Progress Report

Nguyen Dinh Huy*

August 20, 2015

1 Introduction

Robotics has greatly contributed to the increase of global productivity over the past decades through such applications as pick and place, welding, or painting. However, there are still major challenges that prevent the automation of fine assembly tasks (e.g. in the electronics, shoes, food industries). As opposed to heavy industries where sophisticated assembly lines are associated with highly structured environments (see e.g. car assembly lines, where the positions of the car frames are well determined and known in advance to the robot programmer), fine assembly tasks are usually associated with unstructured environments: the small parts to be assembled may come in very diverse positions and orientations. In order to make robots capable of handling those challenges in fine assembly tasks in industrial contexts, we shall develop a intelligent perception system using cheap, off-the-shelf components combined with usual position-controlled industrial manipulators. In particular, we shall develop the theoretical and technological framework to (i) build a general, fast, robust algorithms and implementation, so that a robotic arm can automatically recover and "understand" its working space, (ii) integrate 3D perception, tactile perception. Since the main problem of most 3D perception devices is their relatively rough resolution (which make them unsuitable for fine manipulation tasks), the integration of 3D perception and tactile perception helps refining the environment description.

Therefore, this work will require me a good background on perception especially in 3D reconstruction, object recognitions, object pose estimations, integration of different modes of perception (3D and tactile perceptions); and also on robot control, motion planning. To do so, in the first year of my PhD study, I have focused on developing my understanding of this research area; also building my skills on 3D reconstruction, robot sensor calibration, programming robot algorithms including motion planning, kinematics, dynamics, experiencing with robot simulation environments.

This report will first summarize the status of my progress in the first year of PhD study, including some preliminary studies that I have been accomplished for last two semesters. Although there are still some limitations in the results I have achieved so far, these related works have been the good background for me to complete my primary research. This report is organized as follows. In Section 2, a review and discussion on the 3D scene reconstruction problem are presented. In Section 3, I then report my studies on

*School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore

3D perception using Kinect and Optitrack system, and on robot sensor calibration. Section 4 then presents my work on Time-Optimal Path Parameterization in SO(3) which later on can be applied to many robot motion planning and computer animation problems. Section 5 will conclude and sketch my future works to pursuit this research.

2 3D scene reconstruction

3D scene reconstruction has always been a challenging problem in computer vision. With the increasing demand for robotic automation over the last 20 years, this challenge has become an active area of research which received much attention from many researchers in various aspects of the problem. One of the primary foci of my research is to develop an intelligent sensing system that is able to automatically create a 3D digital representation of a scene of interest (e.g. working space of the robot) without any a priori information. Given a calibrated 3D depth sensor mounted on the end effector of a robotic arm, we shall develop strategies to automatically determine sensor parameters (position and orientation) that help recovering the unknown scene structure. Sensor parameters then can be purposefully selected and changed in order to perform the tasks more flexibly, autonomously, and reliably. The information obtained from such systems can be used to design an automatic robotic system that can handle most of the challenging tasks such as manipulation, assembly, exploration and navigation.

The rest of this section therefore will start with a brief description of a typical sensory system where a 3D sensor mounted on a positioning system. Subsequently, a review and discussion on the classical scene reconstruction cycle are presented.

2.1 The sensory system

When a robot is operating in its working space, the ability to move or manipulate without colliding with any obstacle in the environment is crucial. In order to do so, a robotic system need to be able to "see" and "understand" the surrounding environment. To do so, our approach is to develop strategies to observe all the parts of the environment in the working space to ensure the completeness of the reconstruction. Therefore, it requires a range camera (which will capture visual data about the surrounding environment) and a positioning system (which will transfer the range camera to the next appropriate positions), see Figure 1.

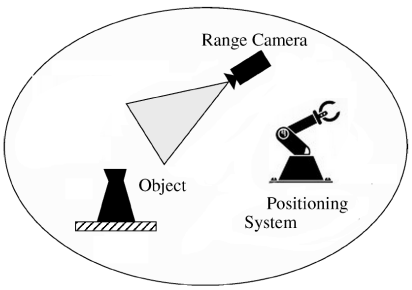

Figure 1: The imaging environment

Range camera is the principal components of a sensory system. Depending on the technology, it can be categorized into active or passive sensor. Passive camera can only provide sparse depth map, and depends mostly on the texture of the objects/environment. Active sensors however are able to provide dense depth map and are less affected by environment conditions (e.g. illumination, objects' texture). In 2010, Microsoft introduced Kinect– a low cost, high-resolution depth and RGB sensor. With a rich data from the combination of depth and RGB information, this device has received much attention from many researchers. Many algorithms and new tools have been developed, that can be applied to solve many 3D vision problems including 3D scene reconstruction.

Observing all parts of the environment requires a positioning system to move the sensor. In this research, a 6 DOFs robot arm will be used to explore its working space using a Kinect mounted on its end effector.

2.2 Reconstruction cycle

A classical 3D scene reconstruction cycle consists of four stages which are planning, scanning, registration and integration (see Figure 2). In any scene reconstruction process, a model of the scene is gradually built by observing the unknown environment using the data acquired from the 3D sensor. At each cycle, the planning step determines new sensor configuration based on a predefined criterion, for example: being able to explore the largest size of the cluttered area. The new information obtained at that new configuration is then registered by using a registration method (e.g. Iterative Closest Point (ICP) [1]). Eventually, registered information must be fused into the model of the scene at integration stages. New configurations are generated and the process continues until all parts of the scene has been explored or until a stopping criterion is satisfied.

Note that each of the stages in reconstruction cycle is a challenging problem on its own and has gained much attention from many researchers. Based on the development of such problems, we can divide the reconstruction cycle into two fundamental steps: the view planning problem and the scanning-registration-integration problem.

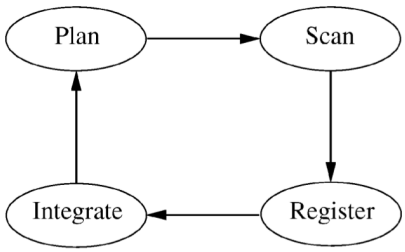

Figure 2: The reconstruction cycle

2.2.1 The view planning problem

Since there is no prior information about the environment and a sensor can only observe a part of the scene from one viewpoint, a viewpoint planner is critically important to plan next viewpoints. This is the problem of finding the next sensor configuration to acquire essential views- commonly known as Next-Best-View

problem (NBV). Connolly [2], Maver and Bajcsy [3], Whaite and Ferrie [4] are one of the first researchers addressing the NBV problem. Connolly [2] proposed an approach based on octree structure to present the seen and unseen area. Planning strategies are made based on ray-casting the model or using the normals of the unseen octree notes. Maver and Bajcy [3] also used the surface normal histogram method to determine the best direction to see the most area of the occluded region. However, these pioneering works only focused on object reconstruction problem.

The robot working space reconstruction (scene reconstruction) problem is more difficult because of a large number of objects that may exist in the target environment, and of many constraints in robot planning problem (e.g. collision checking, reachability of robot, and many others). Most of previous studies are however limited to simple scenes which have some simple geometry objects or have only a single object. Their concepts are commonly based on strategies to explore and label the space of interest. Following Scott's survey [5], view planning method for unknown objects/scene can be categorized into 3 groups that are surface-based [3], [4],[6], global [7] and volumetric methods.

Since its first introduction in the 1970s, volumetric data representations have been developed exponentially and become more practical compared to surface-based representations. In [8], [2],[9], the scanned space is voxelized and labeled based on their positions and visibilities (e.g. voxels are classified into unseen, seen, free or occupied and unoccupied). The next camera configuration that will reveal the largest numbers of unoccupied voxels (or the voxels presenting the hidden area) was defined as the next-best-view, and their approaches are to reduce the number of viewpoints as much as possible. To do so, they evaluated viewpoint quality by employing ray-tracing method to estimate the amount of hidden area. However, the viewpoints were restricted to a predefined area (e.g. a fixed sphere) around the object of interest, which led to many simplifications and limitations compared to robot working space reconstruction. Authors in [10] also presented a volumetric method where a robot followed a predefined trajectory to explore the scanned space. The trajectory then will be modified by following an obstacle-free algorithm. This method is likely an exhaustive search strategy, therefore it is only applicable to scenes with small volume.

2.2.2 The scanning-registration-integration problem

The remaining stages phases of the reconstruction cycle are scanning, registration and integration. As previously mentioned, these problems have gained much attention and obtained adequate solutions which is current applied in many commercial hand-held 3D scanning devices [11], [12], [13]. In 2011, Newcombe [13] proposed an efficiently iterative reconstruction. Dense-point matching and iterative closest point (ICP) algorithm were ran efficiently using parallelized programming, hence enabling the real-time 3D reconstruction.

2.3 Summary

Since the scanning-registration-integration problem has gained adequate solutions, we shall build our implementation by inheriting those results. As previously mentioned, existing approaches for view planning problem still hold many limitations. To our knowledge, no general, fast and stable solution for 3D working space reconstruction has been proposed so far.

3 3D object pose estimation and robot-sensor calibration

As mentioned earlier, the proposed work requires a good background on many areas including robot control and computer vision. This section will briefly present my works and experiments in 3D pose estimation and robot-sensor calibration which have been applied to non-prehensile object transportation experiment on a Denso robot. The experiments were conducted with the objective is to mimic a waiter's work. Hence we require the system have an ability to use a tray (the tray is mounted on the last link of a robot arm) to take an object (i.e. a milk bottle) from a customer (or user), then transfer it to the other position. In order to do so, the system need to be able to acquire the 3D pose of the object (position and orientation) in the robot arm coordinate frame. The rest of this section will present how above problems were solved, and also show some experimental results.

3.1 Object pose estimation using Kinect and PCL

There are several types of depth sensors with a variety of techniques, accuracy, resolutions and prices. Since Kinect sensor enables us to process a rich combination of RGB and depth information, we use a Kinect sensor mounted on a tripod to observe the working space of the robot arm in this experiment.

The 3d data returned from Kinect is in the form of a point cloud, and is processed using Point Cloud Library (PCL) (A point cloud is a set of 3D points which are usually defined by their X, Y, Z coordinates). Our objective is to recognize an object and estimate its position and orientation in the real world. To do so, we followed the local 3D recognition pipeline in PCL [14] widely used in 3D vision community. Figure 3 shows five main steps of the pipeline. The point clouds acquired from the Kinect are first preprocessed to extract key points. The 3D object recognition system then compute descriptors, and compare them with the ones from the object database. The system continues post-processing steps and finally output all the correspondences along with their position and orientation. The post-processing steps are, however, optional but they will help to get better results if they are performed.

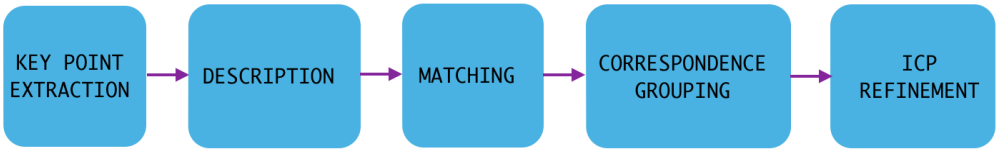

Figure 3: A local 3D recognition pipeline

The first step is to decide which points in the point cloud will have the descriptor computed on them. Note that descriptors can be considered as signatures of a point which contain information about the point itself and the surrounding points. A good descriptor must satisfy following criteria: robust to transformations, robust to noise and resolution invariant. In our experiment, we chose SHOT descriptor [15] since author in [16] observed that SHOT consistently outperformed other approaches in their evaluation experiments.

After we computed descriptor for all of the keypoints, we find in these descriptors correspondences with the ones in the object database. A k-d tree or octree can be used as a search structure to perform a nearest neighbor search efficiently. Finally, we checked the geometric consistency of correspondences

and discarded the ones that do not satisfy. Object pose now can be estimated, however, we then ran a post-processing step (i.e. RANSAC (RANdom SAmple Consensus)) to improve the accuracy of the pose estimation. For example, Figure 4 shows a pose estimation of a milk bottle model that we used in our experiments. The milk bottle model is showed in yellow color, while its estimation is in red color. The green lines connect keypoints and their correspondences.

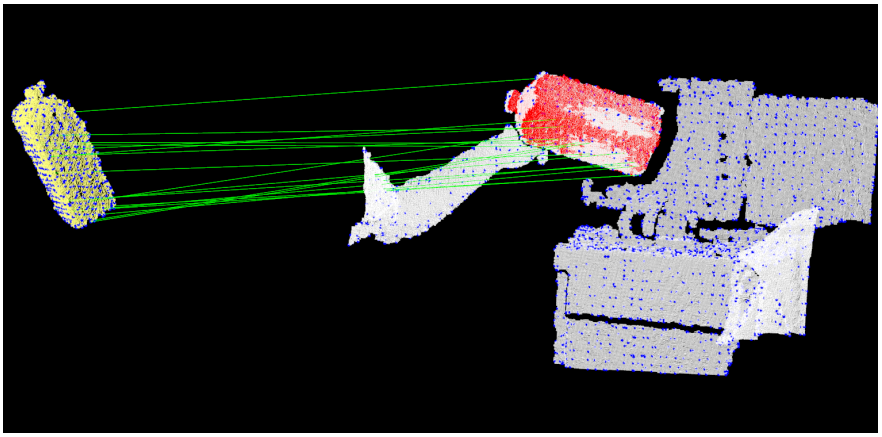

Figure 4: The experimental milk bottle model (in yellow) and its correspondences in a point cloud

3.2 Robot sensor calibration- solving $AX = XB$

It is importance to note that the 3D pose estimation result is commonly not the position and orientation of the real object respect to robot coordinate frame, but rather the transformation from the reference object (in database) to the real object in Kinect frame.Therefore robot sensor calibration is required in order to yield the 3D pose of the object in the robot coordinate frame.

In [17], Park and Martin presented a closed-form least squares solution for the fundamental equation $\mathbf{AX} = \mathbf{XB}$ on the Euclidean group, which commonly arises in the wrist-mounted robotic sensor calibration problem. In our case, the Kinect sensor was not attached on the robot arm but on a tripod to observe the working space. We therefore adopted this method to our problem. The calibration involves performing several measurements $((\mathbf{A}_1, \mathbf{B}_1), (\mathbf{A}_2, \mathbf{B}_2), ..., (\mathbf{A}_n, \mathbf{B}_n))$, where $\mathbf{A}_k$ matrix denotes the position and orientation of the last link relative to the robot base frame after one movement, $\mathbf{B}$ matrix denotes the position and orientation of the last link observe from the Kinect (relative to the Kinect frame), $\mathbf{X}$ matrix then describes the position and orientation of the Kinect frame relative to the robot base frame. We followed the solution in the paper since it is simple, computationally efficient, and can handle noise that usually is present in $\mathbf{A}$, $\mathbf{B}$ measurements. We implemented this method with a generalizable criterion, that is it can be applied to many types of robot arms and sensors.

3.3 Experimental results

The position and orientation of the object relative to the robot base frame are passed to a path planner. The planner then generates a trajectory so that robot arm can safely approach and take the object. The

implementations are then also tested using a motion capture system- Optitrack. The experiment environment is showed in Figure 5.

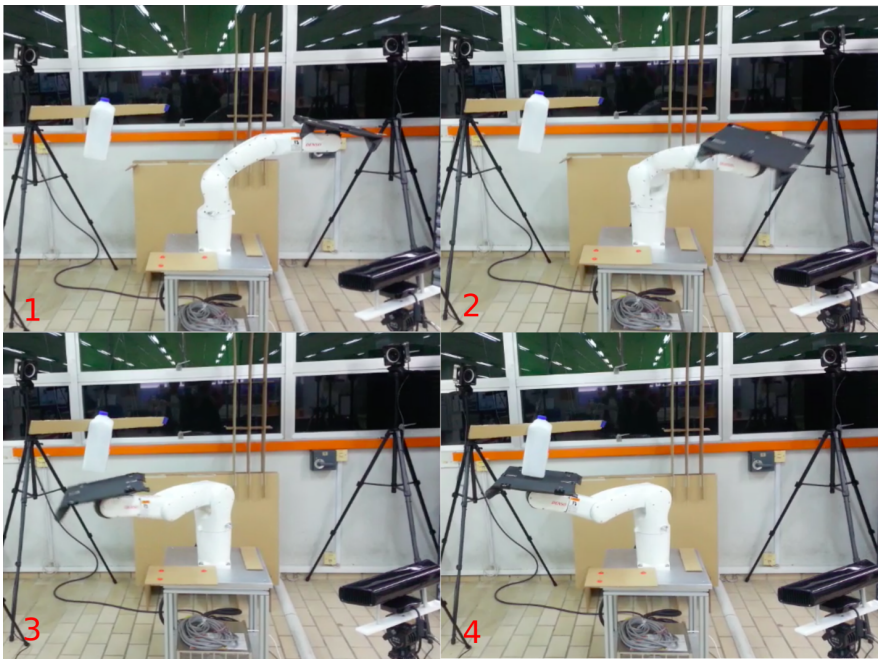

Figure 5: Snapshots of our bottle-tray experiment, video of the experiment can be found at `https://goo.gl/5TYfy5`

4 Time-optimal path parameterization in SO(3)

Another work involves planning fast trajectories in SO(3)[1] under kinodynamic constraints and in a cluttered environment. Here we addressed the time-optimal reorientation of a rigid body problem which 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 [18, 19, 20]. 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 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 SO(3). Regarding component (i), a sampling algorithm based on unit quaternions was proposed in [21]. 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.

As a result, to our knowledge, there are no general methods for planning fast trajectories in SO(3) under

[1]The set of all 3×3 orthogonal matrices with determinant +1 forms a group known as the special orthogonal group SO(3), also known as group of rotation matrices.

kinodynamic constraints and in a cluttered environment. Here we address this problem by following the widely used plan-and-shortcut pipeline, which has shown to be easy to implement, robust and efficient (it is e.g. the default pipeline used in OpenRAVE, a popular robotics software in both academia and industry). This pipeline consists of three steps

1. plan piece-wise "linear" *paths* considering collision constraints using RRT;
2. reparameterize these paths considering time-optimality and kinodynamic constraints;
3. shortcut using time-optimal interpolants under kinodynamic constraints.

Step 1 is the basic RRT in SO(3) [21]. Note that "linear" paths in Euclidean spaces correspond to great-circle arcs in SO(3). Step 2 and step 3 are based on our extension of Time-Optimal Path Parameterization (TOPP, cf. [22]) to SO(3) and constitute the core contribution of this work. 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, SE(3). This work have been submitted to "The Journal of Guidance, Control and Dynamics", and is under review.

4.1 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 expressed in the form (note that all vector inequalities in this paper are element-wise)

$$\ddot{s}\boldsymbol{a}(s) + \dot{s}^2\boldsymbol{b}(s) + \boldsymbol{c}(s) \leq \mathbf{0}, \tag{1}$$

then there exists efficient methods and implementations to find the time-optimal parameterization $s(t)$ (see [22] for references and details).

Consider the case of a rigid body with three independent actuations, such as rigid spacecraft, whose equations of motion are $\mathbb{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbb{I}\boldsymbol{\omega}) = \boldsymbol{\tau}$ 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 $\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max}$.

Following [23, 24], consider a path in the space of rotation matrices $\mathbf{R}(s)_{s\in[0,1]} \in \mathrm{SO}(3)$ given by

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

where $\boldsymbol{r}(s)$ is a 3-dimensional vector. The interest of this types of paths is that they are smooth (if $\boldsymbol{r}$ is smooth), are bi-invariant and interpolate optimally between two rotations (in the case of rest-to-rest motions) and near-optimally between two rotations when initial and final angular velocities are specified. For a detailed discussion, see [24].

Following again [24], we can next write

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

where

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

and

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

Thus, the torque bounds can be put in the form of (1)

4.2 Implementation and simulation results

We experimentally evaluated our approach by solving some spacecraft maneuver problems. Our implementation is open-source and is provided at `https://goo.gl/Ih34Xy`. 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 `https://goo.gl/F2R5Br`.

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

Our goal is first to find a fast trajectory connecting two rotation configurations $(\mathbf{R}_0, \boldsymbol{\omega}_0)$ and $(\mathbf{R}_1, \boldsymbol{\omega}_1)$, subject to velocity and acceleration bounds, as well as obstacle avoidance. As mentioned earlier, we follow the plan-and-shortcut method [25].

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 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 $\mathbf{R}_{t_0}, \mathbf{R}_{t_1} \in \mathrm{SO}(3)$ 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

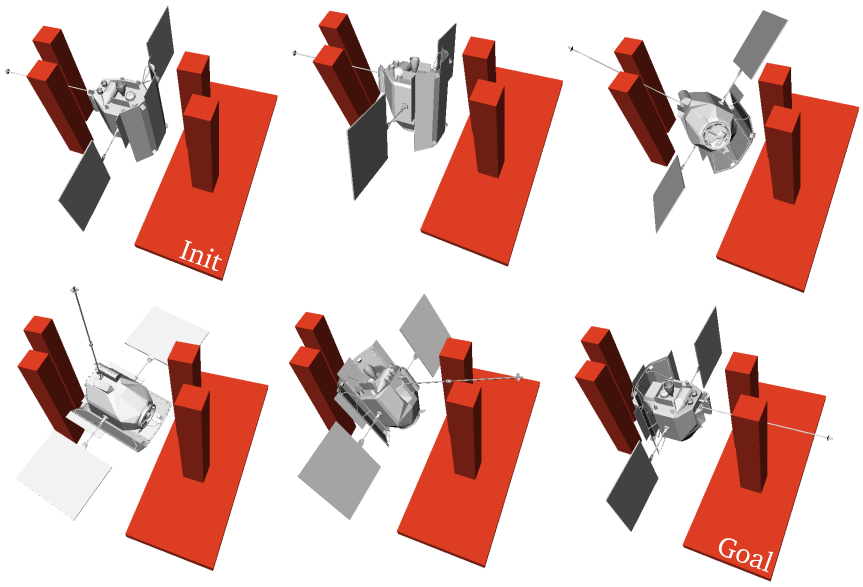

Figure 6: Fast reorientation trajectory for the Messenger spacecraft in a cluttered environment.

discontinuities. If the resulting duration is smaller than T then we replace the original trajectory portion by the shortcut. Fig. 6 shows one typical trajectory found by the algorithm.

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

We then also consider the case of rigid-body motions. While SO(3) is the group of 3D rotations, SE(3) is the special Euclidean group of rigid-body motions. SE(3) includes both rotations and translations, and is of the form

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

where $\mathbf{R} \in \mathrm{SO}(3)$ and $\boldsymbol{q} \in \mathbb{R}^3$. There are a number of available approaches to interpolate trajectories between two elements of SE(3). Park and Ravani [23] exploited the Lie group structure of SE(3) to develop an algorithm analogous to the preceding interpolation in SO(3). The resulting motion, however, is a screw motion, which corresponds to a strange motion in physical space. Moreover, in SE(3) there is in general no bi-invariant interpolations [24]. Therefore, to tackle the problem of generating rigid body motion, we shall interpolate orientations and positions separately. For the orientation part, we proceed as in Section 4.1. For the translation part, we interpolate between two pairs (position, velocity) by a third degree polynomial. Consider two translation configurations $(\boldsymbol{q}_0, \boldsymbol{v}_0)$ and $(\boldsymbol{q}_1, \boldsymbol{v}_1)$, the translation path is given by

$$\boldsymbol{q}(t)_{t\in[0,1]} = \boldsymbol{k}_3 t^3 + \boldsymbol{k}_2 t^2 + \boldsymbol{k}_1 t + \boldsymbol{k}_0, \tag{4}$$

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

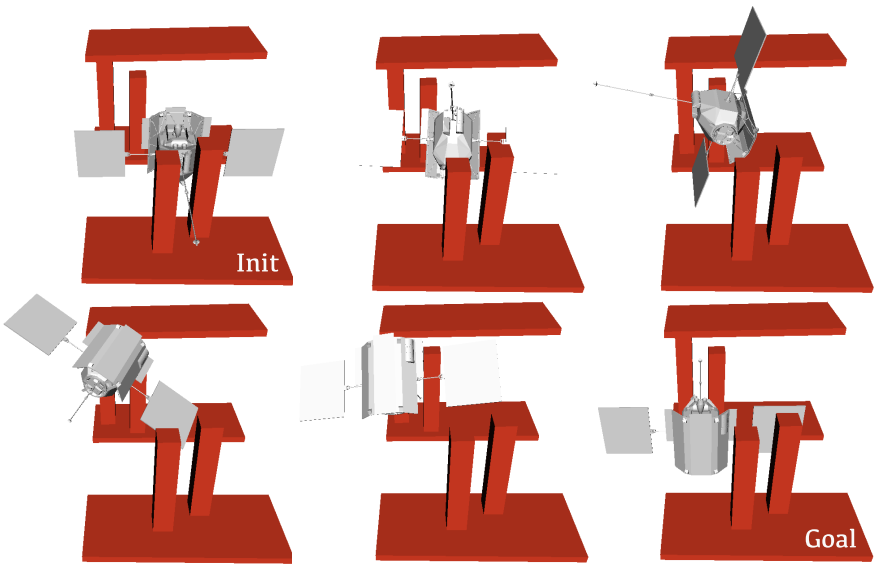

Figure 7: Spacecraft maneuvering in a cluttered environment- SE(3)

Figure 7 shows one typical trajectory in SE(3) found by the algorithm. As the configuration space of the problem is 6D, the complexity of the sampling space in SE(3) is much higher than in SO(3).

5 Conclusion and furture works

This report has summarized the status of my progress in the first year of PhD study, including some preliminary studies that I have been accomplished for the last two semesters. I have focused on developing my understanding, building my skills on 3D reconstruction, robot sensor calibration, programming robot algorithms including motion planning, kinematics, dynamics, and experiencing with robot simulation environments. Although there are still some limitations in the results I have achieved so far, these related works have been the good background for me to complete my primary research.

The goal of my research is to develop techniques that enable robots to actively autonomously explore and "understand" their environments in an incremental way. Such techniques should be robust, efficient, and applicable to a broad class of range sensors, positioning systems and environments. The information obtained from such systems shall be used to design an automatic robotic system that can handle most of the challenging tasks such as fine manipulation and assembly.

As mentioned earlier, my future works, therefore, will first consist in building a view planner to find the best next view for this 3D working space reconstruction problem. Next, this view planner must be integrated to the system where the range sensor mounted on the end effector. This integration is, however, challenging since we have to consider not only the next best view problem but also robot dynamics, robot reachability and collision-free constraints.

Other future work involves integrating 3D perception with tactile perception. For example, tactile sensors placed at the robot end effector can detect contacts, and via the robot kinematics, give an information about the position and orientation of the contact points. We shall propose a framework where 3D perception

gives a rough description of the environment and guides the exploration while tactile perception refines environment description to a level suitable for fine manipulation. The main difficulty here lies in the integration of different modes of perception and their respective uncertainties. We shall make use of and further develop the theories of optimal multi-modal sensor fusion in the case of visual and tactile perceptions.

References

[1] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Tranctions on Pattern Analysis And Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[2] Colnnolly, "The determination of the next best views," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1985, pp. 432–435.

[3] Maver and Bajcy, "Occlusions as a guilde for planning next view," in *IEEE Trans. PAMI*, 1993, pp. 482–496.

[4] Whaite and Ferrie, "Autonomous exploration: Driven by uncertainty," in *IEEE Trans. PAMI*, 1997, pp. 193–205.

[5] Scott, Roth, and Rivest, "View planning for automated 3d object reconstruction inspection," in *ACM Computing Surveys*, 2003, pp. 64–96.

[6] Chen and Li, "Vision sensor planning for 3-d model acquisition." in *IEEE Trans Syst Man Cybernet*, 2005, pp. 894—904.

[7] R. Pito, "A solution to the next best view problem for automated surface acquisition," *IEEE Trans Pattern Anal. Machine Intell.*, vol. 21, pp. 1016—1030, 1999.

[8] Banta and Abidi, "Autonomous placement of a range sensor for acquisition of optimal 3d models," in *Proceedings of the IEEE 22nd International Conference on Industrial Electronics, Control and Instrumentation*, 1996, pp. 1583–1588.

[9] Massios and Fisher, "A best next view selection algorithm incorporating a quality criterion," in *British Machine Vision Conference*, 1998.

[10] Orfanos and Schmitt, "Automatic 3d digitization using a laser rangefinder with a small field of view." in *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, 1997, pp. 60–67.

[11] R. A. Newcombe and A. J. Davison., "Live dense reconstruction with a single moving camera." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[12] J. Stuehmer, S. Gumhold, and D. Cremers, "Real-time dense geometry from a handheld camera," in *Proceedings of the DAGM Symposium on Pattern Recognition*, 2010.

[13] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneax, D. Kim, A. J. Davision, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*, 2011.

[14] R. Rusu and S. Garage, "3d is here: Point cloud library (pcl)," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1–4.

[15] F. Tombari, S. Salti, and L. D. Stefano, "Unique signatures of histograms for local surface description," in *European Conference on Computer Vision (ECCV '10)*, 2010, pp. 356–369.

[16] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Point cloud library: Three-dimensional object recognition and 6 dof pose estimation," in *IEEE Robotics and Automation Magazine*, 2012.

[17] 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, 1994.

[18] 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.

[19] 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.

[20] 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.

[21] 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. IEEE, 2004, pp. 3993–3998.

[22] 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.

[23] 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.

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

[25] 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.