Motion planning and estimation problems in the space of rigid-body motions SE(3)

Huy Nguyen

Contents

Chapter 1

Introduction

A rigid body motion of an object is a motion that preserves the distance between any two points. Study on rigid body motion has long been an interesting field that attracts an extensive attention from many researchers in the scope of robotics. The research fields range from motion planning, mobile robot to computer vision and other related topics. Despite the fact that a number of theoretical treatments for working on rigid body motion have been developed for a long time, Lie theory has proved to be the most effective tool used to resolve many principal issues corresponding to this matter. To be specific, this research will address the crucial problems related to motion planning and estimation problems in the space of rigid body motions. These two problems are not new in the literature, however, by taking advantage of using the analytic tools derived from Lie group theory and Riemannian geometry, we aim at proposing new, general, fast approaches to address these problems.

To begin with, Chapter 2 presents an introduction to this research. We first provide some mathematical backgrounds involving two groups which are of particular interest to us in robotics *i.e.*the special orthogonal group $\boldsymbol{SO}(3)$ representing rotations and the special Euclidean group $\boldsymbol{SE}(3)$ representing rigid body motions. While this topic is very popular such that it can be found in many mathematic and robotic textbooks, it is also perhaps one of the most confusing topics. We will try to introduce this topic in a way that minimize mathematical prerequisites. In this Chapter, we also emphasize the importance of uncertainty in robotics which remains a great challenge that prevent the automation of many complex manipulation tasks such as fine assembly. Then we introduce some interesting problems involved to the presence of uncertainty in such robotic systems. Finally, we provide a review on sampling-based planners and trajectory planning with a special type of constraints named *kinodynamic constraints.*

In Chapter 3 the status of our research including some preliminary work that has been accomplished so far. The first work addresses the problem of planning fast, collision-free, trajectories in the spaces of three-dimensional rotations $\boldsymbol{SO}(3)$ and three-dimensional rigid body motions $\boldsymbol{SE}(3)$ under kinodynamic constraints in cluttered environments. Our main contribution consisted in extending the classical Time-Optimal Path Parameterization (TOPP) algorithm to $\boldsymbol{SO}(3)$. We integrated the algorithm into a plan-and-shortcut pipeline to yield a complete framework (and open-source implementation) for rigid-body motion planning under kinodynamic constraints. The results shows that our implementation could very efficiently find near time-optimal trajectories in a spacecraft maneuver problem. Another work is associating the uncertainty to the solution of hand-eye calibration problem. The distribution of the errors between the estimated hand-eye transformation and the ground truth is assumed to be approximated by a multivariate Gaussian distribution. We formulate the hand-eye calibration problem as a least-squares minimization problem, then utilize propagation of covariance methods from statistics theory to obtain the covariance matrix of the solution. The simulation result shows that our methods can deliver a very good approximation of the real covariance matrix of the hand-eye transformation.

Finally, Chapter 4 sketch the future plan for our research. We also give some brief discussions on how we plan to tackle these challenging but interesting problems.

Chapter 2

Literature Review

2.1 Lie Group Theory

Before addressing some problems by looking at rigid motions as a Lie group, in this Section we will briefly recall some mathematical definitions involving the rotation group $\boldsymbol{SO}(3)$ and the special Euclidean group $\boldsymbol{SE}(3)$, which are important for the subsequent discussions. More details on this topic can be found in [1], [2], [3].

2.1.1 Lie Groups and Lie Algebras

Group and Manifold

In mathematics, a group $\mathbf{G}$ is an algebraic structure consisting of a set of elements equipped with some binary operation (the group operation). The operation satisfies four conditions (*i.e.*closure, associativity, identity and invertibility) called the group axioms. For instances, the most familiar examples of simple groups are the integer numbers $\mathbb{Z}$ under the operation of addition and the special orthogonal group $\boldsymbol{SO}(3)$ under the operation of matrix multiplication.

A manifold M is a topological space where every point $p \in M$ locally resemble Euclidean space near each point. A good way to illustrate this idea is to consider the surface of the Earth as a manifold of dimension two. Although it is round, at a given point, on the small scales that we see, the Earth looks "flat" or a $\mathbb{R}^2$ euclidean space.

Tangent Space of a Manifold

A D-dimensional manifold is a smooth manifold embedded in the $\mathbb{R}^N$ space ($N \geq D$) if every point $p \in \mathbf{M}$ is contained by $\mathbf{U} \subseteq \mathbf{M}$, defined by $\phi : \Omega \longmapsto \mathbf{U}$, $\mathbb{R}^N \longmapsto \mathbf{M}$ where Ω contains the origin and is an open subset of $\mathbb{R}^N$.

A tangent space can be visualized as the vector space of the derivatives at point $p \in \mathbf{M}$ of all possible smooth curves that pass through p. A manifold $\mathbf{M}$ of dimension D in the $\mathbb{R}^N$ space (with $N \geq D$) has associated an N-dimensional tangent space for every point $p \in \mathbf{M}$. This tangent space of $\mathbf{M}$ at a point p is usually denoted as T_pM and has a dimension of D.

Lie Group

A group G is a Lie group when it is a smooth manifold and the group operations $(g,h) \longmapsto gh$ and $g \longmapsto g^{-1}$ are also smooth.

2.1.2 SO(3) and SE(3) as Lie Groups

The special orthogonal group $\boldsymbol{SO}(3)$ is a matrix Lie group representing rotations, and its associated Lie algebra, denoted $\boldsymbol{so}(3)$ with the matrix commutator as Lie bracket. The Lie algebra of $\boldsymbol{SO}(3)$ denoted by $\boldsymbol{so}(3)$ is given by the set of 3×3 real skew-symmetric matrices of the form

$$[\boldsymbol{\omega}] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \tag{2.1}$$

where $\boldsymbol{\omega} = (\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\omega}_3)$ is the axis-angle representation for rotation. Elements of $\boldsymbol{SO}(3)$ are given by the 3×3 real matrices $\boldsymbol{R}$ satisfying $\boldsymbol{R}^\top \boldsymbol{R} = \mathbf{I}$ and $det\boldsymbol{R} = 1$.

The special Euclidean group $\boldsymbol{SE}(3)$ representing rigid body motions is defined to be:

$$\begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}, with\boldsymbol{R} \in \boldsymbol{SO}(3), \boldsymbol{t} \in \mathbb{R}^3. \tag{2.2}$$

A minimal representation of this transformation is defined by the corresponding Lie algebra $\boldsymbol{se}(3)$ which is the tangent space of $\boldsymbol{SE}(3)$ at the identity. The algebra elements are 6-vectors $(\boldsymbol{v}, \boldsymbol{\omega})^\top$ where $\boldsymbol{\omega} = (\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\omega}_3)$ is the axis-angle representation for rotation, and $\boldsymbol{v}$ is a rotated version of the translation $\boldsymbol{t}$.

Elements of the $\boldsymbol{se}(3)$ algebra can be mapped to the $\boldsymbol{SE}(3)$ group using the exponential mapping exp:

$$\exp([(\boldsymbol{v}, \boldsymbol{\omega})]) \overset{\text{def}}{=} \begin{bmatrix} \exp([\boldsymbol{\omega}]) & \mathbf{V}\boldsymbol{v} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}. \tag{2.3}$$

Here,

$$\exp([\boldsymbol{\omega}]) = \mathbf{I} + \frac{\sin \|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|}[\boldsymbol{\omega}] + \frac{1 - \cos \|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|^2}[\boldsymbol{\omega}]^2 \tag{2.4}$$

is the well-known *Rodigues' formula*,

$$\mathbf{V} = \mathbf{I} + \frac{1 - \cos \|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|^2}[\boldsymbol{\omega}] + \frac{\|\boldsymbol{\omega}\| - \cos \|\boldsymbol{\omega}\|}{\|\boldsymbol{\omega}\|^3}[\boldsymbol{\omega}]^2 \tag{2.5}$$

, and [.] is an operator which maps a 3-vector to its skew-symmetric matrix (we also use this operator to turn $(\boldsymbol{\omega}, \boldsymbol{v})^\top$ into a 4×4 member of Lie algebra $\boldsymbol{se}(3)$). Since the exponential mapping exp is surjective, there exists also an inverse relation which is the logarithm mapping log.

2.2 Uncertainty in Robotics

One key issue in a robotic assembly system is dealing with uncertainties that occur in robot motion and sensory information. In many cases such uncertainties may lead to the failure of executing an assembly plan, especially in precision assembly tasks. Uncertainty may arise from five different sources [4]:

-**Environment**: While the working spaces in industrial assembly lines are well-structured, the degree of uncertainty in the working spaces of fine assembly tasks and dynamic environments is considerably high.

-**Sensors**: Even though the rapid development of sensor's manufacturing and computing technologies has brought significant impacts on robotics, sensors are still limited in what they can "sense". They are subject to noise which makes the perceiving measurements perturbed in an unpredictable way. Thus, it limits the information that can be extracted from the sensed information. Also, range and resolution of sensors are still limited by manufacturing and computing technologies.

-**Robots**: The technology of producing robot manipulator has helped to ease this factor. Many manipulators, such as position-controlled industrial arms, are very accurate. However, other low-cost manipulators maybe very inaccurate.

-**Models**: Models are the digital descriptions of the physical world. Essentially, we can only model partially the robot and its environment. Model errors are even worse when robots have to work in unstructured and dynamic environments.

-**Computation**: The amount of computation that can be carried out is limited. Due to the limitations in computing power, most robotics systems must sacrifice accuracy to enable robots response in real-time.

Usually, such uncertainty has not been a major concern for existing applications in robotics and particularly in assembling tasks. However, since there are increasingly more robots operate in unstructured and dynamic environments, the ability to cope with uncertainty is critical for building successful robotic assembling systems.

2.2.1 Uncertainty Presentation and Manipulation for Use in Assembly Tasks

As mentioned earlier, in order to execute successfully the assembling tasks in the presence of uncertainty, a robotic assembly system must be equipped with the abilities of explicitly representing and manipulating the uncertainties. The key challenge in choosing a representation of the pose and its associated uncertainty is that representations either have singularities or constraints; this results from the fact that rotation variables are not vectors but rather members of a non-commutative group. This also causes many difficulties in manipulating poses and their associated uncertainties. As a result, this problem has draw extensive focus from many researchers over the last decades.

In [5], [6], uncertainty is simply represented by worst-case bounds which includes all possible errors (set-oriented representation). This approach is simple but it usually results in conservative estimates which make it difficult to apply in decision-making process. The probabilistic approaches, however, use the calculus of

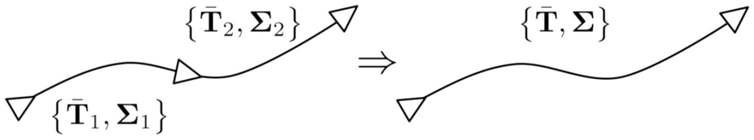

$\{\bar{\mathbf{T}}_2, \mathbf{\Sigma}_2\}$
$\{\bar{\mathbf{T}}_1, \mathbf{\Sigma}_1\}$
$\Rightarrow$
$\{\bar{\mathbf{T}}, \mathbf{\Sigma}\}$

Figure 2.1: Combining a series of two poses, each with its associated uncertainty into a single pose

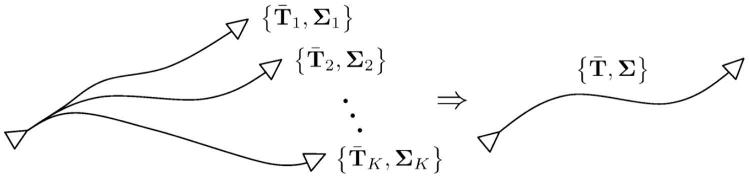

$\{\bar{\mathbf{T}}_1, \mathbf{\Sigma}_1\}$
$\{\bar{\mathbf{T}}_2, \mathbf{\Sigma}_2\}$
$\{\bar{\mathbf{T}}_K, \mathbf{\Sigma}_K\}$
$\Rightarrow$
$\{\bar{\mathbf{T}}, \mathbf{\Sigma}\}$

Figure 2.2: Combining K pose estimates with their associated uncertainty into a single fused estimate

probability theory and assigns probabilities to all potential positions of the object. Real location of the object now becomes a probability distribution over the space [7], [8]. As opposed to the set-oriented representation, the probabilistic representation can make use of probability theory and, thus, provide more uncertainty manipulation methodologies (e.g. propagation, fusion, etc.). In [9],[10], [11], [12], this approach has been utilized to establish a theoretical basis for a popular subfield of the *simultaneous localization and mapping* (SLAM) problem [13].

Recently, this probabilistic approach has been further investigated in [14],[3], [15], [16] where they provided a rigorous treatment of representing and propagating uncertainty on Lie groups (Section 2.1). As we will see, the advantage of this technique is that it not only avoids the need to enforce constraints on rotation variables but also is free of singularities. In this research, we are especially interested in [16] which provided a straightforward, pratical approach for associating uncertainty with transformation matrices. By representing a mean as a 4 × 4 transformation matrix and its associated uncertainty as a 6 × 1 perturbation (with an associated 6 × 6 covariance matrix), the paper attained a simple but useful and easy-to-implement representation:

Random variables on $\boldsymbol{SE}(3)$ is defined as

$$\boldsymbol{T} := \exp_{\boldsymbol{SE}(3)}(\boldsymbol{\xi})\bar{\boldsymbol{T}}, \tag{2.6}$$

where $\bar{\boldsymbol{T}}$ is a “large”, noise-free value (the mean), and $\boldsymbol{\xi} \in \mathbb{R}^6$ is a “small”, noisy perturbation. The small perturbation variable is usually defined to be zero-mean Gaussian variable $p(\boldsymbol{\xi}) = \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\boldsymbol{\sigma}$ is a 6×6 covariance matrix. The authors also provided techniques to manipulate poses and associated uncertainties; for instances, the method of compounding two poses with their associated uncertainty, as illustrated in Figure 2.1, and the fusing of several pose estimates, as illustrated in Figure 2.2. More details on these techniques can be referred at [16].

2.2.2 Some Interesting Problems in Presence of Uncertainty

Since uncertainties usually involve in all components of an assembling system (e.g. robot model and control, estimation object of interest), the ability to estimate these uncertainties is critical to build a robust assembling system. Moreover, being able to obtain these information may greatly contribute to the system knowledge of the surrounding space and the system itself, therefore, enable the system to cope with many complex tasks in various types of environment. In this Section, we give some discussions on two interesting problems that we want to tackle: *How to estimate the uncertainties in an assembling system?* and *How to accomplish the assembly tasks in the presence of uncertainties in the system?*

Estimate the uncertainties in an assembling system

Without loss of generality, in this research, we consider a typical robotic system shown in Fig. 2.3. The coordinate systems $\boldsymbol{c}$, $\boldsymbol{t}$, $\boldsymbol{b}$ and $\boldsymbol{o}$ represent the system of camera, robot end effector, robot base and object/calibration board, respectively.

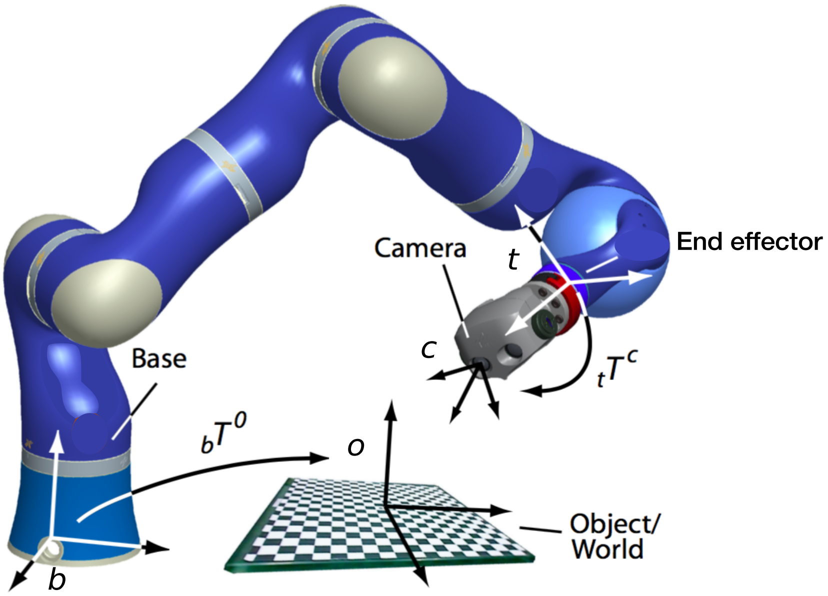


Figure 2.3: Coordinate systems of an assembling system

As from Figure 2.3, ${}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{t}}$ is the homogeneous transformation relating the pose of the base frame $\boldsymbol{b}$ to the pose of the end effector frame $\boldsymbol{t}$. ${}_{\boldsymbol{c}}\boldsymbol{T}^{\boldsymbol{o}}$ is the homogeneous transformation relating the pose of the camera frame $\boldsymbol{c}$ to the pose of the object/board frame $\boldsymbol{o}$. ${}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{t}}$ results from the calibrated forward kinematic model of the robot, the encoder readings of every joint, and possibly its control parameters. With the current technology and advancement of producing robot manipulators, this transformation is not a major concern as its error is relatively small.

The remaining unknown transformations are ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$, the rigid pose of the camera frame $\boldsymbol{c}$ with respect to the robot end effector frame $\boldsymbol{t}$, and ${}_{\boldsymbol{c}}\boldsymbol{T}^{\boldsymbol{o}}$, the rigid pose of the object $\boldsymbol{c}$ relative to the camera frame. Since the errors of these two transformations are usually large, there is a strong need for estimating the poses ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$, ${}_{\boldsymbol{c}}\boldsymbol{T}^{\boldsymbol{o}}$ along with their associated uncertainties. The reasons are of twofolds:

1. to map the sensor measurements into the robot base/world frame. For instance, methods proposed in [16] can be used to compound the series of poses (each with its associated uncertainty) from robot base through the hand and camera to the object. Thus, the transformation of the object with respect to the robot base and its associated uncertainty can be obtained.

2. to allow the system to predict the camera/objects pose(s). Since the working spaces in some cases are highly constrained, this would help the system to perform better in term of avoiding collisions with the environment.

The discussions on the problems of determining these two transformations along with their associated uncertainties are provided after this.

Hand-Eye Calibration Problem

Whenever a sensory device is attached on a robot hand, the problem of determining the relationship between the sensor and the hand *i.e.*the transformation ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$, is crucial. This problem is usually referred as *hand-eye calibration problem*. Even though there are extensive literature on the solution methods for this problem, estimating the associated uncertainty of the resulting hand-eye transformation has not been a major concern for existing applications where measurement noises are relatively small or the tasks do not require a very high accuracy in calibration. Here we give a brief review on this problem.

The hand-eye calibration problem is commonly represented of the form: $\boldsymbol{AX} = \boldsymbol{XB}$. This formulation, in fact, directly reproduces the rigid transformations in the loop base-end effector-camera-object/world-base:

$$ {}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{t}}\, {}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}} = {}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{o}}\, {}_{\boldsymbol{o}}\boldsymbol{T}^{\boldsymbol{c}} \tag{2.7} $$

Due to the fact that ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$ is more often required than ${}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{o}}$, most approaches eliminate the latter by writing Eq. 2.7 at two different instants i and j yielding the well-known hand-eye equation:

$$ {}_{\boldsymbol{b}}\boldsymbol{T}_j^{\boldsymbol{t}^{-1}}\, {}_{\boldsymbol{b}}\boldsymbol{T}_i^{\boldsymbol{t}}\, {}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}} = {}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}\, {}_{\boldsymbol{o}}\boldsymbol{T}_j^{\boldsymbol{c}^{-1}}\, {}_{\boldsymbol{o}}\boldsymbol{T}_i^{\boldsymbol{c}} \Rightarrow \boldsymbol{AX} = \boldsymbol{XB}, \tag{2.8} $$

where the unknown $\boldsymbol{X}$ represents the fixed homogeneous transformation between camera and the end effector ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$, $\boldsymbol{A_{ij}} = {}_{\boldsymbol{b}}\boldsymbol{T}_j^{\boldsymbol{t}^{-1}}\, {}_{\boldsymbol{b}}\boldsymbol{T}_i^{\boldsymbol{t}}$ and $\boldsymbol{B_{ij}} = {}_{\boldsymbol{o}}\boldsymbol{T}_j^{\boldsymbol{c}^{-1}}\, {}_{\boldsymbol{o}}\boldsymbol{T}_i^{\boldsymbol{c}}$. Note that another representation of this problem consists of the form $\boldsymbol{AX} = \boldsymbol{YB}$ which can also be constructed similarly. However, at right now, this is not our particular interest.

In the case that there is no error in the measurements, $\boldsymbol{AX} = \boldsymbol{XB}$ has a one-parameter family of solutions (as first shown by Shiu and Ahmad [17]). Two pairs of $(\boldsymbol{A}_i, \boldsymbol{B}_i)$ satisfying certain constraints are required in order to obtain a unique solution. Unfortunately, in actual sensor calibration applications, noises usually exist in the measured values of $\boldsymbol{A}$ and $\boldsymbol{B}$. Therefore, a more practical approach is to make several measurements $(\boldsymbol{A}_1, \boldsymbol{B}_1), (\boldsymbol{A}_2, \boldsymbol{B}_2), ..., (\boldsymbol{A}_k, \boldsymbol{B}_k)$, and to find an $\boldsymbol{X}$ that is a best fit to the k matrix equations. As a result, this problem is then constructed as a *least-squares optimization problem*, where the objective function is given by:

$$\min_{\boldsymbol{X} \in \boldsymbol{SE}(3)} \sum_i \|\boldsymbol{A}_i \boldsymbol{X} - \boldsymbol{X} \boldsymbol{B}_i\|^2, \tag{2.9}$$

where $\|.\|$ denotes the Frobenius matrix norm, i.e.,$\|A\| = tr(\boldsymbol{A}\boldsymbol{A}^\top)$ for a matrix $\boldsymbol{A}$.

The methods to solve this problem consist of two forms: separable closed-form solutions, simultaneous and iterative solutions. The main concept of separable closed-form solutions [17],[18], [19], [20] is its use of closed form least squares fitting. They calculated the rotation part first, and then the translation part using the rotation result. These methods often yield uncomplex, fast, but error-prone formulations, since rotation estimation errors propagate to the translational part.

On the other hand, the simultaneous solutions for $\boldsymbol{AX} = \boldsymbol{XB}$, therefore, were created in order to minimize these errors. Referring back to Eq. (2.9), there are several choices of $\|.\|$ available. For example, Strobl *et al.*[21] and Ha *et al.*[22] defined $\|.\|$ as

$$\|\boldsymbol{P} - \boldsymbol{Q}\|^2 = \|\boldsymbol{R}_p - \boldsymbol{R}_Q\|^2 + \zeta \|\boldsymbol{t}_P - \boldsymbol{t}_Q\|^2 \tag{2.10}$$

where $\boldsymbol{R}_P, \boldsymbol{R}_Q \in \boldsymbol{SO}(3)$ and $\boldsymbol{t}_P, \boldsymbol{t}_Q \in \mathbb{R}^3$ are the rotations and the translations of $\boldsymbol{P}, \boldsymbol{Q} \in \boldsymbol{SE}(3)$ respectively, $\|.\|$ denotes the Frobenius norm and ζ is a weighting factor for the translation error. In these papers, Newton or Newton-type iterative algorithms are developed to optimally reduce actual system error and allow for a natural weighting of the rotational and translational components.

As extensive works have been investigating this calibration problem, many reliable solutions have been proposed and offer accurate estimations. However, to our knowledge, none of these works has associated the uncertainty with the solution of $\boldsymbol{X}$. As we pointed out earlier, this information is very useful when we need to estimate the object pose and its associated uncertainty. Moreover, the associated uncertainty of the solution can also be used as an evaluation criteria for the calibration process. Therefore, in 3.2 we will presents our approach to associate the uncertainty (through a covariance matrix) with solution of $\boldsymbol{X}$.

Object Pose Estimation in the Presence of Uncertainty

To estimate ${}_{\boldsymbol{c}}\boldsymbol{T}^{\boldsymbol{o}}$, the object pose with respect to the camera frame, there are extensive, fully-developed methods have been proposed in computer vision community. However, when robots operate in a complex environment, uncertainties are unavoidable. Since such uncertainties may lead to the failure of the system, it is important to model the uncertainties of pose estimates provided by commonly used pose estimation algorithms. This problem has draw a lot of attention and has been well-explored in the literature. Here we give a brief review on determining the uncertainties of the pose estimates resulting from the Iterative Closest Point (ICP) algorithm [23]- one of the most well-known pose estimation algorithms.

With the assumption that the object points are correctly matched to the scene points, one of the common approaches is to formulate the uncertainty as isotropic Gaussian noise on the points. Based on this, one can derive analytically the distribution of the estimated rotation. Also, the same concept was applied to derive the pose uncertainty in the context of surface registration using normal-projection ICP [24]. As is well-known, ICP is unable to distinguish between poses with degenerate degrees of freedom. To deal with this, one can introduce a registration index to indicate the near degenerate configuration. In this case, high uncertainty is usually assigned to the near degeneracy.

In [25], it is shown that the covariance of the result of a minimization over a quadratic error function is dependent on the Hessian of the error function. As points matched in ICP algorithm are determined from a nearest-neighbor search, the above result can be applied to the ICP algorithm to find the covariance of the result. The authors in [26] argue that the shape of the error function computed from a given set of observations can not be used to explain the the covariance of the estimated pose an ICP algorithm. Instead they present another analytic expression for the covariance of an ICP pose estimate based on the covariance of observations.

Covariance matrix of the pose estimate has proved to be very useful to evaluate the stability of the ICP algorithm [27]. By applying the Principal Component Analysis (PCA) method on covariance matrix of the pose estimate, one can also select more suitable sampling points, hence, can improve the performance of the ICP algorithm.

Manipulation Planning in the Presence of Uncertainty

A common assumption in planning algorithms for robotic manipulation tasks is that the environment knowledge is perfectly provided beforehand, *i.e.*the geometry and pose of objects are commonly well-defined. This assumption remains true to many sophisticated assembly lines where manipulators carry out the identical task day-by-day in highly structured environments. However, when robots operate in dynamic changing environments, the assumption of well-defined static environments does not hold. For instance, objects are usually moved around due to interactions between robots and the environments; hence, introducing uncertainty into the pose of the objects with respect to the robot. Sensors then can be used to re-localize the objects, however resulting estimations usually are never perfect. Quite often such uncertainties may lead to the failure of executing the task, especially in precision assembly tasks. As depicted in Figure 2.4, a failure may arise due to the presence of uncertainty. Once such uncertainties can be estimated, the knowledge of the surrounding space and the systems can be significantly enhanced based on the collected information. Making a right use of this information will enable the systems to cope with a wide range of more complex tasks in different kinds of environment.

One common solution is to reduce the uncertainties of the estimations of the objects by making use of more accurate sensors *i.e.*tactile, force/torque sensors (these sensors however provide only local views, and possibly deform the objects). For example, Figure 2.5 illustrates a technique based on the sense of touch to improve

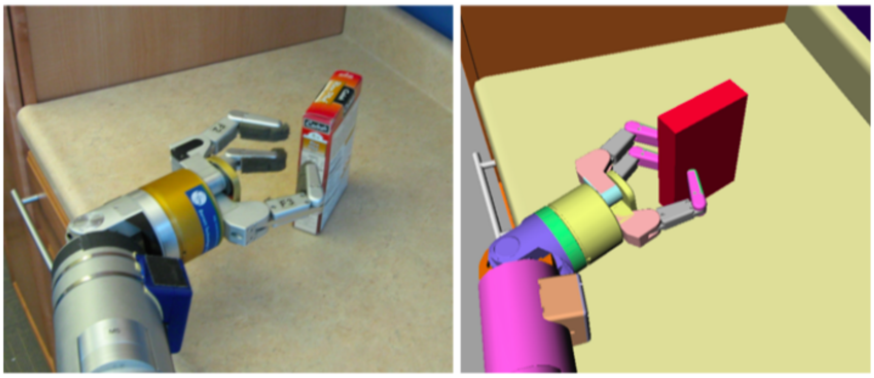


Figure 2.4: A grasp fails due to uncertain object pose estimation.

the object estimation. In the last decade, there has been increasing interest in Bayesian state estimation for the tactile object localization problem [28],[29],[30]. These methods estimate the probability distribution over all possible states (the posterior), which captures the uncertainty resulting from noisy sensors, inaccurate object models or uncertainty occurring due to manipulating process. However, the computational complexity of the problem commonly appears as a nature of these methods. Complexity goes up exponentially with the number of DOFs and the size of the initial uncertainty region. Therefore, many existing approaches simplify the problem by limiting the number of DOFs and/or initial uncertainty. In [31], Petrovskaya *et al.*proposed an efficient approach, termed Scaling Series, that approximates the posterior by particles. It performs the estimation by successively refining the high probability region and scaling granularity of estimation from low to high. Their approach does not utilize any special properties of the manipulated objects and can be easily applied to any object represented as a polygonal mesh. There are also a number of approaches that combine visual and tactile information to refine the model of an *unknown* object [32], [33]. This refined model will then allow the planning of more complex manipulation tasks. As pointed out in [32], in this area, there are a number of open questions: "How can tactile feedback be interpreted to choose an appropriate corrective action independent of the object, the task and environment? and How can visual and tactile information be fused in the controller?"

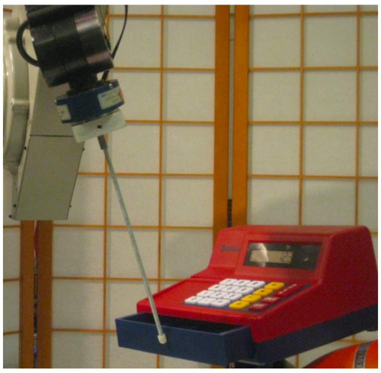
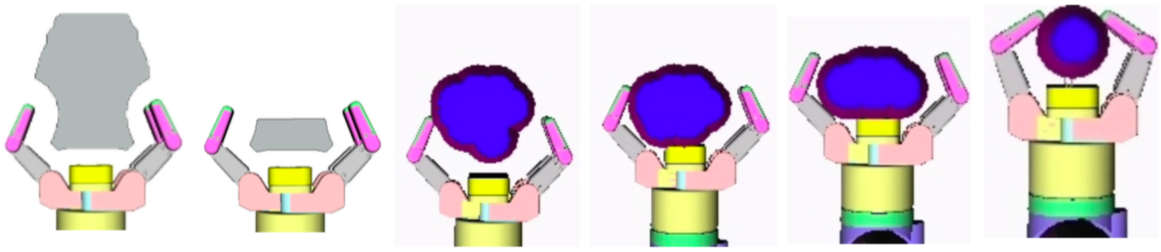


Figure 2.5: A technique based on the sense of touch can be used to improve the object estimation [31].

On the other hand, another solution is to develop a fully flexible, robust planning algorithm that can operate successfully and stably in the presence of uncertainty. The idea of motion planning under the presence



(a) Capture region (b) Object poses after pushing (c) Uncertainty (d) Push grasp (e) Push grasp (f) Push grasp

Figure 2.6: Push-grasping method presented in [34]. (a) The capture region of a for a rotationally symmetric bottle for a push-grasp. Every point corresponds to a bottle position where the coordinate frame of the bottle is at its center. (b) Uncertainty region of the object after the push, before closing the fingers. (c-f) A push-grasp funneling the uncertainty into the hand.

of uncertainty is not a new problem to be considered in robotics. One of the very first papers on this topic can be referred back to the seminal work of Lozano-Perez *et al.*[35] on *preimage planning* framework. A preimage is defined as an area of configuration space where a certain motion command can guarantee to archive a given goal. The algorithm use the preimage as a building block to produce a sequence of commands that guarantee to succeed in the presence of uncertainty. However, authors in [36],[37] have shown that to construct preimages is very computationally expensive. These early works then led to the development of more efficient and general planning algorithms based on the idea of probabilistically sampling a robot's configuration space. Recent works presented in [38] and [39] are able to incorporate sensor uncertainty into the planning stage in high-dimensional problems. Other approaches can be referred to *non-prehensile manipulation* where non-prehensile actions, such as pushing, flipping, squeezing, and more others, are used to manipulate/reduce the uncertainty. As depicted in Figure 2.6, [34] has shown that the system can use some pushing action primitives to reduce the uncertainty of the object pose. Another non-prehensile manipulation primitive, which is toppling, is also used in [40]. The system introduced in [41] tries to rotate an object on the support surface, before grasping it. However, most work in this topic makes assumptions on object geometry, such as symmetry, which makes their methods not generalizable. Moreover, they concerns only on finding feasible solutions, as opposed to optimal solutions.

2.3 Motion planning

In this Section, we will first provide a brief review on motion planners. We then will highlight the importance of the sampling-based planners that use randomized search strategies to explore sampling spaces. Next, two main streams of sampling-based planners which are PRM [42] and RRT [43] will be presented. Finally, we introduce the *motion planning in* $\boldsymbol{SE}(3)$ problem where we aim to plan fast, collision-free, trajectories in the spaces of three-dimensional rotations $\boldsymbol{SO}(3)$ and three-dimensional rigid body motions $\boldsymbol{SE}(3)$ under "kinodynamic" constraints in cluttered environments.

2.3.1 Sampling-based planners

Over the past decades, motion planning has been widely used in robotics and automation. Although it first appeared as a field of robotics, motion planning has also been successfully applied to numerous of fields aside from robotic applications[44].

Path planning, a purely geometric process which only concerns on finding a collision-free path regardless of the feasibility of the path, is perhaps the most basic form of the problem. For example, the algorithm has to find a collision-free path for a robot to move from its initial position and orientation (initial configuration) to the goal position and orientation (goal configuration). In this case, due to the fact that the robot workspace has low dimension *i.e.*, $\mathbf{R}^2$ or $\mathbf{R}^3$, solving this problem can be done by using simple methods *i.e.*swept volume. However, when different types of robots (e.g. many-DOFs robots) are involved and/or kinodynamic constraints are considered, the problem becomes much more complex.

In 1979, Lozano *et al.*[45] introduced a revolutionary idea of planning in the robot *configuration space* or $\mathcal{C}$-space. It was then formally described in [46] in 1983. By representing a robot stage as a point in the $\mathcal{C}$-space (e.g. a vector contains all the joint angles of the robot), as depicted in Figure 2.7, it simplifies a motion planning problem for a robot into a motion planning problem for a point in $\mathcal{C}$. This idea is very general that it can be applied to many kinds of problem involving different types of robots. It therefore has become a foundation for motion planning theory.

Many algorithms has been proposed for solving an motion planning problem. However, early efforts such as works in [47], [45], [48], [49], [50] and [51] are usually computationally expensive, therefore, not capable for planning in high-dimensional $\mathcal{C}$-spaces. Sampling-based algorithms, hence, were developed with the aim of being able to overcome the complexity of robot planning algorithms for high-dimensional problems. This randomized approach has demonstrated their efficacy for solving difficult problems. Nevertheless, due to randomness, sampling-based planners cannot guarantee the completeness (i.e not guaranteed to find a solution if one exists). Instead, they possess a weaker notion of completeness, *probabilistic completeness* [52], which guarantees to return a solution whenever one exists, given sufficient runtime of the algorithm.

Sampling-based planners can be classified into two families: probabilistic roadmaps (PRM) and rapidly-exploring random trees (RRT). Next, some background on these two approaches will be reviewed.

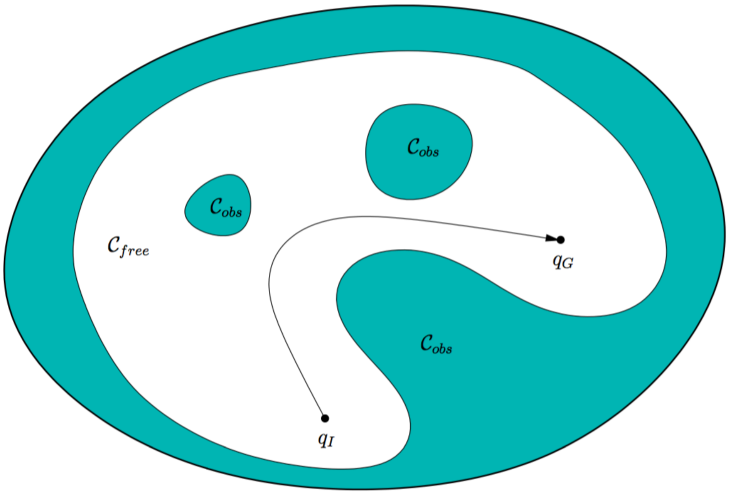


Figure 2.7: The basic motion planning problem is conceptually very simple using $\mathcal{C}$-space ideas. The task is to find a path from $\boldsymbol{q}_{\text{init}}$ to $\boldsymbol{q}_{\text{goal}}$ in $\mathcal{C}_{\text{free}}$. The entire blob represents $\mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{obs}}$ where $\mathcal{C}_{\text{obs}}$ denotes obstacle space and $\mathcal{C}_{\text{free}}$ represents the free space

2.3.2 Probabilistic roadmap (PRM)

The basic idea behind PRM is to generate a probabilistic roadmap to represent the connectivity of the free space [42]. PRM consists of two phases which are construction phase and query phase. The construction phase takes random samples from the $\mathcal{C}$space for certain amount of time. The planner then checks and collects samples that are in $\mathcal{C}_{\text{free}}$ space (discards ones that lie in obstacle space), and use a local planner try to connect them together. Once the roadmap is built, the query phase occurs next. In the query phase, the planner will define initial and final configurations, then attempt to connect them to the roadmap. As a result of maintaining the roadmap, PRM is able to solve problems where several queries involving the same system. It is also known as a multiple-query method. Most of the planning time is spent on generating the roadmap; a query, hence, can be found very quickly.

As a good roadmap is the core of a PRM planner, various algorithms have been proposed to improve sampling and making connections in roadmaps. In [42], vertex enhancement approach gives more effort on vertices that were difficult to connect to other vertices in the roadmap construction algorithm. Authors in [53], [54] showed that it is sometimes better to sample along the boundary of $\mathcal{C}_{\text{free}}$. Likewise, Gaussian sampling strategy proposed in [55] manages to increase the probability of sampling around obstacles. Bridge-test sampling [56] uses three samples along a short segment attempt to identify narrow corridors in the environment. [57], [58] introduced medial axis sampling methods which force the samples to be as far from the boundary as possible.

Nowadays PRM planners are still of great interest. Some analysis of PRM planners are provided in [59], [60]. Further reading on the probabilistic foundations of PRM can be found in [61]

2.3.3 Rapidly-exploring Random Tree (RRT)

The rapidly-exploring random tree (RRT) is another class of sampling-based planners,i.e. single-query planners. It was first proposed in [43] as a new randomized data structure for motion planning problem. RRT planners can easily handle various systems involving different contraints, and have been widely used in robotic motion planning.

In most of RRT planners, a tree rooted from start configuration is incrementally grown towards goal configuration. Once a new sample is selected, if it lies in $\mathcal{C}_{\text{free}}$ space, a connection is attempted to the nearest vertex of the tree. This new sample and its connection are added to the current tree if the connection lies entirely in $\mathcal{C}_{\text{free}}$ space and satisfies all the constraints. In single-query problems, RRT planners are usually faster compared to PRM planners since they do not need preprocessing. Another important property of RRT planners is that it introduces Voronoi bias, i.e. the tree tend to expand towards large unsearched areas. Therefore, the planner can efficiently and quickly explore high-dimensional spaces.

To improve the performance of RRT planners, it is suggested that the search can be performed in bidirectional directions. This strategy constructs *two* trees instead of one, i.e one tree grown from start configuration and another grown from goal configuration. The solution is found when the two trees can be successfully connected. There exist many other approaches to improve performance of the planners [62], [63], [64], [65], [66], [67], [68]. However, the recommended choice mainly depends on the characteristics of problem to be solved. Some analyses of RRT planners can be found in [69] and [70].

2.3.4 Some Discussions on Sampling-based Planners

It is admitted that both probabilistic roadmap planners and RRT-based planners have been widely applied in various practical aspects of science with a high efficiency. There are still a number of problems which arise naturally in these sampling-based approaches, but cause serious effects to these planners' performance. The report is carried on with a discussion sketching out some issues along with making several suggestions to deal with them.

Undoubtedly, the choice of distance metric plays a crucial role in how well an RRT-based planner can perform in reality. The planner will hardly reach the set target in case the distance metric are not accurate enough to indicate the cost-to-go. However, the challenge of developing an ultimate metric which can display the true cost-to-go is considered as hard as of working on the original problem [71]. In regard of this issue, [72] proposed an approach in order to enhance the performance of the planner. In general, the method is based on collecting some additional information, for instance, constraint violation frequency while exploring the surrrounding space and utilize these additional information in order to diminish as much as possible the sensitivity of the distance metric. On the other hand, an comparative analysis on comparing and assessing the differences from using a range of metrics has been made in [73].

In addition, narrow passage, a conventional matter of sampling-based planners, is also seen as an important issue that needs to be discussed. Generally speaking, a narrow passage is a small-volume region in $\mathcal{C}_{\text{free}}$ which can appear either as the physical narrow passage in the robot' s working environment or from the joint limits of the robot its self. Obviously, it is considered a tough challenge to have a random configuration in a narrow passage because of the small volume. Since the configuration space can not be explicitly described, it is hard to ascertain if the sampled configuration is in a narrow passage. However, the local planner may still unable to connect the sampled configuration with the tree even when it actually rises from a narrow passage. As a result, the performance of the sampling-based planners will be considerably affected when a solution path for a query $(qinit, qgoal)$ need to encounter a narrow passage. A demonstration of a narrow passage in a configuration is illustrated in Figure 2.8.

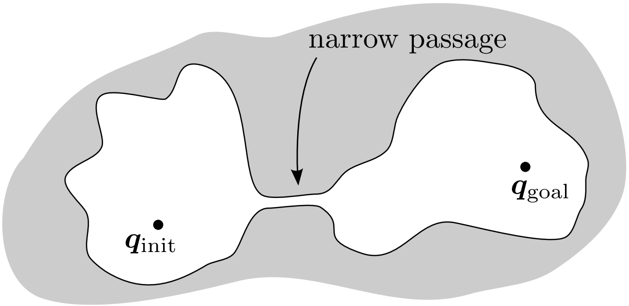

narrow passage
$\boldsymbol{q}_{\text{goal}}$
$\boldsymbol{q}_{\text{init}}$

Figure 2.8: $\mathcal{C}_{\text{free}}$ is represented as the white area in the figure and $\mathcal{C}_{\text{obs}}$ is represented as the grey area. Here $\mathcal{C}_{\text{free}}$ comprises two large volumes containing $\boldsymbol{q}_{\text{init}}$ and $\boldsymbol{q}_{\text{goal}}$ which are connected together by a narrow passage.

A number of various experiments and heuristics have been carefully conducted to enhance the general performance of the families of planners when they confront different narrow passages. The authors of [74] proposed a modified probabilistic roadmap planner which allows to create a new configuration only in the situation that it passes the *bridge test*. Many RRT-based planners mentioned in the literature operates to appoint a weight w to each of the nodes of the tree. Besides, it also chooses a suitable node to be extended with an expected probability of $1/w$. The weight w of a node indicates the possibility of how much the surrounding region of the node has been investigated. In that way, it can be concluded that the area which has been less explored is likely to be an area created by a narrow passage. Authors in [75] and [76] utilize information of narrow passages in the physical workspace as the purpose of guiding the sampling in the configuration space. The planner in [55] has followed Gaussian sampling distribution method to make the sampling as close as possible to the obstacles in $\mathcal{C}$-space due to the fact that these configurations can contribute to narrow passages.

2.3.5 Trajectory planning with kinodynamic constraints

As mentioned earlier, there are numerous motion planners that can handle problems with different types of constraints. When planning motions subject to only *kinematic constraints* (e.g. robot joint limits and obstacle avoidance), solutions are purely geometrical objects, or in other words, path descriptions contain no time information. However, to execute the robot motions after planning, time information along the path is required. Other constraints which include higher-order time derivatives of the robot configuration (e.g. bounds on the robot velocity, acceleration, and force/torque) must also be taken into account. Those constraints are called *kinodynamic constraints* [77].

In overall, there are two principal approaches used to cope with the problems related to trajectory planning with kinodynamic constraints. The first approach is to plan directly a motion in the *state space* of the robot, a space of robot configurations together with their first derivatives. This method, hence, is considered as a state space approach. A trajectory is achieved in just a single shot. On the other hand, the second approach (also known as a configuration approach) separates the problem into two sub-problems, including a path-planning problem and a time-parameterization problem.

State Space Approach

A particular illustration of the planner using State Space Approach is an RRT-based planner shown in [71]. This study is also known as the first one to point out some issues in regard to kinodynamic motion planning by applying a sampling-based planner. According to this method, an exploration tree is established in the state space of robot $\mathcal{X} \in \mathbf{R}^{2n}$ instead of the configuration space $\mathcal{C} \in \mathbf{R}^{n}$ to consider the dynamics of the general system while the tree is extended. As a result, the work space is enlarged to double in dimension. The other type of kinodynamic RRT-based planner has been mentioned in [78] that is an extension of a path planner previously suggested [79]. The search space is the state × time space, $\mathcal{X} \times \mathcal{T}$, where $\mathcal{T} = [0, \infty)$, .The dimension in this case is even higher than it of the mentioned planner. In place of using the conventional method that samples randomly a state and then extends the tree so as to approach it to obtain a new state, a vertex on the tree is chosen in a random way so that it can be extended from with some possibilities inversely proportional to the density of its surrounding area.

Configuration Space Approach

In contrast, configuration space approach is based on the idea that separates the principal problem into two different sub-problems. Introduced in the early 1986 in [80], this idea is in fact familiar in scope of robotics. Seeing a trajectory as a constitution of a path and a time-parameterization, *i.e.*, a *velocity profile*, a trajectory planning problem can be categorized into two separated problems, namely a path planning one and a time-parameterization one. The former subproblem, *i.e.*, path planning problem, is entirely geometric. Only kinematic constraints; for instance, the joint limits of the robot and the obstacle avoidance; are investigated carefully in this phase. Moreover, the choice of path planners that can be used is diversified.

After getting a path from the planner, the latter subproblem is implemented to deal with a *velocity profile* along the path that the trajectory can be dynamically executed. There are a number of algorithms that are insured to find the optimal velocity profile in case the path is time-parameterizable, *i.e.*, at least one attainable velocity profile exists.

If the configuration space approach is implemented, its strong point is that the search space of the planner is only a half compared to it of the state space approach. As a result, the expected operating time of the planner is significantly reduced. However, it is not certain that the obtained path will be time-parameterizable by taking into account only kinematic constraints in the first phase. This is a prominent issue due to the fact that a non-parameterizable path needs to be removed and the planning has to get back to the starting point.

A hands-on and common method applied to resolve the mentioned issue is to add several supplementary constraints into path planning to make sure that the output path from the planner will be perfectly executable *quasi-statically*, *i.e.*, at any arbitrarily low speed. A benefit of using this method is that we do not need to check any extra collision. As a case in point, the authors in [81] mention a kinodynamic motion planning problem for humanoid robots. In the planning phase, Quasi-static constraints are taken into account by computing the planner to sample a totally new configuration from a precomputed set $\mathcal{Q}_{\text{stable}} \subset \mathcal{C}$ of quasi-statically stable configurations.

However, it is too limited if we embed quasi-static constraints into path planning problem. A prime example is a human doing his natural walking motions. When there is only one foot on the ground, the center of mass projection is scarcely in the supporting area but that person can still stand as long as the zero-moment point constraint is not violated. This limitation hinders the experiments in [81] in comprising any walking motion.

The Admissible Velocity Propagation algorithm (AVP), lately, has been recommended in 2013 in [82] to particularly deal with this issue. The crucial idea of the algorithm is to plan a kinodynamically practicable motions while still remaining in the configuration space.

Trajectory Planning with Kinodynamic Constraints in $\boldsymbol{SE}(3)$

We are now interested in finding fast, collision-free, motions for transferring a rigid body (e.g. a rigid spacecraft) from an initial configuration (position and orientation) to a goal configuration, subject to bounds on angular velocities, angular accelerations or torques. This problem is difficult because of the non-trivial natures of the *kinodynamic bounds*, and of the underlying space, the space of rigid body configurations, or $\boldsymbol{SE}(3)$.

When only rotations are involved, the problem is known as *time-optimal reorientation of a rigid body'*, and is an interesting fundamental problem that arises in a number of applications including spacecraft or robot motion planning, computer animation, etc. There are two main directions of research in the literature. The aerospace community has considered the problem of finding *exact* time-optimal reorientation trajectories under kinodynamic constraints by applying directly Pontryagin's minimum principle [83, 84, 85]. However, this approach could only deal with the relatively simple cases of rest-to-rest motions in an obstacle-free environment.

In the robotics community, popular sampling-based motion planning methods such as RRT or PRM have been extended to $\boldsymbol{SO}(3)$. Such methods require three components: (i) random sampling, (ii) distance metric, and (iii) interpolation between two points. While straightforward in Euclidean spaces, these three components involve considerable challenges in $\boldsymbol{SO}(3)$. Regarding component (i), a sampling algorithm based on unit quaternions was proposed in [86]. As for components (ii) and (iii), methods based on unit quaternions or on rotation matrices exist, but have rarely been considered in the contexts of kinodynamic constraints and time-optimality.

Yet, the ability to reorientate a rigid body without colliding with the environment is crucial in space stations maneuver and in many robotics or computer graphics applications. To our knowledge, there are no general methods for planning fast trajectories in $\boldsymbol{SO}(3)$ under kinodynamic constraints and in a cluttered environment.

We will introduce our approach that provide an efficient method to find fast maneuvers in $\boldsymbol{SO}(3)$ considering collision avoidance in Chapter 3.1. We also present an extension to solve the problem the space of three-dimensional rigid body motions, $\boldsymbol{SE}(3)$.

Chapter 3

Progress

In this Chapter, we present the status of our research including some preliminary work that has been accomplished so far.

In Section 3.1, we present our studies on motion planning problems in the space of rigid-body motions $\boldsymbol{SE}(3)$. In particular, we propose a new approach for the problem of planning fast, collision-free, trajectories in the spaces of three-dimensional rotations $\boldsymbol{SO}(3)$ and three-dimensional rigid body motions $\boldsymbol{SE}(3)$ under kinodynamic constraints in cluttered environments. Our main contribution consisted in extending the classical Time-Optimal Path Parameterization (TOPP) [87, 88, 89] algorithm to $\boldsymbol{SO}(3)$. We integrated the algorithm into a plan-and-shortcut pipeline to yield a complete framework (and open-source implementation) for rigid-body motion planning under kinodynamic constraints. We also showed that our implementation could very efficiently find near time-optimal trajectories in a spacecraft maneuver problem. This work has been published in Journal of Guidance Control and Dynamics (JGCD), 2016 [90].

Another work presented in Section 3.2 shows our studies on associating the uncertainty to the solution of hand-eye calibration problem. The distribution of the errors between the estimated hand-eye transformation and the ground truth is assumed to be approximated by a multivariate Gaussian distribution. We formulate the hand-eye calibration problem as a least-squares minimization problem, then utilize propagation of covariance methods from statistics theory to obtain the covariance matrix of the solution. The simulation result shows that our methods can deliver a very good approximation of the real covariance matrix of the hand-eye transformation.

3.1 Motion Planning in SE(3)

3.1.1 Introduction

In this Section, we are interested in finding fast, collision-free, motions for transferring a rigid body (e.g. a rigid spacecraft) from an initial configuration (position and orientation) to a goal configuration, subject to bounds on angular velocities, angular accelerations or torques. This problem is difficult because of the non-trivial natures of the *kinodynamic bounds* [1] and of the underlying space, the space of rigid body configurations, or $\boldsymbol{SE}(3)$ [2].

When only rotations are involved (no translation), the space of possible motions is $\boldsymbol{SO}(3)$ [3]. Li and Bainum studied maneuvers based on rotations around a principal axis (or eigenaxis rotations), which they thought to be time-optimal for an inertially symmetric body with independent three-axis control [83]. Bilimoria and Wie showed that, even for inertially symmetric bodies, time-optimal maneuvers actually include significant nutational components [84]. More recently, Bai and Junkins refined Bilimoria and Wie's results by describing precisely various classes of time-optimal maneuvers [85]. The cited works have in common that they try to compute *exact* time-optimal motions by applying directly Pontryagin's minimum principle. While the theoretical interest of such an approach is evident, in practice, it is very computationally intensive and does not seem to be generalizable beyond rest-to-rest motions in obstacle-free environments.

Yet, the ability to move without colliding with the environment is crucial in cluttered space stations. In some applications, the satellite or spacecraft must avoid going through some orientations (in order e.g. to keep communication uninterrupted with the base station), which can be also modelled as obstacle in $\boldsymbol{SO}(3)$ or $\boldsymbol{SE}(3)$. Beyond the aerospace field, planning fast collision-free rigid body motions is also important in many robotics or computer graphics applications.

We present an efficient method to find fast maneuvers in $\boldsymbol{SO}(3)$ considering collision avoidance. For that, we adopt the plan-and-shortcut method [91] widely used in the robotics community

1. Plan a collision-free *geometric path* between the initial and the final orientation;

2. Time-parameterize that geometric path to obtain a *trajectory* that respects the kinodynamic constraints (bounds on angular velocities, angular accelerations or torques);

3. Repeatedly apply *shortcuts* to decrease the time duration of the obtained trajectory.

Step 1 of the above algorithm has been addressed, in particular by Kuffner [86], by extending the classical Rapidly exploring Random Tree (RRT) algorithm [71] to $\boldsymbol{SO}(3)$. Regarding steps 2 and 3, the crucial requirement is the ability to optimally *time-parameterize* a given path in $\boldsymbol{SO}(3)$ under kinodynamic constraints. We do so by extending the classical [87, 88, 89] Time-Optimal Path Parameterization (TOPP)

[1] In robotics, "kinodynamic constraints" refer to constraints that can only be expressed based on the time derivatives of the robot configuration, for instance, bounds on the robot velocity, acceleration, and force/torque [71].

[2] Special Euclidean group in dimension 3.

[3] Special Orthogonal group in dimension 3.

algorithm to the case of $\boldsymbol{SO}(3)$, which constitutes the main contribution of this work. We show that, overall, the method we propose can find fast maneuvers for a satellite model in a cluttered environment in less than 10 seconds. We also present an extension to the space of three-dimensional rigid body motions, $\boldsymbol{SE}(3)$.

This Section is organised as follows. In Section 3.1.2, we recall some backgrounds regarding the problem of interpolation in $\boldsymbol{SO}(3)$ and $\boldsymbol{SE}(3)$. Choosing a good method to interpolate paths between two elements of $\boldsymbol{SO}(3)$ or $\boldsymbol{SE}(3)$ is particularly important: we need paths that are "short" and smooth, with easy-to-compute first and second derivatives, so that the subsequent time-parameterization can efficiently yield high-quality trajectories. In Section 3.1.3, we present the extension of TOPP to $\boldsymbol{SO}(3)$. In Sections 3.1.4, we present the implementation of the planning method and simulation results on several spacecraft maneuver problems. Section 3.1.5 concludes and sketches some future research directions.

3.1.2 Interpolation in SO(3) and SE(3)

Notations

We first recall some useful notations in the study of $\boldsymbol{SO}(3)$ [92]. Given a vector $\boldsymbol{r} \in \mathbb{R}^3$, one defines the skew-symmetric matrix

$$[\boldsymbol{r}] \overset{\text{def}}{=} \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}.$$

The exponential of a skew-symmetric matrix defined as above is given by

$$\boldsymbol{e}^{[\boldsymbol{r}]} \overset{\text{def}}{=} \mathbf{I} + \frac{\sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|}[\boldsymbol{r}] + \frac{1-\cos\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^2}[\boldsymbol{r}]^2. \tag{3.1}$$

Next, for any matrix $\mathbf{R} \in \boldsymbol{SO}(3)$, there exists $\boldsymbol{r}$ such that $\mathbf{R} = \boldsymbol{e}^{[\boldsymbol{r}]}$, and one denotes $\log \mathbf{R} \overset{\text{def}}{=} [\boldsymbol{r}]$. More precisely, if $\operatorname{tr}(\mathbf{R}) \neq 1$, then

$$\log \mathbf{R} = \frac{\phi}{2\sin\phi}(\mathbf{R} - \mathbf{R}^\top), \tag{3.2}$$

where ϕ satisfies $1 + 2\cos\phi = \operatorname{tr}(\mathbf{R})$. If $\operatorname{tr}(\mathbf{R}) = 1$, then

$$\log \mathbf{R} = \pm\pi[\hat{\boldsymbol{v}}], \tag{3.3}$$

where $\hat{\boldsymbol{v}}$ is a unit length eigenvector of $\mathbf{R}$ associated with the eigenvalue 1.

Interpolation in SO(3)

Consider two orientation matrices $\mathbf{R}_0, \mathbf{R}_1 \in \boldsymbol{SO}(3)$ and two vectors $\boldsymbol{\omega}_0, \boldsymbol{\omega}_1 \in \mathbb{R}^3$. We are interested in the problem of finding a smooth, "short" curve $(\mathbf{R}(t))_{t\in[0,1]}$ in $\boldsymbol{SO}(3)$ such that

$$\mathbf{R}(0) = \mathbf{R}_0,\ \mathbf{R}(1) = \mathbf{R}_1,\ \boldsymbol{\omega}(0) = \boldsymbol{\omega}_0,\ \boldsymbol{\omega}(1) = \boldsymbol{\omega}_1,$$

where $\boldsymbol{\omega}(t)$ denotes the angular velocity vector computed in the moving (body) frame.

Consider a vector $\boldsymbol{r}_1$ such that $[\boldsymbol{r}_1] = \log(\mathbf{R}_0^\top \mathbf{R}_1)$. One possible interpolation method is given by [92, 93]

$$\mathbf{R}(t) = \mathbf{R}_0 \boldsymbol{e}^{[\boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t]}, \tag{3.4}$$

where $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3 \in \mathbb{R}^3$ are constant vectors satisfying

(i) $\boldsymbol{a}_1 = \boldsymbol{\omega}_0$;

(ii) $\boldsymbol{a}_3 + \boldsymbol{a}_2 + \boldsymbol{a}_1 = \boldsymbol{r}_1$;

(iii) $3\boldsymbol{a}_3 + 2\boldsymbol{a}_2 + \boldsymbol{a}_1 = \mathbf{A}^{-1}(\boldsymbol{r}_1)\boldsymbol{\omega}_1$ [the definition of $\mathbf{A}$ is given in equation (3.7) below].

This interpolation method has many advantages. First, it approximates the minimum acceleration interpolation, which implies that the underlying path is smooth and short. In particular, if $\boldsymbol{\omega}_0 = \boldsymbol{\omega}_1 = \mathbf{0}$, then this interpolation indeed yields the shortest path in $\boldsymbol{SO}(3)$ [93]. This interpolation is also bi-invariant: the resulting orientation trajectories are invariant with respect to the choice of fixed or moving reference frames. Finally, the first and second derivatives are smooth and easy to compute, which is useful for TOPP where we need smooth and easy-to-compute first and second derivatives. The angular velocity and acceleration are indeed given by

$$\begin{aligned} \boldsymbol{\omega}(t) &= \mathbf{A}(\boldsymbol{r})\dot{\boldsymbol{r}}, && (3.5)\\ \dot{\boldsymbol{\omega}}(t) &= \mathbf{A}(\boldsymbol{r})\ddot{\boldsymbol{r}} + \mathbf{C}(\boldsymbol{r}, \dot{\boldsymbol{r}}), && (3.6) \end{aligned}$$

where

$$\mathbf{A}(\boldsymbol{r}) \stackrel{\text{def}}{=} \mathbf{I} - \frac{1 - \cos\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^2}[\boldsymbol{r}] + \frac{\|\boldsymbol{r}\| - \sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^3}[\boldsymbol{r}]^2, \tag{3.7}$$

and

$$\begin{aligned} \mathbf{C}(\boldsymbol{r}, \dot{\boldsymbol{r}}) \stackrel{\text{def}}{=}\ & \frac{\|\boldsymbol{r}\| - \sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^3}\dot{\boldsymbol{r}} \times (\boldsymbol{r} \times \dot{\boldsymbol{r}})\\ -\ & \frac{2\cos\|\boldsymbol{r}\| + \|\boldsymbol{r}\|\sin\|\boldsymbol{r}\| - 2}{\|\boldsymbol{r}\|^4}\boldsymbol{r}^\top\dot{\boldsymbol{r}}(\boldsymbol{r} \times \dot{\boldsymbol{r}})\\ +\ & \frac{3\sin\|\boldsymbol{r}\| - \|\boldsymbol{r}\|\cos\|\boldsymbol{r}\| - 2\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^5}\boldsymbol{r}^\top\dot{\boldsymbol{r}}(\boldsymbol{r} \times (\boldsymbol{r} \times \dot{\boldsymbol{r}})). \end{aligned} \tag{3.8}$$

Interpolation in SE(3)

While $\boldsymbol{SO}(3)$ is the group of 3D rotations, $\boldsymbol{SE}(3)$ is the special Euclidean group of rigid-body motions. $\boldsymbol{SE}(3)$ includes both rotations and translations, and is of the form

$$\begin{bmatrix} \mathbf{R} & \boldsymbol{q} \\ 0 & 1 \end{bmatrix},$$

where $\mathbf{R} \in \boldsymbol{SO}(3)$ and $\boldsymbol{q} \in \mathbb{R}^3$. There are a number of available approaches to interpolate trajectories between two elements of $\boldsymbol{SE}(3)$. Park and Ravani [92] exploited the Lie group structure of $\boldsymbol{SE}(3)$ to develop an algorithm analogous to the preceding interpolation in $\boldsymbol{SO}(3)$. The resulting motion, however, is a screw motion, which corresponds to a strange motion in physical space. Moreover, in $\boldsymbol{SE}(3)$ there is in general no bi-invariant interpolations [93]. Therefore, we use a simple method, which consists in interpolating orientations and positions separately. For the orientation part, we proceed as in Section 3.1.2. For the translation part, we interpolate between two pairs (position, linear velocity) by a third degree polynomial. Consider two pairs $(\boldsymbol{p}_0, \boldsymbol{v}_0)$ and $(\boldsymbol{p}_1, \boldsymbol{v}_1)$, the interpolant is given by

$$\boldsymbol{p}(t)_{t\in[0,1]} = \boldsymbol{k}_3 t^3 + \boldsymbol{k}_2 t^2 + \boldsymbol{k}_1 t + \boldsymbol{k}_0, \tag{3.9}$$

where $\boldsymbol{k}_3, \boldsymbol{k}_2, \boldsymbol{k}_1, \boldsymbol{k}_0$ can be easily found using the boundary conditions $\boldsymbol{p}(0) = \boldsymbol{p}_0, \boldsymbol{v}(0) = \boldsymbol{v}_0, \boldsymbol{p}(1) = \boldsymbol{p}_1, \boldsymbol{v}(1) = \boldsymbol{v}_1$ ($\boldsymbol{v}(t)$ denotes the linear velocity vector). Note that the first and second derivatives of the interpolant are trivial to compute.

3.1.3 Time-Optimal Path Parameterization in SO(3)

Consider a path $\mathcal{P}$ – represented as the underlying path of a trajectory $\boldsymbol{r}(s)_{s\in[0,s_{\text{end}}]}$ – in the configuration space. Assume that $\boldsymbol{r}(s)_{s\in[0,s_{\text{end}}]}$ is C^1- and piecewise C^2-continuous. We are interested in *time-parameterizations* of $\mathcal{P}$, which are increasing *scalar functions* $s : [0, T] \to [0, s_{\text{end}}]$, under kinodynamic constraints.

If the constraints can be reduced to the form (note that all vector inequalities in this Section are element-wise)

$$\ddot{s}\boldsymbol{a}(s) + \dot{s}^2\boldsymbol{b}(s) + \boldsymbol{c}(s) \leq \mathbf{0}, \tag{3.10}$$

then efficient methods and implementations allow finding the time-optimal parameterization $s(t)$ (see e.g. [89]).

Consider now a rigid body with three independent actuations, such as a rigid spacecraft whose equation of motion is [83, 84, 85]

$$\mathbb{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbb{I}\boldsymbol{\omega}) = \boldsymbol{\tau}, \tag{3.11}$$

where $\mathbb{I}$ is the 3×3 inertia matrix of the spacecraft and $\boldsymbol{\tau}$ the 3-dimensional torque vectors. The actuation bounds are given by

$$\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max}. \tag{3.12}$$

Consider now an orientation trajectory $\mathbf{R}(s)_{s\in[0,1]} \in \boldsymbol{SO}(3)$ given by

$$\mathbf{R}(s) = \mathbf{R}_0 \boldsymbol{e}^{[\boldsymbol{r}(s)]}. \tag{3.13}$$

Substituting the expressions of $\dot{\boldsymbol{r}}$ and $\ddot{\boldsymbol{r}}$ in terms of the path parameter s

$$\dot{\boldsymbol{r}} = \dot{s}\boldsymbol{r}_s, \quad \ddot{\boldsymbol{r}} = \ddot{s}\boldsymbol{r}_s + \dot{s}^2\boldsymbol{r}_{ss} \tag{3.14}$$

into (3.5) and (3.6), one obtains

$$\boldsymbol{\omega} = \dot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s, \tag{3.15}$$

$$\dot{\boldsymbol{\omega}} = \ddot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s)\right\}, \tag{3.16}$$

$$\boldsymbol{\tau} = \ddot{s}\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s)\right\}. \tag{3.17}$$

Thus, the condition $\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max}$ can be put in the form of (3.10) with

$$\boldsymbol{a} = \begin{bmatrix} \mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s \\ -\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s \end{bmatrix}, \tag{3.18}$$

$$\boldsymbol{b} = \begin{bmatrix} \mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \\ -(\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s)) \end{bmatrix}, \tag{3.19}$$

$$\boldsymbol{c} = \begin{bmatrix} -\boldsymbol{\tau}_{\max} \\ \boldsymbol{\tau}_{\min} \end{bmatrix}. \tag{3.20}$$

The motion of the spacecraft is generally constrained, not only by the spacecraft itself (for example, torque limits) but also by task constraints (e.g. bounds on angular accelerations). The form (3.10) allows easily combining different type of constraints by concatenating the vectors $\boldsymbol{a}(s), \boldsymbol{b}(s), \boldsymbol{c}(s)$ corresponding to the different constraints. For instance, bounds on angular accelerations ($\dot{\boldsymbol{\omega}}_{\min} \leq \dot{\boldsymbol{\omega}} \leq \dot{\boldsymbol{\omega}}_{\max}$) can be put in the form of (3.10) by noting that $\dot{\boldsymbol{\omega}} = \ddot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s)\right\}$.

Bounds on angular velocities can be expressed as $\boldsymbol{\omega}^\top\boldsymbol{\omega} \leq \omega_{\max}$, where $\omega_{\max}$ is a scalar. Such constraints were first addressed in [94]. The treatment was completed and implemented in [89].

For time-parameterization a given path in $\boldsymbol{SE}(3)$ under kinodynamic constraints, we separate the process into 2 steps. First, we re-time the orientation part as presented earlier. Regarding the translation part, since we interpolate two pairs $(\boldsymbol{p}_0, \boldsymbol{v}_0)$ and $(\boldsymbol{p}_1, \boldsymbol{v}_1)$ by a third degree polynomial (Section II 3.1.2), the first and second derivatives are trivial to compute. Therefore, contraints on linear accelerations $(\boldsymbol{\alpha}_{\min} \leq \boldsymbol{\alpha} \leq \boldsymbol{\alpha}_{\max})$ or forces $(\boldsymbol{f}_{\min} \leq \boldsymbol{f} \leq \boldsymbol{f}_{\max})$ can be easily expressed in the form of (3.10).

3.1.4 Implementation and evaluation

We experimentally evaluated our approach on some spacecraft maneuver problems. Our implementation is open-source and can be found online at [95]. All experiments were run on a machine with an Intel i7 3.40 GHz processor, 4GB RAM. Videos of the resulting simulations can be found at [96].

Planning fast, collision-free trajectories in SO(3) under kinodynamic constraints

Here, we present the algorithm to find a fast, collision-free, trajectory connecting two rotations $\mathbf{R}_0$ $\mathbf{R}_1$, subject to velocity and torque bounds. As mentioned earlier, we follow the plan-and-shortcut method [91].

In step 1, we use an RRT-based path planner to find a collision-free piecewise "linear" path that connects $(\mathbf{R}_0, \boldsymbol{\omega}_0)$ and $(\mathbf{R}_1, \boldsymbol{\omega}_1)$ (each "linear" segment is actually a great-circle arc in $\boldsymbol{SO}(3)$).

Next, in step 2, we optimally time-parameterize each of the "linear" segment. To avoid discontinuities of the velocity vector at the junctions of the "linear" segments, we ensure that the velocities at the beginning and the end of each segment are zero.

In step 3, at each shortcut iteration, we select two random time instants t_0, t_1 along the trajectory, which correspond to two orientation matrices $\mathbf{R}_{t_0}, \mathbf{R}_{t_1} \in \boldsymbol{SO}(3)$ and two angular velocity vectors $\boldsymbol{\omega}_{t_0}, \boldsymbol{\omega}_{t_1} \in \mathbb{R}^3$. We then find the interpolation path $\mathbf{R}^*(t)_{t \in [0,T]}$, where $T = t_1 - t_0$, such that

$$\mathbf{R}^*(0) = \mathbf{R}_{t_0},\ \mathbf{R}^*(T) = \mathbf{R}_{t_1},\ \boldsymbol{\omega}^*(0) = \boldsymbol{\omega}_{t_0},\ \boldsymbol{\omega}^*(T) = \boldsymbol{\omega}_{t_1}.$$

If this path involves any collision, we discard it. Otherwise, we use TOPP to optimally time-parameterize it, while ensuring that the initial and final velocities of the shortcut portion are unchanged to avoid velocity discontinuities. If the resulting duration is smaller than T then we replace the original trajectory portion by the shortcut.

We tested the proposed algorithm on a reorientation problem for the Messenger spacecraft subject to bounds on angular velocities and torques in a cluttered environment (model downloaded from [97]), see Fig. 3.1. We ran 100 trials, with 200 shortcut iterations per trial. The average running time for the three steps of the algorithm are shown in Table 3.1. Overall, the algorithm can find fast trajectories for the satellite model in this cluttered environment in 7.42 ± 4.58s. The average number of successful shortcuts per trial was 11.12 ± 6.31. Fig. 3.1 shows one typical trajectory found by the algorithm. Fig. 3.2 shows the corresponding angular velocities and torques as functions of time. Note that, $(\omega^1, \omega^2, \omega^3)$ are the three elements of the angular velocity vector $\boldsymbol{\omega}$, while (τ^1, τ^2, τ^3) are the three elements of the torque vector $\boldsymbol{\tau}$. One can see that, in agreement with time-optimality, at least one constraint is saturated at any time instant.

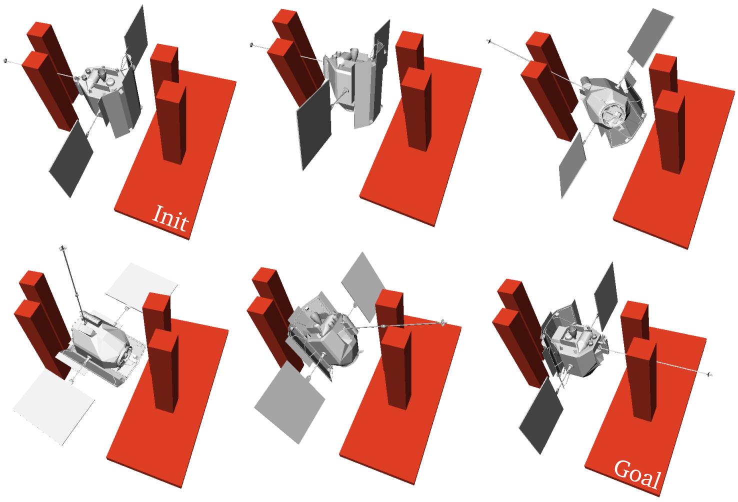


Figure 3.1: A fast reorientation trajectory in $\boldsymbol{SO}(3)$ for the Messenger spacecraft in a cluttered environment.

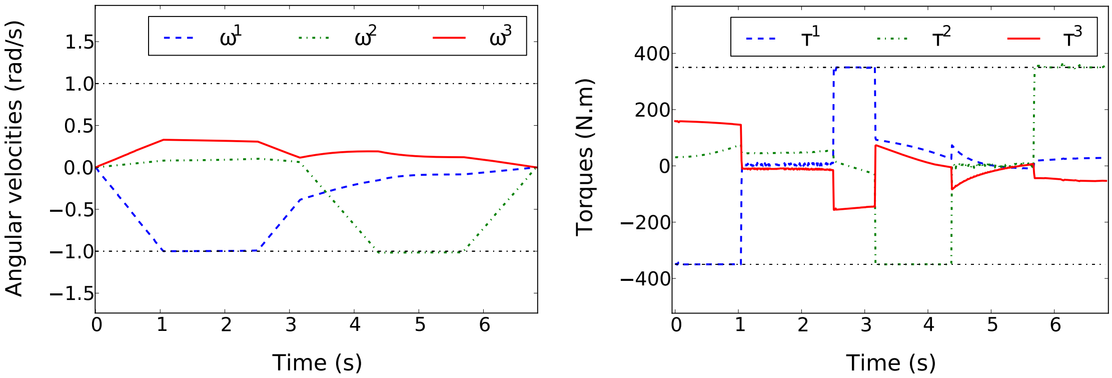


Figure 3.2: Angular velocities and torques of the trajectory in Fig. 3.1.

Planning Fast, Collision-free Trajectories in SE(3) Under Kinodynamic Constraints

We now consider the case of rigid-body motions. Figure 3.3 shows the environment we used to test the proposed algorithm. We ran 100 trials, with 200 shortcut iterations per trial. The average running time for the three steps of the algorithm is showed in Table 3.1. Overall, the algorithm can find fast trajectories for the satellite model in this cluttered environment in 32.21 ± 6.12s. The average number of successful shortcuts per trial was 18.32 ± 7.34. The total running time for problems in $\boldsymbol{SE}(3)$ was longer than in $\boldsymbol{SO}(3)$, because of the higher problem dimension and environment complexity. One typical resulting trajectory found by the algorithm is showed in Figure 3.3. The corresponding velocities, torques and forces are showed in Figure 3.4. Note that, (v^1, v^2, v^3) are the three elements of the linear velocity vector $\boldsymbol{v}$, while (f^1, f^2, f^3) are the three elements of the force vector $\boldsymbol{f}$. One can also see here that, in agreement with time-optimality, at least one constraint is saturated at any time instant.

	RRT	TOPP on “linear” segments	Shortcutting	Total running time
$\boldsymbol{SO}(3)$	1.87±3.01	0.09±0.07	5.34±2.35	7.42±4.58
$\boldsymbol{SE}(3)$	6.18±5.23	0.32±0.20	25.54±6.53	32.21±6.12

Table 3.1: Average running time, in seconds, for the three steps of the plan-and-shortcut method in $\boldsymbol{SO}(3)$ and $\boldsymbol{SE}(3)$

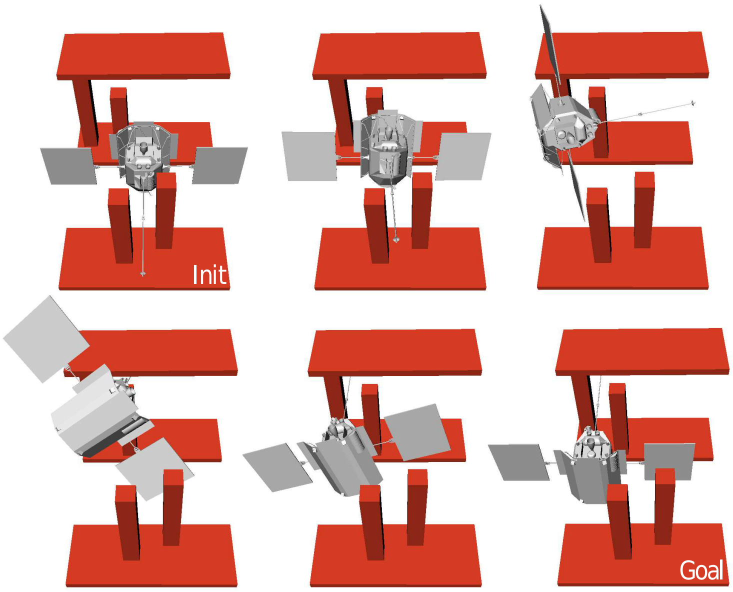


Figure 3.3: A fast maneuver trajectory in $\boldsymbol{SE}(3)$ for the Messenger spacecraft in a cluttered environment.

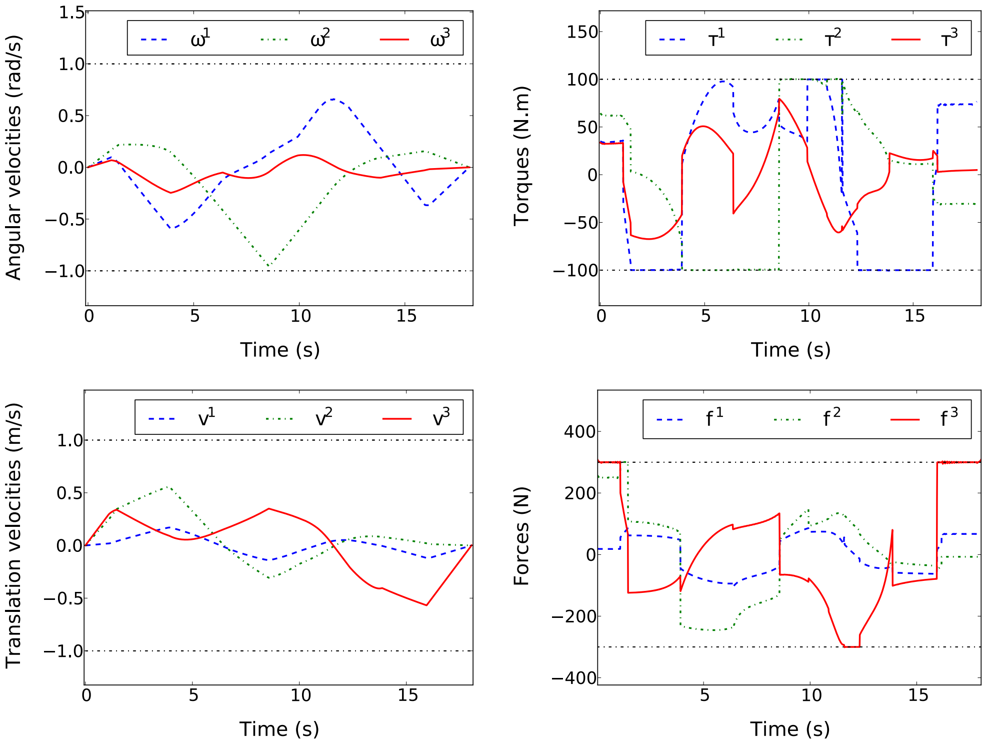


Figure 3.4: Velocities, torques and forces of the trajectory in Fig. 3.3.

3.1.5 Conclusion

We have addressed the problem of planning fast, collision-free, trajectories in the spaces of three-dimensional rotations $\boldsymbol{SO}(3)$ and three-dimensional rigid body motions $\boldsymbol{SE}(3)$ under kinodynamic constraints in cluttered environments. Our main contribution consisted in extending the classical Time-Optimal Path Parameterization (TOPP) algorithm to $\boldsymbol{SO}(3)$. We integrated the algorithm into a plan-and-shortcut pipeline to yield a complete framework (and open-source implementation) for rigid-body motion planning under kinodynamic constraints. We showed that our implementation could very efficiently find near time-optimal trajectories in a spacecraft maneuver problem. We are currently considering applications to multi-spacecraft coordination, humanoid robots and computer animation.

3.2 On the Uncertainty of the Classical Hand-Eye Calibration Problem

3.2.1 Introduction

Whenever a sensory device is attached on a robot hand, the problem of determining the relationship between the sensor and the hand is crucial. This problem is usually referred as *hand-eye calibration problem* and has long been an interesting topic which draws an extensive attention from many robotic researchers.

Despite of extensive literature on the solution methods for this problem, estimating the associated uncertainty of the resulting hand-eye transformation has not been a major concern for existing applications. However, in some applications that rely on low-cost and noisy sensors or require very high accuracy, some measure of confident or uncertainty of solution is also required. The Airbus Shopfloor Challenge that we participated in at ICRA 2016 can be considered as a prime example in this case. In the challenge, competing teams have to built systems that are able to perform accurate drilling on an artefact representing part of the aircraft fuselage, as depicted in Figure 3.5. To map the artefact pose measured in camera frame into the robot base frame, it is very important to have an accurate calibration along with the associated uncertainty. Moreover, having the estimated uncertainty of the hand-eye transformation will enable the system to predict accurately the camera pose according to the robot motion. Since the working space in this case is highly constrained, this would help the system to perform better in term of avoiding collisions with the environment. In respect of the hand-eye calibration problem, the uncertainty of the hand-eye transformation can also be used as an evaluation criteria for the calibration process.

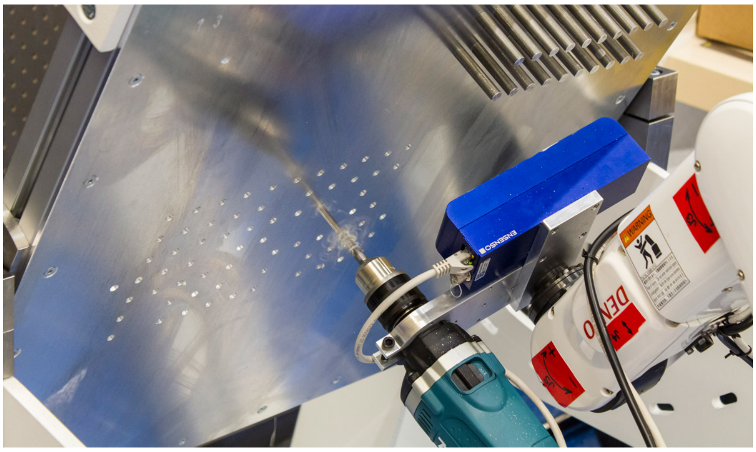


Figure 3.5: Airbus Shopfloor Challenge-ICRA 2016, where competing teams designed and built robotic systems that are able to perform accurate drilling compliant with aeronautic standards.

In this Section, we, hence, will present our approach to address this interesting problem. The organization of this Section is as follows. We first give a brief discussion on the hand-eye calibration problem in Section 3.2.2. Next, Section 3.2.3 recall some important mathematics background on methods for assessing and quantifying the results of estimation algorithms. We then present how we adopt these methods to this hand-eye calibration problem in Section 3.2.4. Simulation results and discussions are provided in Section 3.2.5. Finally, Section 3.2.6 concludes and sketches some future research directions.

3.2.2 Description of the hand-eye calibration problem

The hand-eye calibration problem is commonly represented of the form: $\boldsymbol{AX} = \boldsymbol{XB}$. This formulation, in fact, directly reproduces the rigid transformations in the loop base-end effector-camera-object/world-base (Figure 2.3):

$$ {}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{t}}\,{}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}} = {}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{o}}\,{}_{\boldsymbol{o}}\boldsymbol{T}^{\boldsymbol{c}} \tag{3.21} $$

Due to the fact that ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$ is more often required than ${}_{\boldsymbol{b}}\boldsymbol{T}^{\boldsymbol{o}}$, most approaches eliminate the latter by writing Eq. 3.21 at two different instants i and j yielding the well-known hand-eye equation:

$$ {}_{\boldsymbol{b}}\boldsymbol{T}_j^{\boldsymbol{t}^{-1}}\,{}_{\boldsymbol{b}}\boldsymbol{T}_i^{\boldsymbol{t}}\,{}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}} = {}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}\,{}_{\boldsymbol{o}}\boldsymbol{T}_j^{\boldsymbol{c}^{-1}}\,{}_{\boldsymbol{o}}\boldsymbol{T}_i^{\boldsymbol{c}} \Rightarrow \boldsymbol{AX} = \boldsymbol{XB}, \tag{3.22} $$

where the unknown $\boldsymbol{X}$ represents the fixed homogeneous transformation between camera and the end effector ${}_{\boldsymbol{t}}\boldsymbol{T}^{\boldsymbol{c}}$, $\boldsymbol{A_{ij}} = {}_{\boldsymbol{b}}\boldsymbol{T}_j^{\boldsymbol{t}^{-1}}\,{}_{\boldsymbol{b}}\boldsymbol{T}_i^{\boldsymbol{t}}$ and $\boldsymbol{B_{ij}} = {}_{\boldsymbol{o}}\boldsymbol{T}_j^{\boldsymbol{c}^{-1}}\,{}_{\boldsymbol{o}}\boldsymbol{T}_i^{\boldsymbol{c}}$. Note that another representation of this problem consists of the form $\boldsymbol{AX} = \boldsymbol{YB}$ which can also be constructed similarly. However, at right now, this is not our particular interest.

In actual sensor calibration applications, some noise is usually present in the measurements of $\boldsymbol{A}$ and $\boldsymbol{B}$. Therefore, a practical approach is to make several measurements $(\boldsymbol{A}_1, \boldsymbol{B}_1), (\boldsymbol{A}_2, \boldsymbol{B}_2), ..., (\boldsymbol{A}_k, \boldsymbol{B}_k)$, and to find an $\mathbf{X}$ that is a best fit to the k matrix equalities. This problem is usually formulated as a *least-squares optimization problem*, in which the objective function is given by:

$$ \min_{\boldsymbol{X}\in\boldsymbol{SE}(3)} \sum_i \|\boldsymbol{A}_i\boldsymbol{X} - \boldsymbol{X}\boldsymbol{B}_i\|^2 \tag{3.23} $$

where $\|.\|$ denotes the Frobenius matrix norm.

There exists extensive work on solving this classical problem, which we give a detailed literature review in Section 2.2.2. What is worth emphasizing is that Eq. 3.23 is highly non-linear and non-convex, whose the number and properties of local minima are greatly affected by the level of uncertainty in the measurements $(\boldsymbol{A}_i, \boldsymbol{B}_i)$. Most existing methods only succeed in finding a local minimum that close to the ground truth of $\boldsymbol{X}$. The errors between the estimated poses and the ground truth, therefore, are distributed according to some unknown distribution. To our knowledge, none of these methods has addressed the problem of determining the uncertainty of the solution of $\boldsymbol{X}$ or been able to show how the level of error in the measurements influence the level of error in the solution.

With regard to this problem, we assume the distribution of the errors between the estimated $\boldsymbol{X}$ and the ground truth to be approximated by a multivariate Gaussian distribution. Moreover, the algorithms used to solve a least-squares optimization problem can be seen as the most feasible solution to maximize a likelihood function (Maximum Likelihood Estimate). In this way, the uncertainty of the estimated parameters given by a Maximum Likelihood Estimation can be computed by applying *propagation of uncertainty* method from the statistics theory. The background of this theory, paricularly on the methods for assessing and quantifying the results of the estimation algorithms, will be mentioned with more details in the next Section.

3.2.3 Error Analysis: Variance of Estimated Parameters

There exists much theory in parameter estimation, dealing with covariance of the estimate. In [98], Hartley and Zisserman provided a discussion on two methods for computing the uncertainty, *i.e.*Monte Carlo method, and a Jacobian based method. They simplify the problem by only considering first-order Taylor expansions, and using an assumption of Gaussian probability distributions. Similar approaches and other extensions developed for other distributions (i.e. Fisher Information matrix) can also be found in [99], [100], and many other statistics textbooks. Here we are particularly interested in the method presented in [98] since it is well suited for our approach. Derivations through out this Section, therefore, will follow the instruction provided in [98].

Forward Propagation of Covariance

In statistics, when variables have uncertainties, these errors will affect the uncertainty of a function constructed from them.

Theorem 1 *Let $\boldsymbol{v}$ be a random vector in parameter space $\mathbb{R}^M$ with its mean $\bar{\boldsymbol{v}}$ and covariance matrix $\mathbf{\Sigma}$. Considering a function $\boldsymbol{f} : \mathbb{R}^M \to \mathbb{R}^N$ which is an affine mapping of the form $\boldsymbol{f}(\boldsymbol{v}) = \boldsymbol{f}(\bar{\boldsymbol{v}}) + \mathbf{A}(\boldsymbol{v} - \bar{\boldsymbol{v}})$, then $\boldsymbol{f}(\boldsymbol{v})$ is a random variable with mean $\boldsymbol{f}(\bar{\boldsymbol{v}})$ and its covariance matrix $\mathbf{A\Sigma A}^\top$*

This is the most general expression for the forward propagation of covariance from one set of variables onto the combination of them in the linear case. When $\boldsymbol{f}(\boldsymbol{v})$ is a non-linear function, we may compute the linear approximation of $\boldsymbol{f}(\boldsymbol{v})$ defined by $\boldsymbol{f}(\boldsymbol{v}) \approx \boldsymbol{f}(\bar{\boldsymbol{v}}) + \boldsymbol{\mathcal{J}}(\boldsymbol{v} - \bar{\boldsymbol{v}})$, where $\boldsymbol{\mathcal{J}} = \partial \boldsymbol{f} / \partial \boldsymbol{v}$ is the $N \times M$ Jacobian matrix evaluated at $\bar{\boldsymbol{v}}$.

Theorem 2 *Let $\boldsymbol{v}$ be a random vector in parameter space $\mathbb{R}^M$ with its mean $\bar{\boldsymbol{v}}$ and covariance matrix $\mathbf{\Sigma}$. Considering a function $\boldsymbol{f} : \mathbb{R}^M \to \mathbb{R}^N$ which is differentiable in a neighbourhood of $\bar{\boldsymbol{v}}$, to the first order approximation, $\boldsymbol{f}(\boldsymbol{v})$ is a random variable with mean $\boldsymbol{f}(\bar{\boldsymbol{v}})$ and its covariance matrix $\boldsymbol{\mathcal{J}}\mathbf{\Sigma}\boldsymbol{\mathcal{J}}^\top$, where $\boldsymbol{\mathcal{J}}$ is the Jacobian matrix of $\boldsymbol{f}$ at $\bar{\boldsymbol{v}}$*

Note that, how closely this approximation to the real mean and variance of $\boldsymbol{f}(\bar{\boldsymbol{v}})$ is relied on how good is the linear approximation of the function $\boldsymbol{f}$.

Backward Propagation of Covariance

As opposed to forward propagation, backward propagation is usually applied in an estimation problem, in which we try to estimate a vector of parameters $\mathbf{P}$ given a set of measurements $\mathbf{V}$. We first consider the case where $\boldsymbol{f} : \mathbb{R}^M \to \mathbb{R}^N$ is an affine mapping of the form $\boldsymbol{f}(\mathbf{P}) = \boldsymbol{f}(\bar{\mathbf{P}}) + \boldsymbol{\mathcal{J}}(\mathbf{P} - \bar{\mathbf{P}})$ where $\boldsymbol{\mathcal{J}}$ has rank M. Let $\mathbf{V}$ have a Gaussian probability distribution with its mean $\bar{\mathbf{V}} = \boldsymbol{f}(\bar{\mathbf{P}})$ and its covariance matrix $\mathbf{\Sigma}$. The vector $\mathbf{P}$ in the parameter space is a parameterization of the point $\boldsymbol{f}(\mathbf{P})$ on the image $\mathbf{S}_M$ of the mapping. As we do not consider the over-parameterized case, we assume that parameter space has higher dimension than measurement space (i.e $M < N$) and that the image $\mathbf{S}_M$ of the mapping has dimension of M.

Let $\boldsymbol{\eta} : \mathbb{R}^N \to \mathbf{S}_M$ be the mapping that maps the "closest" point on $\mathbf{S}_M$ to a point $\mathbf{V}$ in $\mathbb{R}^N$. We assume that $\boldsymbol{f}$ is invertible on $\mathbf{S}_M$, and let $\boldsymbol{f}^{-1} : \mathbf{S}_M \to \mathbb{R}^M$ be the inverse mapping. One may obtain the mapping $\boldsymbol{f}^{-1} \circ \boldsymbol{\eta} : \mathbb{R}^N \to \mathbb{R}^M$ by composing $\boldsymbol{\eta} : \mathbb{R}^N \to \mathbf{S}_M$ and $\boldsymbol{f}^{-1} : \mathbf{S}_M \to \mathbb{R}^M$. This mapping in fact assigns to a measurement vector $\mathbf{V}$, the set of parameters $\mathbf{P}$ corresponding to the maximum likelihood estimate $\hat{\mathbf{V}}$. In principle we may propagate the covariance of probability distribution in the measurement space $\mathbb{R}^N$ to compute a covariance matrix for the set of parameters $\mathbf{P}$ corresponding to maximum likelihood estimation. We aim to apply Theorem 1 and Theorem 2.

We will prove that $\boldsymbol{f}^{-1} \circ \boldsymbol{\eta}$ is also an affine mapping. Therefore, we can apply Theorem 1 to find the covariance of the estimated parameters $\hat{\mathbf{P}} = \boldsymbol{f}^{-1} \circ \boldsymbol{\eta}(\mathbf{V})$.

When seeking for $\hat{\mathbf{P}}$, we try to minimize $\|\mathbf{V} - \hat{\mathbf{V}}\| = \|\mathbf{V} - \boldsymbol{f}(\hat{\mathbf{P}}\|$. The later quantity can be written as follows:

$$\|\mathbf{V} - \boldsymbol{f}(\hat{\mathbf{P}}\| = \|(\mathbf{V} - \bar{\mathbf{V}}) - \boldsymbol{\mathcal{J}}(\hat{\mathbf{P}} - \bar{\mathbf{P}})\| \tag{3.24}$$

Minimizing the right hand side of Eq. 3.24 is a classical weighted linear least squares problem. It is minimized when

$$(\hat{\mathbf{P}} - \bar{\mathbf{P}}) = (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} (\mathbf{V} - \bar{\mathbf{V}}). \tag{3.25}$$

Note that covariance matrix $\boldsymbol{\Sigma}$ appears in Eq. 3.25 as a weighting matrix when $\mathbf{V}$ has a Gaussian probability distribution with its covariance matrix $\boldsymbol{\Sigma}$.

The true value and estimated value of parameters are $\bar{\mathbf{P}} = \boldsymbol{f}^{-1}(\bar{\mathbf{V}})$ and $\hat{\mathbf{P}} = \boldsymbol{f}^{-1}(\hat{\mathbf{V}})$, respectively. Then the estimated value of parameters $\hat{\mathbf{P}}$ can be expressed as follows:

$$\begin{aligned} \boldsymbol{f}^{-1} \circ \boldsymbol{\eta}(\mathbf{V}) &= \hat{\mathbf{P}} \qquad (3.26) \\ &= (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} (\mathbf{V} - \bar{\mathbf{V}}) + \mathbf{f}^{\mathbf{-1}}(\bar{\mathbf{V}}) \\ &= (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} (\mathbf{V} - \bar{\mathbf{V}}) + \mathbf{f}^{\mathbf{-1}} \circ \boldsymbol{\eta}(\bar{\mathbf{V}}) \end{aligned}$$

The above equation has shown that $\boldsymbol{f}^{-1} \circ \boldsymbol{\eta}$ is affine and $(\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}}$ is its linear part. Applying Theorem 1, the covariance matrix of the estimated $\hat{\mathbf{P}}$ is given by

$$\begin{aligned} [(\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}}] \boldsymbol{\Sigma} [(\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}}]^\top &= (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\Sigma} \boldsymbol{\mathcal{J}} (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \qquad (3.27) \\ &= (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{\mathbf{-1}} \end{aligned}$$

Hence, we have proved the following theorem:

Theorem 3 *Consider a function $\boldsymbol{f} : \mathbb{R}^M \rightarrow \mathbb{R}^N$ which is an affine mapping $\boldsymbol{f}(\mathbf{P}) = \boldsymbol{f}(\bar{\mathbf{P}}) + \boldsymbol{f}$, $\boldsymbol{\mathcal{J}}$ is of rank M. $\mathbf{V}$ is a vector of measurement with mean $\bar{\mathbf{V}}$ and covariance matrix $\boldsymbol{\Sigma}_\mathbf{V}$. Let $\boldsymbol{f}^{-1} \circ \boldsymbol{\eta}\mathbb{R}^N \rightarrow \mathbb{R}^M$ be the mapping that maps a measurement $\mathbf{V}$ to a set of parameters corresponding to the maximum likelihood estimate $\hat{\mathbf{V}}$. Then $\hat{\mathbf{P}} = \boldsymbol{f}^{-1} \circ \boldsymbol{\eta}(\mathbf{V})$ is a random variable with mean $\bar{\mathbf{P}}$ and covariance matrix*

$$\boldsymbol{\Sigma}_\mathbf{P} = (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}_\mathbf{V}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{-\mathbf{1}}. \tag{3.28}$$

When $\boldsymbol{f}$ is not an affine mapping, similar procedure can be used to approximate the mean and covariance. One may approximate $\boldsymbol{f}$ by and affine function, then we can obtain other theorem for backward propagation of covariance for non-linear case.

Theorem 4 *Consider a function $\boldsymbol{f} : \mathbb{R}^M \rightarrow \mathbb{R}^N$ which is differentiable mapping. Let $\boldsymbol{\mathcal{J}}$ be Jacobian matrix of $\boldsymbol{f}$ evaluated at a point $\bar{\mathbf{P}}$. Let $\mathbf{V}$ be a vector of measurement with mean $\bar{\mathbf{V}}$ and covariance matrix $\boldsymbol{\Sigma}_\mathbf{V}$. Let $\boldsymbol{f}^{-1} \circ \boldsymbol{\eta}\mathbb{R}^N \rightarrow \mathbb{R}^M$ be the mapping that maps a measurement $\mathbf{V}$ to a set of parameters corresponding to the maximum likelihood estimate $\hat{\mathbf{V}}$. Then to the first order approximation, $\hat{\mathbf{P}} = \boldsymbol{f}^{-1} \circ \boldsymbol{\eta}(\mathbf{V})$ is a random variable with mean $\bar{\mathbf{P}}$ and covariance matrix*

$$\boldsymbol{\Sigma}_\mathbf{P} = (\boldsymbol{\mathcal{J}}^\top \boldsymbol{\Sigma}_\mathbf{V}^{\mathbf{-1}} \boldsymbol{\mathcal{J}})^{-\mathbf{1}} \tag{3.29}$$

So far, we have discussed analytical methods of estimating covariance. These methods are based on an assumption of linearity, which the surface of $\boldsymbol{f}(\mathbf{P})$ is assumed to be locally flat. Therefore, the estimate of the covariance may not good when $\boldsymbol{f}$ is not locally flat in the vicinity of the estimated point.

Monte Carlo Estimation of Covariance

Monte Carlo method is a general approach to estimate the covariance. It relies on an exhaustive simulation to obtain numerical results, hence this method is computationally expensive. However, Monte Carlo method is very useful for estimating the actual covariance, and its results can be used as Ground truth data.

A Monte Carlo algorithm may start with a set of measurements that matches perfectly to the given parameters. It is usually assumed that the noise distribution of measurements is known beforehand. First, the noise drawn from the given distribution is added to the "perfect" measurements to obtain a synthetic corrupted measurements. The corresponding parameters are then computed by using the estimation procedure. This step is repeated for a number of times to obtain multiple trials of estimated parameters. Finally, the covariance matrix can be computed *statistically* from those trials.

3.2.4 Associate Uncertainty to a Hand-Eye Calibration Solution

In previous Section, we have discussed analytical and numerical methods for estimating covariance of a Maximum Likelihood Estimate. This Section will present how we adopt these methods to find the uncertainty of the hand-eye calibration solution.

In actual sensor calibration applications, due to presence of noise in the measurements, hand-eye calibration problems are usually formulated as *least-squares optimization problems*. Therefore, we can applied Theorem 4 and Theorem 2 to estimate the covariance matrix. As stated earlier, these two theorems are only valid when the mapping from parameter space to measurement space is locally flat in the vicinity of the estimated point. To prove this, it may require another work in the future. Instead, we will derive an analytical solution for estimating the covariance matrix and validate our method by comparing our results to the Ground truth data from a Monte Carlo method. The main challenge is that we now have to adopt this solution to the space of rigid body motion $\boldsymbol{SE}(3)$ (i.e when given optimization algorithms are only involved in the search on the space of $\boldsymbol{SE}(3)$ [21]) and/or to the space of rotation group $\boldsymbol{SO}(3)$ (i.e. when optimization algorithms are used to calculate the rotation part of the hand-eye transformation first, and then the translation part using the rotation result [18], [19], [20]). In this work, we will introduce our studies on adopting the covariance propagation methods to an existing solution of hand-eye calibration problem, which is a separable closed-form solution proposed in [20]. This method is fast but often yields error-prone solutions, that is because rotation estimation errors usually propagate to the translational part. The error-prone solutions, however, are more suitable for our studies since bigger errors in the solutions are easier to be compared when we validate our method.

A Separable Closed-Form Solution of Hand-Eye Calibration Problem

We first recall here the hand-eye solution proposed in [20]. The Eq. 3.22 can be decomposed into two equations as follows

$$\boldsymbol{R_A R_X} = \boldsymbol{R_X R_B}, \tag{3.30}$$

$$\boldsymbol{R_A t_X} + \boldsymbol{t_A} = \boldsymbol{R_X t_B} + \boldsymbol{t_X}, \tag{3.31}$$

where $\boldsymbol{R_A}, \boldsymbol{R_X}, \boldsymbol{R_B} \in \boldsymbol{SO}(3)$ are rotation parts, and $\boldsymbol{t_A}, \boldsymbol{t_X}, \boldsymbol{t_B} \in \mathbb{R}^3$ are translation parts of $\boldsymbol{A}, \boldsymbol{X}, \boldsymbol{B}$ transformations respectively.

Let $[\boldsymbol{\alpha}]$ and $[\boldsymbol{\beta}]$ denote logarithms of $\boldsymbol{R_A}$ and $\boldsymbol{R_B}$ respectively. Via logarithm mapping, Eq. 3.30 can also be written as

$$\log \boldsymbol{R_A} = \log \boldsymbol{R_X R_B R_X^\top} = \boldsymbol{R_X}[\boldsymbol{\beta}]\boldsymbol{R_X^\top}. \tag{3.32}$$

Applying an easy convention $\boldsymbol{R}[\boldsymbol{\omega}]\boldsymbol{R}^\top = [\boldsymbol{R\omega}]$ for $\boldsymbol{R} \in \boldsymbol{SO}(3)$ and $[\boldsymbol{\omega}] \in \boldsymbol{so}(3)$, it follows that

$$\boldsymbol{\alpha} = \boldsymbol{R_X \beta}. \tag{3.33}$$

Suppose that we have a set of k measurements $(\boldsymbol{A}_1, \boldsymbol{B}_1), (\boldsymbol{A}_2, \boldsymbol{B}_2), ..., (\boldsymbol{A}_k, \boldsymbol{B}_k)$. The first objective is to find $\boldsymbol{R_X}$ that minimizes

$$\psi_1 = \sum_i^k \|\boldsymbol{R_X}\boldsymbol{\beta}_i - \boldsymbol{\alpha}_i\|^2. \tag{3.34}$$

As suggested in [20], the optimal solution of $\boldsymbol{R_X}$ can be computed as

$$\boldsymbol{R_X} = (\boldsymbol{M}^\top \boldsymbol{M})^{-1/2}\boldsymbol{M}^\top, \tag{3.35}$$

where $\boldsymbol{M} = \sum \boldsymbol{\beta}_i \boldsymbol{\alpha}_i^\top$.

After obtaining the rotation solution, we then find $\boldsymbol{t_X}$ that minimizes

$$\psi_2 = \sum_i^k \|(\boldsymbol{R}_{\boldsymbol{A}_i} - \boldsymbol{I})\boldsymbol{t_X} - \boldsymbol{R_X}\boldsymbol{t}_{\boldsymbol{B}_i} + \boldsymbol{t}_{\boldsymbol{A}_i}\|^2. \tag{3.36}$$

The optimal solution of $\boldsymbol{R_X}$ is the standard least squares solution

$$\boldsymbol{t_X} = (\boldsymbol{C}^\top \boldsymbol{C})^{-1}\boldsymbol{C}^\top \boldsymbol{g}, \tag{3.37}$$

where

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{I} - \boldsymbol{R}_{\boldsymbol{A}_1} \\ \boldsymbol{I} - \boldsymbol{R}_{\boldsymbol{A}_2} \\ \dots \\ \boldsymbol{I} - \boldsymbol{R}_{\boldsymbol{A}_k} \end{bmatrix}, \boldsymbol{g} = \begin{bmatrix} \boldsymbol{t}_{\boldsymbol{A}_1} - \boldsymbol{R_X}\boldsymbol{t}_{\boldsymbol{B}_1} \\ \boldsymbol{t}_{\boldsymbol{A}_2} - \boldsymbol{R_X}\boldsymbol{t}_{\boldsymbol{B}_2} \\ \dots \\ \boldsymbol{t}_{\boldsymbol{A}_k} - \boldsymbol{R_X}\boldsymbol{t}_{\boldsymbol{B}_k} \end{bmatrix}. \tag{3.38}$$

Associate Uncertainty to the Hand-Eye Calibration Solution

It worths emphasizing that we now separate the objective function 3.23 into two least squares optimization problems, i.e. Eq. 3.34 and Eq. 3.36. Therefore, the uncertainty of the hand-eye transformation can be found by applying Theorem 4 to these two optimization problems.

Because of the presence of noise in the measurements, a set of noisy data $((\boldsymbol{A}_1, \boldsymbol{B}_1), ..., (\boldsymbol{A}_k, \boldsymbol{B}_k))$ is acquired. The optimization algorithm then try to find an $\mathbf{X}$ that minimizes the objective function 3.23. Assuming that measurements of $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ are perturbed by Gaussian noises [16]

$$\boldsymbol{A}_i = \exp([\boldsymbol{\xi_A}])\bar{\boldsymbol{A}}_i \tag{3.39}$$

$$\boldsymbol{B}_i = \exp([\boldsymbol{\xi_B}])\bar{\boldsymbol{B}}_i. \tag{3.40}$$

where $\bar{\boldsymbol{A}}_i$ and $\bar{\boldsymbol{B}}_i \in \boldsymbol{SE}(3)$ are the means of measurements $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$, respectively; $\boldsymbol{\xi_A}$ and $\boldsymbol{\xi_B} \in \mathbb{R}^6$ are noisy perturbations subjecting to zero-mean Gaussian with covariance matrices $\mathbf{\Sigma_A}$ and $\mathbf{\Sigma_B}$, respectively. This implies that each measurement of $\boldsymbol{A}_i$ (or $\boldsymbol{B}_i$) has exactly the same error distribution (i.e. when measurements are made using same devices). However, it worth emphasizing that our approach is still valid when uncertainty of all measurements $\boldsymbol{A}_i$ (or $\boldsymbol{B}_i$) are different and independent.

Further, in our case, we also assume that uncertainties of rotation component and translation component in the measurements are independent

$$\mathbf{\Sigma_{A_i}} = \mathbf{\Sigma_A} = \begin{bmatrix} \mathbf{\Sigma_{t_A}} & \mathbf{0_{3\times3}} \\ \mathbf{0_{3\times3}} & \mathbf{\Sigma_{R_A}} \end{bmatrix}, \mathbf{\Sigma_{B_i}} = \mathbf{\Sigma_B} = \begin{bmatrix} \mathbf{\Sigma_{t_B}} & \mathbf{0_{3\times3}} \\ \mathbf{0_{3\times3}} & \mathbf{\Sigma_{R_B}} \end{bmatrix}. \tag{3.41}$$

We first consider the computation of the covariance of rotation obtained from equations $\boldsymbol{\alpha}_i = \boldsymbol{R_X}\boldsymbol{\beta}_i$. Now, consider the simple case when error appears only in the measurements of $\boldsymbol{\alpha}_i$. We define the function $\boldsymbol{f}$ as a mapping from $\boldsymbol{R_X}$ matrix to the measurements $\boldsymbol{\alpha}_i = \boldsymbol{R_X}\bar{\boldsymbol{\beta}}_i$, where $\bar{\boldsymbol{\beta}}_i$ denotes the true measurement (error-free) of $\boldsymbol{\beta}_i$. The points $\boldsymbol{\alpha}_i$ construct a composite vector denoted by $\mathbf{V}$. We suppose that all measurements of $\boldsymbol{\alpha}_i$ are independent with their covariance matrices $\mathbf{\Sigma_{R_A}}$. Hence the covariance matrix of the measurement vector $\mathbf{V}$ has the block-diagonal form $\mathbf{\Sigma_V} = \mathbf{diag}(\mathbf{\Sigma}_{\boldsymbol{\alpha}_1}, \mathbf{\Sigma}_{\boldsymbol{\alpha}_2}, ..., \mathbf{\Sigma}_{\boldsymbol{\alpha}_\mathbf{k}}) = \mathbf{diag}(\mathbf{\Sigma_{R_A}}, \mathbf{\Sigma_{R_A}}, ..., \mathbf{\Sigma_{R_A}})$. The pipeline for computing the covariance of the estimated rotation is as follows

1. Estimated the $\hat{\boldsymbol{R}}_{\boldsymbol{X}}$ from the given data using Eq. 3.34;
2. Compute the Jacobian matrix $\boldsymbol{\mathcal{J}_f}$ evaluated at $\hat{\boldsymbol{R}}_{\boldsymbol{X}}$;
3. Compute the Covariance matrix $\mathbf{\Sigma_{R_X}}$ of the estimated rotation matrix by applying Theorem 4.

The first step in the pipeline is straightforward, thus we investigate the two last steps in a bit more details. The Jacobian matrix has a block structure of the form $\boldsymbol{\mathcal{J}_f} = \partial\mathbf{V}/\partial\boldsymbol{R_X} = (\boldsymbol{\mathcal{J}}_1^\top, \boldsymbol{\mathcal{J}}_2^\top, ..., \boldsymbol{\mathcal{J}}_k^\top)^\top$ where $\boldsymbol{\mathcal{J}}_i = \partial\boldsymbol{\alpha}_i/\partial\boldsymbol{R_X}$ and can be computed as follows

$$\boldsymbol{\mathcal{J}}_i = \left.\frac{\partial\boldsymbol{R_X}\boldsymbol{\beta}_i}{\partial\boldsymbol{R_X}}\right|_{\boldsymbol{R_X}=\hat{\boldsymbol{R}}_{\boldsymbol{X}}} = \left.\frac{\partial\exp([\boldsymbol{\xi}_{\boldsymbol{R_X}}])\hat{\boldsymbol{R}}_{\boldsymbol{X}}\boldsymbol{\beta}_i}{\partial\boldsymbol{\xi}_{\boldsymbol{R_X}}}\right|_{\boldsymbol{\xi}_{\boldsymbol{R_X}}=\mathbf{0}} = -[\hat{\boldsymbol{R}}_{\boldsymbol{X}}\boldsymbol{\beta}_i]. \tag{3.42}$$

Then by applying directly Theorem 4, one can compute

$$\mathbf{\Sigma_{R_X}} = (\boldsymbol{\mathcal{J}}_\mathbf{f}^\top\mathbf{\Sigma_V^{-1}}\boldsymbol{\mathcal{J}}_\mathbf{f})^{-1} = (\sum_\mathbf{i}^\mathbf{k}\boldsymbol{\mathcal{J}}_\mathbf{i}^\top\mathbf{\Sigma_{R_A}}\boldsymbol{\mathcal{J}}_\mathbf{i})^{-1} \tag{3.43}$$

In the case of error presenting in both $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ measurements, we can still follow the pipeline proposed previously to compute the covariance of the $\boldsymbol{R_X}$ solution. However, the two last steps are more complicated.

Suppose that the set of parameters $\mathbf{P}$ may be partitioned into two sets: $\boldsymbol{R_X}$ is the set of parameters describing the rotation matrix of interest, and a set of k parameters describing the estimates of $\boldsymbol{\beta}_i$. Let $\mathbf{V}$ denote the total measurement vector. The function $\boldsymbol{f}$ maps parameters $\mathbf{P}$ to $\mathbf{V} = (\mathbf{V}_1^\top, ..., \mathbf{V}_k^\top)^\top$ where each measurement vector $\mathbf{V}_i = (\boldsymbol{\beta}_i^\top, \boldsymbol{\alpha}_i^\top)^\top = (\boldsymbol{\beta}_i^\top, (\boldsymbol{R_X}\boldsymbol{\beta}_i)^\top)^\top$.

Once again, we assume that all measurements of both $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are independent with their covariance matrices $\mathbf{\Sigma_{R_A}}$ and $\mathbf{\Sigma_{R_B}}$ respectively. It is also assumed that the errors of $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ measurements are independent. Hence the covariance matrix $\mathbf{\Sigma_V} = \mathbf{diag}(\mathbf{\Sigma_{V_1}}, \mathbf{\Sigma_{V_2}}, ..., \mathbf{\Sigma_{V_k}})$ where $\mathbf{\Sigma_{V_i}} = \mathbf{diag}(\mathbf{\Sigma_{R_B}}, \mathbf{\Sigma_{R_A}})$. It is obvious that each $\boldsymbol{\alpha}_i$ is only dependent on $\boldsymbol{R_X}$ and $\boldsymbol{\beta}_i$, and not on the other parameters $\boldsymbol{\beta}_j$ where $i \neq j$.

Therefore, $\partial\boldsymbol{\alpha}_i/\partial\boldsymbol{\beta}_j = \mathbf{0}$ for $i \neq j$. Then one can compute the Jacobian matrix $\boldsymbol{\mathcal{J}}_{\boldsymbol{f}} = \partial\mathbf{V}/\partial\mathbf{P}$ as follows

$$\boldsymbol{\mathcal{J}}_{\boldsymbol{f}} = \left[\boldsymbol{\mathcal{J}}^{\boldsymbol{R_X}}|\boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}}\right] = \left[\begin{array}{c|cccc} \boldsymbol{\mathcal{J}}_1^{\boldsymbol{R_X}} & \boldsymbol{\mathcal{J}}_1^{\boldsymbol{\beta}} & & & \\ \boldsymbol{\mathcal{J}}_2^{\boldsymbol{R_X}} & & \boldsymbol{\mathcal{J}}_2^{\boldsymbol{\beta}} & & \\ \vdots & & & \ddots & \\ \boldsymbol{\mathcal{J}}_k^{\boldsymbol{R_X}} & & & & \boldsymbol{\mathcal{J}}_k^{\boldsymbol{\beta}} \end{array}\right],$$

where $\boldsymbol{\mathcal{J}}^{\boldsymbol{R_X}} = (\boldsymbol{\mathcal{J}}_1^{\boldsymbol{R_X}\top}, ..., \boldsymbol{\mathcal{J}}_k^{\boldsymbol{R_X}\top})^\top$, $\boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}} = diag(\boldsymbol{\mathcal{J}}_1^{\boldsymbol{\beta}}, ..., \boldsymbol{\mathcal{J}}_k^{\boldsymbol{\beta}})$, and

$$\boldsymbol{\mathcal{J}}_i^{\boldsymbol{R_X}} = \partial\mathbf{V}_i/\partial\boldsymbol{R_X} = \begin{bmatrix} \mathbf{0} \\ -[\hat{\boldsymbol{R}}_{\boldsymbol{X}}\boldsymbol{\beta}_i] \end{bmatrix}, \tag{3.44}$$

$$\boldsymbol{\mathcal{J}}_i^{\boldsymbol{\beta}} = \partial\mathbf{V}_i/\partial\boldsymbol{\beta}_i = \begin{bmatrix} \mathbb{I} \\ \hat{\boldsymbol{R}}_{\boldsymbol{X}} \end{bmatrix}. \tag{3.45}$$

Applying Theorem 4, one computes

$$\boldsymbol{\mathcal{J}}_{\boldsymbol{f}}^\top \boldsymbol{\Sigma}_{\mathbf{V}}^{-1} \boldsymbol{\mathcal{J}}_{\mathbf{f}} = \begin{bmatrix} \boldsymbol{\mathcal{J}}^{\boldsymbol{R_X}\top}\boldsymbol{\Sigma}_{\mathbf{V}}^{-1}\boldsymbol{\mathcal{J}}^{\mathbf{R_X}} & \boldsymbol{\mathcal{J}}^{\boldsymbol{R_X}\top}\boldsymbol{\Sigma}_{\mathbf{V}}^{-1}\boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}} \\ \boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}\top}\boldsymbol{\Sigma}_{\mathbf{V}}^{-1}\boldsymbol{\mathcal{J}}^{\mathbf{R_X}} & \boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}\top}\boldsymbol{\Sigma}_{\mathbf{V}}^{-1}\boldsymbol{\mathcal{J}}^{\boldsymbol{\beta}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{U} & \boldsymbol{W} \\ \boldsymbol{W}^\top & \boldsymbol{Z}. \end{bmatrix}. \tag{3.46}$$

The covariance matrix $\boldsymbol{\Sigma}_{\mathbf{P}}$ of the parameter set is the pseudo-inverse of this matrix. And the covariance matrix $\boldsymbol{\Sigma}_{\mathbf{R_X}}$ is located at the top left block of this pseudo-inverse. To calculate this pseudo-inverse is, however, very computationally expensive. We show next how to simplify the computation by making use of the block form of the Jacobian matrix.

One can easily simplify the submatrices in Eq. 3.46 according to

$$\boldsymbol{U} = \sum_i^k \boldsymbol{\mathcal{J}}_i^{\boldsymbol{R_X}\top}\boldsymbol{\Sigma}_{\mathbf{V_i}}^{-1}\boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\mathbf{R_X}} \tag{3.47}$$

$$\boldsymbol{W} = [\boldsymbol{W}_1, ..., \boldsymbol{W}_k] \text{ where } \boldsymbol{W}_i = \boldsymbol{\mathcal{J}}_i^{\boldsymbol{R_X}\top}\boldsymbol{\Sigma}_{\mathbf{V_i}}^{-1}\boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\boldsymbol{\beta}} \tag{3.48}$$

$$\boldsymbol{Z} = diag(\boldsymbol{Z}_1, ..., \boldsymbol{Z}_k) \text{ where } \boldsymbol{Z}_i = \boldsymbol{\mathcal{J}}_i^{\boldsymbol{\beta}\top}\boldsymbol{\Sigma}_{\mathbf{V_i}}^{-1}\boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\boldsymbol{\beta}}. \tag{3.49}$$

Furthermore, it is easy to see that $\boldsymbol{Z}$ is invertible. Therefore, the matrix in Eq. 3.46 can be diagonalized as follows

$$\boldsymbol{\mathcal{J}}_{\boldsymbol{f}}^\top \boldsymbol{\Sigma}_{\mathbf{V}}^{-1} \boldsymbol{\mathcal{J}}_{\mathbf{f}} = \begin{bmatrix} \boldsymbol{U} & \boldsymbol{W} \\ \boldsymbol{W}^\top & \boldsymbol{Z} \end{bmatrix} = \begin{bmatrix} \mathbb{I} & \boldsymbol{W}\boldsymbol{Z}^{-1} \\ \mathbf{0} & \mathbb{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{U} - \boldsymbol{W}\boldsymbol{Z}^{-1}\boldsymbol{W}^\top & \mathbf{0} \\ \mathbf{0} & \mathbb{Z} \end{bmatrix} \begin{bmatrix} \mathbb{I} & \mathbf{0} \\ (\boldsymbol{W}\boldsymbol{Z}^{-1})^\top & \mathbb{I} \end{bmatrix}. \tag{3.50}$$

Here, we assume the following identity $(\boldsymbol{G}\boldsymbol{H}\boldsymbol{G}^\top)^+ = \boldsymbol{G}^{-\top}\boldsymbol{H}^+\boldsymbol{G}^{-1}$ where $\boldsymbol{G}$ is an invertible matrix and $\boldsymbol{L}^+$ denote the pseudo-inverse of a matrix $\boldsymbol{L}$. Note that the derivations of conditions and proof for this identity are too complicated to be briefly explained here, more details can be found at Section A6 in [98]. Applying this identity to 3.50, one may obtain

$$\mathbf{\Sigma_P} = (\boldsymbol{\mathcal{J}}_\mathbf{f}^\top \mathbf{\Sigma_V^{-1}} \boldsymbol{\mathcal{J}}_\mathbf{f})^+ = \begin{bmatrix} \mathbf{\Sigma_{R_X}} & -\mathbf{\Sigma_{R_X} W Z^{-1}} \\ -(\mathbf{\Sigma_{R_X} W Z^{-1}})^\top & (\boldsymbol{W}\boldsymbol{Z}^{-1})^\top \mathbf{\Sigma_{R_X} W Z^{-1}} + \mathbf{Z^{-1}} \end{bmatrix} \tag{3.51}$$

where

$$\mathbf{\Sigma_{R_X}} = (\mathbf{U} - \mathbf{W Z^{-1} W^\top})^+ = (\mathbf{U} - \sum_{\mathbf{i}}^{\mathbf{k}} \mathbf{W_i Z_i}{}^{\mathbf{-1}} \mathbf{W_i}{}^\top)^+, \tag{3.52}$$

and $\boldsymbol{U}$, $\boldsymbol{W}$, $\boldsymbol{Z}$ are defined at Eq. 3.47, 3.48, 3.49, respectively.

We investigate now the computation of the covariance of the translation obtained from equations $\boldsymbol{R_X t_{B_i}} - \boldsymbol{t_{A_i}} = (\boldsymbol{R_{A_i}} - \boldsymbol{I})\boldsymbol{t_X}$, where $\boldsymbol{R_X}$ has just been computed using 3.34. By denoting $\boldsymbol{q}_i$ and $\boldsymbol{Q}_i$ as $\boldsymbol{R_X t_{B_i}} - \boldsymbol{t_{A_i}}$ and $(\boldsymbol{R_{A_i}} - \boldsymbol{I})$ respectively, these equations are simplified to

$$\boldsymbol{q}_i = \boldsymbol{Q}_i \boldsymbol{t_X}. \tag{3.53}$$

As errors also appear in $\boldsymbol{q}_i$ and $\boldsymbol{Q}_i$, we follow the same procedure as previous. Let $\mathbf{V}$ denote the total measurement vector. Suppose that the set of parameters $\mathbf{P}$ may be partitioned into two sets: $\boldsymbol{t_X}$ is the set of parameters describing the translation of interest, and a set of k parameters describing the estimates of $\boldsymbol{Q}_i$. The function $\boldsymbol{f}$ maps parameters $\mathbf{P}$ to $\mathbf{V} = (\mathbf{V}_1^\top, ..., \mathbf{V}_k^\top)^\top$ where each measurement vector $\mathbf{V}_i = (\boldsymbol{Q}_i^\top, \boldsymbol{q}_i^\top)^\top = (\boldsymbol{Q}_i^\top, (\boldsymbol{Q}_i \boldsymbol{t_X})^\top)^\top$.

Based on the assumption in 3.41, the covariance matrix $\mathbf{\Sigma_V} = \mathbf{diag}(\mathbf{\Sigma_{V_1}}, \mathbf{\Sigma_{V_2}}, ..., \mathbf{\Sigma_{V_k}})$ where $\mathbf{\Sigma_{V_i}} = \mathbf{diag}(\mathbf{\Sigma_{Q_i}}, \mathbf{\Sigma_{q_i}})$. By applying the forward propagation of covariance (*i.e.*Theorem 2), the covariance matrices $\mathbf{\Sigma_{Q_i}}$ and $\mathbf{\Sigma_{q_i}}$ can be computed as follows

$$\mathbf{\Sigma_{Q_i}} = \mathbf{\Sigma_{R_A}}, \tag{3.54}$$

$$\mathbf{\Sigma_{q_i}} = \mathbf{\Sigma_{t_A}} + \mathbf{\hat{R}_X \Sigma_{t_B} \hat{R}_X^\top} + [\mathbf{\hat{R}_X} \boldsymbol{\beta}_\mathbf{i}] \mathbf{\Sigma_{R_X}} [\mathbf{\hat{R}_X} \boldsymbol{\beta}_\mathbf{i}]^\top \tag{3.55}$$

Then the Jacobian matrix is of the form $\boldsymbol{\mathcal{J}_f} = \partial \mathbf{V} / \partial \mathbf{P} = \left[\boldsymbol{\mathcal{J}^{tx}} | \boldsymbol{\mathcal{J}^{Q}}\right]$ where

$$\boldsymbol{\mathcal{J}^{tx}} = (\boldsymbol{\mathcal{J}}_1^{\boldsymbol{tx}\top}, ..., \boldsymbol{\mathcal{J}}_k^{\boldsymbol{tx}\top})^\top, \tag{3.56}$$

$$\boldsymbol{\mathcal{J}}_i^{\boldsymbol{tx}} = \partial \mathbf{V} / \partial \boldsymbol{t_X} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{Q}_i \end{bmatrix}, \tag{3.57}$$

$$\boldsymbol{\mathcal{J}^{Q}} = diag(\boldsymbol{\mathcal{J}}_1^{\boldsymbol{Q}}, ..., \boldsymbol{\mathcal{J}}_k^{\boldsymbol{Q}}), \tag{3.58}$$

$$\boldsymbol{\mathcal{J}}_i^{\boldsymbol{Q}} = \partial \mathbf{V} / \partial \boldsymbol{Q} = \begin{bmatrix} \mathbb{I} \\ -[\boldsymbol{Q}_i \boldsymbol{t_X}] \end{bmatrix}. \tag{3.59}$$

Applying Theorem 4, one can compute the covariance matrix $\mathbf{\Sigma_P} = (\boldsymbol{\mathcal{J}}_\mathbf{f}^\top \mathbf{\Sigma_V^{-1}} \boldsymbol{\mathcal{J}}_\mathbf{f})^+$. Similar to previous derivation for the covariance of the estimated rotation (Eq. 3.52), the covariance matrix of the estimated translation is also located at the top left block of this pseudo-inverse, and is given as follows

$$\boldsymbol{\Sigma}_{\mathbf{t_X}} = (\mathbf{U} - \mathbf{W}\mathbf{Z}^{-\mathbf{1}}\mathbf{W}^{\top})^{+} = (\mathbf{U} - \sum_{\mathbf{i}}^{\mathbf{k}} \mathbf{W_i}\mathbf{Z_i}^{-\mathbf{1}}\mathbf{W_i}^{\top})^{+}. \tag{3.60}$$

where

$$\boldsymbol{U} = \sum_{i}^{k} \boldsymbol{\mathcal{J}}_{i}^{\boldsymbol{tx}\top} \boldsymbol{\Sigma}_{\mathbf{V_i}}^{-\mathbf{1}} \boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\mathbf{tx}} \tag{3.61}$$

$$\boldsymbol{W} = [\boldsymbol{W}_1, ..., \boldsymbol{W}_k] \text{ with } \boldsymbol{W}_i = \boldsymbol{\mathcal{J}}_{i}^{\boldsymbol{tx}\top} \boldsymbol{\Sigma}_{\mathbf{V_i}}^{-\mathbf{1}} \boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\mathbf{Q}} \tag{3.62}$$

$$\boldsymbol{Z} = diag(\boldsymbol{Z}_1, ..., \boldsymbol{Z}_k) \text{ with } \boldsymbol{Z}_i = \boldsymbol{\mathcal{J}}_{i}^{\boldsymbol{Q}\top} \boldsymbol{\Sigma}_{\mathbf{V_i}}^{-\mathbf{1}} \boldsymbol{\mathcal{J}}_{\mathbf{i}}^{\mathbf{Q}}. \tag{3.63}$$

3.2.5 Simulation Results

To validate the method proposed in previous Section, we show here a numerical experiment in which we compared the results from our method with the benchmark drawn from Monte Carlo simulations. Our implementation is open-source and can be found online at [101]

Suppose that the trues value of the rotation and translation of the hand-eye transformation $\boldsymbol{X}$ are as follows

$$\bar{\boldsymbol{R}}_X = \begin{bmatrix} -0.17556632 & 0.869125 & 0.46238318 \\ 0.95274827 & 0.03173922 & 0.30209826 \\ 0.24788547 & 0.49357306 & -0.83362967 \end{bmatrix}, \bar{\boldsymbol{t}}_X = \begin{bmatrix} 0.341213 & 0.123214 & -0.2 \end{bmatrix} \tag{3.64}$$

We first generate k set of uncorrupted pairs of $\bar{\boldsymbol{A}}_i, \bar{\boldsymbol{B}}_i$ that satisfy the equation $\bar{\boldsymbol{A}}_i\boldsymbol{X} = \boldsymbol{X}\bar{\boldsymbol{B}}_i$ where the rotation component of $\bar{\boldsymbol{B}}_i$,which is $\bar{\boldsymbol{\beta}}_i$, are sampled uniformly over the solid ball of radius π, and each entry of the translation part $\bar{\boldsymbol{t}}_{\boldsymbol{B}_i}$ is selected in the range of $[-1, 1]$.

Each pair of $\bar{\boldsymbol{A}}_i, \bar{\boldsymbol{B}}_i$ are now perturbed by Gaussian noises as follows

$$\boldsymbol{A}_i = \exp([\boldsymbol{\xi}_{\boldsymbol{A}}])\bar{\boldsymbol{A}}_i \tag{3.65}$$

$$\boldsymbol{B}_i = \exp([\boldsymbol{\xi}_{\boldsymbol{B}}])\bar{\boldsymbol{B}}_i. \tag{3.66}$$

where $\boldsymbol{\xi}_{\boldsymbol{A}}$ and $\boldsymbol{\xi}_{\boldsymbol{B}} \in \mathbb{R}^6$ are noisy perturbations subjecting to zero-mean Gaussian with covariance matrices $\boldsymbol{\Sigma}_{\mathbf{A}}$ and $\boldsymbol{\Sigma}_{\mathbf{B}}$, respectively.

$$\boldsymbol{\Sigma}_{\mathbf{A}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{t_A}} & \mathbf{0_{3\times3}} \\ \mathbf{0_{3\times3}} & \boldsymbol{\Sigma}_{\mathbf{R_A}} \end{bmatrix}, \boldsymbol{\Sigma}_{\mathbf{B}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{t_B}} & \mathbf{0_{3\times3}} \\ \mathbf{0_{3\times3}} & \boldsymbol{\Sigma}_{\mathbf{R_B}} \end{bmatrix}. \tag{3.67}$$

Suppose the covariance matrices are

$$\mathbf{\Sigma_A} = \lambda \times \text{diag}(4\ , 9,\ 3,\ 0.2,\ 0.5,\ 0.3) \tag{3.68}$$

$$\mathbf{\Sigma_B} = \lambda \times \text{diag}(3\ , 6,\ 8,\ 0.3,\ 0.7,\ 0.5) \tag{3.69}$$

where $\lambda \in [0, 1] * 10^{-5}$ is a scaling parameters that allows us to change the manitude of the uncertainty.

The estimated covariance matrices are then computed as proposed in Eq. 3.52 and 3.60. This are then compared to the results from Monte Carlo method, in which we keep regenerating the random data and finding the estimates using Eq. 3.35, 3.37 for a lagre number (500) of trials. The *true* covariances are computed statistically from the errors between the estimated rotations/translations and the true ones.

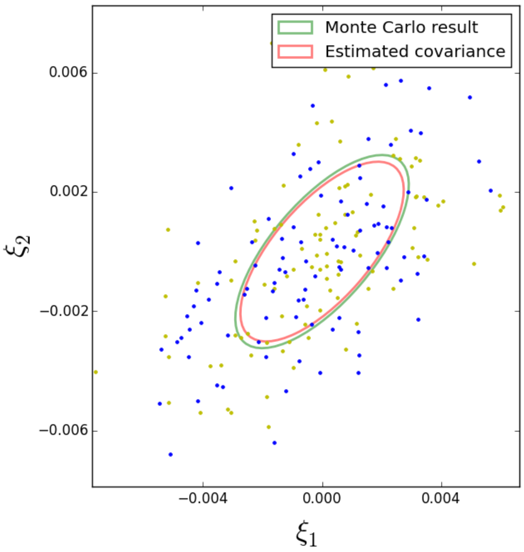


Figure 3.6: One-standard-deviation covariance ellipses, given by $\mathbf{\Sigma_{t_X}}$, shown for Monte Carlo and our results.

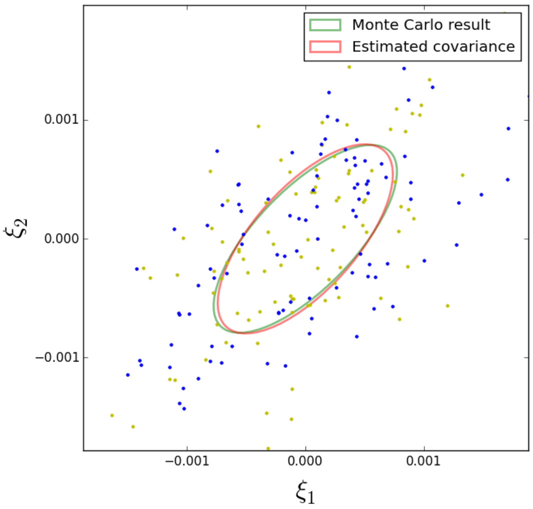


Figure 3.7: One-standard-deviation covariance ellipses, given by $\mathbf{\Sigma_{R_X}}$, shown for Monte Carlo and our results

Figure 3.6 illustrates one-standard-deviation covariance ellipses, given by $\mathbf{\Sigma}_{\mathbf{t_X}}$, shown for Monte Carlo and our results. Note that although we obtain a 3×3 covariance matrix, it is confined to a plane for the ease of illustration and plotting. Hence, we only show here the covariance ellipse corresponding to the first two elements of the translation. Similarly, Figure 3.7 compares the one-standard-deviation covariance ellipses, given by $\mathbf{\Sigma}_{\mathbf{R_X}}$, obtained by Monte Carlo and our results. It can be seen that our method can provide very good estimations of the covariance matrices of the estimated hand-eye transformation.

We did not compare the computational cost since our method is extremely efficient as compared to Mote Carlo method.

3.2.6 Conclusion

In this Section, we has presented our studies on adopting the covariance propagation methods to an existing solution of hand-eye calibration problem *i.e.*a separable closed-form solution proposed in [20]. The hand-eye calibration problem is formulated as a least-squares minimization problem, then propagation of covariance methods from statistics theory are utilized to obtain the covariance matrix of the solution. The simulation result shows that our methods can deliver a very good approximation of the real covariance matrix of the hand-eye transformation.

Chapter 4

Future plan

Study on rigid body motion has long been an interesting field that attracts an extensive attention from many researchers in the scope of robotics. The research fields range from motion planning, mobile robot to computer vision and other related topics. Despite the fact that a number of theoretical treatments for working on rigid body motion have been developed for a long time, Lie theory has proved to be the most effective tool used to resolve many principal issues corresponding to this matter. To be specific, this research focuses on the crucial problems related to motion planning and estimation in the space of rigid body motion. As mentioned earlier, these two problems are not new in the literature, however, by taking the advantage of using the analytic tools derived from Lie group theory and Riemannian geometry, we aim at proposing new, general, fast approaches to address these problems.

In regard of motion planning, through implementing numerous experiments and investigation, we expect to discover a variety of practical applications, namely multi-spacecraft coordination, humanoid robots and computer animation, based on our work proposed in Section 3.1. On the other hand, there are various potential directions in both practical and theoretical orientation to be explored in the estimation problems under the presence of uncertainty. Herewith are

1. Associating uncertainty to the estimate of hand-eye transformation;
2. Associating uncertainty to the pose of an object estimated by using sensory devices;
3. Using the information of estimated uncertainty to build a more robust, autonomous robotic systems which are able to cope with many complex tasks in various types of environment. In other words, knowledge of the uncertainties estimated from the first two problems will be incorporated into the system, in order to enhance the robustness of the automation system. This, also, can be considered as the ultimate goal for our research.

The rest of this Chapter will discuss these research directions in more details and give a statement of our objective as well as how we plan to tackle these challenging but interesting problems.

4.1 On the Uncertainty of Hand-Eye Calibration Problems

In Section 3.2 we has presented our studies on adopting the covariance propagation methods to an existing solution of hand-eye calibration problem *i.e.*a separable closed-form solution proposed in [20]. Until now, the proposed theory mentioned in the research has just been validated through a number of simulation experiments implemented. We, hence, expect to conduct more practical experiments on the real robotic systems with the aim of testifying and proving our methods.

Due to the propagation of the error from rotation to translation, the method proposed in [20] usually yields error-prone solutions. Our next plan is to extent our work to other solutions of the hand-eye calibration problem which may give us a better estimated transformation. To be more specific, the optimization algorithms directly exploiting the matrix Lie group structure of $\boldsymbol{SE}(3)$; for instance, Newton and Newton-type iterative algorithms that evolve directly on the search space of $\boldsymbol{SE}(3)$; will be taken into account in this case. As mentioned earlier, the main challenges mainly lie on how to apply propagation covariance methods into that optimization algorithm while still maintaining the search on the space of rigid body motions $\boldsymbol{SE}(3)$.

Other issues may correspond to avoid the instability and the poor convergence of the optimization methods. Moreover, since many current applications do not require to calibrate frequently, computational efficiency has not been a major concern in some previous works. We may also aim to propose a fast and robust algorithm that provides more accurate and informative solutions.

4.2 On the Uncertainty of Object Estimation Problems

To estimate the object pose with respect to the sensory device frame, there are extensive, fully-developed methods have been proposed in computer vision community. However, when robots operate in a complex environment, uncertainties are unavoidable. Since such uncertainties may lead to the failure of the system, it is important to model the uncertainties of pose estimates provided by commonly used pose estimation algorithms. As mentioned in Section 2.2.2, this problem has draw a lot of attention and has been well-explored in the literature. Therefore, we will analyze existing methods with regard to practical aspect, then adopt these methods into our framework.

4.3 Dealing with Uncertainty in Assembling Systems

Thanks to the estimated uncertainties that may appear when the assembling systems operate, the knowledge of the surrounding space and the systems can be significantly enhanced based on the collected information. As a result, the systems are able to cope with a various range of more complex tasks in different kinds of environment. This is also set as one of the crucial objectives of our further work in this field.

To tackle this problem, one common solution is to reduce the uncertainties of the estimations of the objects by making use of more accurate sensors *i.e.*tactile, force/torque sensors. There are many extensive works regarding to the methods used to solve this problem. A prime example is to utilize touch information to reduce the initial uncertainty of the object pose given by vision [28],[29],[30], [31]. After investigating various methods by different researchers, we find the approach by Petrovskaya *et al.*(can be seen at [31]) the most efficient one in this case. Therefore, in order to achieve the initial set targets, we have embarked on adopting this approach in some ways for the mentioned problem. This adoption has been particularly applied into dealing with the problem related to pin insertion. The demonstration video can be found at `https://youtu.be/xk9J_zznCVk` for more reference.

In addition, another solution is to develop a fully flexible, robust planning algorithm that can operate successfully and stably in the presence of uncertainty. As reviewed in Section 2.2.2, there are various potential directions in both practical and theoretical aspects to be explored in the manipulation planning under the presence of uncertainty. For instance, works presented in [38] and [39] are able to incorporate sensor uncertainty into the planning stage in high-dimensional problems; and non-prehensile actions can also be used to reduce the uncertainty of the object pose, as presented in 2.6, [34]. We, therefore, can also study in more depth forwards these directions.

Bibliography

[1] J. Gallier, "Notes on differential geometry and lie groups," *University of Pennsylvannia*, 2012.

[2] F. C. Park, *The optimal kinematic design of mechanisms.* PhD thesis, Havard University, 1991.

[3] G. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 1: Classical Results and Geometric Methods*, vol. 1. Springer Science & Business Media, 2009.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics.* MIT press, 2005.

[5] R. H. Taylor, *The Synthesis of Manipulator Control Programs from Task-level Specifications.* PhD thesis, Stanford University, Stanford, CA, USA, 1976. AAI7707174.

[6] R. A. Brooks, "Symbolic error analysis and robot planning," *The International Journal of Robotics Research*, vol. 1, no. 4, pp. 29–78, 1982.

[7] S. Su and C. Lee, "Uncertainty manipulation and propagation and verification of applicability of actions in assembly tasks," in *IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2471–2476, 1991.

[8] S.-F. Su and C. G. Lee, "Manipulation and propagation of uncertainty and verification of applicability of actions in assembly tasks," *Systems Science and Cybernetics, IEEE Transactions on*, vol. 22, no. 6, pp. 1376–1389, 1992.

[9] R. A. Brooks, "Visual map making for a mobile robot," in *Robotics and Automation, 1985. Proceedings. IEEE International Conference on*, vol. 2, pp. 824–829, IEEE, 1985.

[10] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *IEEE Journal of Robotics and Automation*, vol. 5, no. 4, pp. 56–68, 1986.

[11] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous robot vehicles*, pp. 167–193, Springer, 1990.

[12] H. F. Durrant-Whyte, "Uncertainty geometry in robotics," *IEEE Journal of Robotics and Automation*, vol. 4, no. 1, pp. 23–31, 1988.

[13] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 2, pp. 99–110, 2006.

[14] Y. Wang and G. S. Chirikjian, "Nonparametric second-order theory of error propagation on motion groups," *International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1258–1273, 2008.

[15] G. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, vol. 2. Springer Science & Business Media, 2011.

[16] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Transactions on Robotics*, vol. 30, pp. 679 – 693, June 2014.

[17] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 1, 1989.

[18] C. C. Wang, "Extrinsic calibration of a vision sensor mounted on a robot," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 2, pp. 161–175, 1992.

[19] R. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, 1989.

[20] F. C. Park and B. J. Martin, "Robot sensor calibration: solving ax= xb on the euclidean group," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, 1994.

[21] K. H. Strobl and G. Hirzinger, "Optimal hand-eye calibration," in *Robotics: Science and Systems*, pp. 4647–4653, 2006.

[22] J. Ha, D. Kang, and F. C. Park, "A stochastic global optimization algorithm for the two-frame sensor calibration problem," *IEEE Transactions on Industrial Electronics*, 2015.

[23] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Robotics-DL tentative*, pp. 586–606, International Society for Optics and Photonics, 1992.

[24] A. J. Stoddart, S. Lemke, A. Hilton, and T. Renn, "Estimating pose uncertainty for surface registration," *Image and Vision Computing*, vol. 16, no. 2, pp. 111–120, 1998.

[25] O. Bengtsson and A.-J. Baerveldt, "Robot localization based on scan-matching—estimating the covariance matrix for the idc algorithm," *Robotics and Autonomous Systems*, vol. 44, no. 1, pp. 29–40, 2003.

[26] A. Censi, "An accurate closed-form estimate of icp's covariance," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3167–3172, IEEE, 2007.

[27] N. Gelfand, L. Ikemoto, S. Rusinkiewicz, and M. Levoy, "Geometrically stable sampling for the icp algorithm," in *3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, pp. 260–267, IEEE, 2003.

[28] K. Gadeyne and H. Bruynickx, "Markov techniques for object localization with force-controlled robots," in *International Conference on Advanced Robotics*, 2001.

[29] S. R. Chhatpar and M. S. Branicky, "Particle filtering for localization in robotic assemblies with position uncertainty," in *IEEE/RSJ International Conference on Intelligent Robots and SystemsIntelligent Robots and Systems, 2005 (IROS 2005)*, pp. 3610–3617, 2005.

[30] K. K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez, "Task-driven tactile exploration," in *Robotics: Science and Systems*, 2010.

[31] A. Petrovskaya and O. Khatib, "Global localization of objects via touch," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 569–585, 2011.

[32] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis - a survey," *IEEE Transactions on Robotics*, vol. 30, pp. 289 – 309, 2014.

[33] J. Illonen, J. Bohg, and V. Kyrki, "3-d object reconstruction of symmetric objects by fusing visual and tactile sensing," *The International Journal of Robotics Research*, vol. 33, pp. 321–341, Oct. 2013.

[34] M. Dogar and S. Srinivasa, "A framework for push-grasping in clutter," *Robotics: Science and systems VII*, vol. 1, 2011.

[35] T. Lozano-Perez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *The International Journal of Robotics Research*, vol. 3, no. 1, pp. 3–24, 1984.

[36] M. Erdmann, "Using backprojections for fine motion planning with uncertainty," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 19–45, 1986.

[37] J. Canny, "On computability of fine motion plans," in *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pp. 177–182, IEEE, 1989.

[38] B. Burns and O. Brock, "Sampling-based motion planning with sensing uncertainty," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3313–3318, IEEE, 2007.

[39] D. Berenson, S. S. Srinivasa, and J. J. Kuffner, "Addressing pose uncertainty in manipulation planning using task space regions," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1419–1425, IEEE, 2009.

[40] K. M. Lynch, "Toppling manipulation," in *Robotics and Automation (ICRA), 1999 IEEE International Conference on*, pp. 2551–2557, 1999.

[41] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, "Planning pre-grasp manipulation for transport tasks," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2697–2704, IEEE, 2010.

[42] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566–580, 1996.

[43] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," tech. rep., Iowa State University, 1998.

[44] J.-C. Latombe, "Motion planning: A journey of robots, molecules, digital actors, and other artifacts," *The International Journal of Robotics Research*, vol. 18, no. 11, pp. 1119–1128, 1999.

[45] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[46] T. Lozano-Pérez, "Spatial planning: A configuration space approach," *Computers, IEEE Transactions on*, vol. 100, no. 2, pp. 108–120, 1983.

[47] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility-polygon search and euclidean shortest paths," in *Foundations of Computer Science, 1985., 26th Annual Symposium on*, pp. 155–164, IEEE, 1985.

[48] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *Systems Science and Cybernetics, IEEE Transactions on*, vol. 4, no. 2, pp. 100–107, 1968.

[49] A. Stentz, "Optimal and efficient path planning for unknown and dynamic environments," tech. rep., DTIC Document, 1993.

[50] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proceedings of the 20th Annual IEEE Conference on Foundations of Computer Science*, pp. 421–427, 1979.

[51] J. T. Schwartz and M. Sharir, "On the "piano movers" problem. ii. general techniques for computing topological properties of real algebraic manifolds," *Advances in Applied Mathematics*, vol. 4, no. 3, pp. 298–351, 1983.

[52] S. LaValle, *Planning algorithms.* Cambridge Univ Press, 2006.

[53] N. M. Amato, O. B. Bayazit, and L. K. Dale, "Obprm: An obstacle-based prm for 3d workspaces," 1998.

[54] N. M. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 1, pp. 113–120, IEEE, 1996.

[55] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The gaussian sampling strategy for probabilistic roadmap planners," in *Robotics and Automation, 1999. Proceedings. ICRA '99. IEEE International Conference on*, vol. 2, pp. 1018–1023, 1999.

[56] D. Hsu, T. Jiang, J. Reif, and Z. Sun, "The bridge test for sampling narrow passages with probabilistic roadmap planners," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, 2003.

[57] C. Holleman and L. E. Kavraki, "A framework for using the workspace medial axis in prm planners," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2, pp. 1408–1413, IEEE, 2000.

[58] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *Robotics and Automation, 1999. Proceedings. ICRA '99. IEEE International Conference on*, vol. 2, pp. 1024–1031, 1999.

[59] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe, "Analysis of probabilistic roadmaps for path planning," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 1, pp. 166–171, 1998.

[60] A. M. Ladd and L. E. Kavraki, “Measure theoretic analysis of probabilistic path planning,” *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 2, pp. 229–242, 2004.

[61] D. Hsu, J.-C. Latombe, and H. Kurniawati, “On the probabilistic foundations of probabilistic roadmap planning,” *The International Journal of Robotics Research*, vol. 25, no. 7, pp. 627–643, 2006.

[62] P. Cheng and S. M. LaValle, “Resolution complete rapidly-exploring random trees,” in *Robotics and Automation, 2002. Proceedings. ICRA ’02. IEEE International Conference on*, pp. 267–272, 2002.

[63] G. Sánchez and J.-C. Latombe, “A single-query bi-directional probabilistic roadmap planner with lazy collision checking,” in *Robotics Research*, pp. 403–417, 2003.

[64] S. R. Lindemann and S. M. LaValle, “Incrementally reducing dispersion by increasing voronoi bias in RRTs,” in *Robotics and Automation, 2004. Proceedings. ICRA ’04. IEEE International Conference on*, vol. 4, pp. 3251–3257, 2004.

[65] A. Yershova, L. Jaillet, T. Siméon, and S. M. LaValle, “Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain,” in *Robotics and Automation, 2005. Proceedings. ICRA ’05, IEEE International Conference on*, pp. 3856–3861, 2005.

[66] M. Akinc, K. E. Bekris, B. Y. Chen, A. M. Ladd, E. Plaku, and L. E. Kavraki, “Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps,” in *Robotics Research*, pp. 80–89, Springer, 2005.

[67] N. Vahrenkamp, D. Berenson, T. Asfour, J. Kuffner, and R. Dillmann, “Humanoid motion planning for dual-arm manipulation and re-grasping task,” in *Intelligent Robots and Systems (IROS), 2009 IEEE/RSJ International Conference on*, pp. 2464–2470, 2009.

[68] L. Keselman, E. Verriest, and P. A. Vela, “Forage RRT-an efficient approach to task-space goal planning for high dimensional systems,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 1572–1577, 2014.

[69] J. Kuffner and S. LaValle, “RRT-connect: An efficient approach to single-query path planning,” in *IEEE International Conference on Robotics and Automation*, 2000.

[70] S. M. LaValle and J. J. Kuffner Jr, “Rapidly-exploring random trees: Progress and prospects,” 2000.

[71] S. LaValle and J. Kuffner, “Randomized kinodynamic planning,” *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[72] P. Cheng and S. M. LaValle, “Reducing metric sensitivity in randomized trajectory design,” in *Intelligent Robots and Systems (IROS), 2001 IEEE/RSJ International Conference on*, vol. 1, pp. 43–48, 2001.

[73] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo, “Choosing good distance metrics and local planners for probabilistic roadmap methods,” in *Robotics and Automation, 1998. Proceedings. ICRA ’98. IEEE International Conference on*, vol. 1, pp. 630–637, 1998.

[74] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. H. Reif, “Narrow passage sampling for probabilistic roadmap planning,” *Robotics, IEEE Transactions on*, vol. 21, no. 6, pp. 1105–1115, 2005.

[75] H. Kurniawati and D. Hsu, “Workspace importance sampling for probabilistic roadmap planning,” in *Intelligent Robots and Systems (IROS), 2004 IEEE/RSJ International Conference on*, vol. 2, pp. 1618–1623, 2004.

[76] S. Rodriguez, X. Tang, J.-M. Lien, and N. M. Amato, “An obstacle-based rapidly-exploring random tree,” in *Robotics and Automation, 2006. Proceedings. ICRA ’06, IEEE International Conference on*, pp. 895–900, 2006.

[77] B. Donald, P. Xavier, J. Canny, and J. Reif, “Kinodynamic motion planning,” *Journal of the ACM (JACM)*, vol. 40, pp. 1048–1066, 1993.

[78] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, “Randomized kinodynamic motion planning with moving obstacles,” *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 233–255, 2002.

[79] D. Hsu, J.-C. Latombe, and R. Motwani, “Path planning in expansive configuration spaces,” in *Robotics and Automation, 1997. Proceedings. ICRA ’97. IEEE International Conference on*, vol. 3, pp. 2719–2726, 1997.

[80] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: the path-velocity decomposition," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 72–89, 1986.

[81] J. J. Kuffner Jr, S. Kagami, K. Nishiwaki, M. Inaba, and H. Inoue, "Dynamically-stable motion planning for humanoid robots," *Autonomous Robots*, vol. 12, no. 1, pp. 105–118, 2002.

[82] Q.-C. Pham, S. Caron, and Y. Nakamura, "Kinodynamic planning in the configuration space via admissible velocity propagation," in *Proceedings of Robotics: Science and Systems*, 2013.

[83] F. Li and P. M. Bainum, "Numerical approach for solving rigid spacecraft minimum time attitude maneuvers," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 1, pp. 38–45, 1990.

[84] K. D. Bilimoria and B. Wie, "Time-optimal three-axis reorientation of a rigid spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 3, pp. 446–452, 1993.

[85] X. Bai and J. L. Junkins, "New results for time-optimal three-axis reorientation of a rigid spacecraft," *Journal of guidance, control, and dynamics*, vol. 32, no. 4, pp. 1071–1076, 2009.

[86] J. J. Kuffner, "Effective sampling and distance metrics for 3d rigid body path planning," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 4, pp. 3993–3998, IEEE, 2004.

[87] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

[88] K. Shin and N. McKay, "Minimum-time control of robotic manipulators with geometric path constraints," *Automatic Control, IEEE Transactions on*, vol. 30, no. 6, pp. 531–541, 1985.

[89] Q.-C. Pham, "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *IEEE Transactions on Robotics*, vol. 6, pp. 1533–1540, 2014.

[90] H. Nguyen and Q.-C. Pham, "Time-optimal path parameterization of rigid-body motions: applications to spacecraft reorientation," *Journal of Guidance, Control, and Dynamics*, pp. 1–5, 2016.

[91] R. Geraerts and M. Overmars, "Creating high-quality paths for motion planning," *The International Journal of Robotics Research*, vol. 26, no. 8, pp. 845–863, 2007.

[92] F. Park and B. Ravani, "Bezier curves on riemannian manifolds and lie groups with kinematics applications," *Journal of Mechanical Design*, vol. 117, no. 1, pp. 36–40, 1995.

[93] F. C. Park and B. Ravani, "Smooth invariant interpolation of rotations," *ACM Transactions on Graphics (TOG)*, vol. 16, no. 3, pp. 277–295, 1997.

[94] L. Žlajpah, "On time optimal path control of manipulators with bounded joint velocities and torques," in *Robotics and Automation, 1996. Proceedings. ICRA '96, IEEE International Conference on*, vol. 2, pp. 1572–1577, 1996.

[95] H. Nguyen and Q.-C. Pham, "Topp for rigid-body motions," https://github.com/dinhhuy2109/TOPP-SO3 [retrieved 30 Oct. 2015].

[96] H. Nguyen and Q.-C. Pham, "Topp for rigid-body motions," https://youtu.be/heM7uxGrfVc [retrieved 30 Oct. 2015].

[97] NASA-3D-Resources, "Messenger 3d model," http://nasa3d.arc.nasa.gov/detail/eoss-messenger [retrieved 30 Oct. 2015].

[98] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision.* Cambridge university press, 2003.

[99] K. Kanatani, *Statistical optimization for geometric computation: theory and practice.* Courier Corporation, 2005.

[100] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes in C*, vol. 2. Citeseer, 1996.

[101] H. Nguyen, "On the uncertainty of the hand-eye calibration problem," https://github.com/dinhhuy2109/axxb-covariance 2016.