorithm used in this literature will be discussed in Section 5.5.2.

Special treatment has to be done for the collision constraint. It does not translate easily to the joint variable constraint since it depends on the geometry of each link. A posterior check is much easier instead of formulating collision constraints into the experiment design problem. More specifically, after the experiment design is done and a group of designed joint variables q_{des} is obtained. Perform a posterior check algorithm, which attaches a convex shape to each link and checks if collision happens for each designed joint variable. If true, pick that joint variable out. One of the algorithms for collision detection of convex shapes can be found in [GJK88].

5.3 Identifiability Analysis

Non-identifiability falls into 2 categories [WHR21]. One is structurally unidentifiable, which means the model is locally unidentifiable at a given model parameter β on every input x. Structural non-identifiability can be eliminated by remodeling the function. The other

type of Non-identifiability is practical non-identifiability. The definition of practical non-identifiability, in this literature, is defined as the model being locally unidentifiable at a given model parameter β and at a given input x. Practical non-identifiability should not happen with good experiment design.

Detect non-identifiability

Identifiability analysis by model function (Section 3.3.1) can be performed once the modeling step is done without even needing to do experiment design. It can be used to detect structural unidentifiable parameters. Since most industrial robots mostly adopt a similar design, their FK model function and a few common unidentifiable cases are already well-studied. These well-studied cases can be spotted right away if given the DH parameters. Some of them are explained later in this section.

Identifiability analysis by SVD (introduced in Section 3.3.3) can be performed only after obtaining a set of designed joint variables q_{des} from the experiment design step. However, it can detect both structural and practical non-identifiability. In addition, it can conveniently spot the non-identifiability of a robot with a novel geometry or a newly designed robot. This is especially useful for the surgical robot presented in 7, which has a geometry unlike that of a common industrial robot.

Counter non-identifiability

In general, practical non-identifiability is caused by negligence in the design of the experiment. It may happen to every parameter. To counter it, spread the designed joint variables and the end-effector position/pose in Cartesian space as much as possible. Then, the practical identifiability of geometric parameters, meaning DH parameters and CT parameters, should improve. For the joint compliance parameter, spread the joint torque/force as much as possible. This is most easily done by mounting payloads with various weights to the robot.

However, since most industrial robots have a joint axis 1 parallel to the direction of gravity, there will be no substantial joint deflection on joint 1; therefore, C_1 becomes redundant.

For structural non-identifiability, it can be decided depending on the type of unidentifiable parameters whether to remodel or not. If the unidentifiable parameter is the CT parameter (most likely r_{mb}), a different model (e.g. choose a different sequence S for $H_S(\cdot)$) can be used instead. If the unidentifiable parameter is DH parameter, care is needed before proceeding to change the model formulation. Since DH convention is used in almost every robot design and its accompanying control program, it could cause quite an inconvenience to employ another model. Therefore, in this literature, when encountering non-identifiability with the DH parameter, remodeling is never adopted.

If both remodeling and redesign experiment is not acceptable, fixing the remaining unidentifiable parameter during identifying step is another option. To decide whether to fix a parameter, Use Theorem 3.4.1 or the more useful version given in (3.50). It is advised as discussed in Section 3.4.2 to fix unidentifiable or close to unidentifiable parameters since the precision for the identification of those parameters is bad. It would be better to trade the large variance for a small bias by fixing those parameters to lower MSE. If there is an unidentifiable parameter with a bad initial guess, neither fixing it nor leaving it is more beneficial to the other. In this case, both options are acceptable.

After all the above countermeasures are executed, if there are still some unidentifiable parameters left, use LM method given in 2.3.2 to identify them in the identifying step. It will always give a parameter estimation although it might have a large MSE. In comparison, GN method would have numerical issues when trying to calculate the matrix inversion in (2.22).

A few common examples of structural non-identifiability are given below. The underlying cause for most of them arises from the arbitrary selection in the DH convention.

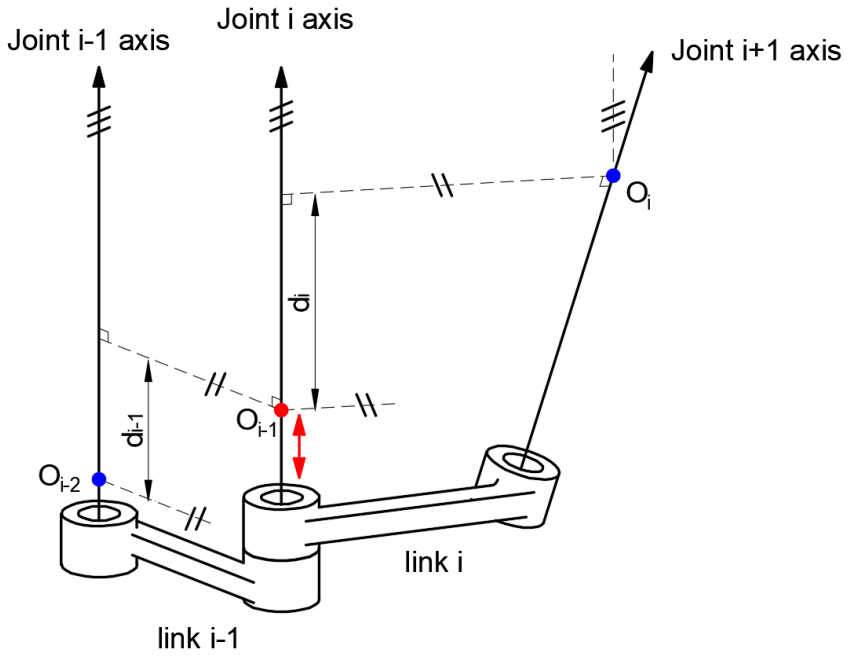


Figure 5.6: illustration of parallel axis non-identifiability

5.3.1 Parallel Axis

If there are consecutive joints i and $i-1$ that are parallel (meaning $\alpha_{i-1} = 0$) or near parallel to each other, then d_i and d_{i-1} are dependent. In [Hay83], Hayati tried to tackle this by remodeling FK and introducing a new Hayati parameter. This issue comes from the non-unique common normal of two parallel lines. See Figure 5.6, the origin of each frame is marked with a colored dot. Since joint axis i is parallel to joint axis $i-1$, frame $\{i-1\}$ can be selected to be any point on joint axis i (red dot can be moved along joint axis i) without changing the previous and next frame (blue dots are fixed). An infinite amount of (d_i, d_{i-1}) pair corresponds to the same FK model, which makes them dependent and thus unidentifiable.

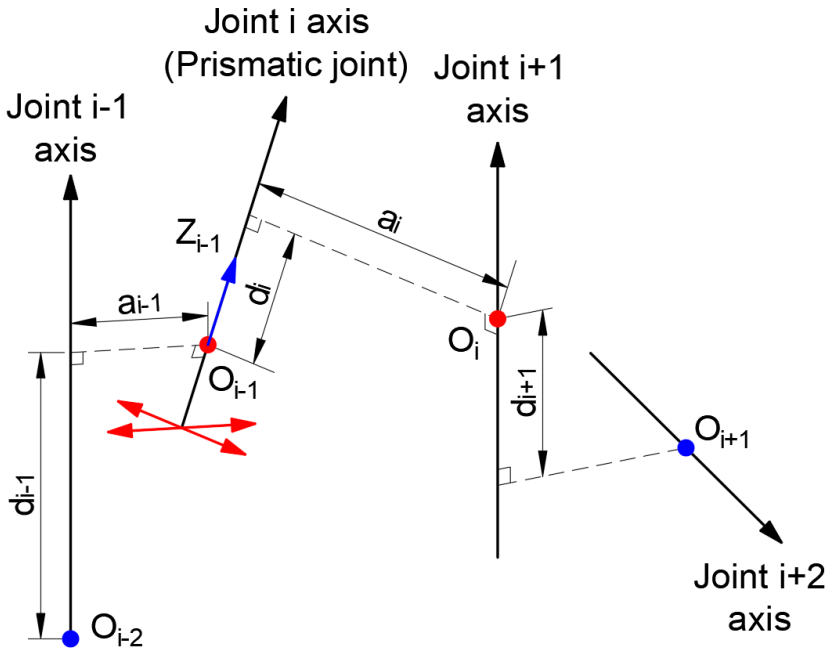


Figure 5.7: illustration of prismatic joint non-identifiability

5.3.2 Prismatic Joint

If joint i is prismatic joint, then $(a_i, a_{i-1}, d_{i+1}, d_i, d_{i-1})$ are dependent on each other. This non-identifiability arises from the arbitrary selection for the location of the prismatic joint axis or Z_{i-1}. As long as the direction of Z_{i-1} is the same as joint axis i, it can be attached to any location on link i. See Figure 5.7. The location of Z_{i-1} and O_{i-1} can move perpendicular to its direction (red arrow), which causes O_i to move along the joint axis $i+1$. However, frame $\{i-2\}$ and $\{i+1\}$ and other frames remain unchanged. Since the axes of all joints remain in the same direction, link twist α and joint angle θ of all joints remain unchanged. Only the said five unidentifiable parameters are changed. This means there is an infinite amount of parameter set $(a_i, a_{i-1}, d_{i+1}, d_i, d_{i-1})$ which represent the same model, making them dependent on each other. Since there are 2 dof to decide where to place Z_i, to eliminate the non-identifiability, 2 of the 5 parameters $(a_i, a_{i-1}, d_{i+1}, d_i, d_{i-1})$ have to be fixed.

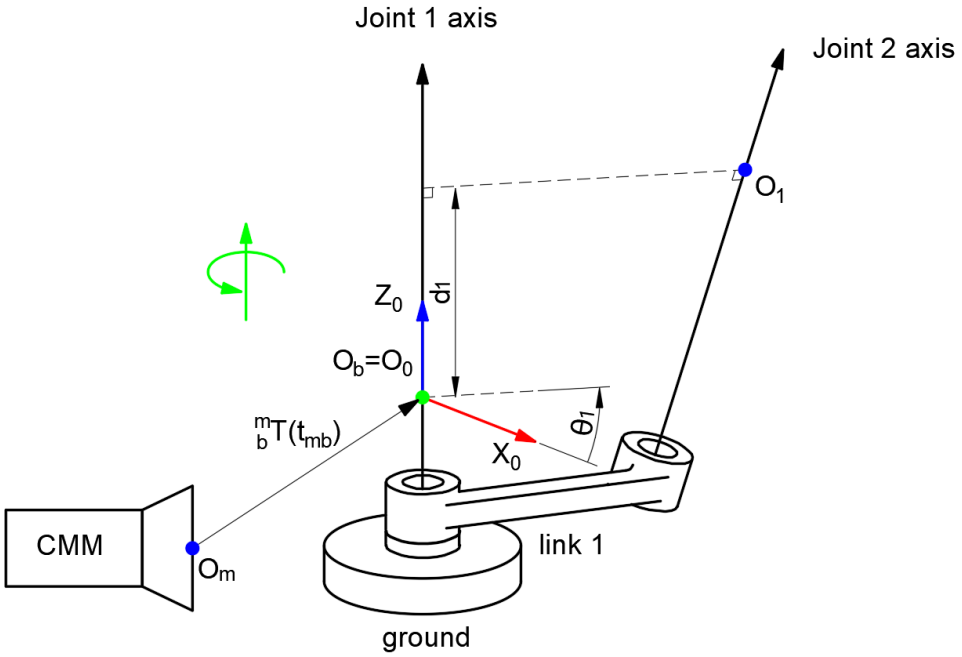


Figure 5.8: illustration of base frame non-identifiability

5.3.3 base frame

This will happen during every robot FK calibration. (t_{mb}, d_1, θ_1) are dependent on each other. This non-identifiability comes from the arbitrary selection of base frame $\{b\}$ along joint axis 1. See Figure 5.8. Base frame $\{b\}$ can be chosen to be on any point along joint axis 1, and it can also rotate around the joint axis 1 (green dot, X_0, and Z_0 can move/rotate in the direction of green arrow). Following similar arguments as in previous sections, this means there is an infinite amount of parameter set (t_{mb}, d_1, θ_1) which represents the same model; thus they are dependent. Since there are 2 dof to decide where to place $\{b\}$, to eliminate the non-identifiability, 2 of the 8 parameters (t_{mb}, d_1, θ_1) have to be fixed. Usually, (d_1, θ_1) are selected, since there is generally no good initial guess on t_{mb}.

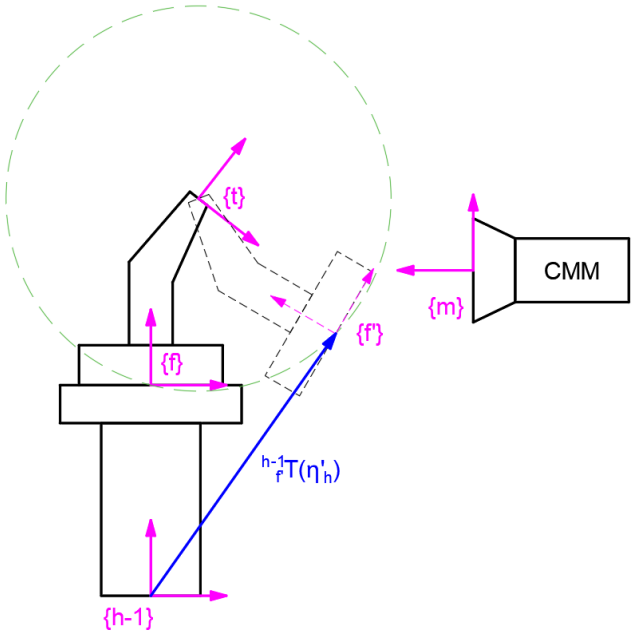


Figure 5.9: illustration of flange frame non-identifiability

5.3.4 flange frame

This happens when CMM only has partial pose measurement. The last link parameter η_h would become dependent. For example, suppose that CMM can only measure the position of the end-effector. See Figure 5.9. Since there is no way to know the orientation of the tool just by the CMM measurement alone, it is undetermined as to where the flange frame is relative to the second last link frame $\{h-1\}$. Any point on the green sphere is a potential flange frame $\{f'\}$, and any last link DH parameter η'_h corresponding to a given potential $\{f'\}$ describes the same model behavior (same TCP). Therefore, η_h is dependent on each other.

One method to eliminate such non-identifiability is to design the tool so that multiple different tool transformation matrices ${}^f_tT$ are used. This would impose more constraint and implicitly determine the flange frame location relative to $\{h-1\}$. For example, in [NB13], 8 different ${}^f_tT$ are used to obtain full parameter calibration on the last link. The minimum

Table 5.1: minimum amount of ${}^{f}_{t}T$ to achieve full last link calibration N_T

CMM type	EDH	DH
position only	3	1
position + partial orientation	1	0
full measurement	0	0

Table 5.2: amount of unidentifiable last link parameter U

CMM type	EDH	DH
position only	3	1
position + partial orientation	1	0
full measurement	0	0

amount of ${}^{f}_{t}T$ that should be used to obtain full last link calibration is denoted N_T and given in Table 5.1. If only one ${}^{f}_{t}T$ is used, the amount of unidentifiable last link parameters, U, is given in Table 5.2. Both N_T and U depend on the last link parameterization (EDH or DH) and CMM type (position only, position + partial orientation, or full pose measurement). A simple way to derive U is. For example, if EDH is used (6 unknown η_h), and CMM only measures position (3 constraints), that means there are 3 degrees of freedom left to place the location of $\{f\}$, thus $U = 6 - 3 = 3$.

5.4 Measurement

During the measuring step of FK calibration, a Coordinate Measuring Machine (CMM) is used. The robot will be commanded to N points using the designed joint variable q_{des}, and

Table 5.3: total r_t and effective r dimension of CMM measurement

CMM type	r_t	r	measurement	description
P	3	3	p_m	position only
PZ	6	5	p_m, z_m	position + partial orientation
T	12	6	${}^m_tT$	full measurement

payloads would be mounted onto the robot to apply joint torque. For the i-th point, measure the joint variable $q_i \in R^h$ using the encoder, measure the joint torque/force $\tau \in R^h$ using motor current, and measure the end-effector pose ${}^m_tT_i \in R^{4\times4}$ using CMM .

Depending on the CMM used, it may measure the full pose of ${}^m_tT$ (T-type), position + partial orientation (p_m, z_m) (PZ-type), or simply position p_m (P-type), where (p_m, z_m) are the TCP position and tool centerline (See (5.5)). Note that the total dimension of the measurement (r_t) is not necessarily the same as the effective dimension (r) due to additional constraints given on the norm of orientation measurement. For example, the total dimension of PZ-type measurement (p_m, z_m) is $r_t = 3 + 3 = 6$. However, since there is an additional constraint $\|z_m\| = 1$, the effective dimension is $r = 3 + 2 = 5$. The total and effective dimensions of all CMM types are listed in Table 5.3.

The combined effective measurement r of all N data points has to be larger than the parameter number n; otherwise, practical non-identifiability would occur. This was mentioned in Section 4.1.2 and (4.10), the minimum amount of data points required is N_{min}

$$N_{min} r \geq n$$

The following text in this section will be introducing 2 types of CMM, which will be used in Chapter 6 and 7. The introduction will be focusing on their relevance with FK calibration, rather than its working principle.

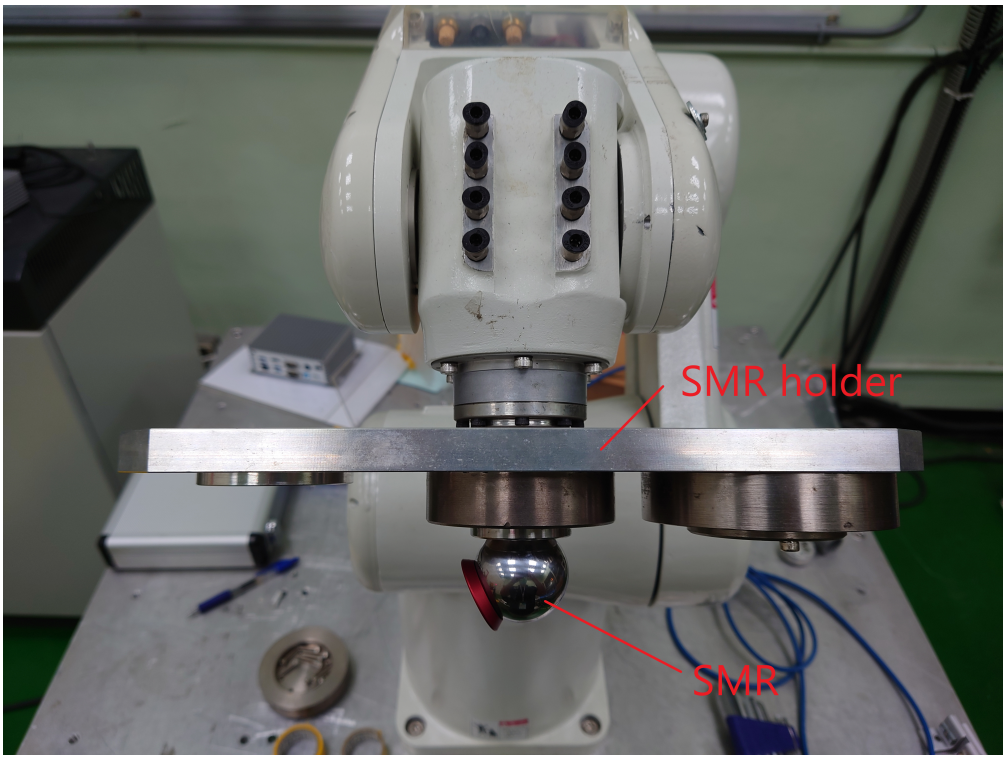


Figure 5.10: SMR and SMR holder

Laser Tracker

See Figure 5.10. A laser tracker can track the position of a spherical mounted reflector (SMR), which can be mounted on the robot flange with an SMR holder. The SMR holder is precisely machined and its dimensions are assumed known. This implies that the transformation matrix ${}^{f}_{t}T$ is known, where the tool frame $\{t\}$ is the frame attached to the position of the SMR. A laser tracker is, in general, a P-type CMM.

OCT

Optical Coherence Tomography device (OCT) is another type of CMM. It can measure a 3D point cloud of the object placed in its working range. This is referred to as a Volume scan or V scan. Each volume scan can be divided into multiple slices of cross-sections which

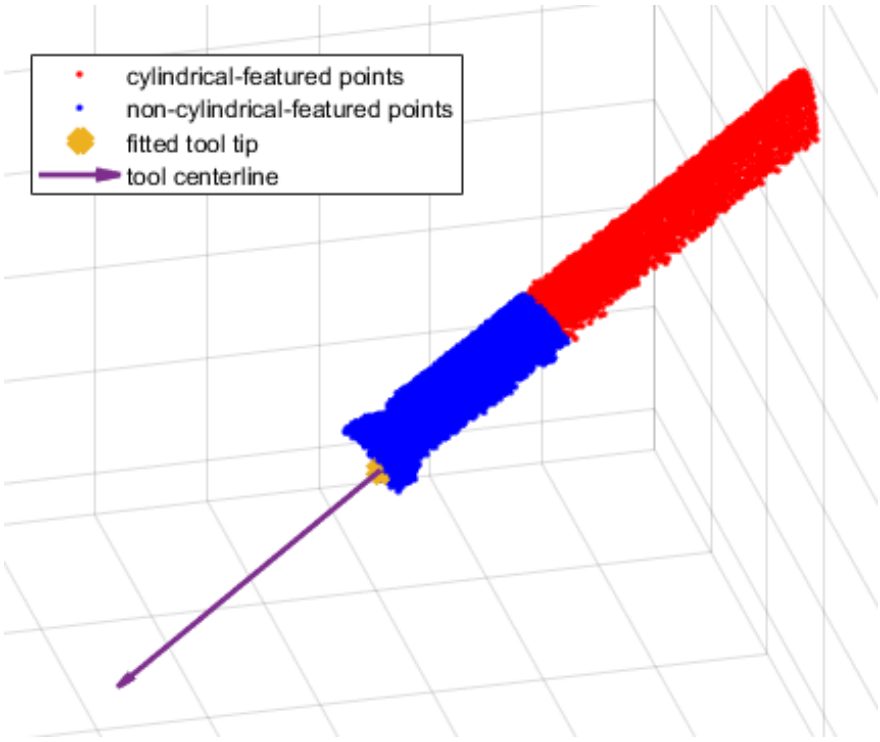


Figure 5.11: tool localization using OCT V scan

are referred to as B scans. If the geometrical topology of the object is known (e.g. cylinder), the point cloud can then be fitted using the following proposed algorithm to obtain the orientation and position of the object.

See Figure 5.11, a tool localization algorithm is developed to obtain the tooltip position (yellow dot) and tool axis direction (purple arrow) using an OCT volume scan, which essentially makes OCT a PZ-type CMM. The tool is assumed to have a cylindrical shaft (red dots) and may have non-cylindrical features at its tip (blue dots). The localization algorithm workflow is drawn in Figure 5.12 and discussed below

1. Principal component analysis is performed to create a bounding box to exclude outlier points and to obtain an initial guess of tool axis location and direction. Outlier rejection is performed on 3 principal axes separately. See Figure 5.13, the distribution of points along each axis is obtained and smoothed by the moving average. Then, the lower and

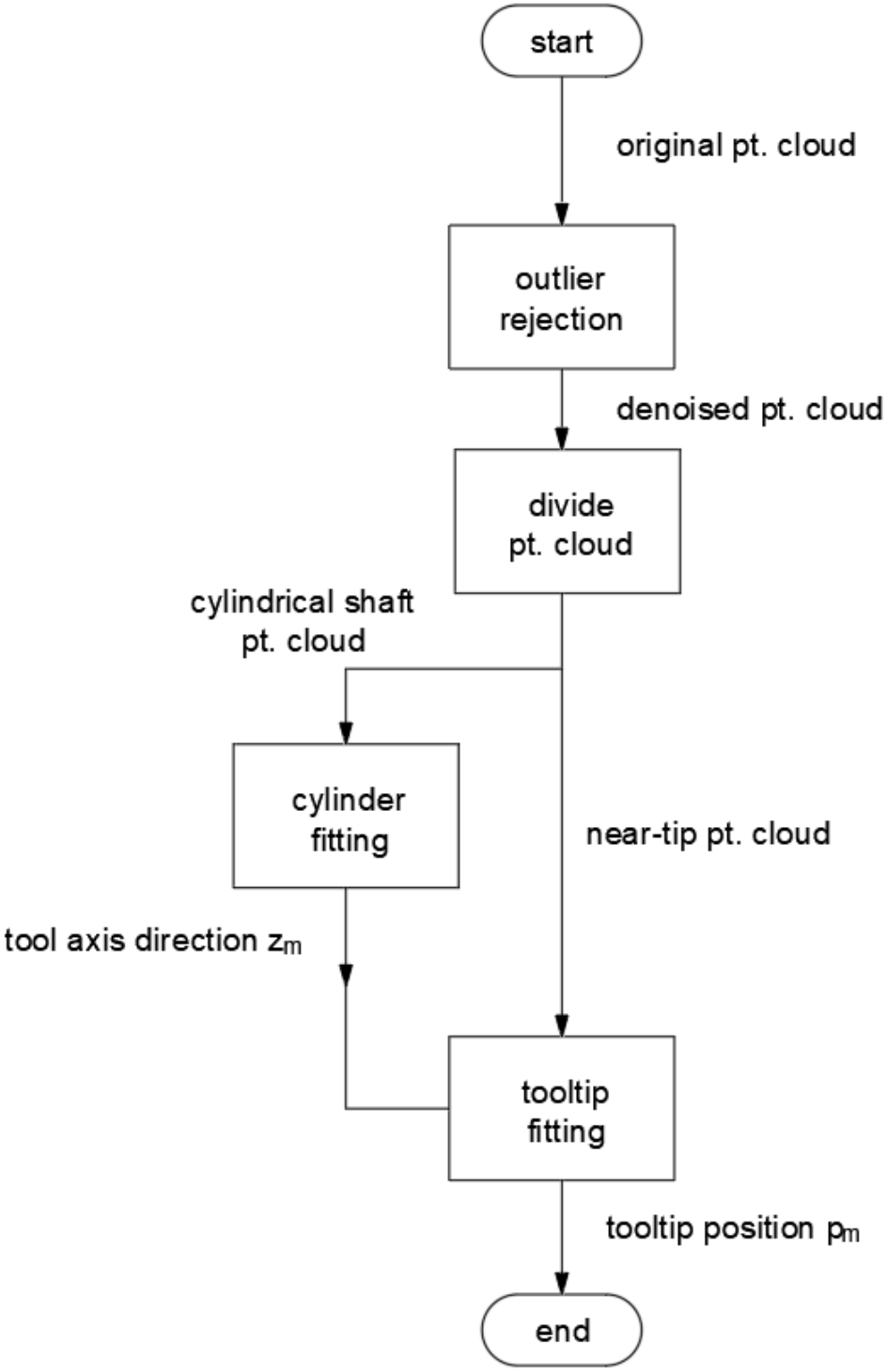


Figure 5.12: tool localization workflow. "pt." stands for "point".

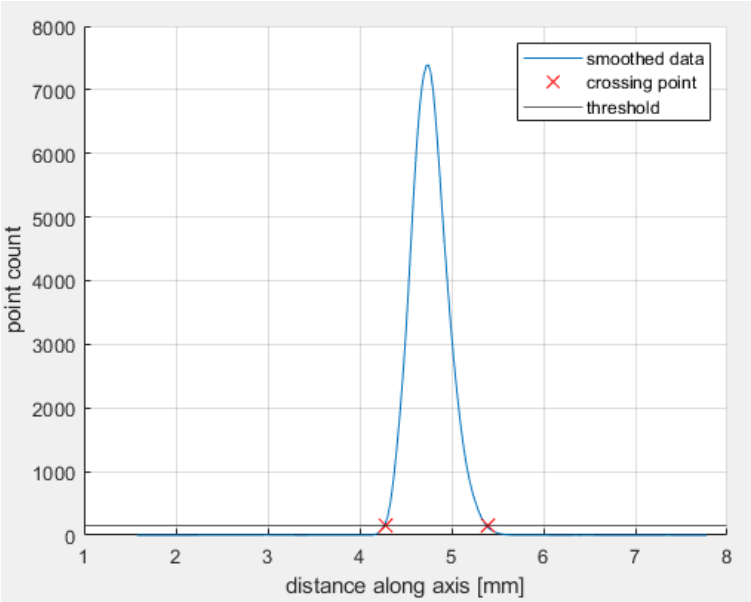


Figure 5.13: OCT point distribution along principal axis

upper bound of the bounding box along each axis is selected to be the crossing point of the highest peak and a threshold line of 2% peak height.

2. Divide the point cloud into 2 groups. The first group is the points of the cylindrical tool shaft. The other is the points of the non-cylindrical feature points near the tip. The plane of division is selected to be at a fixed distance from the outermost point of the point cloud.

3. To obtain tool axis direction, perform cylinder fitting on the points of the cylindrical shaft using the initial guess of the tool axis obtained in the first step.

4. To obtain the tooltip position, a value called d is assigned to each point. d is the location of the associated point along the tool axis z_m with $d = 0$ being the outermost point (See Figure 5.14). Draw a cumulative percentage plot vs d (Figure 5.15). The cumulative percentage stands for the percentage of points which has a less d value than the current d.

   Fit a line using the points of $d > -0.15mm$. Outlier is rejected during line fitting.

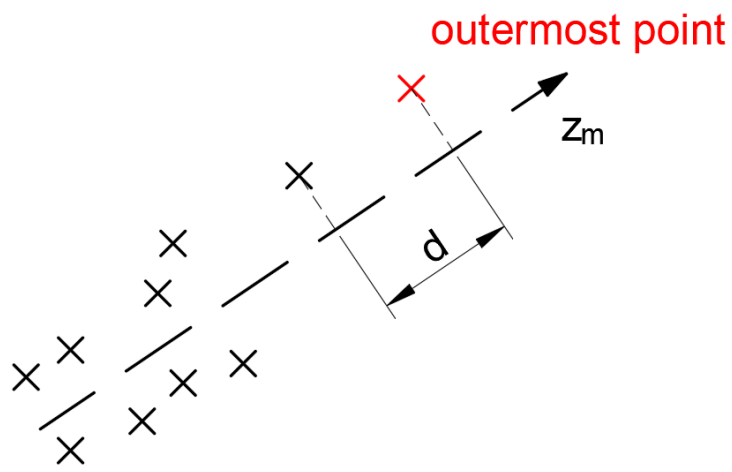


Figure 5.14: definition of d

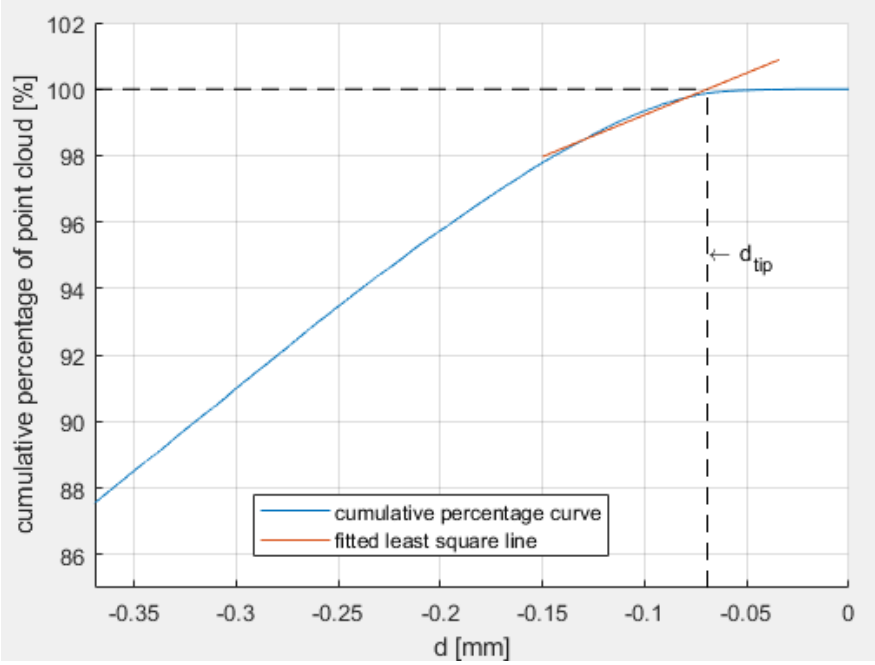


Figure 5.15: cumulative percentage vs d

Find the intersection of the fitted line and 100% cumulative percentage. The intersected point has d value of $d = d_{tip}$, which is considered the fitted tooltip location along the tool axis.

5.5 Parameter Identification

The nonlinear model regression technique introduced in Chapter 2 is used to identify the unknown DH parameters η, CT parameters t_{mb}, and joint compliance coefficient C. Using the notation in that chapter, the model parameter of interest, the model input, and the output are

$$\beta = \begin{bmatrix} \eta \\ t_{mb} \\ C \end{bmatrix} \in R^n, \quad x = \begin{bmatrix} q \\ \tau \end{bmatrix} \in R^p, \quad y = \begin{bmatrix} p_m \\ x_m \\ y_m \\ z_m \end{bmatrix} \in R^m \tag{5.23}$$

, where $q = [q_1; q_2; \cdots; q_N] \in R^{Nh}$, supposing the robot has h joints, and N points are measured during the measuring steps. For i-th points, the joint encoder measurement is denoted $q_i \in R^h$. The notation for joint torque/force measurement $\tau \in R^{Nh}$, tool frame position measurement $p_m \in R^{3N}$, and tool frame orientation measurement x_m,y_m,$z_m \in R^{3N}$ are defined in a similar fashion. Depending on the CMM type, a subset of tool pose measurement (p_m, x_m,y_m,z_m) may be unavailable and y might change. Also, depending on whether it is desired or feasible to calibrate the joint compliance coefficient, τ and C may be taken out of x and β.

FK model function is as derived in Section 5.1, and the overhead symbol $\hat{}$ refers to any quantity estimated by the FK model. For example, $\hat{p}_m(x, \beta)$ means the TCP position relative to CMM estimated by FK.

5.5.1 Generic Optimization Scheme

Depending on the CMM type, the objective function may include position error and/or orientation error. The following would describe how to formulate the objective for the P, PZ, and T type CMM. Then, the LM method described in Section 2.3.2 can be employed to estimate the model parameters.

The position error and the orientation error for the 3 tool frame axes are defined as

$$\begin{aligned} e_p(\beta) &= p_m - \hat{p}_m(x, \beta) \\ e_x(\beta) &= x_m - \hat{x}_m(x, \beta) \\ e_y(\beta) &= y_m - \hat{y}_m(x, \beta) \\ e_z(\beta) &= z_m - \hat{z}_m(x, \beta) \end{aligned} \tag{5.24}$$

P-type

$$S(\beta) = e_p^T e_p \tag{5.25}$$

PZ-type

$$S(\beta) = e_p^T e_p + w^2 e_z^T e_z \tag{5.26}$$

T-type

$$S(\beta) = e_p^T e_p + w^2 (e_x^T e_x + e_y^T e_y + e_z^T e_z) \tag{5.27}$$

, where w is the weight between orientation error and position error. It is selected to be $w = \sigma_p/\sigma_z$. σ_p and σ_z are the standard deviations of the position and orientation measurement, which are obtained by some testing on the CMM.

5.5.2 Registration using vector field

The task of finding the CT parameters t_{mb} alone while other parameters are given is called a registration problem. This problem can be interpreted as finding the relative position and orientation of the robot base to CMM, ${}^{m}_{b}T$. Registration can be done explicitly using point cloud registration [HHN88], which is a technique to find the spatial transformation between 2 groups of points by matching them. Application of this method means matching the end-effector position measurements from CMM to its estimated positions from FK.

However, some CMMs can also measure the orientation (or part of orientation) of the end-effector. The problem now becomes matching] two vector fields instead of just two point clouds. In this case, an extended version of the registration method with vector fields is proposed to take advantage of the additional orientation measurement. This can also be solved explicitly. In another word, it is desired to find an estimator for transformation ${}^{m}_{b}T^{*}$ such that it is closed-form and has the following form

$$ {}^{m}_{b}T^{*} = g(y, x|\beta_1) \tag{5.28} $$

, where y may have none of, part of, or all of the orientation measurements, and (β_1,β_2) are

$$ \beta_1 = \begin{bmatrix} \eta \\ C \end{bmatrix}, \quad \beta_2 = t_{mb} \tag{5.29} $$

The above uses the notation convention given in Section 2.4. Depending on whether it is desired or feasible to calibrate C, C and τ may be taken out of β_1 and x. Note that CT parameter t_{mb} is never explicitly calculated using this method since it is actually ${}^{m}_{b}T$ that is of interest. There is no need to calculate its parameterization as is needed when doing numerical solving.

The estimator $g(\cdot)$ begins by calculating the tool frame $\{t\}$ relative to base frame $\{b\}$ by FK model functions for all N data points. For the i-th data point

$$ {}^{b}_{t}\hat{T}_i(x_i, \beta_1) = \begin{bmatrix} {}^{b}_{t}\hat{R}_i & \hat{p}_{b,i} \\ 0 & 1 \end{bmatrix}, \quad {}^{b}_{t}\hat{R}_i = \begin{bmatrix} \hat{x}_{b,i} & \hat{y}_{b,i} & \hat{z}_{b,i} \end{bmatrix} \tag{5.30} $$

Then the estimator aims to find the optimal ${}^{m}_{b}R$ and p_{mb} (both represented by t_{mb}) such that the objective in (5.25), (5.26), or (5.27) is minimized.

For errors given in (5.24), they turn into

$$ \begin{aligned} e_{p,i}(\beta_2|\beta_1) &= p_{m,i} - {}^{m}_{b}R\hat{p}_{b,i} - p_{mb} \\ e_{x,i}(\beta_2|\beta_1) &= x_{m,i} - {}^{m}_{b}R\hat{x}_{b,i} \\ e_{y,i}(\beta_2|\beta_1) &= y_{m,i} - {}^{m}_{b}R\hat{y}_{b,i} \\ e_{z,i}(\beta_2|\beta_1) &= z_{m,i} - {}^{m}_{b}R\hat{z}_{b,i} \end{aligned} \tag{5.31} $$

P-type

The objective in (5.25) becomes

$$ S(\beta_2|\beta_1) = \sum_{i=1}^{N} \left\| p_{m,i} - {}^{m}_{b}R\hat{p}_{b,i} - p_{mb} \right\|^2 \tag{5.32} $$

In [HHN88], it is explained that the above objective can be simplified into

$$ S_1(R_{mb}) = \sum_{i=1}^{N} \left\| p'_{m,i} - {}^{m}_{b}R\hat{p}'_{b,i} \right\|^2 \tag{5.33} $$

, where $p'_{m,i}$ and $\hat{p}'_{b,i}$ are the position measurement and position estimation by FK shifted to their respective average point, or

$$ \begin{aligned} p'_{m,i} &= p_{m,i} - \overline{p}_m, \qquad \overline{p}_m = \frac{1}{N}\sum_{i=1}^{N} p_{m,i} \\ \hat{p}'_{b,i} &= \hat{p}_{b,i} - \overline{p}_b, \qquad \overline{p}_b = \frac{1}{N}\sum_{i=1}^{N} \hat{p}_{b,i} \end{aligned} \tag{5.34} $$

, and the optimal p^*_{mb} is

$$p^*_{mb} = \overline{p}_m - {}^m_b R^* \overline{p}_b \tag{5.35}$$

${}^m_b R^*$ is found by minimizing the above $S_1({}^m_b R)$. Then, it is shown that minimizing $S_1({}^m_b R)$ is the same as maximizing $S_2({}^m_b R)$

$$S_2({}^m_b R) = Tr({}^m_b R^T M), \quad M = \sum_{i=1}^{N} p'_{m,i} \hat{p'}^T_{b,i} \tag{5.36}$$

Finally, it is shown that the optimal ${}^m_b R^*$ is

$${}^m_b R^* = M(M^T M)^{-\frac{1}{2}} \tag{5.37}$$

PZ-type

The objective given in (5.26) can be written as

$$S(\beta_2|\beta_1) = \sum_{i=1}^{N} \|p_{m,i} - {}^m_b R \hat{p}_{b,i} - p_{mb}\|^2 + w^2 \sum_{i=1}^{N} \|z_{m,i} - {}^m_b R \hat{z}_{b,i}\|^2 \tag{5.38}$$

By [HHN88], the first summation in the above equation can be turned into the form in (5.33) if optimal $p_{mb}*$ is given by (5.35). Then, the objective becomes minimizing the following

$$S_1(R_{mb}) = \sum_{i=1}^{N} \left\|p'_{m,i} - {}^m_b R \hat{p}'_{b,i}\right\|^2 + \sum_{i=1}^{N} \|w z_{m,i} - {}^m_b R w \hat{z}_{b,i}\|^2 \tag{5.39}$$

Comparing the term in the two summations in the above equation, it can be seen that the pairs, $(p'_{m,i}, \hat{p}'_{b,i})$ and $(wz_{m,i}, w\hat{z}_{b,i})$ take similar structure. Following the same deduction in [HHN88] by expanding the squared norm and using the cyclic property of trace, the objective becomes maximizing $S_2({}^m_b R) = Tr({}^m_b R^T M)$, where M is

$$M = \sum_{i=1}^{N} p'_{m,i} \hat{p'}^{T}_{b,i} + w^2 \sum_{i=1}^{N} z_{m,i} \hat{z}^{T}_{b,i} \tag{5.40}$$

The optimal ${}^{m}_{b}R^*$ is then given by calculating (5.37).

T-type

Following the previous deduction, the objective is to maximize $S_2({}^{m}_{b}R)$ as in (5.36) but M is given by the following

$$\begin{aligned} M &= \sum_{i=1}^{N} p'_{m,i} \hat{p'}^{T}_{b,i} + w^2 (\sum_{i=1}^{N} x_{m,i} \hat{x}^{T}_{b,i} + \sum_{i=1}^{N} y_{m,i} \hat{y}^{T}_{b,i} + \sum_{i=1}^{N} z_{m,i} \hat{z}^{T}_{b,i}) \\ &= \sum_{i=1}^{N} p'_{m,i} \hat{p'}^{T}_{b,i} + w^2 (\sum_{i=1}^{N} {}^{m}_{t}R_i \, {}^{b}_{t}\hat{R}^{T}_{i}) \end{aligned} \tag{5.41}$$

, and the optimal p^*_{mb} and ${}^{m}_{b}R^*$ can then be calculated by (5.35) and (5.37).

5.5.3 Nested Optimization Scheme

As discussed in Section 2.4, nested optimization scheme can be employed in the identification of FK parameters ($\beta_1 = [\eta; C]$) since part of the model parameter ($\beta_2 = t_{mb}$) can be found in a closed-form solution given the rest, as demonstrated in the previous section. This would improve the robustness of the optimization by eliminating the need to provide an initial guess for t_{mb}, which is usually very inaccurate. In addition, it can be shown empirically that this will improve the efficiency of LM method while simultaneously reaching the same result as the generic LM method.

Suppose ${}^{m}_{b}T^*$ given in (5.28) can be partitioned into the following

$${}^{m}_{b}T^* = g(y, x|\beta_1) = \begin{bmatrix} g_R(y, x|\beta_1) & g_p(y, x|\beta_1) \\ 0 & 1 \end{bmatrix} \tag{5.42}$$

Substitute the above into (5.31), and the errors given that the optimal CT parameter is used become the following. For simplicity and to emphasize β_1, y and x are dropped.

$$\begin{aligned} e_{p,i}(\beta_1) &= p_{m,i} - g_R(\beta_1)\hat{p}_{b,i}(\beta_1) - g_p(\beta_1) \\ e_{x,i}(\beta_1) &= x_{m,i} - g_R(\beta_1)\hat{x}_{b,i}(\beta_1) \\ e_{y,i}(\beta_1) &= y_{m,i} - g_R(\beta_1)\hat{y}_{b,i}(\beta_1) \\ e_{z,i}(\beta_1) &= z_{m,i} - g_R(\beta_1)\hat{z}_{b,i}(\beta_1) \end{aligned} \tag{5.43}$$

The nonlinear model regression problems in (5.25), (5.26), and (5.27) thus become the following forms depending on the CMM type. Note that all of the following objectives are only a function of FK parameters $\beta_1 = [\eta; C]$. This problem can then be solved using LM method.

P-type

$$S(\beta_1, {}^{m}_{b}T^*(\beta_1)) = \sum_{i=1}^{N} e_{p,i}^T e_{p,i} \tag{5.44}$$

PZ-type

$$S(\beta_1, {}^{m}_{b}T^*(\beta_1)) = \sum_{i=1}^{N} e_{p,i}^T e_{p,i} + w^2 e_{z,i}^T e_{z,i} \tag{5.45}$$

T-type

$$S(\beta_1, {}^{m}_{b}T^*(\beta_1)) = \sum_{i=1}^{N} e_{p,i}^T e_{p,i} + w^2 (e_{x,i}^T e_{x,i} + e_{y,i}^T e_{y,i} + e_{z,i}^T e_{z,i}) \tag{5.46}$$

CHAPTER 6

Application to an industrial robot

6.1 experiment design

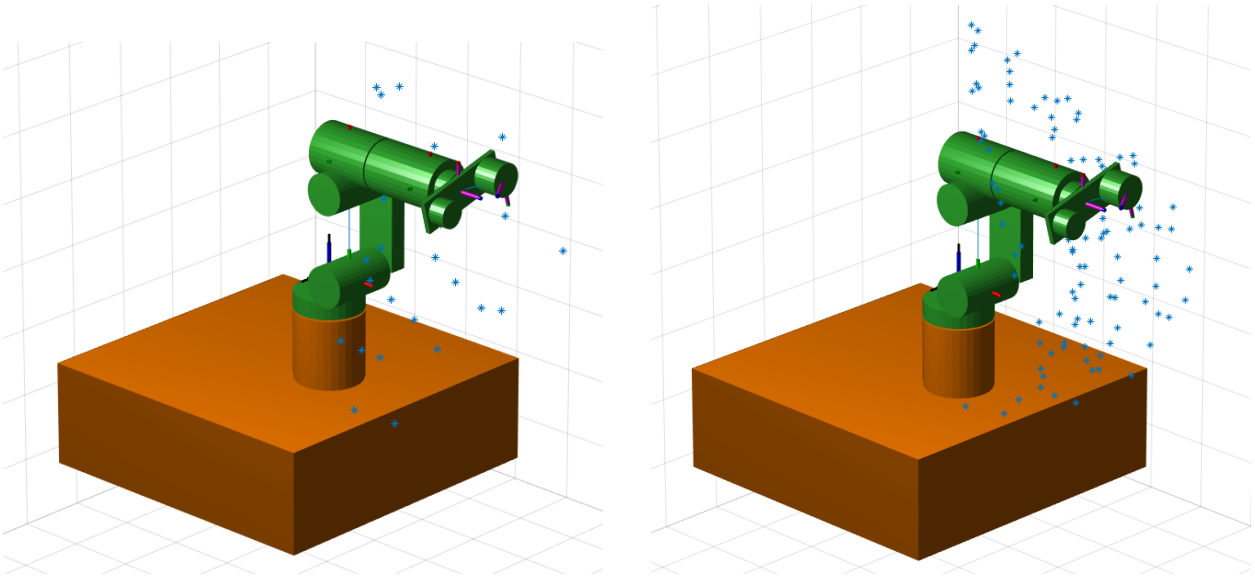
In this chapter, a 6-joint serial manipulator AR-607 (Figure 6.1, developed by Industrial Technology Research Institute, Taiwan) is used for demonstration. The CMM used is a laser tracker (AT901, Leica), which is a P-type CMM and can measure the position of an SMR mounted on the robot. To avoid the non-identifiability of the last link parameters (Section 5.3.4) 3 different tool transformation matrices ${}^{f}_{t}T$ are used. More specifically, a special SMR holder (See Figure 6.2) is mounted on the robot flange. It can hold SMR in 3 different mounting positions. In addition, payloads can be easily mounted onto this SMR holder. In total three different payload weights (0N, 14N, 28N) are used in this experiment. These weights of the payload are selected to span over the designed working range of the robot.

See Figure 6.3. The robot is commanded to 27 joint configurations for measurements. They are used in the calibration algorithm to identify the model parameters. In total, 27 joint configurations $\times$ 3 SMR holders $\times$ 3 payloads constitute 243 data points. They are referred to as calibration points. Additionally, another 100 joint configurations are used to validate the result. They are not used in the calibration algorithm. In total, 100 joint configurations $\times$ 3 SMR holders $\times$ 3 payloads constitute 900 data points. These are referred to as validation points. For each data point, the position measurement is taken by CMM, the joint value is recorded by the encoder, and the joint torque is recorded through motor current.

Figure 6.1: industrial robot AR-607 developed by ITRI

(a) mounting 1 (b) mounting 2 (c) mounting 3

Figure 6.2: SMR holder



(a) calibration points (b) validation points

Figure 6.3: data points used in experiment

The Cartesian coordinate of both calibration and validation points is chosen so as to span the workspace of the robot. Another requirement for these data points is that their pose must allow visibility of SMR to the laser tracker. To achieve this, an initial experiment is conducted to register the robot with CMM. The position and encoder measurements of 8 data points are collected. These measurements are used in the registration method in Section 5.5.2 to infer the location of CMM relative to the robot, assuming the DH parameters to be their nominal value and $C_i = 0$. Then, to obtain joint angles, inverse kinematics is performed to enforce the SMR opening pointing toward CMM within a certain angle tolerance. Note that optimal experiment design is not performed in this calibration scheme. It is demonstrated in Chapter 7.

6.2 modeling and identifiability analysis

The FK model follows the definition given in Section 5.1. The joint compliance model is assumed, and the extended DH parameter is assumed for the last link. The Euler angle sequence for CT parameter is $S = ZYX$. Under these conditions, there are only 4 unidentifiable parameters in this model, d_1, θ_1, d_2 and C_1. See Section 5.3.3, (d_1, θ_1) are typical unidentifiable parameters for they are dependent on the CT parameters. Since joint 2 and join 3 are parallel, d_2 is dependent on d_3 (Section 5.3.1). Lastly, C_1 is unidentifiable because the direction of gravity is parallel to the joint 1 axis, meaning the addition of payload would not apply torque to joint 1. Thus, the small deviation of C_1 will not cause any movement on the position of SMR. By the argument provided in Section 3.3.1, C_1 is unidentifiable. Since these 4 parameters are unidentifiable, by the argument given in Section 3.4.2, as long as their nominal values have a finite bias bound B (the nominal value is close to the actual value within the bias bound B), they should be fixed at nominal value. This means they are assumed known. Note that the nominal value of $C_1 = 0$. In total, there are 34 parameters to be identified (6 CT parameters + 21 DH parameters + 2 EDH parameters for the last link + 5 Joint compliance coefficients).

Table 6.1: identified DH parameter deviation and Joint compliance coefficient

joint	1	2	3	4	5	6
δd	-	-	-0.22	0.56	-0.17	-0.55
$\delta\theta$	-	-0.06	-0.16	-0.04	0.01	-0.76
δa	-0.17	0.19	0.06	-0.15	0.21	-0.55
$\delta\alpha$	-0.02	-0.02	0.04	0.06	-0.02	-0.09
δb	N.A.	N.A.	N.A.	N.A.	N.A.	-0.92
$\delta\beta$	N.A.	N.A.	N.A.	N.A.	N.A.	-0.37
C	-	0.88	1.92	2.57	10.62	5.92

[a] Unit for angle is in deg, length in mm., C in 10^{-3} deg/Nm

[b] "δ" means deviation of the calibrated value from the nominal value

[c] hyphen "-" means fixed parameter. Its value is 0

6.3 Result

6.3.1 calibrated parameters and position error

The initial guess of DH parameters and joint compliance coefficient uses the nominal value. Note that the nominal value of the joint compliance coefficient is 0 ($C_i = 0$). Unlike the nested scheme, the generic scheme requires an initial guess of CT parameters. It is obtained by performing the registration method provided in Section 5.5.2 on the 243 calibration points. This initial registration assumes the DH parameters and joint compliance coefficients to be nominal values. The calculated initial guess of CT parameter is listed in Table 6.2.

The calibrated values of model parameters are shown in Table 6.1 and 6.2. It is observed that the calibrated values and the error are completely the same for both schemes. This means that both schemes can achieve the same results and the same residual error.

Table 6.2: identified CT parameter

	$p_{mb,x}$	$p_{mb,y}$	$p_{mb,z}$	$r_{mb,z}$	$r_{mb,y}$	$r_{mb,x}$
initial guess	2.6795	2.0032	-0.4972	-178.32	-3.61	0.88
calibrated value	2.6789	2.0020	-0.4973	-178.41	-3.57	1.16

[a] Unit for angle is in deg, length in m

To have a better visualization of error distribution, see Figure 6.4. The median error of the validation points is 0.232 mm (red band) and that of the calibration points (0.138 mm). See Table 6.3, the residual error after 3 different kinds of calibration scenarios are reported. The 3 scenarios use the same data (243 calibration points) to identify model parameters. The only difference is the model parameter they identify. For example, scenario CT only identifies CT parameters, while assuming DH parameters and joint compliance coefficient are at nominal value.

6.3.2 Joint compliance model

See Table 6.3, compared to scenario CT+DH, full calibration scenario CT+DH+C has smaller errors in every criterion. For example, validation points has an rms error of 0.490mm after full calibration CT+DH+C, while that after partial calibration CT+DH is 0.64 0mm. This suggests incorporating joint compliance into the model could improve accuracy. This can also be seen in Figure 6.5. This figure shows the position error distribution of validation points under various loads (0N, 14N, and 28N) and different calibration scenarios (CT+DH or CT+DH+C). If joint compliance is not considered (CT+DH), the points having larger loads have noticeably larger median and maximum errors. However, if joint compliance is considered (CT+DH+C), the error is smaller and less sensitive to load. This shows that the incorporation of the joint compliance model into FK calibration would improve the accuracy

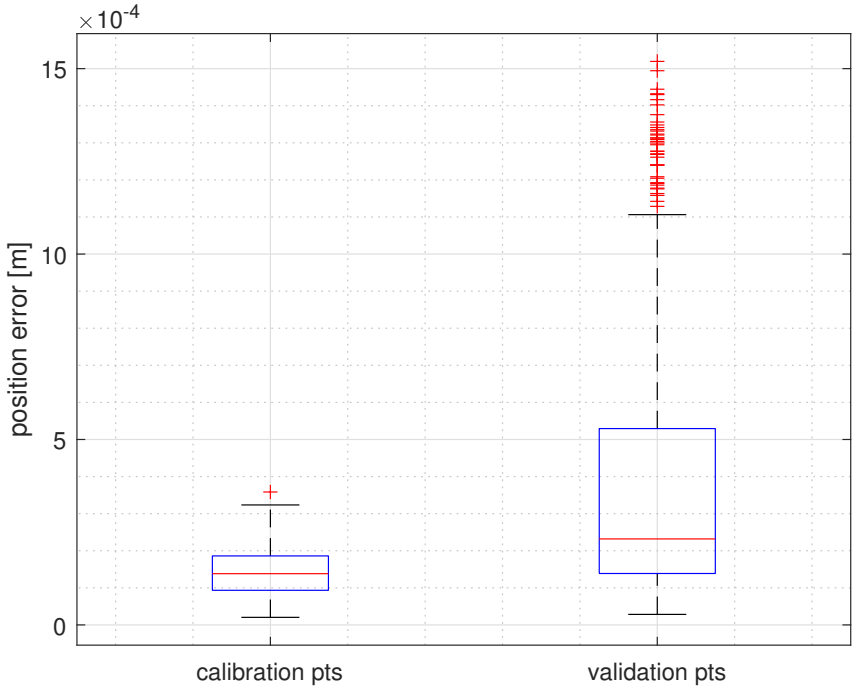


Figure 6.4: box plot of position error, the box represents the 25th and 75th percentile, the red band represents the median, the whisker represents maximum or minimum value, and the red crosses represent outliers. Points with position error greater than 75th percentile plus 1.5 times the interquartile distance are considered outliers.

Table 6.3: residual error

point type	scenario[b]	rms	max	std
calibration points	CT	1.539	3.697	0.615
	CT + DH	0.218	0.460	0.088
	CT + DH + C	0.154	0.358	0.064
validation points	CT	1.664	3.646	0.717
	CT + DH	0.640	2.030	0.422
	CT + DH + C	0.490	1.519	0.328

[a]Unit is mm

[b] indicates whether a specific group of parameter is identified

of the FK estimation against varying loads.

6.3.3 efficiency and robustness improvement of nested scheme

To demonstrate the efficiency improvement of the nested scheme, for the nested scheme, the averaged calculation time for each LM iteration is 0.2226 ± 0.0001 s, and, for the generic scheme, that is 0.2644 ± 0.0002 s. The generic scheme is around 18% slower. For both schemes, the Jacobian calculation time takes 96% of the time for each iteration. See Figure 6.6 (a), to reach the same error change convergence criteria with $\epsilon_f = 10^{-9}$ (Section 2.3.1), the generic scheme needs to perform 12 LM iterations, while the nested scheme only needs 8 iterations. That is around 33% less iteration needed. In conclusion, The nested scheme requires not only less time on each iteration but also fewer iterations to achieve the same convergence criteria.

To demonstrate the robustness of the nested scheme, see Figure 6.6. If the generic scheme has a very good initial guess not far from calibrated value (like the one in Table

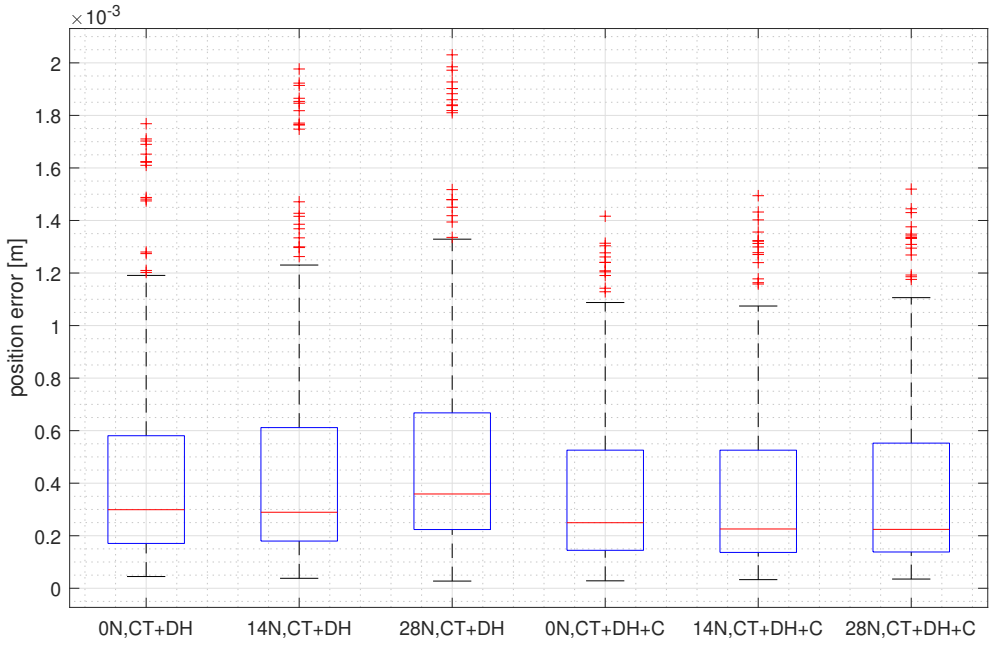
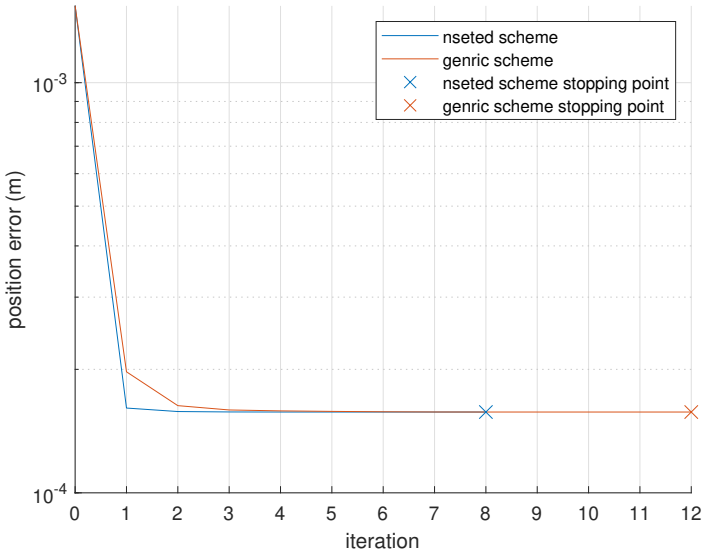
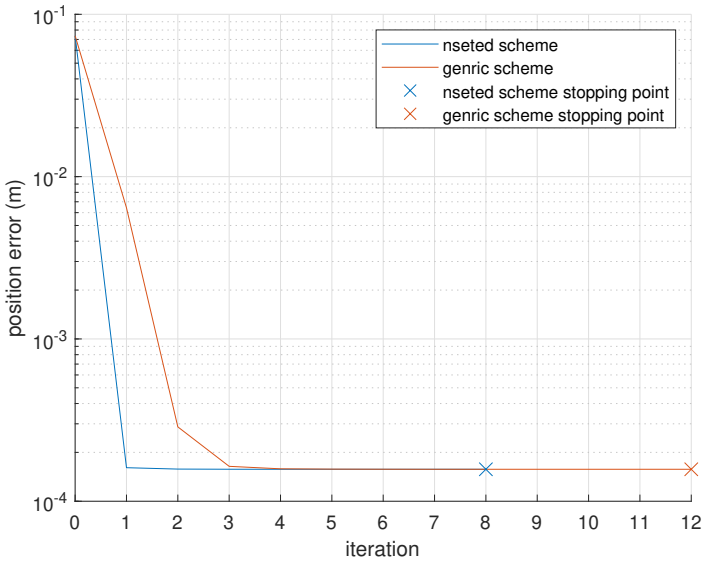


Figure 6.5: box plot of position error of the validation points under various loads and calibration scenarios



(a) good CT initial guess

$$t_{mb0} = [2.6795\ 2.0023\ -0.4972\ -178.32\ -3.61\ 0.88]$$



(b) bad CT initial guess

$$t_{mb0} = [2.6805\ 2.0023\ -0.4972\ -168.31\ -3.61\ 0.88]$$

Figure 6.6: convergence curve

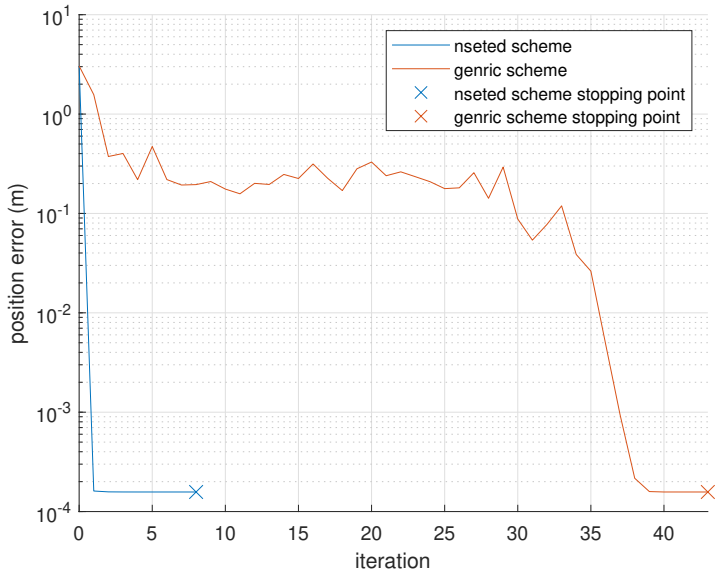


Figure 6.7: convergence curve with a very bad CT initial guess

$$t_{mb0} = [0\ 0\ 0\ 0\ 0\ 0]$$

6.2), the convergence of the nested scheme is only slightly faster than that of the generic scheme (Figure 6.6 (a)). If there is an unpredictable disturbance in the initial guess of the CT parameter, the convergence for the generic scheme will deteriorate (See Figure 6.6 (b)), while that of the nested scheme remains unaffected. To demonstrate an extreme case, see Figure 6.7. In this case, the CT initial guess is so bad that it needs 43 iterations to converge, while the nested scheme needs just 8 iterations. These show that the nested scheme is robust against the inaccuracy of the CT initial guess.

CHAPTER 7

Application to a surgical robot

7.1 Full calibration on entire Kinematic chain

7.1.1 experiment setup

See Figure 7.1, the surgical robot called Intraocular Robotic Intervention Surgical System Version 2 (IRISS v2) is developed by UCLA. It is a 4-joint robot. The first 2 joints and the last joint are revolute joints. The 3rd joint is a prismatic joint. It has an exchangeable tool cartridge (See Figure 7.2), which can easily be dismounted to replace the tool and the last joint of the robot. The CMM used in calibration and also widely used in eye surgery is an Optical coherence tomography device (OCT). The OCT device used in this experiment is Telesto II 1060LR with objective lens LSM04BB (Thorlabs). Its volume scan precision is 9.2 μm along the lens axis and 25 μm along the lateral direction. The OCT V scan can be used to infer the TCP and tool direction through the tool localization method mentioned in Section 5.4. This makes OCT a PZ-type CMM. A precision test of the tool localization method is performed with an I/A handpiece (92-IA21 Handle, Millennium Surgical). During the test, the I/A tool is fixed by a swivel clamp under the OCT lens, and the volume scan of its tip is taken n=30 times. The precision error is rms error of the measured tip location and tool direction by the tool localization method, which is 15 μm and 0.035 deg respectively.

During the calibration experiment, the I/A tool cartridge is mounted on the robot. The plastic cap on the I/A tooltip is removed. Then, the tip of the I/A tool is adjusted to be placed under OCT sensing workspace (See Figure 7.3). For each data point, the robot

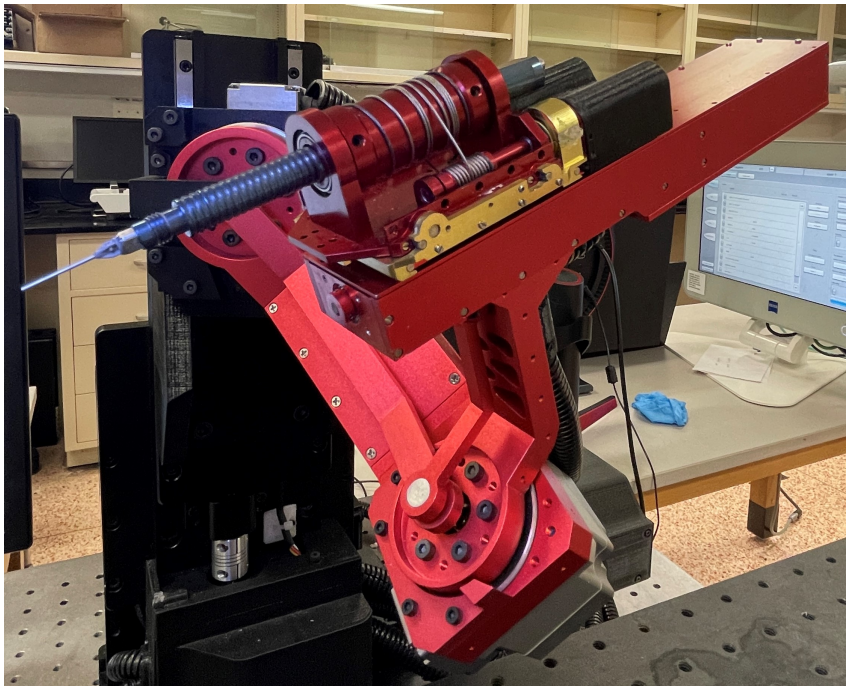


Figure 7.1: IRISS v2 robot

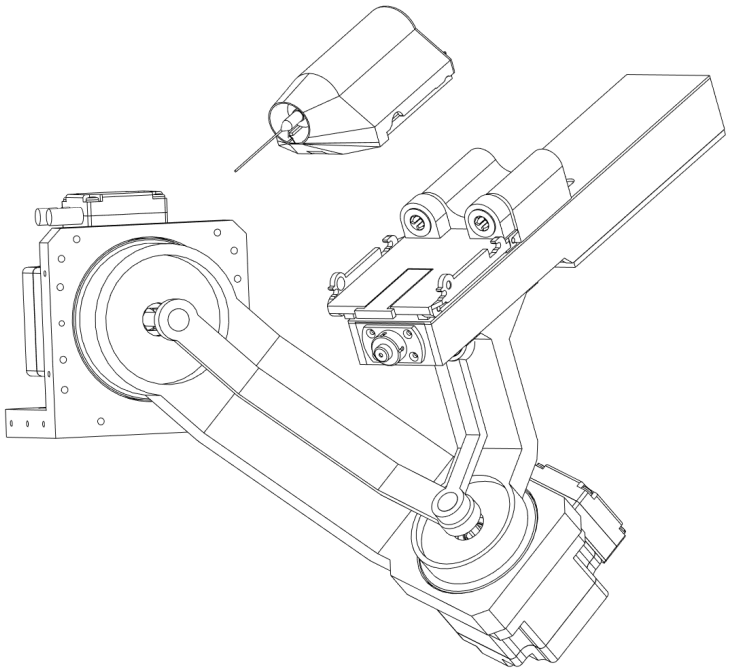


Figure 7.2: IRISS v2 robot when tool cartridge is detached

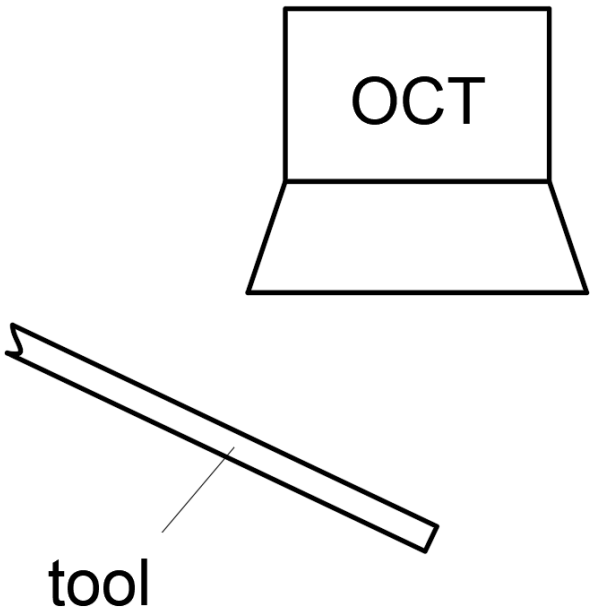


Figure 7.3: IRISS v2 and OCT setup

is commanded to a designated joint configuration. Then, the OCT volume scan and joint encoder value are recorded. A total of 3 groups of data points are taken. The first group (N_1=60) is random points within the OCT working range. They are used as calibration points. The second group (N_2=60) is optimal points derived using optimal experiment design and also used as calibration points (See Section 7.1.3 for detail). The last group is random points but used as validation points (N_3=60). Therefore, a total of 180 data points are taken. These points, though generated differently, all fit certain Cartesian space constraints described in Section 7.1.3.

7.1.2 modeling and identifiability analysis

Since the surgical robot would only handle light load, it is not needed to use the joint compliance model to compensate for angle deflection induced by heavy load. Also, the extended DH parameter is used for the last link. ZYZ Euler angle representation is used to

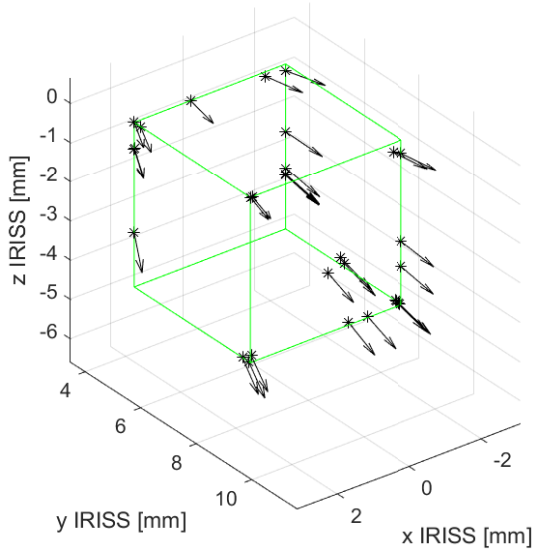
represent CT parameters. Lastly, for this special robot, the tool frame is considered the same as the flange frame, since there is no robot flange. This means that the tool transformation matrix is ${}^{f}_{t}T = I_4$.

For Identifiability analysis, there are 6 unidentifiable parameters, $d_1, \theta_1, \theta_3, \theta_4, d_4$, and a_2, assuming nominal DH parameter. These unidentifiable parameters can be found by examining the model function. θ_3 and θ_4 are redundant because their deviation would only cause rotation along the tool axis, making no difference in the TCP and tool direction. (d_1, θ_1) are unidentifiable due to the non-identifiability of the base frame. d_4 is dependent with d_3 since joint 3 and 4 are parallel ($\alpha_3 = 0$, Parallel axis non-identifiability). Lastly, a_2 and a_3 are dependent since their deviation happens to cause the TCP to move in the same direction.

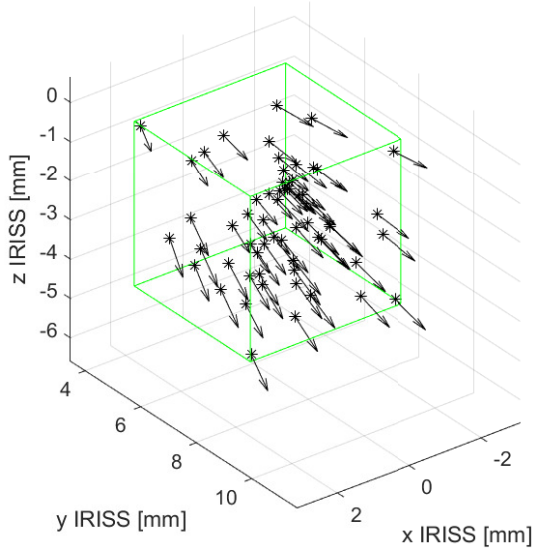
The identifiability of parameters can also be determined by SVD of the Jacobian matrix. To do that, 30 sets of joint variables, q_i, are randomly selected. Then, the Jacobian matrix is calculated using those joint configurations. Redundant parameters have to be determined first by examining the norm of each Jacobian column. Redundant parameters would have close to 0 norms. Recall that redundant parameters refer to a special kind of non-identifiability, where the parameter is unidentifiable by itself and not dependent on other parameters. In the case of the IRISS robot, the Jacobian column of θ_3 and θ_4 are 0 vectors; thus, they are redundant. Secondly, dependent parameters are found by calculating the scaled Jacobian matrix and examining its singular values and right singular vectors (Section 3.3.3). These unidentified parameters should then be fixed at their nominal values while solving the calibration problem.

7.1.3 experiment design

For the optimal points, the point-by-point search method discussed in Section 4.1.2 is employed. The obtained optimal design points are shown in Figure 7.4 along with the random points. Note that the optimal points tend to gather near the edge of the boundary. The



(a) optimal calibration points



(b) random calibration points

Figure 7.4: calibration points for full calibration. The green cube is the specified boundary, which is within OCT sensing workspace. The points are presented in the world frame (also called the IRISS frame here).

point-by-point method iteratively solves the following problem until a given data point number N is reached.

$$\begin{aligned} &\min_{q_{k+1}} -log(det(\sum_{i=1}^{k} F_i + F_{k+1}(q_{k+1}))) \\ &s.t.\ C(q_{k+1}) \leq 0 \end{aligned} \tag{7.1}$$

For the objective of the optimization problem, D-optimality is chosen. Recall that the Fisher information matrix $F_i(q_i)$ is.

$$F_i(q_i) = J^T V_e^{-1} J \tag{7.2}$$

As discussed in Section 5.2, the calculation of Jacobian requires both joint value and model parameter as input. However, the model parameter is unknown. Therefore, for the DH parameter, the nominal value is used instead. For the CT parameter, an experiment is done ahead to obtain the initial guess (Section 5.2 and Section 5.5.2). Recall that, V_e is (Section 2.2)

$$V_e = V_y + J_x V_x J_x^T \tag{7.3}$$

See (2.7) for the definition of V_y and V_x. V_y is the OCT measurement noise covariance matrix, which is the position and orientation measurement noise of the tool localization method using OCT V-scan (Section 5.4). A precision test is done ahead to determine V_y. The test shows that the tooltip position measurement noise has a standard deviation of 0.015 mm, and an orientation noise standard deviation of 0.035 deg. As for V_x, it is the input noise covariance matrix, which arises from the uncertainty due to the resolution of the joint encoder. Such uncertainty (4.5e-4 deg for joint 1 and 2, 0.07 um for joint 3, and 0.09 deg for joint 4) and its effect on tool movement are assumed small compared to OCT measurement noise. Therefore, V_x is assumed 0 for simplicity.

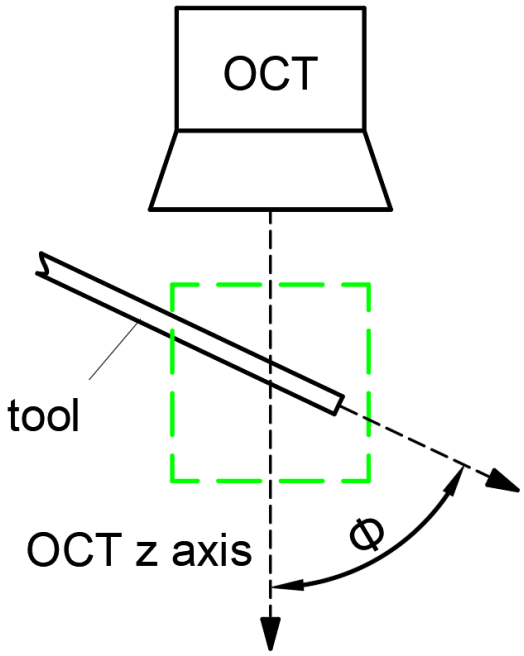


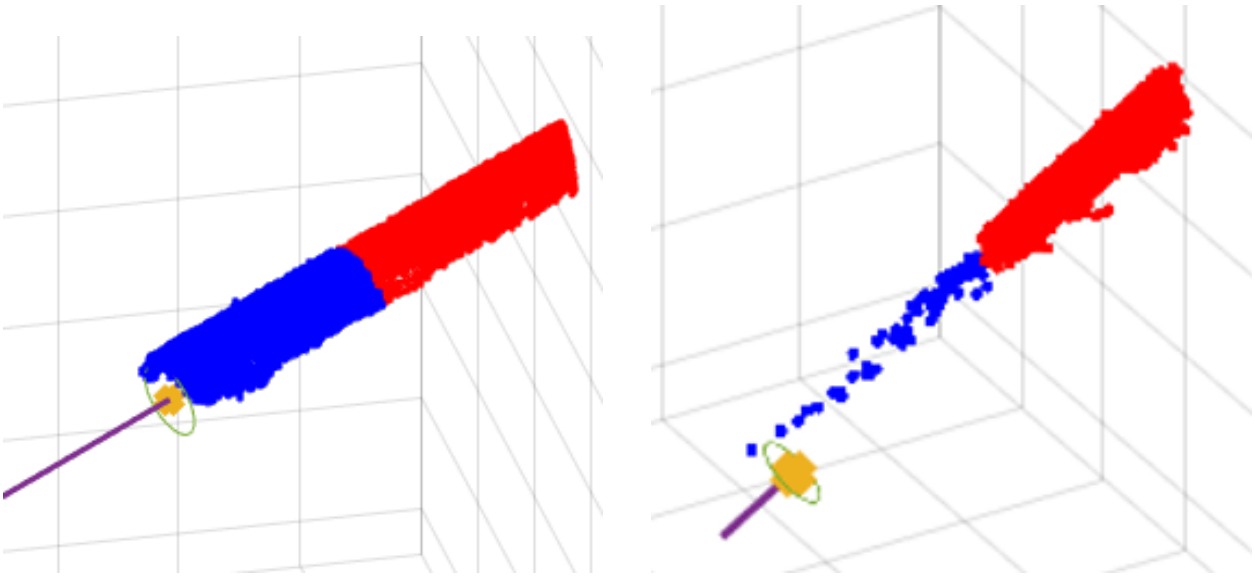
Figure 7.5: definition of tool angle ϕ and cube constraint

For the constraint function $C(q)$, there are three constraints that the data point must satisfy in order to obtain valid OCT measurement. The first constraint is the joint angle constraint, meaning the joint angle must be within some fixed bound for safety reasons. The second constraint is that the tooltip position must be within the OCT working range. This is done by enforcing its location estimated by FK to be within a cube (L = 4 mm) defined relative to the OCT frame. This is demonstrated by Figure 7.5.

$$\hat{p}_m(q) \in S_{cube} \tag{7.4}$$

The third constraint is that the angle ϕ between the tool axis and the OCT axis can not be smaller than a given angle ϕ_{th} (See Figure 7.5); otherwise, the V-scan would have bad quality (See Figure 7.6).

$$\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \hat{z}_m(q) - cos(\phi_{th}) \leq 0 \tag{7.5}$$



(a) good quality OCT V-scan (b) bad quality OCT V-scan

Figure 7.6: OCT V-scan comparison. Note that the bad OCT V-scan has loose point cloud density (See blue points).

For random points, each coordinate of the point is selected by uniform distribution within the cube. IK is then performed to obtain the joint configuration. If the angle constraint mentioned above is not satisfied, then discard the point and repeat the process until the specified amount of points is found.

7.1.4 Result

Nested scheme is used to identify the model parameters. The weight between is chosen to be the ratio between the standard deviations of position noise and orientation noise.

$$w = \frac{\sigma_p}{\sigma_z} = \frac{0.015mm}{0.035deg} = 0.024m \tag{7.6}$$

The identified parameters are shown in Table 7.1 and 7.2. The residual error of the validation points after calibration is shown in Table 7.3. It can be seen that the results for both random

Table 7.1: identified DH parameter deviation after full calibration of surgical robot

calibration point	joint	1	2	3	4
optimal point	δd	-	0.15	-0.46	-
	$\delta\theta$	-	0.87	-	-
	δa	-0.32	-	1.13	-0.34
	$\delta\alpha$	0.91	-3.13	1.50	0.18
	δb	N.A.	N.A.	N.A.	-0.04
	$\delta\beta$	N.A.	N.A.	N.A.	-1.50
random point	δd	-	0.20	-0.45	-
	$\delta\theta$	-	0.76	-	-
	δa	-0.33	-	1.12	-0.34
	$\delta\alpha$	1.00	-2.83	1.33	0.17
	δb	N.A.	N.A.	N.A.	-0.04
	$\delta\beta$	N.A.	N.A.	N.A.	-1.43

[a] Unit for angle is in deg, length in mm.

[b] hyphen "-" means fixed parameter. Its value is 0

points and optimal points are similar

To further compare the result between optimally designed points and random points. See Figure 7.7, Figure 7.8 and Figure 7.9. With the increase in the number of calibration points used, the rms error of the validation points using optimal design decreases slightly faster than that using the random points.

Table 7.2: identified CT parameter after full calibration of surgical robot

calibration point	$p_{mb,x}$	$p_{mb,y}$	$p_{mb,z}$	$r_{mb,z}$	$r_{mb,y}$	$r_{mb,z}$
optimal point	5.96	11.16	2.64	26.39	-89.03	-179.21
random point	5.98	11.22	2.64	26.24	-89.08	-179.51

[a] Unit for angle is in deg, length in mm

Table 7.3: residual error of validation points after full calibration of surgical robot

calibration point	error type	rms	max	std
optimal	position [μm]	44	93	19
	orientation [deg]	0.23	0.55	0.11
random	position [μm]	45	93	21
	orientation [deg]	0.24	0.77	0.13

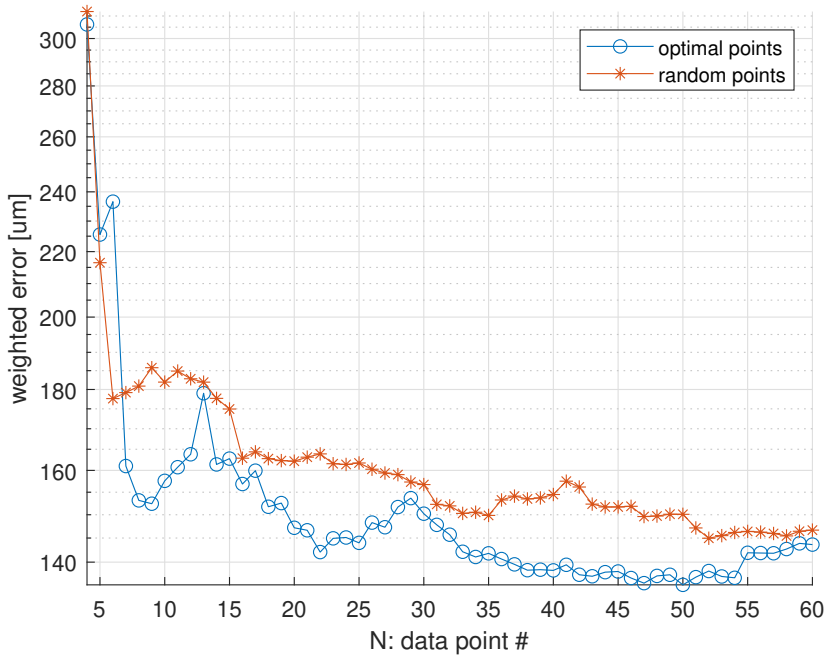


Figure 7.7: weighted error ($e_w = e_p + we_z$) vs N for full calibration

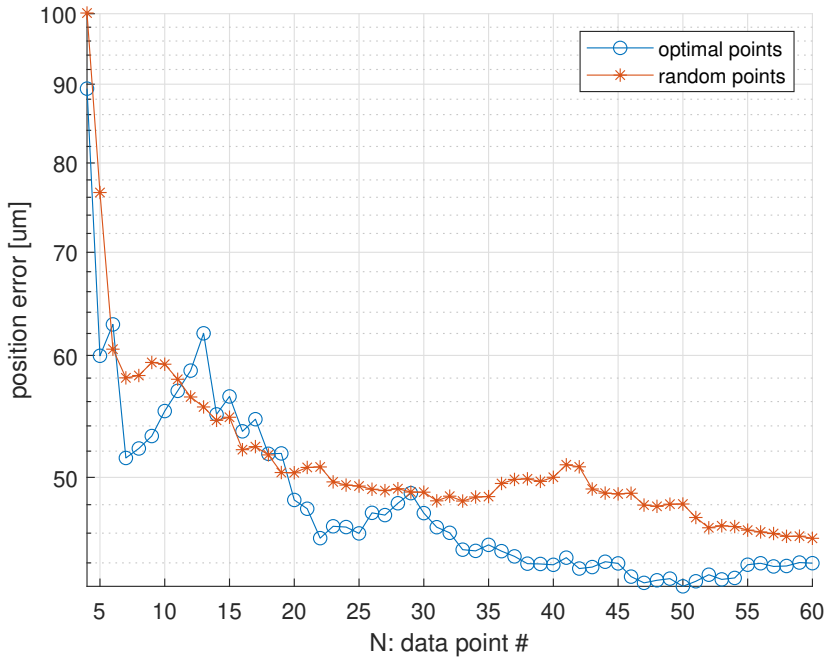


Figure 7.8: position error (e_p) vs N for full calibration

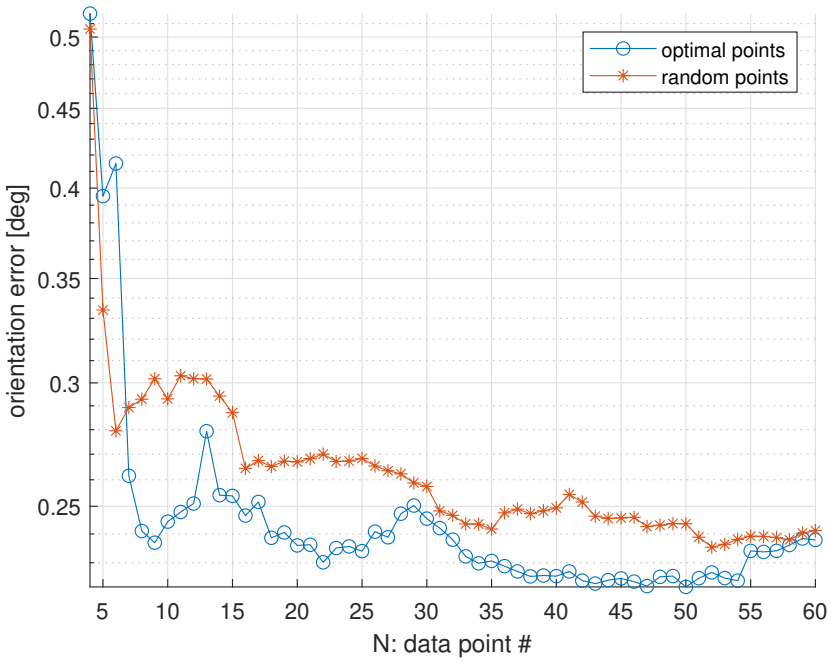


Figure 7.9: orientation error (e_z) vs N for full calibration

7.2 In situ partial calibration on instrument

7.2.1 background

The goal of in situ partial calibration is to calibrate the DH parameters of the last 2 links. This need comes from the special mechanism of IRISS v2. Its tool cartridge can be detached and replaced with other tools in order to perform various surgical operations. After a tool exchange happens, the axis of the last joint and the tool would not be the same as before, which corresponds to the change in DH parameters of the last 2 links (η_3 and η_4). This could lead to an unacceptably large position error (1 mm) for most surgical operations. The proposed calibration workflow is listed as followed. This workflow is also drawn in a flowchart in Figure 7.10.

1. perform full calibration to identify all DH parameters (η_i, $i = 1 \sim 4$) and CT parameters (t_{mb})

2. perform a surgical operation

3. continue if there are still operations left undone

4. exchange tool for next surgical operation

5. perform in situ partial calibration to identify DH parameters of the last 2 links (η_3 and η_4)

6. go back to step 2.

The FK model and experiment procedure to collect data for in situ operation are the same as full calibration (Section 7.1). The difference is that full calibration only needs to be performed once prior to all surgical operations, while partial calibration needs to be performed each time tool exchange happens. Another difference is that partial calibration only calibrates the last 2 sets of DH parameters while the CT parameter and first 2 sets of DH parameters are assumed known from full calibration. This setup reduces the number of parameters to be calibrated in partial calibration and, in theory, reduces the data points needed to achieve the same accuracy. Ideally, the less time spent on partial calibration, the better. Therefore, this reduction of data points needed is beneficial to streamlining the surgical operation.

7.2.2 experiment setup and design

The experiment setup including the tool, the CMM, and the robot are the same as in Section 7.1. To simulate the workflow presented in Figure 7.10, full calibration has to be done ahead. This is achieved by using the identified parameters η_1 η_2 and t_{mb} from the previous section (Table 7.2 and Table 7.1). Then, the tool cartridge is dismounted and replaced with a different I/A tool. After that, three groups of points, including 60 optimal calibration points, 60 random calibration points, and 60 validation points, are taken.

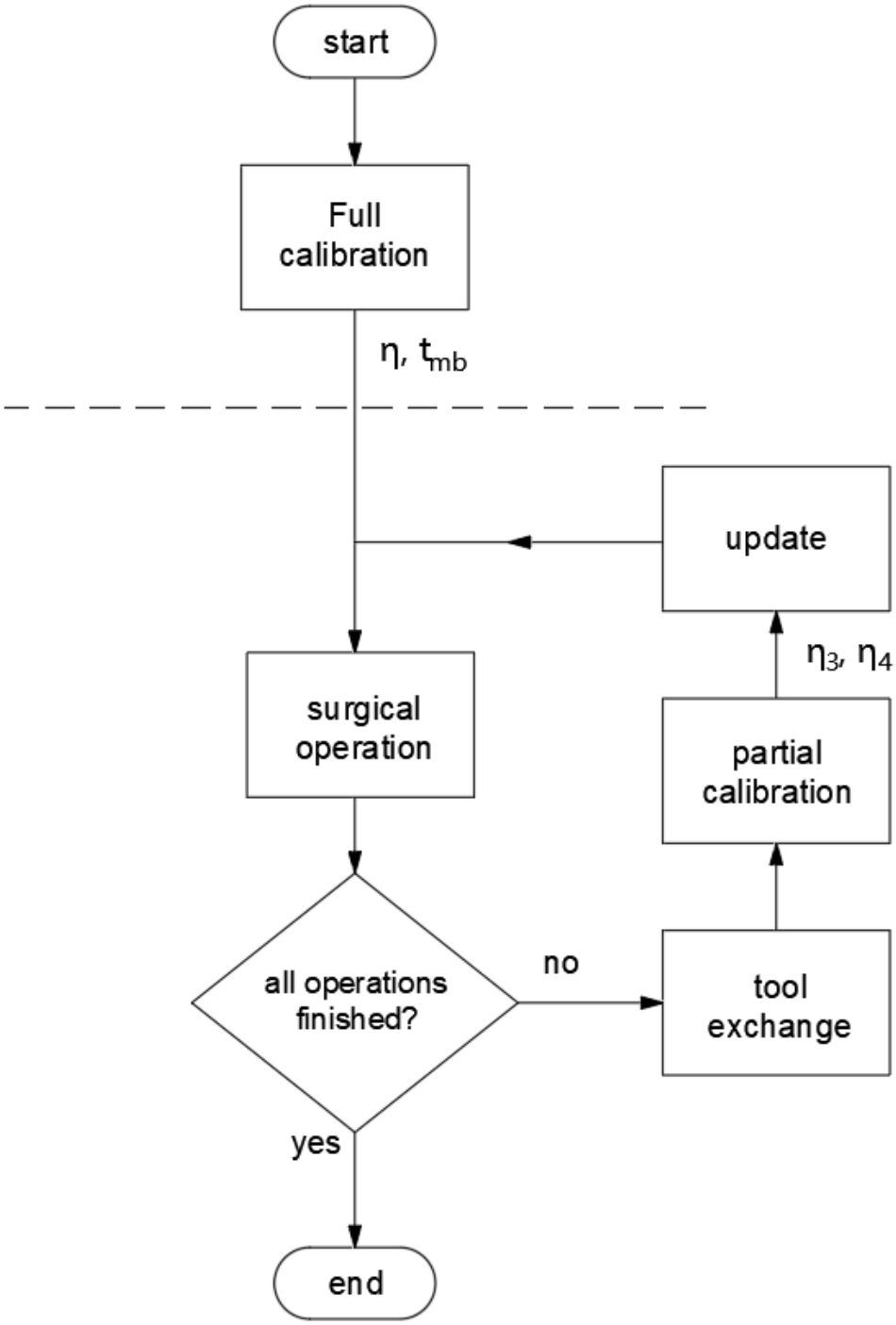


Figure 7.10: proposed calibration workflow

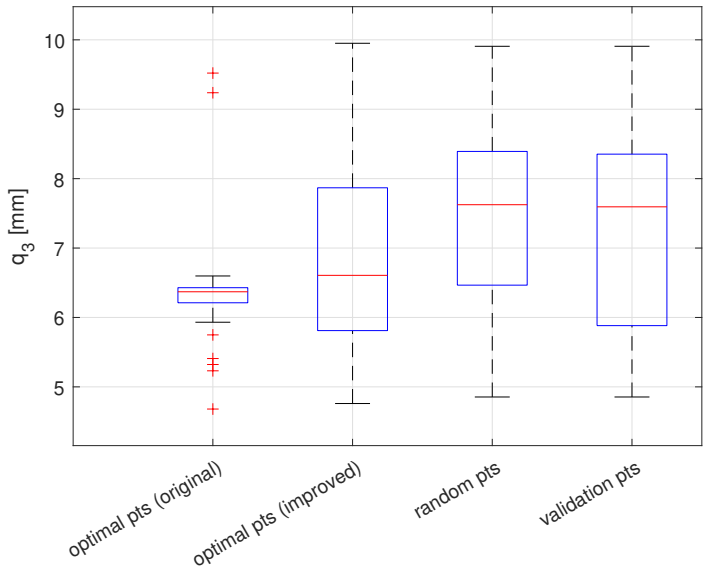


Figure 7.11: comparison of joint 3 range

The constraint for all the points remains the same as in Section 7.1.3. Also, the method to generate the random points is the same as in Section 7.1.3. However, for optimal points, an extra term is added to the objective of the original optimal design problem (7.1). Its purpose is to increase the randomness of joint 3. If not, joint 3 of all the optimal points would have a very small distribution range (See original optimal points in Figure 7.11). It has been observed that this would cause the calibration with optimal points to have a much worse result than random calibration points. The reason is suspected to be that concentrated joint 3 value would not be able to capture all the unmodeled behavior of joint 3.

In light of this joint 3 issue, an improved optimal design problem incorporating an extra term is proposed below.

$$\begin{aligned} &\min_{q_{k+1}} -log(det(\sum_{i=1}^{k} F_i + F_{k+1}(q_{k+1}))) - h\ log(c(\sigma_3(q), \sigma_{3,th}, r)) \\ &s.t.\ C(q_{k+1}) \leq 0 \end{aligned} \tag{7.7}$$

, where $\sigma_3(q)$ means the standard deviation of the joint 3 value of all joint configurations, and

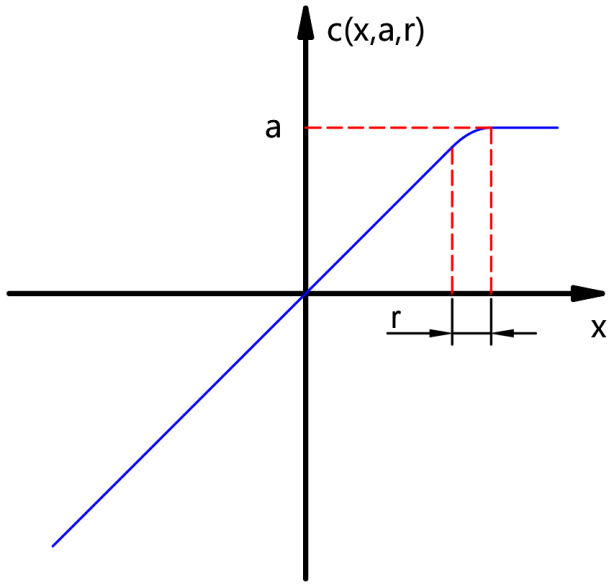


Figure 7.12: illustration of clip function $c(\cdot)$

h is a weighting factor set as $h = 0.04$. $(\sigma_{3,th}, r) = (1.3\text{mm}, 0.2\text{mm})$ is the clipping threshold and margin of smoothness of the clipping function $c(\cdot) : R \times R \times R \to R$. The illustration of $c(\cdot)$ is drawn in Figure 7.12, and its expression is listed below

$$c(x,a,r) = \begin{cases} x & x \leq a - \frac{r}{2} \\ -\frac{1}{2r}(x - a - \frac{r}{2})^2 + a & a + \frac{r}{2} > x > a - \frac{r}{2} \\ a & x \geq a + \frac{r}{2} \end{cases} \tag{7.8}$$

The main purpose of this clip function is to clip the input x to a specified threshold a, while still maintaining first-order differentiability by introducing the margin of smoothness r. If $r = 0$, the clip function is no longer first-order differentiable. The overall effect on the optimal design problem is that, after σ_3 has been increased to $\sigma_{3,th}$, the optimization stops considering maximizing the randomness of joint 3 as an objective. This is to prevent the optimization from over-stretching the distribution range of joint 3 and ignoring the original objective. After this improvement, the joint 3 distribution range of the optimal points increases and is similar to that of the random points (See Figure 7.11). The generated optimal and random

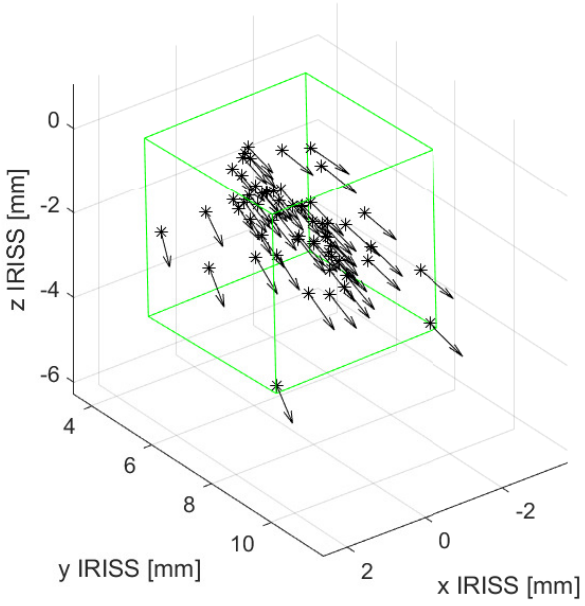
points are shown in Figure 7.13. Note that the optimal points have a more concentrated point spread than random points.

7.2.3 result

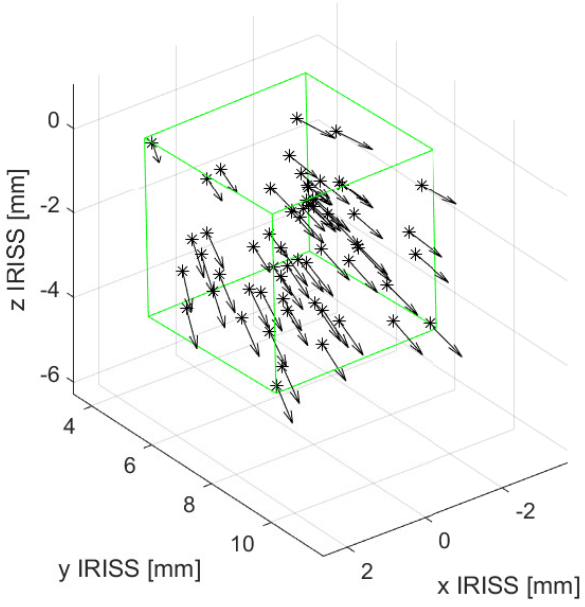
Instead of the nested scheme, the generic optimization scheme is employed for partial calibration since the CT parameter t_{mb} has already been given from the full calibration (Table 7.2). Also, the DH parameter of joint 1 and 2 is given from full calibration as well (Table 7.1). The identified parameters of partial calibration are shown in Table 7.4. The residual error of the validation points after calibration is shown in Table 7.5. As in the full calibration, the results for both random points and optimal points are similar.

See Figure 7.14, Figure 7.15, and Figure 7.16. It can be seen that, despite the initial quicker error convergence with increasing data point numbers, the optimal points eventually have slightly larger errors (weighted error difference $\sim 7\mu m$). This shows that the optimal design is better for small amount of calibration points but worse for large amount of calibration points.

This outcome may be due to the trade-off between information drawn from data points and the randomness of those data points. Ideally, suppose the model is correct, the optimally designed point would definitely have a larger fisher information matrix (in terms of D-optimality) and better parameter estimation precision. However, there is always some unmodeled error. In order to account for this unmodeled behavior, the calibration points should be spread wide enough (to increase randomness), whether in Cartesian space or in joint space, to capture all the behavior of the robot. When the calibration point number is small, the random calibration points might not have enough randomness to outperform the larger information obtained from the optimal points. However, with the increase in data points number, the randomness of the random points would eventually prevail. It can be seen that in Figure 7.4 the spread of random points in Cartesian space is larger than that of optimal points. This might be the reason why optimal points fail to outperform random



(a) optimal calibration points



(b) random calibration points

Figure 7.13: calibration points for partial calibration

Table 7.4: identified DH parameter deviation after partial calibration of surgical robot

calibration point	joint	3	4
optimal point	δd	-0.42	-
	$\delta\theta$	-	-
	δa	1.09	0.62
	$\delta\alpha$	1.45	2.17
	δb	N.A.	-0.44
	$\delta\beta$	N.A.	2.07
random point	δd	-0.42	-
	$\delta\theta$	-	-
	δa	1.07	0.62
	$\delta\alpha$	1.39	2.17
	δb	N.A.	-0.43
	$\delta\beta$	N.A.	2.07

[a] Unit for angle is in deg, length in mm.

[b] hyphen "-" means fixed parameter. Its value is 0.

Table 7.5: residual error of validation points after partial calibration of surgical robot

calibration point	error type	rms	max	std
optimal	position [μm]	59	108	23
	orientation [deg]	0.22	0.48	0.10
random	position [μm]	56	100	20
	orientation [deg]	0.21	0.49	0.10

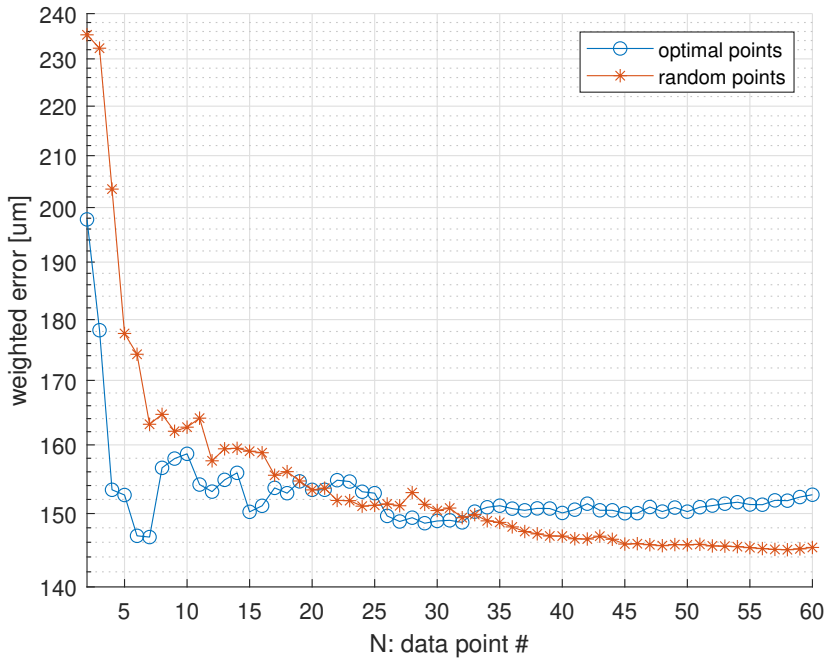


Figure 7.14: weighted error ($e_w = e_p + we_z$) vs N for partial calibration

points when the data point number is large.

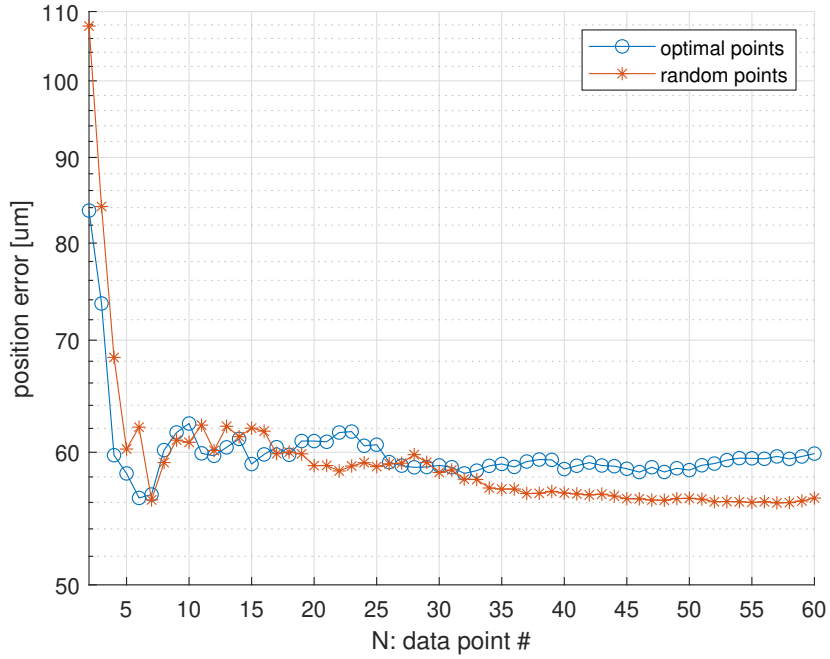


Figure 7.15: position error (e_p) vs N for partial calibration

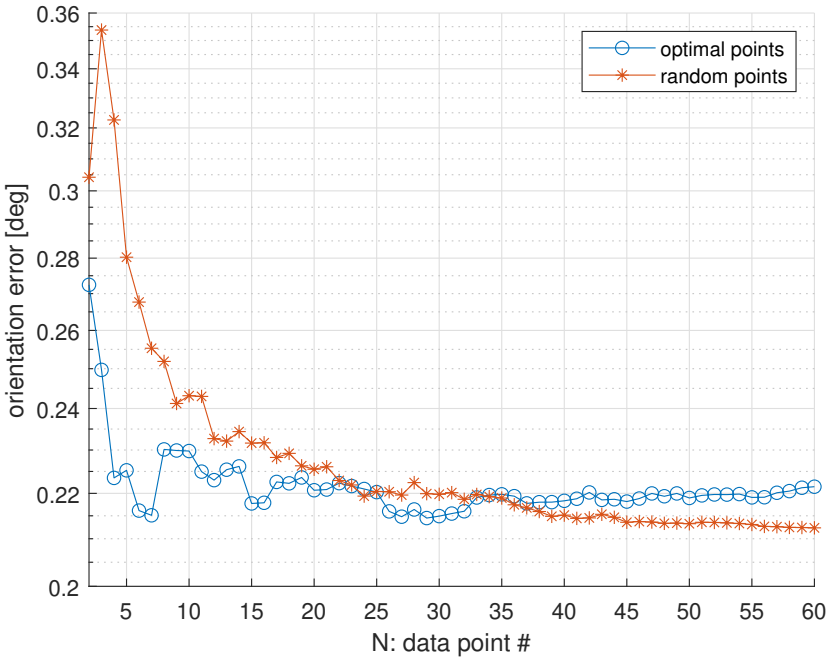


Figure 7.16: orientation error (e_z) vs N for partial calibration

REFERENCES

[AWR07] E. Abele, M. Weigold, and S. Rothenbücher. "Modeling and Identification of an Industrial Robot for Machining Applications." *CIRP Annals*, **56**(1):387–390, 2007.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*, pp. 385–390. Cambridge university press, 2004.

[Cra46] Harald Cramér. *Mathematical Methods of Statistics*, pp. 477–487. Princeton Univ. Press, Princeton, NJ, 1946.

[DH55] J. Denavit and R. S. Hartenberg. "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices." *Journal of Applied Mechanics*, **22**(2):215–221, 06 1955.

[DMB93] Keith L. Doty, Claudio Melchiorri, and Claudio Bonivento. "A Theory of Generalized Inverses Applied to Robotics." *The International Journal of Robotics Research*, **12**(1):1–19, 1993.

[GG09] Carol Fridericus Gauss and Carl Friedrich Gauss. *Theoria Motus Corporum Coelestium in sectionibus conicis Solem Ambientium*. Sumtibus Frid. Perthes et I.H. Besser, 1809.

[GH90] J.D. Gorman and A.O. Hero. "Lower bounds for parametric estimation with constraints." *IEEE Transactions on Information Theory*, **36**(6):1285–1301, 1990.

[GJK88] E.G. Gilbert, D.W. Johnson, and S.S. Keerthi. "A fast procedure for computing the distance between complex objects in three-dimensional space." *IEEE Journal on Robotics and Automation*, **4**(2):193–203, 1988.

[GM95] Christian Gourieroux and Alain Monfort. "Maximum Likelihood Estimation." In QuangTranslator Vuong, editor, *Statistics and Econometric Models*, volume 1 of *Themes in Modern Econometrics*, pp. 170–171. Cambridge University Press, 1995.

[GYN00] Chunhe Gong, Jingxia Yuan, and Jun Ni. "Nongeometric error identification and compensation for robotic system by inverse calibration." *International Journal of Machine Tools and Manufacture*, **40**(14):2119–2137, 2000.

[Hay83] Samad A. Hayati. "Robot arm geometric link parameter estimation." In *The 22nd IEEE Conference on Decision and Control*, pp. 1477–1483, 1983.

[HHN88] Berthold K. P. Horn, Hugh M. Hilden, and Shahriar Negahdaripour. "Closed-form solution of absolute orientation using orthonormal matrices." *J. Opt. Soc. Am. A*, **5**(7):1127–1135, Jul 1988.

[Jor06] Stephen J. Wright Jorge Nocedal. *Numerical Optimization*, pp. 563–594. Springer, New York, NY, USA, 2nd edition, 2006.

[Kay93] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*, volume 1, pp. 73–76. Prentice Hall, 1st edition, 1993.

[KB19] Kaveh Kamali and Ilian A. Bonev. "Optimal Experiment Design for Elasto-Geometrical Calibration of Industrial Robots." *IEEE/ASME Transactions on Mechatronics*, **24**(6):2733–2744, 2019.

[KJB16] Kaveh Kamali, Ahmed Joubair, Ilian A. Bonev, and Pascal Bigras. "Elasto-geometrical calibration of an industrial robot under multidirectional external loads using a laser tracker." In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4320–4327, 2016.

[LC98] E. L. Lehmann and George Casella. *Theory of Point Estimation*, p. 24. Springer New York, New York, NY, 2nd edition, 1998.

[LEV44] KENNETH LEVENBERG. "A METHOD FOR THE SOLUTION OF CERTAIN NON-LINEAR PROBLEMS IN LEAST SQUARES." *Quarterly of Applied Mathematics*, **2**(2):164–168, 1944.

[LHS08] Chris Lightcap, Samuel Hamner, Tony Schmitz, and Scott Banks. "Improved Positioning Accuracy of the PA10-6CE Robot with Geometric and Flexibility Calibration." *IEEE Transactions on Robotics*, **24**(2):452–456, 2008.

[LLW21] Lingxiao Li, Lu Li, Yanan Wang, Baolin Feng, and Guojiang Li. "Kinematics Redundancy Identification of Arbitrary Serial Robot." In *2021 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 956–962, 2021.

[Mar63] Donald W. Marquardt. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters." *Journal of the Society for Industrial and Applied Mathematics*, **11**(2):431–441, 1963.

[Mat12] Ravishankar Mathur. *An analytical approach to computing step sizes for finite-difference derivatives*. PhD thesis, University of Texas at Austin, 2012.

[MD00] M.A. Meggiolaro and S. Dubowsky. "An analytical method to eliminate the redundant parameters in robot calibration." In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 4, pp. 3609–3615 vol.4, 2000.

[Mis64] Richard von Mises. *Mathematical theory of probability and statistics*, pp. 427–429. Academic Press, New York, 1964.

[NB13] Albert Nubiola and Ilian A. Bonev. "Absolute calibration of an ABB IRB 1600 robot using a laser tracker." *Robotics and Computer-Integrated Manufacturing*, **29**(1):236–245, 2013.

[ODB12] Adel Olabi, Mohamed Damak, Richard Bearee, Olivier Gibaru, and Stephane Leleu. "Improving the accuracy of industrial robots by offline compensation of joints errors." In *2012 IEEE International Conference on Industrial Technology*, pp. 492–497, 2012.

[Olv14] Peter J. Olver. *Introduction to partial differential equations*, pp. 182–184. Springer, 2014.

[Rao92] C. Radhakrishna Rao. "Information and the Accuracy Attainable in the Estimation of Statistical Parameters." In Samuel Kotz and Norman L. Johnson, editors, *Breakthroughs in Statistics: Foundations and Basic Theory*, pp. 235–247. Springer New York, New York, NY, 1992.

[Sag09] Guillaume Sagnol. "Computing Optimal Designs of multiresponse Experiments reduces to Second-Order Cone Programming.", 2009.

[Sch93] Klaus Schröer. "Theory of Kinematic Modelling and Numerical Procedures for Robot Calibration." In Roger Bernard and S. Albright, editors, *Robot Calibration*, pp. 157–196. Chapman & Hall, London, 1st edition, 1993.

[WHR21] Franz-Georg Wieland, Adrian L. Hauber, Marcus Rosenblatt, Christian Tönsing, and Jens Timmer. "On structural and practical identifiability." *Current Opinion in Systems Biology*, **25**:60–69, 2021.

[WZF09] Jianjun Wang, Hui Zhang, and Thomas Fuhlbrigge. "Improving machining accuracy with robot deformation compensation." In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3826–3831, 2009.

[ZRH90a] H. Zhuang, Z.S. Roth, and F. Hamano. "A complete and parametrically continuous kinematic model for robot manipulators." In *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 92–97 vol.1, 1990.

[ZRH90b] Hanqi Zhuang, Zvi S. Roth, and Fumio Hamano. "Observability Issues in Kinematic Error Parameter Identification of Manipulators." In *1990 American Control Conference*, pp. 2287–2293, 1990.

[ZWH96] Hanqi Zhuang, Jie Wu, and Weizhen Huang. "Optimal planning of robot calibration experiments by genetic algorithms." In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pp. 981–986 vol.2, 1996.

[ZWR93] Hanqi Zhuang, Luke K. Wang, and Zvi S. Roth. "Error-model-based robot calibration using a modified CPC model." *Robotics and Computer-Integrated Manufacturing*, **10**(4):287–299, 1993.

[ZWR94] Hanqi Zhuang, Kuanchih Wang, and Z.S. Roth. "Optimal selection of measurement configurations for robot calibration using simulated annealing." In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 393–398 vol.1, 1994.