POLITECNICO DI TORINO

Master Degree course in Computer Engineering

Master Degree Thesis

3D Pose Estimation for Robot Mikado

Candidate:

Tengang Vini Marconie
s255221

Advisor:

Prof. Marina Indri

External advisors at TU-Darmstadt:

Prof. Jan Peters

Mr. Pascal Klink

Mr. Tuan Dam

Abstract

There is a persistent need to establish a spatial mapping between two or more 2D or 3D representations of an object. This task is very common in the field of science, medicine, engineering, as well as in the field of robotics. This spatial mapping is called object registration or object pose (position and orientation) estimation and its goal is to determine a spatial transformation that best aligns two object descriptions.
This thesis work focuses on the implementation and evaluation of methods for free-form object registration from 3D point clouds. Enhancement to the registration accuracy is achieved by using more informative features (color) to existing methods (ICP, PPF). This thesis, for example, describes a system for accurate object registration from a 3D point cloud which is vital in the task of robotic grasping.
3D pose estimation experiments demonstrate that proper down-sampling and the use of color information in our setup significantly increase registration accuracy.
Besides having multiple areas in which object registration can be applied, the chosen field of application for this thesis is oriented to a future robotic grasping task.
Key words: 3D point cloud, 3D pose estimation, ICP, PPF

Acknowledgements

I would like to gratefully acknowledge all my supervisors for their prolific suggestions. My special thanks go to Mr. Pascal Klink and Prof. Marina Indri for their infinite patience and guidance. All the support I had was invaluable. I am grateful to all my friends for being a surrogate family to me throughout my studies abroad. My final words of gratitude to my family for all the love, support and guidance I receive in whatever I pursue.

Contents

Figures

List of Figures

Abbreviations and Symbols

List of Abbreviations

2D	2-dimensional.
3D	3-dimensional.
BnB	Branch-and-Bound.
CAD	Computer-aided Design.
dist	Distance.
dof	Degrees of freedom.
e.g	for example.
EIH	Eye-in-Hand.
GHT	Generalized Hough transform.
Go-ICP	Globally optimal ICP.
i.e	that is.
ICP	Iterative closest Point.
IR	InfraRed.
k-d	K-dimensional.
PC	Point Cloud.
PPF	Point Pair Feature.
RGB	Red Green Blue.
RGB - D	Red Green Blue Depth.
RMSD	Root Mean squared Distance.
RMSE	Root Mean squared Error.
std	Standard deviation.
SVD	Singular Value Decomposition.
thr	Threshold.
w.r.t	with respect to.

List of Symbols

C_m	Centroid of point cloud (point-set) M.
C_s	Centroid of point cloud (point-set) S.
$\boldsymbol{F}$	Feature vector.
$\boldsymbol{H}$	Covariance matrix.
$\boldsymbol{K}$	Camera calibration matrix.
$\boldsymbol{R}$	Rotation matrix.
$\boldsymbol{T}$	Transformation matrix.

$\boldsymbol{t}$	Translation vector.
$\overline{E(R)}$	Mean rotation error.
$\overline{E(t)}$	Mean translation error.
σ_R	Rotation std.
σ_t	Translation std.
C	Set of point pair correspondences.
M	Model point cloud (unless otherwise stated).
N_m	Cardinality of model point cloud.
N_s	Cardinality of scene point cloud.
N	Cardinality of correspondence set.
R_x	Rotation about the x-axis.
$SE(3)$	Spatial Euclidean group in $\mathbb{R}^3$, consisting of rotations and translations in $\mathbb{R}^3$.
$SO(3)$	Rotation group in $\mathbb{R}^3$.
S	Scene point cloud (unless otherwise stated).
$\mathbb{R}_x + \mathbb{R}_0^+ y$	Half-plane defined by x-axis and $+y$-axis.
θ_{xyz}	Rotation about the x-axis, y-axis, z-axis.
f	Camera focal length.
t_{xyz}	Translation along the x-axis, y-axis, z-axis.

1 Introduction

The need to establish a spatial mapping between two or more 2D or 3D representations of an object is very common in the field of science, medicine, engineering, as well as in the field of robotics. This spatial mapping is referred to as object registration or object pose (position and orientation) estimation. The registration problem is that of determining a spatial transformation that best aligns two object descriptions based on a suitable cost metric. Several tasks can be carried out using the spatially aligned object descriptions. Some of these tasks include:

• 3D object recognition:

This is a well-known research problem in the field of computer vision. Most approaches to this object recognition problem are based on 3D pose estimation. See Figure 1.1.

• Real-time pose tracking:

Tracking the position and orientation of a free-form object in real-time requires high-speed registration capabilities. Real-time human pose tracking for example as in Figure 1.1, is achieved by rapid processing and registration of motion and depth sequences.

• Model-based localization:

The location of a desired physical scene object from its model description is achieved by spatially registering the model object to the physical scene. For example in automatic localization and grasping, once registration is done properly, the grasping task can be performed accurately. See Figure 1.1.

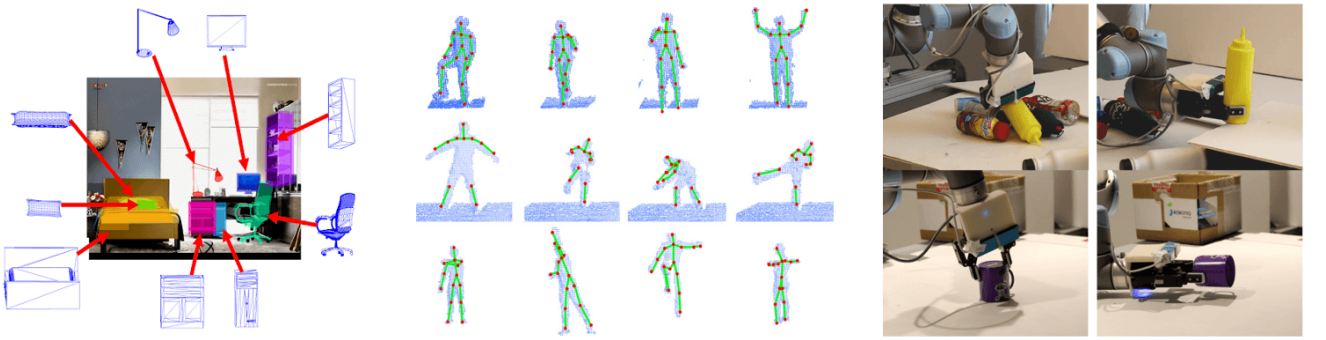

Figure 1.1: Tasks based on pose estimation.
Recognition and location of objects from 3D image datasets (left) [1].
Real-time human pose tracking scenario (middle) [2]. A grasping task
performed after accurate model localization (right) [3].

1.1 Motivation

Although there are numerous fields in which registration can be carried out as mentioned earlier, the field of interest for this thesis work is that of robotics. The ideas and approaches described in this work can be applied to a broad range of pose estimation problems. However, it is particularly

beneficial in robotics since the object to be registered must not always be necessarily at the exact same location.
The problem of robotic grasping being a difficult one today was even more complicated in the past due to the lack of sensory capabilities. Most of the grasp planning methods could not act intelligently in recognizing objects and perceiving the workspace. To perform an effective grasp operation, for example, the target object presented to the robot needed to be placed in a predefined precise location and with a known fixed orientation. Such systems, therefore, lacked automatic modification capability and flexibility to adjust for changes in the assigned task. Further flexibility enhancement can, therefore, be obtained by the integration of vision sensors.
The area of robotic grasping based on visual information has emerged within the past years as more informative visual sensors can be integrated with the robotic grasping system to meet flexibility needs [4], [5], [6], [7]. This research area combines mechanical actuation, vision sensing, image processing, simulation, and high-performance computing for proper planning and execution of grasping tasks. Central to the grasping task is the registration operation, objective of this thesis work, which relays on the sensing capabilities of the vision system and good processing of the sensed information (colored and depth images) to generate the registration data. A point cloud is used as registration data to describe the scene and model objects in this work.

1.2 Thesis Objective

This thesis mainly focuses on the problem of achieving and ensuring accurate 3D pose estimation results. Figure 1.2 sketches a scheme for achieving this goal. The example application used in this scheme is the estimation of Mikado stick poses which are necessary for a future grasping task. However as suggested earlier, the proposed approach can be applied to other fields. The method consists of two stages. The first stage consists of data generation and collection, i.e, a computer-aided design (CAD) generation of the model point cloud and the critical collection of visual information (colored and depth images) of the real scene to create the scene point cloud.
The data obtained from the first stage then serves as the entry point into the second stage centred around various registration algorithms, the ICP algorithm (and its variants) and the PPF registration algorithm. ICP directly performs the online registration of the model object taking as input the scene and model point clouds and outputs the optimal registration pose. Contrarily to ICP, PPF consists of 2 phases, an offline phase which uses the model point cloud to generate a global model description of the object to be registered; this global model description is then used in the online phase together with the scene point cloud to carry out the registration operation.

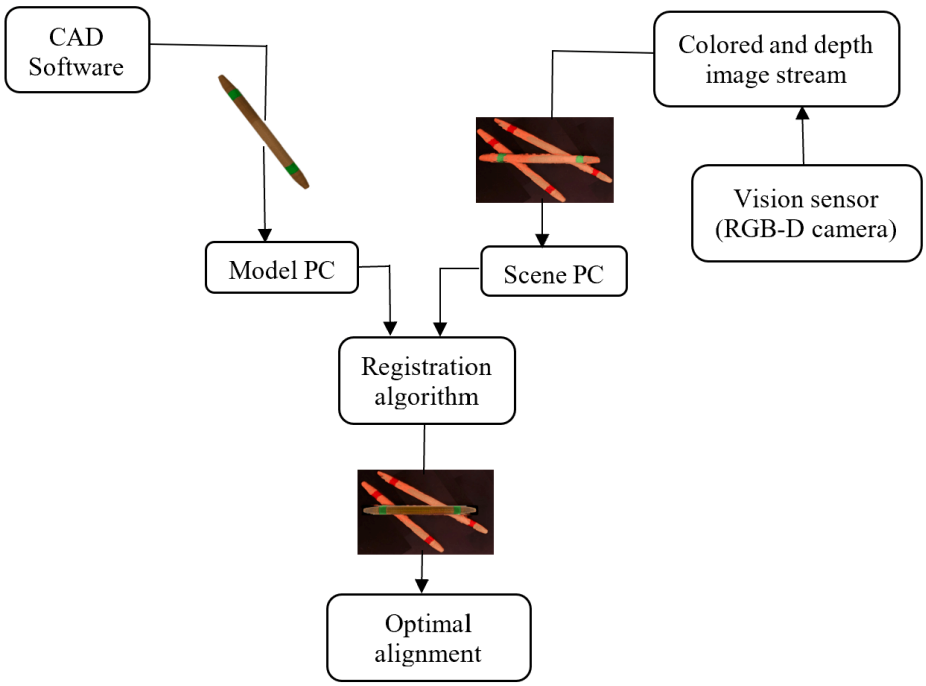

Figure 1.2: Free-from point cloud Registration scheme.
In stage 1, quality model and scene point clouds (inputs) are generated from CAD software and vision sensor data (colored and depth images), respectively. The inputs are later passed to the pose estimation algorithm. The execution of the registration algorithm then outputs the optimal registration pose.

In stage 1 of the registration scheme in Figure 1.2, quality model and scene point clouds (inputs) are generated from CAD software and vision sensor data (colored and depth images), respectively. The inputs are later passed to the pose estimation algorithm. The execution of the registration algorithm then outputs the optimal registration pose which describes the spatial location of the object of interest in the scene.

1.3 Thesis Layout

The rest of the thesis work is organized as follows.
Chapter 2 provides the foundation of the work, the background, and the description of the object registration problem. Herein is equally presented details on the choice of the vision sensor configuration and its calibration for our specific setup.
Chapter 3 presents the registration algorithms used as solution methods in this thesis. This chapter equally provides some enhancements to speed up the registration algorithms in order to obtain reasonable registration time. The important and subjective nature of the ICP algorithm to local minima is discussed and possible methods to overcome the problem are equally provided.
Chapter 4 demonstrates how both the synthetic and quality real-world data are generated. Dealing with a data-driven scheme, the quality of the real-world data has a great influence on the registration results. To this aim, an approach to obtain good quality real-world data to describe the scene is provided herein.

Chapter 5 shows how the proposed registration scheme presented in Figure 1.2 can be applied to the 3D pose estimation problem using the collected synthetic and real-world data. The overall registration setup and experimental results and plots are also described in this chapter.
Chapter 6 contains the conclusion and proposes directions for eventual future work.

2 Foundations

This chapter lays down the foundations necessary to achieve our goal of accurate object registration. Details on the pose estimation problem description are provided in this section. Herein is equally presented the basic concepts of the image generation process

2.1 Vision System

The quality of the images plays an important rule in our registration scheme as the scene point cloud is generated from these images (See Figure 1.2). Having clear in mind that judicious collection of quality visual information gives better registration results, a proper view of the object to be registered is therefore mandatory for the vision system to be integrated with the registration scheme. To this end, a preexisting pinhole camera model is used as the main component of the vision system, with the sole purpose of generating quality data for object pose estimation.

2.1.1 Eye-In-Hand Camera Location

Several studies have been carried out in the field of eye-in-hand (EIH) visual servoing methodology [8], [9]. This is because eye-in-hand configuration plays an important role in industrial assembly operations, object tracking [10], object pose estimation, as well as in many other fields of application. An eye-in-hand configuration can be defined as a robot end effector equipped with a camera (in general a close-range camera), as shown in Figure 2.1. Depending on the complexity of the task to be performed, a specific camera will be selected. To counteract the effect of changes in light intensity at some points in the real-world scene and guarantee caption of good images even in dim light, an illuminating source could equally be attached to the gripper. Such systems are mainly used to guide the robot end effector (gripper) to perform a specific task. In our set-up, the task will be that of collecting clear images of the scene from different poses of the camera to generate the point cloud. This point cloud will be used to estimate the poses of objects (Mikado sticks) in the scene. The pose information can then serve as guidance for robot movement to a specific location in the workspace for eventual operations.

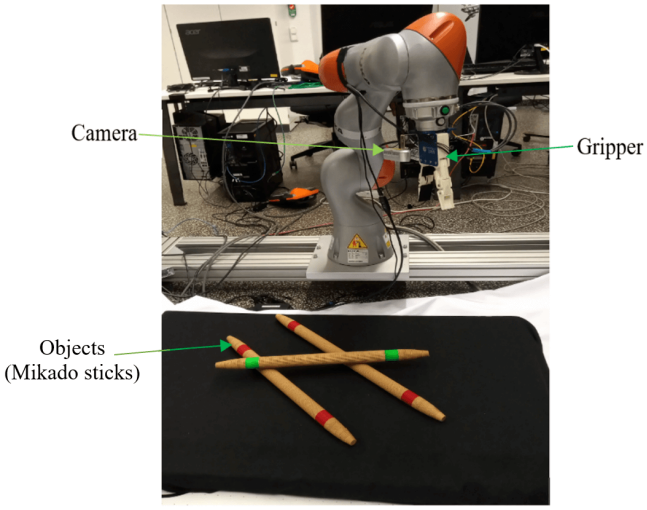

Figure 2.1: Eye-in-hand configuration.
Experimental setup showing the Eye-in-hand location of the camera. This configuration ensures the collection of clear scene images from different poses of the camera.

The eye-in-hand camera configuration is beneficial for our scenario since it guarantees that the camera will not be occluded by the manipulator itself, as it follows its trajectory, taking images of the scene from different poses.

2.1.2 Pinhole Camera Model

A camera can be described as a system that performs the non-invertible mapping from real-space coordinates to image plane coordinates ($\mathbb{R}^3 \mapsto \mathbb{R}^2$) [11]. An analytical model for this transformation is therefore necessary for determining image plane measurements in the real-world coordinates system. The analytical model therefore plays a major role in pose estimation as well. Due to its central projection and its capability to provide good approximations for other lensed cameras, the pinhole camera model is one of the most adopted in computer vision. This model has equally been adopted for our set-up.
The projection of a 3D point $M[X, Y, Z]^T$ in real space onto the image plane at $m[x, y]^T$ for a simple pinhole camera located at the origin of the world frame is represented in Figure 2.2. The non-linear transformation expressing this projection is given by:

$$x = \frac{fX}{Z} \tag{2.1}$$

$$y = \frac{fY}{Z} \tag{2.2}$$

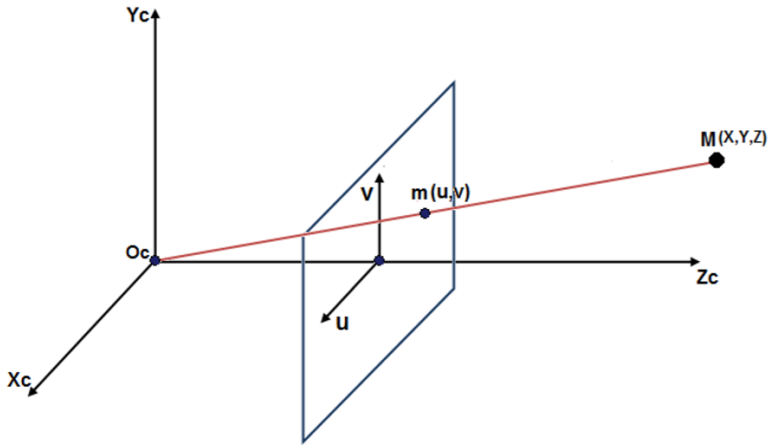

Figure 2.2: Pinhole camera model. Projection of a 3D point $M[X,Y,Z]^T$ in real space onto the image plane at $m[x,y]^T$ for a simple pinhole camera located at the origin of the world frame.

However, the assumption that the camera is located in the center of the world frame is not always true. If this is not the case, M must be transformed from the world frame to the camera frame before applying the projection. This camera operation can be described more compactly by the linear projective transformation proposed by Hartley (1999) [12] as:

$$m = \boldsymbol{P}M \tag{2.3}$$

where $M[X,Y,Z]^T$ and $m[x,y]^T$ for simplicity of notation are written as M and m, respectively, as will be the case in other equations of this section. M and m are homogeneous coordinates in the world frame and image plane, respectively, and $\boldsymbol{P} \in \mathbb{R}^{3\times4}$ is the 3×4 projection matrix of the camera. By definition, the position and orientation of the camera in the world frame are given by its extrinsic parameters. Representing these extrinsic parameters by an inhomogeneous translation vector $\boldsymbol{t}$ and rotation matrix $\boldsymbol{R}$, the projection matrix which performs both the coordinate transformation and projection onto the image plane can be expressed as:

$$\boldsymbol{P} = \boldsymbol{K}(\boldsymbol{R}^T | -\boldsymbol{R}^T\boldsymbol{t}) \tag{2.4}$$

where $(\boldsymbol{R}^T | -\boldsymbol{R}^T\boldsymbol{t}) \in \mathbb{R}^{3\times4}$ is the 3×4 extrinsic camera parameters and $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ is the 3×3 camera calibration matrix. The calibration matrix for a general pinhole camera is:

$$\boldsymbol{K} = \begin{pmatrix} af & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

where f is the focal length, (x_0, y_0) is the principal point (located at the intersection of the optical axis and image plane), a is the pixel aspect ratio and s is the pixel skew description. These quantities are independent of the position and orientation of the camera and are collectively

known as the intrinsic camera parameters. When accurate calibration is not necessary, it is quite reasonable to assume squared pixels (unity aspect ratio and zero skew) and the principal point at the origin of the image plane. In such cases, the focal length f is the only parameter of the calibration matrix:

$$\boldsymbol{K} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.6}$$

This camera focal length and its principal focus point can be computed by following a standard calibration approach proposed by Chen and He (2007) [13]
The projection of a 3D point $M[X, Y, Z]^T$ on to the image plane at a point $m[x, y]^T$ can thus be represented in the homogeneous matrix form, and its complete pixel position can now be written as:

$$m = \boldsymbol{K}(\boldsymbol{R}^T | -\boldsymbol{R}^T \boldsymbol{t})M \tag{2.7}$$

2.2 Object Registration

This section provides details on how the registration problem is formulated and presents a brief related background on what has already been done as far the registration tasks are concerned. It equally clarifies how the characteristics of the information extracted from the data can guide the choice of the registration approach.

2.2.1 Problem Description and Related Background

Object pose estimation (also known as object registration) is the problem of finding the relative pose (position and orientation) between two descriptions of the same object. These descriptions of the object may be either geometric, i.e, based on the geometry (shape) of the object or photometric, i.e, based on the object image, or they can be equally based on point cloud representation of the object, and could be 2D or 3D.
The goal of registration is to determine a spatial transformation which aligns the two object descriptions based on a suitable cost metric. The determined transformation can either be rigid or deformable, and may or may not include a scale term. This thesis work however focuses on the problem of rigid registration without scale of 3D geometric object descriptions.
Related background research on object pose estimation suggests that it can be tackled either through a learning-based approach or following a classical approach. In the former, natural features of the object are extracted from the image to create a 3D metric model, which is used for offline training of the neural network used for registration [14]. In the later, iterative algorithms directly operate online and repetitively find correspondence between model points and scene object points until convergence is achieved. Other approaches equally extract features from the model object not to train a network but rather to generate a global model description in the offline phase of the registration; this description of the model is then queried during the online phase of the registration [15]. A close examination of 2D photo-metric image registration which has applications

in the fields of object recognition and medical processing can be found in [16]. A huge amount of work which aims at registering 2D geometric descriptions to 3D geometric models in computer vision can be found in [17], [18].
Estimating the relative pose between 2 sets of points with known correspondences is a common 3D to 3D registration problem. Two general solutions to this problem are the singular value decomposition (SVD) solution method [19], [20]. and the quaternion-based solution method [21], [22]. The former solution method is adopted in this thesis work in the implementation of the ICP algorithm. Several other approaches have been used to solve variants of 3D to 3D pose estimation problems. These approaches differ in the search mechanism of the optimal transformation pose and the choice of the cost function. For example, the quaternion and SVD solution methods mentioned above are both formulated as least-squares minimization problem. A nice outline of other methods that register 3D surfaces to point-sets can be found in [23].
Research from the literature suggests that a minimum of 3 points are necessary for solving the 3D correspondences point problem. The presence of additional points renders the problem overconstrained and makes it possible to remove outlier correspondences. The elimination of outliers requires to specify which points to eliminate. In most methods, this is done by setting a residual error threshold which partitions the point-set into inlier and outlier correspondences. Methods for robust determination of residual thresholds are proposed in [24], [25].

2.2.2 Shape-based Registration

The geometric description of an object as mentioned earlier encodes information about the object's shape. Volumetric shape descriptions on the other hand explicitly represent the regions of space that are occupied by an object. The transition between an object and its surrounding (i.e, the object's surface) is explicitly represented by boundary shape descriptions. All this information on the shape of the object and its surrounding leads us to a form of registration based on geometric data: Given a 3D data-set in sensor coordinate system, which describes a data shape that may match with a model shape, and given different geometric shape representations of a model shape in a model coordinate system, the goal is to estimate the optimal rotation and translation that best aligns (registers) the model shape and the data shape minimizing the distance between the shapes through a suitably chosen cost metric. As mentioned earlier, Several cost functions exist. The choice of the cost function in most cases differentiate a solution method from another. For example, the shape-based registration problem solved in [22] minimizes the mean-square distance metric. Primitives which can be used to describe the geometric data (object shape) include: sets of points, sets of lines, sets of curves, sets of polygons (e.g., triangle meshes), implicit surfaces, or parametric surfaces.
A general class of shape registration problems is the 3D to 3D registration, which includes the registration of a Model composed of point-sets, line-sets, curve-sets or surfaces, to Data composed of one of these representations. A common example is the registration of point-sets to surfaces. More formally, the objective of these problems is to solve the minimization equation:

$$\min_{T} d[M, T(S)] \tag{2.8}$$

where M is a Model description, S is a Data description, T is a rigid transformation, and d is a measure of distance (similarity) between the object descriptions. A solution method for this class of problems as proposed in [22] is based on the iterative closest point (ICP) algorithm. The ICP

algorithm being the basis for most registration algorithms will be described in detail in Section 3.1. It is, however, important to point out that the shape-based registration approach requires no extraction of features, no curve or surface derivatives and no preprocessing of the 3D data apart from the elimination of outliers if necessary.

2.2.3 Registration from 3D Point Cloud

A point cloud is a collection of data points in space defined by a given coordinate system. In a 3D coordinates system, for example, a point cloud may define the shape of some real or simulated physical system. They are generally produced by 3D scanners but can equally be generated from colored and depth images taken by an RGB-D camera. The later point cloud generation method will be adopted in this work as shown in Figure 2.3.

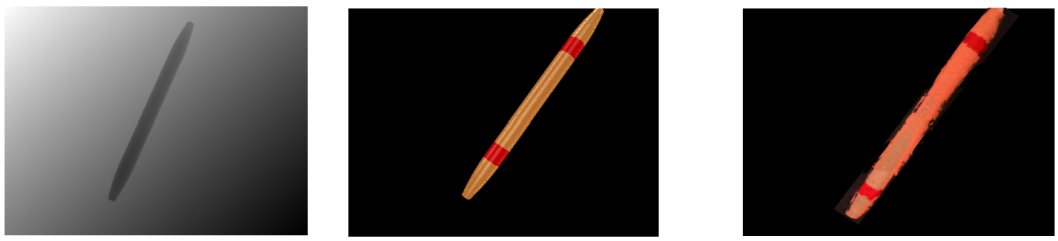

Figure 2.3: Generated point cloud. Processing of the depth image on the left and the colored image in the middle outputs the resulting Point cloud on the right.

Pose estimation from point cloud data and shape-based registration share the same concept. The difference between the 2 approaches only resides at the level of the data structure. In point cloud-based registration, the scene and model objects are both represented by 3D point cloud data-sets. The problem can, therefore, be formulated as in the previous case choosing a suitable cost function to be minimized. Besides, being a model-based registration method like the one mentioned in Section 2.2.2, object registration from point cloud data provides the additional advantage that peculiar features can be extracted from the point clouds and used for registration of free-form 3D objects.
A mixed-integer problem formulation of this point cloud registration problem focuses on finding the best configuration of a rigid-body model to explain the sensed point cloud data [26]. The model configuration is parameterized by the rotation $\boldsymbol{R} \in SO(3)$ and translation $\boldsymbol{t} \in \mathbb{R}^3$ of the rigid body. The point cloud sensor samples a set of N_s points $S = \{s_i\}$ from the geometry of the world. The model can be described as a collection of N_m point features, leading to an optimization penalizing a norm.

$$\min_{\boldsymbol{R},\boldsymbol{t},C} \sum_{i \in [1,N_s]} \|\boldsymbol{R}s_i + \boldsymbol{t} - m_{C(i)}\|, \tag{2.9}$$

$$C(i) = \underset{j \in [1,N_m]}{\operatorname{argmin}} \|\boldsymbol{R}s_i + \boldsymbol{t} - m_j\|, \tag{2.10}$$

where $C(i)$ relates each scene point to the closest model feature according to the desired norm.

The crucial point in solving the pose estimation problem is finding the right correspondences between the 2 object descriptions. This fundamental step in the registration process is generally problematic because correspondence search can easily get trapped into local minima. When this happens, the registration fails to return the optimal solution. This correspondence problem can, however, be solved by using Go-ICP. Based on the aforementioned mixed-integer formulation (solved using the BnB method), Go-ICP is capable of returning the optimal registration pose. The drawback of this method is that it involves a very high computation cost. To avoid this huge computation, alternative approaches such as PPF algorithm and other enhancements to the basic ICP algorithm will be used in this thesis work to find correspondences and minimize the registration error.
The alternative formulation of this problem is based on the assumption that both the scene and the model point clouds are represented as a finite set of oriented points, where a normal is associated with each point (point pair feature; PPF) [15]. This formulation is based on a global model description and a voting scheme. Nevertheless, regardless of the formulation, proper management of outliers improves the registration performance.

3 Algorithmic Description of the Investigated Algorithms

Object pose estimation being a very fertile research argument, several algorithmic approaches to achieve this goal are present in the literature. This chapter provides some of these algorithms with specific emphasis on the ones investigated for our setup. Eventual improvements to these approaches to enhance pose estimation performances are also herein presented.

3.1 Iterative Closes Point (ICP) Algorithms

Considering the more general class of registration problems, i.e, 3D-to-3D registration, the individual point pairs (s_i, m_i) correspondences are unknown a priori. The iterative closes point (ICP) algorithm proposed in [22] is an approach for solving this class of problems since, it handles the full 6 degrees of freedom (dof) and requires neither derivative estimation nor feature extraction. This algorithm is the basis of most registration algorithms and is very popular because of its simplicity and clarity. However, the algorithm in its base form works only in ideal cases. This will, therefore, lead us to further modifications and improvements, that will be discussed in Section 3.1.3

3.1.1 Mathematical Background

Some mathematical background related to the ICP algorithm is hereafter recalled. Let's consider in general two point-sets M and S in a finite-dimensional vector space $\mathbb{R}^d$ with N_m and N_s points respectively. What we want to do is to determine the rigid (or non-rigid) transformation $T: \mathbb{R}^d \rightarrow \mathbb{R}^d$ that minimizes the distance (mean squared distance) between the two point-sets:

$$RMSD(T(M), S) = \frac{\sum_{m \in T(M)} \sum_{s \in S} |m - s|^2}{n} \tag{3.1}$$

In presence of outliers, i.e, situations where the two sets do not have the same number of points, only partial overlapping can be performed. This leads us to setting a threshold parameter that eliminates point pairs which are greater than the user-specified threshold (thr).
Thus, for a set of pairs of points $C = (m_i, s_j)$, where $m_i \in M$, $s_i \in S$, $dist(\cdot)$ is the euclidean distance function, it must hold that:

$$\forall s_k \in S: dist(m_i, s_k) \geq dist(m_i, s_j), dist(m_i, s_j) < thr$$

Hence (3.1) can be rewritten to manage outliers as:

$$RMSD(T(M), \mu(S)) = \frac{\sum_C dist(T(m_i), s_j)^2}{|C|}, m_i, s_j \in C \tag{3.2}$$

where $\mu(S)$ is the point-set such that $\forall m_i, s_j \in M, S, dist(m_i, s_j) < thr$. The distance $dist(T(m_i), s_j)$ is minimized and $|C|$ is the cardinality of set C.

3.1.2 ICP Formulation

• ICP mathematical description

Let M and S be two point clouds with finite number of points N_m and N_s, respectively. Firstly, we compute the nearest point in S for every point in M. The points can be chosen randomly. The ICP algorithm assumes that corresponding points are the nearest ones. For these pairs, we compute the rotation matrix and translation vector and try to minimize the error:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_m} w_{ij} \|s_i - (\boldsymbol{R} m_j + \boldsymbol{t})\|^2 \tag{3.3}$$

where w_{ij} are the weights of corresponding points. If point m_i is the nearest point to s_j, we set the weights as $w_{ij} = 1$, otherwise $w_{ij} = 0$.
We can then rewrite (3.3) as:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \frac{\sum_{i=1}^{N} \|s_i - (\boldsymbol{R} m_j + \boldsymbol{t})\|^2}{N} \tag{3.4}$$

where

$$N = \sum_{i=1}^{N_s} \sum_{j=1}^{N_m} w_{ij} \tag{3.5}$$

and the corresponding matrix of weights w_{ij} is substituted by the sum of corresponding points. Singular Value Decomposition (SVD) is then used in the optimization step to find the optimal transformation $T(\boldsymbol{R}, \boldsymbol{t})$. However, we need the centroid of alignment to compute SVD. Having determined point pair correspondences, the centroids of both point clouds can be computed as:

$$C_M = \frac{\sum_{i=1}^{N} m_i}{N} \tag{3.6}$$

$$C_S = \frac{\sum_{i=1}^{N} s_i}{N} \tag{3.7}$$

After determining the centroids, we align all the points of M and S using their corresponding centroid and we have:

$$M' = \{m_i' = m_i - C_M\}_{i \ldots N}$$
$$S' = \{s_i' = s_i - C_S\}_{i \ldots N}$$

With their corresponding matrix forms given by $\boldsymbol{M}'$, $\boldsymbol{S}'$ respectively.
The covariance matrix $\boldsymbol{H}$ is calculated as:

$$\boldsymbol{H} = \boldsymbol{M}'\boldsymbol{S}'^{T} \tag{3.8}$$

The decomposition of $\boldsymbol{H}$ into its Singular value components gives:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{T} \tag{3.9}$$

The rotation matrix is then given by:

$$\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^{T} \tag{3.10}$$

And the translation vector is calculated as:

$$\boldsymbol{t} = C_S - \boldsymbol{R}C_m \tag{3.11}$$

• Execution steps of ICP

Hereafter is presented an overview of the ICP algorithm which considers management of outliers [20]: Given two input point clouds M and S, where M is the model point cloud with N_m data points and S is the scene point cloud with N_s data points, the goal is to find the rigid transformation that guarantees the best correspondence between these point clouds. This can be formulated in the following steps:

1. Computation of the nearest point in S for every point (eventually part of points) in M, e.g, using the Euclidian distance:

$$dist_i = \min\left\{\sqrt{m_i^2 - s_j^2}\right\},\ j = 1...N_s \tag{3.12}$$

2. Eliminate the point pair if $dist_i > thr$

3. Assignment of weight to individual point pairs: typically the weight is $w_{ij} = 1$. The weights could, however, be set as the dot product between the normal vectors to the point pairs:

$$w_{ij} = n_i n_j \tag{3.13}$$

4. Computation of the rotation matrix $\boldsymbol{R}$ and translation vector $\boldsymbol{t}$ using distance minimization least square method in equations (3.6)-(3.11).

5. Computation of the transformation of M using the values of $\boldsymbol{R}$ and $\boldsymbol{t}$, from the previous step as: $\boldsymbol{R}m_i + \boldsymbol{t}$

6. Computation of the registration error $E(\boldsymbol{R}, \boldsymbol{t})$ using (3.4).

7. Repeat steps 1-6 until required accuracy or maximum number of iterations (if set) is attained.

The computation of the closest point in step 1 equally outputs the set of point pair correspondences $C = \{(m_i, s_i)\}_{i=1\ldots N}$, since each point in S is paired to N_m points from M. Hence N will turn out to be equal to N_s in case no point pair was eliminated due to the threshold distance constrain. Figure 3.1 Shows a Flowchart and pseudocode of the ICP algorithm:

Algorithm 1 Algorithm_ICP

1: **procedure** ICP(M, S, T_0)
2: $T \leftarrow T_0$
3: **while not** *Registered* **do**
4: $C \leftarrow \emptyset$
5: **for** $m_i \in Transform(M, T)$ **do**
6: $\hat{s}_j \leftarrow$ *Closest point in* S *to* m_i
7: $C \leftarrow C + (m_i, \hat{s}_j)$
8: $T \leftarrow SVD(C)$
9: **return** T

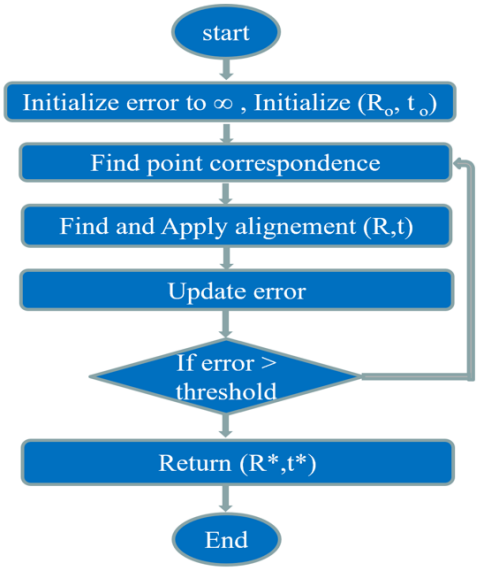

Figure 3.1: Pseudocode (left) and Flowchart (right) of ICP. Herein, $\boldsymbol{T}$ is the transformation matrix (rotation matrix $\boldsymbol{R}$ and translation vector $\boldsymbol{t}$), $\boldsymbol{T_0}$ is the initial transformation matrix ($\boldsymbol{R_0} \in \mathbb{R}^{3\times3}$ identity matrix and $\boldsymbol{t_0} \in \mathbb{R}^3$ zero-translation vector), set C contains all point pair correspondences that cover the closest points in S to points in M. We then iteratively calculate the new transformation $\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{t})$ using SVD ((3.6)-(3.11)) and apply the calculated $\boldsymbol{T}$ to the model points M and the error is computed. The iterations continue until convergence ($(\boldsymbol{R}^*, \boldsymbol{t}^*)$) or maximum number of iterations (if set) is reached.

3.1.3 Enhancements to ICP Algorithm

Due to convergence issues and the high computational cost ($O(N_m N_s)$) of the closest neighbor search in the ICP algorithm, several approaches have been developed to speed up the algorithm. Some of them are hereafter presented:

- Guidance to convergence region:

  The ICP algorithm works quite well, especially when there exists an approximate estimate for initialization that can reduce the number of registration iterations. However, this algorithm is subject to the problem of local minima for which a theorem exists and is proven in [22]:

  "The iterative closes point algorithm always converges monotonically to a local minimum w.r.t. the mean-square distance objective function."

  There is, therefore, no guarantee that the ICP algorithm will converge to the global minimum. In such situations when non-global convergence is a problem it is necessary to develop

techniques for guiding the algorithm to the region of global minimum. One of these techniques is presented in [22]. The idea behind this technique is based on the assumption that there exist a finite set of local minima for the 2 point-sets to be registered. This set of local minima can be partitioned into equivalence classes such that one of theses classes maps to the correct global minimum. To guarantee that the global minimum is found during the registration, it is necessary to use an appropriate set of initial transformations such that transforming the model object by at least one of these initial transformations will place it into the correct equivalence class. This will then guide the registration to the global minimum. Methods for determining the equivalence classes and the set of initial transformations are discussed in [22].

- Using a k-dimensional tree:

  Finding the closest points (step 1 in the execution steps of ICP algorithm above) is computationally expensive. In general, having N_m points in the model point cloud and N_s points in the scene point cloud, the complexity of a single closest point query is $O(N_m N_s)$ in the worst case and $O(N_s \log(N_m))$ in the average case. However as suggested in [22], an expected cost of $O(\log(N_m))$ using a k-dimensional tree (simply k-d tree) can be achieved. The k-d tree being a special case of the binary partitioning reduces the closest point computation to a binary tree search. At each node of the tree, a test is performed to determine on which side of a hyperplane the closest point lies.

- Weighting of points:

  In the trivial case all points have the same weight (ideally $w_i = 1$). However, it is suitable to assign a weight to points based for instance on their relative distance. Depending on the distance of the points, we can decide to either eliminate the point if it exceeds a predefined threshold or assign a weight to the point pair based on the formula proposed in [27]:

  $$weight = 1 - \frac{Dist(s_1, m_2)}{Dist_{max}} \tag{3.14}$$

  Alternatively, weight can be assigned based on the direction of the normal vectors to the points, as their scalar product varies depending on their spatial orientation. If the normal vectors at both points are parallel, the dot product is larger while it decreases with increasing size of angle-normal and reduces to zero in the perpendicular case. The weight can thus be assigned as:

  $$weight = n_1^s n_2^m \tag{3.15}$$

- Multiresolution scheme:

  The idea behind using a multiresolution approach to improve the performance of ICP is to use a down-sampled data-set for the first few iterations and increase the resolution of the data-set later in the next iterations. For example, one could progressively scale up the resolution from $\frac{1}{5}$ of the data-set in the initial iterations up to the entire data set towards the final iterations thereby achieving a coarse to fine registration.

  This approach reduces the computational cost, since the duration of each iteration carried out at low resolution is considerably reduced. As suggested in [28], where multiresolution ICP is used for image registration, the final matching precision is expected to be the same as in the case of using all the data-set throughout the whole registration. Besides, the total number of iterations is expected not to be greater than the one in the single resolution case.

3.2 Colored ICP

The use of an RGB-D camera in our setup provides additional information (colour) besides the 3D location of the point cloud. Dealing with a coloured scenario, this photometric information could help increase the accuracy of the registration by using colour along with the geometric location of points. Each point in the point cloud has an associated RGB colour.
The simpler 3D ICP problem formulation presented in Section 3.1.2 is therefore raised to a higher 6D dimensional space parameterized by both position and colour. Correspondences between the scene and model point clouds for our point-to-point registration are therefore established in a 6D space rather than a 3D space. The registration goal is that of optimizing the joint photometric and geometric cost function represented below:

$$E(\boldsymbol{R},\boldsymbol{t}) = (1-\sigma)E_C(\boldsymbol{R},\boldsymbol{t}) + \sigma E_G(\boldsymbol{R},\boldsymbol{t}) \tag{3.16}$$

where E_C and E_G are the photometric and geometric terms, respectively, and $\sigma \in [0,1]$ is a weight used to balance the two terms.
This formulation of the problem presented by Park(2017) [29] and used for scene reconstruction provides a rather complex definition of the photometric term E_C. The photometric term in the latter case is defined as a continuous and differentiable function whose gradient is an indication of how colour varies as a function of position, whereas in our case-study, we simply define the photometric term as the RMSE variation of RGB colours associated to scene and model points.

$$E(\boldsymbol{R},\boldsymbol{t}) = \sum_{i=1}^{N_s}\sum_{j=1}^{N_m} w_{ij}\|s_i - (\boldsymbol{R}m_j + \boldsymbol{t})\|^2 \tag{3.17}$$

where,

$$w_{ij} = \begin{cases} 1, & \text{if } \|(m_i - s_j) + \gamma(c_i^m - c_i^s)\|^2 < thr \\ 0, & \text{otherwise} \end{cases}$$

where, c_i^s and c_i^m are RGB colours associated to scene point s_i and model point m_i, respectively, and $\gamma \in [0,1]$ is the color scale.
The complex definition of the photometric term in [29] is aimed at reducing the dimensionality of the problem and the error due to distant correspondences that have a similar colour. This is however not the case in our setup since the Mikado sticks have distinctive colour strands to help identify them. Hence the decision to bear the 6D parameterization cost is made.
The colour information plays a vital role in orienting the model Mikado stick toward the correct Mikado stick in the scene, since geometrically all the sticks are identical. Obviously, all the possible improvements to ICP algorithm discussed in Section 3.1.3 equally hold in this case, since this formulation is merely an extension of the former one raised to a higher dimensional space.

3.3 Globally Optimal ICP (Go-ICP)

Due to its iterative nature, the ICP algorithm is well known for its problem of local minima. This imposes the necessity of a good initialization. Without such an initialization, it easily gets stuck into local minima which may be very far from the optimal solution.

The necessity to overcome these weaknesses for our specific setup made us investigate an alternative algorithm: the Globally optimal iterative closest point algorithm, which guarantees global optimality without the need for a good initialization. Just as the regular ICP, GO-ICP relies on the search of closest points at each iteration.

Overview of Go-ICP

Go-ICP achieves global optimality by solving the registration problem formulated as a mixed-integer problem. The underlying mixed-integer problem is solved using the branch-and-bound (BnB) technique.
The BnB technique is not a solution method limited only to the class of mixed-integer problems since it can be applied to several problems of different nature.
This technique is based on the principle that the total set of feasible solutions can be partitioned into smaller subsets of solutions. These smaller subsets are then evaluated systematically until the best solution is found. BnB uses a tree diagram of nodes and branches to organize the solution partitioning. An optimal integer solution is reached when a feasible integer solution is achieved at a node that has an upper bound greater than or equal to the upper bound at any other ending node.
Go-ICP is summarized in [30] as follows:
"Use BnB to search the space of $SE(3)$

> Whenever a better solution is found, call ICP (initialized at this solution) to refine (reduce) the objective function value.
>
> Use ICP's result as an updated upper bound to continue the above BnB search.

Until convergence."
$SE(3)$ is the spatial euclidean group in $\mathbb{R}^3$ that consists of rotations and translations in $\mathbb{R}^3$.
The Go-ICP using a BnB scheme efficiently searches the entire 3D motion space $SE(3)$. By exploiting the special structure of $SE(3)$ geometry, it continuously derives novel upper and lower bounds for the registration error function. The local ICP is integrated into the nested BnB scheme, which speeds up the registration method while guaranteeing global optimality as shown in Figure 3.2

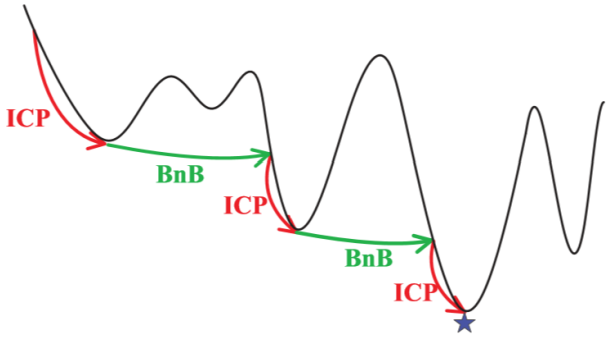

Figure 3.2: Collaboration between BnB and ICP to achieve global minimum. Whenever the outer BnB finds an upper bound better (less) than the so-far lower bound, the classical ICP algorithm is called to take over with new initial parameters. By doing so, the ICP jumps out of the previous local minima and the function value is progressively decreased until global optimality is achieved.

Despite its ability to achieve global minimum, due to its computational cost, Go-ICP is especially useful for practical scenarios where having an exact optimal solution is highly desirable. Go-ICP does not apply to real-time critical application. Also, as experimented in [30] for pose estimation of an object in a cluttered scene (which is the situation for our specific setup), Go-ICP has a registration time which is quite long compared to real-time expectations. For these reasons the algorithm is only briefly described and will not be tested in our registration scenario. However, more details on Go-ICP can be found in [30].

3.4 Point Pair Feature (PPF) Registration Algorithm

The classical ICP algorithm performs well for registration purposes mostly in ideal situations such as: the absence of clutter in the scene, absence of occlusion, good initialization $(\boldsymbol{R_0}, \boldsymbol{t_0})$, and to some extent it requires the scene and model point clouds to have the same number of points. Go-ICP on the other hand does not necessarily need a good initialization to guarantee optimal registration, but unfortunately it is not suitable for real-time applications due to its high computational cost. All these limitations lead us into examining another registration algorithm: the Point pair feature (PPF) registration algorithm. This algorithm will be implemented for our specific setup in an attempt to have better 3D pose estimation performances.
The alternative approach we adopted for solving the problem of pose determination of free-form 3D objects in point clouds as proposed in [15] is based on oriented point pair features. Compared to traditional approaches based on point descriptors, whereby the registration is performed by directly exploiting local information between points in the two point clouds, the PPF algorithm creates a global model description based on oriented point pair features and matches that model locally using a fast voting scheme. The global model description consists of all model point pair features and represents a mapping from the point pair feature space to the model, where similar features on the model are grouped together. This approach allows the use of much sparser model and scene point clouds. Registration is performed locally on a reduced two-dimensional search space using an efficient voting scheme. Conceptual description and in-depth overview of the algorithm are presented in the following subsections.

3.4.1 Point Pair Feature (PPF)

The PPF algorithm is based on the withdrawal of point pair features from the model and scene point clouds, which are then used for computing the feature vectors. The idea presented here is similar to the surflet-pair feature described in [31], whereby local information on the surface is provided by the relative position and orientation of surface point pairs. A schematic representation of the information is presented in Figure 3.3

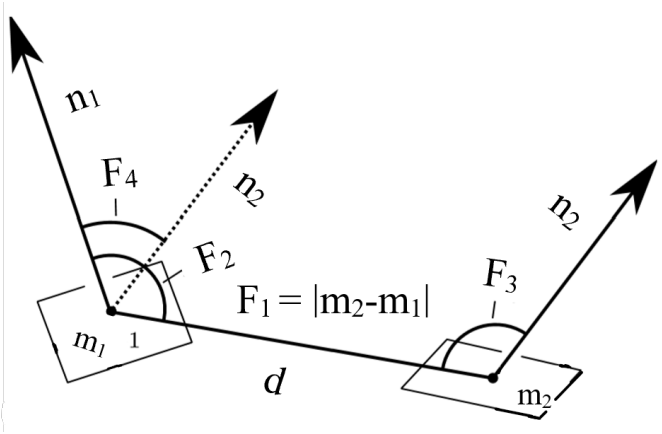

Figure 3.3: Point pair feature F of two oriented points. m_1 and m_2 are oriented points with normals n_1 and n_2 respectively. The distance between the point pair is given by d. The distance, the angles between both normals and the distance form the features of the point pair.

Given a pair of oriented points m_1, m_2, with normals n_1, n_2, the distance between them is set as $d = m_2 - m_1$ and the feature vector $\boldsymbol{F}$ is defined as:

$$\boldsymbol{F}(m_1, m_2) = (\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2)) \tag{3.18}$$

where $\angle(a, b) \in [0; \pi]$ denotes the vector angle between a and b. This feature $\boldsymbol{F}$ assumed symmetric ($\boldsymbol{F}(m_1, m_2) = \boldsymbol{F}(m_2, m_1)$) is then used for creating the global model description and later for estimating the pose of the object of interest in the scene.

3.4.2 Global Description of the Model

In the offline phase of the algorithm, a global description of the model is built using the point pair feature described in Section 3.4.1. The global model is defined by a set of point pair features, whereby similar feature vectors are grouped together. This grouping of feature vectors to describe the model is done by calculating the feature vector $\boldsymbol{F}$ of equation (3.18) for all point pairs $m_i, m_j \in M$ (the model point cloud). The distances are then sampled in steps of d_{dist} and the angles in steps of $d_{angle} = 2\pi / n_{angle}$. Feature vectors of same distance and angles are then grouped together. The global model description is therefore represented as a mapping from the sampled point pair feature space to the model: $L: \mathbb{R}^4 \rightarrow A \subset M^2$ where the four dimensional point pair features in equation (3.18) are mapped to set A of all pairs $(m_i, m_j) \in M^2$ that define an equal

feature vector. This is better illustrated in Figure 3.4, which shows an example of point pairs with similar features on a single object.

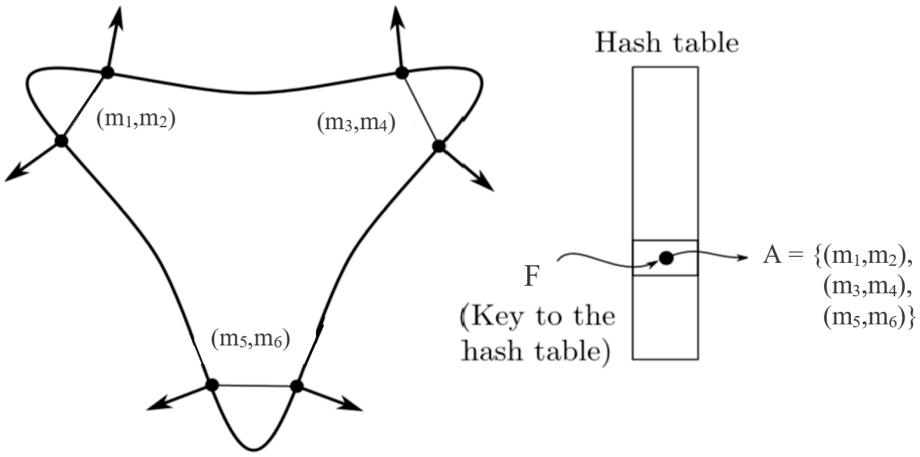

Figure 3.4: Global description of the model and hash table representation. Same features on the model object are collected in a set A and stored in the same location in the hash table.

Same features on the object are collected in a set A. From an implementation point of view, this description of the model is represented as a hash table which can be accessed directly using the sampled feature $\boldsymbol{F}$ as key. Point pairs on the model surface with the same feature vector $\boldsymbol{F}$ are stored in the same slot in the hash table. Therefore by using $\boldsymbol{F_s}$ as key, all model features $\boldsymbol{F_m(m_i, m_j)}$ matching a given scene feature $\boldsymbol{F_s(s_i, s_j)}$ can be retrieved on average in constant time and used for pose determination.

3.4.3 Local Coordinate and Voting Scheme

Once the global model description is computed, it can then be queried in the online phase of the registration process to determine the final object pose. To achieve this, local poses are first determined for the set of scene point pairs that have similar feature vectors as the model point pairs. These local poses are then grouped (clustered) in a systematic way after voting for each of them. Details on Pose clustering and the voting scheme are presented in the later sections. Central to local pose determination is the concept of local coordinates described hereafter.

Local Coordinates

Let's assume an arbitrary reference $s_r \in S$ (scene point cloud) lies on the object whose pose is to be estimated. If this assumption is correct, then there exists a point $m_r \in M$ (model point cloud) that corresponds to s_r. Alignment of s_r, m_r and their normals leads to a decrease in the number of dof for the pose of the model in the scene. This is because, after such an alignment, only a rotation of the model around the normal of s_r is needed to align it to the scene. Thus, the rigid transformation from the model space into the scene space can be represented by a point on the model and a rotation angle α. The pair (m_r, α) is defined as the local coordinates of the model w.r.t the reference point s_r. A point pair $(m_r, m_i) \in M^2$ is therefore aligned to a scene pair

$(s_r, s_i) \in S^2$ whenever both pairs have the same feature vector $\boldsymbol{F}$. The transformation from the local model coordinates to the scene coordinates is given by:

$$s_i = T_{s\to g}^{-1} R_x(\alpha) T_{m\to g} m_i \tag{3.19}$$

This transformation is explained in [15] as sketched in Figure 3.5.

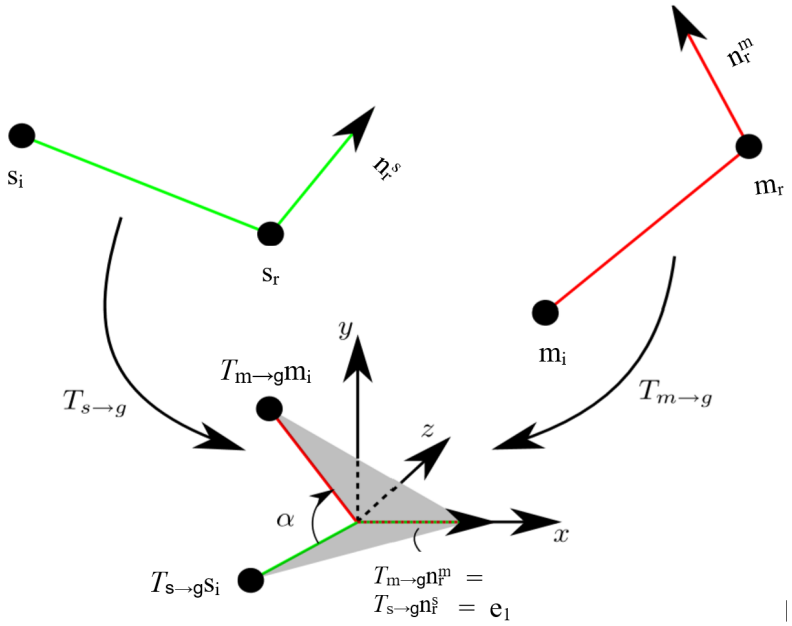

Figure 3.5: Model and scene Coordinates transformation. Set of transformations necessary to align a model point pair to a scene point pair. $T_{m\to g}$ is the transformation that translates m_r into the origin and rotates its normal n_r^m onto the x-axis. $T_{s\to g}$ is the transformation that translates s_r into the origin and rotates its normal n_r^s onto the x-axis. $R_x(\alpha)$ is the rotation around the x-axis of angle α to match the s_i and m_i

The angle α is computed as $\alpha = \alpha_m - \alpha_s$ where $\alpha_m = \angle(T_{m\to g} m_i, y)$ is the angle between $T_{m\to g} m_i$ and the y-axis and $\alpha_s = \angle(T_{s\to g} s_i, y)$ is the angle between $T_{s\to g} s_i$ and the y-axis.
Important to note here is the fact that compared to a general rigid transformation which has 6dof in 3D, the local coordinates transformation have only 1dof (the rotation angle α)

Voting Scheme

Pose voting is crucial since the global object pose can easily be retrieved once the optimal local coordinates are found. The scheme hereafter used for voting is very close to the Generalized Hough Transform (GHT). GHT solves the problem of finding the transformation's parameter that maps a given model onto an image. Given the value of the transformation's parameter, the position of the model in the image can be determined. However, the voting scheme used in this work could be more promising since the local coordinates has only 1dof. The goal of the voting scheme is, therefore, that of finding optimal local coordinates (m_r, α) that maximize the number points in

the scene that lies on the model, given a fixed reference point s_r on the scene point cloud. To this aim, a 2D-array $N_m \times N_{angle}$ is created to accumulate the votes, where N_m is equal to the number of model sample points $|M|$, and N_{angle} is the number of sample steps of the rotation angle α. The discrete space of local coordinates of a fixed reference point is defined by this accumulator array. The Actual voting is performed by pairing every other point $s_i \in S$ in the scene with the reference point s_r to form the feature vector $\boldsymbol{F_s}(s_r, s_i)$, and the model surface is searched to find point pairs (m_r, m_i) that have the same feature vectors as $\boldsymbol{F_s}$. This search is done using the precomputed global model description and answers the question of where on the model the scene point pair (s_r, s_i) could be located. The computed feature $\boldsymbol{F_s}(s_r, s_i)$ is used as key to access the hash table of the global model description which in turn outputs the set of similar features on the model.
For every possible location of (s_r, s_i) on the model surface, the rotation angle α that corresponds to the local coordinate (m_r, α) that maps (m_r, m_i) to (s_r, s_i) is calculated using equation (3.19), as depicted in Figure 3.5. A vote is cast for the local coordinates (m_r, α). The steps involved in the voting scheme proposed in [15] are depicted in Figure 3.6. Once all the scene points $s_i \in S$ are processed, the entries of the accumulator array with maximum values correspond to the optimal local coordinates. However, to improve registration stability, all entries with values (number of votes) relative to the maximum value based on a predefined threshold can be used. From these optimal local coordinates, a rigid movement can be performed to estimate the 3D pose of the corresponding model object in the scene.

Enhancement to the Voting Scheme

Improvement of the PPF algorithm is done by making the voting stage of the registration faster. We obtain this by speeding up the solving of equation (3.19): for every point pair in the list, α is split into two parts, $\alpha = \alpha_m - \alpha_s$, such that α_m and α_s depend only on the point pair on the model and scene, respectively. $R_x(\alpha)$ is equally split into $R_x(\alpha) = R_x(-\alpha_s)R_x(\alpha_m)$ and $R_x^{-1}(-\alpha_s) = R_x(\alpha_s)$ is used to obtain:

$$t = R_x(\alpha_s)T_{s\to g}s_i = R_x(\alpha_m)T_{m\to g}m_i \in \mathbb{R}_x + \mathbb{R}_0^+ y \tag{3.20}$$

Equation (3.20) suggests that t lies on the half-plane defined by the x-axis and the positive part of the y-axis. This guarantees the uniqueness of t (i.e, solving for t will give a unique solution) for each point pair in the model or scene point cloud. Hence, it is possible to include an additional parameter in the global description of the model by precomputing α_m for every model point pair in the off-line phase of the algorithm. In this way, α_s can be calculated only once for every scene point pair (sr, si) and the final rotation angle α is a trivial difference in the values of α_m and α_s.

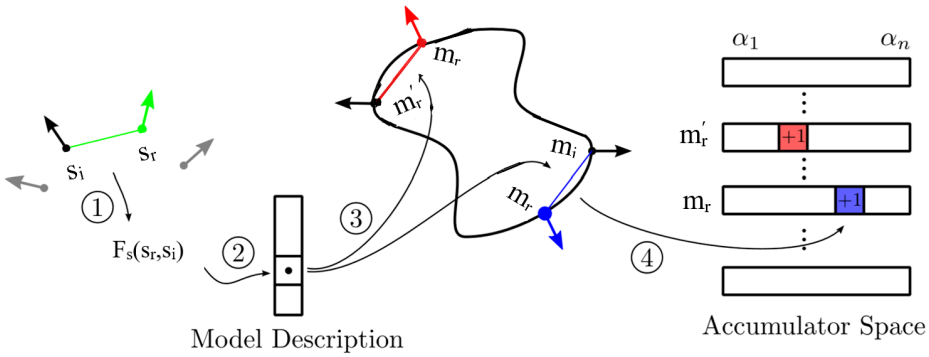

Figure 3.6: Steps of the voting scheme. After pairing the reference point s_r with every other point s_i in the scene, calculating and matching the feature vector $\boldsymbol{F}$ to the global model description, the rotation angle α is computed for each scene-model point pair match. A vote is then cast for the local coordinate (m_i, α).

The outline of the voting scheme (Figure 3.6) includes:

① Pairing of the reference point s_r with every other point s_i in the scene and calculation of the feature vector $\boldsymbol{F}$.

② Matching of feature $\boldsymbol{F}$ to the global model description and the consequent return of a set of point pairs on the model that have the same feature vector (similar distance and orientation).

③ Calculation of the rotation angle α for each point pair on the model matched to the point pair in the scene.

④ Voting for the local coordinate (m_i, α)

3.4.4 Pose Clustering

Since the voting scheme proposed in section 3.4.3 successfully determines the pose of the object of interest in the scene only if the reference point s_r lies on the surface of the object, multiple reference points are therefore necessary to ensure that one of them lies on the object whose pose is to be determined. As described earlier, every single reference point returns a set of possible object poses that correspond to the maximum vote in its accumulator array. The returned set of possible object poses only approximates the ground truth pose. This is because the sampling rates of the scene and model, and even the sampling rate of the rotation in the local coordinates are all different.
All the retrieved poses are then clustered and filtered. Filtering further improves the final pose estimate, since it eliminates from each cluster poses that differ in translation and rotation for more than a predefined threshold. After evaluating the scores of all clusters, the final object pose is determined by taking the average of all the poses contained in the cluster with the maximum score. While the score of a pose is simply the number of votes it obtained in the voting scheme, the score of a cluster is given by the sum of the number of votes of the individual poses contained in the cluster. Pose averaging further enhances the overall pose of the object. It is important to point out that several clusters with the same maximum value can be returned by this approach since multiple instances of the same object can be present in the scene.

3.4.5 Phases of the PPF Algorithm

After the in-depth overview of the various building blocks that constitute the PPF algorithm, it can be noticed that it is composed of an offline and an online phase as summarized in Figure 3.7. The off-line phase of the PPF approach for free form object pose estimation in point clouds is briefly composed of:

- Creation of a global model description through point pair feature extraction from the model point cloud to generate feature vectors.
- Grouping of model point pairs with similar feature vectors $\boldsymbol{F_m}$ into sets of point pairs.
- Storage of point pair sets in an Hash table for efficient query during the online phase.

The online phase on the other hand is briefly composed of:

- Set of reference points selection in the scene point cloud.
- Creation of point pair features $\boldsymbol{F_s}$ by pairing all other points in the scene with every single reference point ($\boldsymbol{F_s}(s_r, s_i)$).
- Hash table query using $\boldsymbol{F_s}$ as key to perform matching with global model description.
- Calculation of local coordinates (m_i, α) for each match between $\boldsymbol{F_s}$ and $\boldsymbol{F_m}$
- Local pose voting
- Pose clustering and final object pose determination

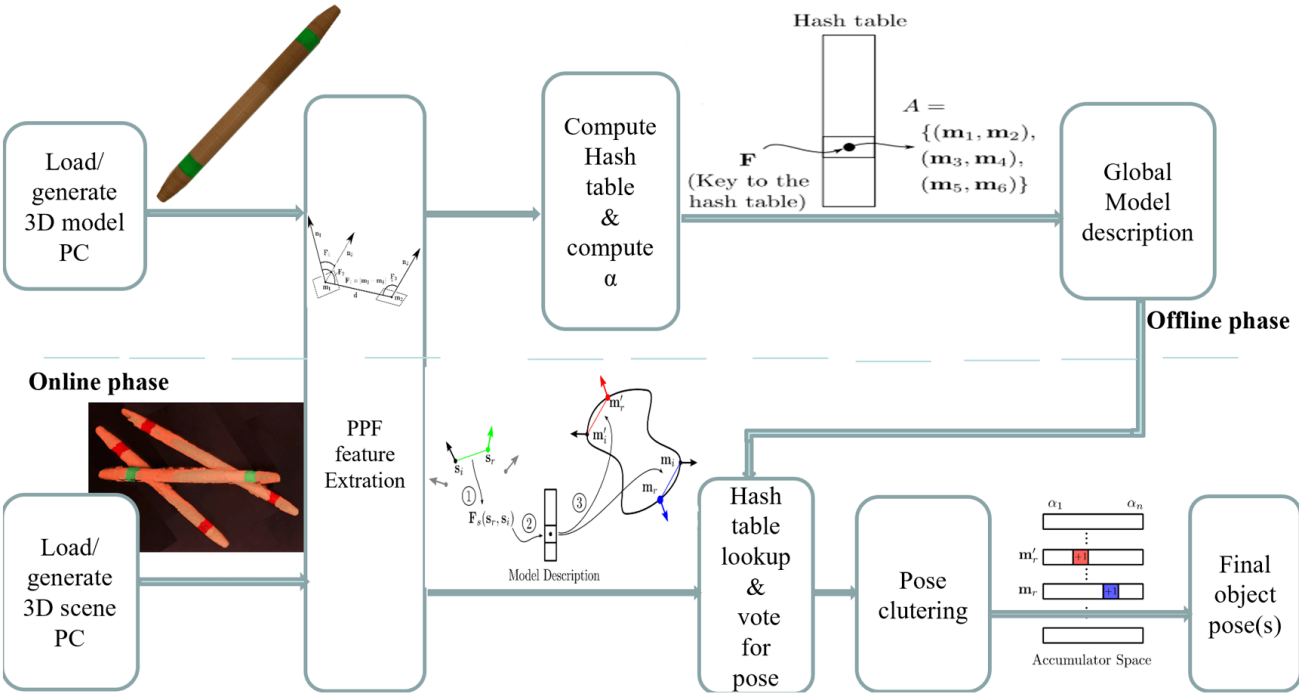

Figure 3.7: Flow chart of PPF algorithm. The offline phase consists of loading the model point cloud and extracting point pair features from it to generate the global model description. In the online phase, after selecting the reference points from the loaded scene point cloud and pairing them with all other points in the scene, feature vectors are computed. Corresponding model-scene point pair matches are retrieved by accessing the model Hash table. For each of these model-scene point pair correspondences, the local pose is computed and voted for. Following is the clustering of local poses, which leads to the computation of the final pose by averaging the poses of the cluster with the maximum score.

A summary of how various phases of PPF algorithm are executed is shown in Figure 3.7. In the offline phase, the model point cloud is loaded and point pairs are extracted from it to compute feature vectors. The value of α is also partially computed by computing α_m. Point pairs with similar feature vectors are grouped and saved in the same slot together with their values of α_m and the transformations necessary to perform local coordinates transformation (see Figure 3.5). In the online phase, the selected reference points from the scene point cloud are paired with all the other points in the scene, and the feature vectors are computed together with the values of α_s and the transformations necessary to perform local coordinates transformation(see Figure 3.5). These feature vectors are used as keys to retrieve all sets of point pairs that have similar feature vectors from the hash table. For each of the model-scene point pair match, local poses are computed and voted for. Once all the points have been retrieved, the local poses are clustered based on a predefined threshold in rotation and translation between poses of the same cluster. From the cluster with the maximum score, the final pose is calculated by averaging all the poses in that cluster.

An eventual improvement to this pose estimation algorithm in addition to speeding up the voting scheme as mentioned previously, could be the addition of a new parameter in the feature vector in

equation (3.18) that describes the color. This leads us to the coloured point pair feature algorithm (C-PPF).
C-PPF shares the same underlying concepts as PPF. The only difference resides in the dimension of the feature vector $\boldsymbol{F}$ which is 5 ($\boldsymbol{F} \in \mathbb{R}^5$) in this case. The norm of the RGB color values constitutes the fifth dimension in the feature vector.

$$\boldsymbol{F}(m_1, m_2) = (\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2), \|u\|_2,) \tag{3.21}$$

where $\angle(a, b) \in [0; \pi]$ denotes the vector angle between a and b and $\|u\|_2$ is the norm of the RGB color values of the reference point (s_r / m_r).
This will be implemented and evaluated in our specific setup in an attempt to improve registration performances.
One can conclude after the description of all the aforementioned registration algorithms that they all have in common the search of correspondences between model and scene point clouds. While ICP and its variants search for model-scene point correspondences to achieve registration, the PPF algorithm and its variant rather focus on the search of model-scene oriented point pair feature correspondences to achieve pose estimation.

4 Evaluation Data Generation

4.1 Model Objects and Point Cloud

The Mikado sticks used in our setup are modeled using a computer-aided design (CAD) software. Using this software, a 3D project file is created in a simulation environment where the 3D cylindrical central parts and the 3D conical extreme with flat ends are separately modeled. These individual parts are later put together to create complete Mikado sticks. The designed complete Mikado sticks are saved in a point cloud format and used as our model point cloud.

4.2 Scene Objects and Point Cloud

The real scene objects (Mikado sticks) used in our set up are created by 3D printing the CAD model sticks projected in Section 4.1.

• Synthetic scene point cloud.

The synthetic scene point cloud is merely a collection of different model Mikado sticks (point clouds) assembled in space with different positions and orientations as in Figure 5.5 (Section 5.2 of Chapter 5).

• Real scene point cloud.

Dealing with a model-based registration where the fitting error between the model and the scene point clouds is heavily influenced by how good the collected real-world data is, accurate collection of real-world data is, therefore, crucial to obtain good registration performance.

Strategic positioning of the robot end-effector (to which the RGB-D camera is attached) to guarantee a proper view of the scene is therefore mandatory. To achieve this, a grid of end-effector poses is hardcoded. The path followed by the end-effector is such that it covers all the cells of the grid. The end-effector pose, when positioned in each cell, is such that the focal length of the RGB-D camera attached to it always points toward the center of the scene.

A 3×3 grid of end-effector poses is used for real-world data collection. As the end-effector follows its trajectory from one cell to another, a pair of depth and colored images are taken and used to generate an instance of the scene point cloud. All the 9 instances of the point clouds are then adequately transformed and merged so that they properly fit together to form a unique and dense scene point cloud used for registration.

5 Experiments and Results

5.1 Equipment and Set-up

Robot:

The robot used in our setup is a Kuka LBR iiwas R820 manipulator, a lightweight robotic arm with 7 degrees of freedom. All motor units and current-carrying cables (except the cable connecting to the camera in our setup and the connecting cable to the cabinet robot controller) are protected beneath cover plates. Each axis is protected utilizing axis range sensors and can be adjusted using internal sensors. Each joint is equipped with a position sensor on the input side, torque sensors on the output side and temperature sensors. This allows the robot to be operated with position and impedance control. The temperature sensors prevent thermal overloading as the robot automatically stops if the threshold motor temperature is surpassed.

The robotic system (Figure 5.1) is composed of all the components of an industrial robot, including the manipulator (mechanical system and electrical installations), controller, connecting cables and a smart PAD control panel.

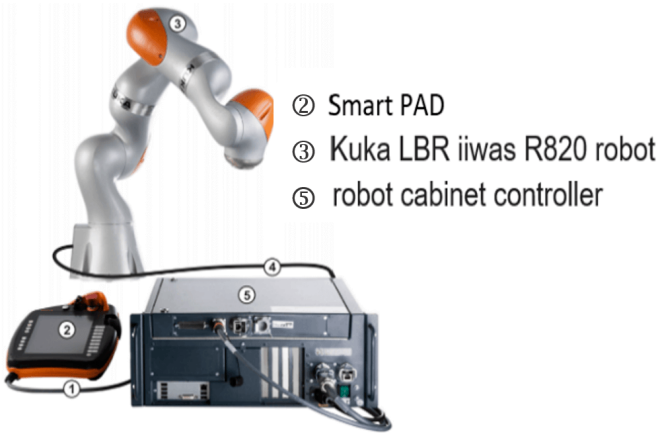

Figure 5.1: Robotic system. Composed of all components of an industrial robot, including the manipulator (mechanical system and electrical installations), controller, connecting cables and a smart PAD control panel.

Gripper:

The gripper used hereafter is the EZ robotic gripper due to its capability to be easily bolted to a wide variety of robotic arms, its gentle force, and firm grip. Its prehensile fingers enable pick up of objects ranging from small objects like pencils to much larger objects. This gripper adapts to the objects it has to pick up and works with very gentle force or a very firm

grip. These grip properties could be useful only in a future grasping task since in the object registration task the gripper only serves as a support for the camera as depicted below.

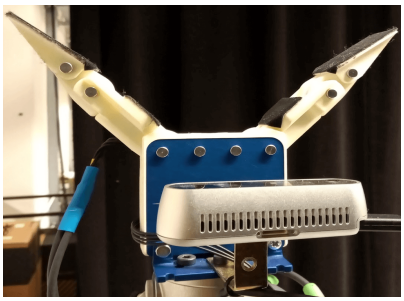

Figure 5.2: Gripper supporting camera. Serves mainly as a support for the camera in the registration task.

Vision sensor (Camera):

We used an Intel RealSense D435 RGB-D camera in our setup (Figure 5.3). This camera uses stereo vision to calculate depth. The stereo vision consists of a left imager and right imager, and an optional infrared projector. The infrared projector improves depth accuracy in scenes with low texture by projecting a non-visible static IR pattern. The left and right imagers capture the scene and send the captured data to the depth imaging processor, which calculates depth values for each pixel in the image. The depth pixel values are then processed to generate a depth image frame. The Depth Module, besides having the left and right imagers with the IR projector, equally incorporates an RGB color sensor. The RGB color sensor data are sent to the vision processor via the color Image signal processor generating a colored image. The pair of colored and depth images can then be used to generate the point cloud.

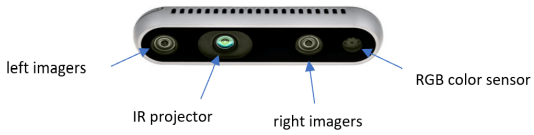

Figure 5.3: RGB-D Camera. Equipped with various modules necessary to collect the depth and colored images used to generate the point cloud.

Depicted in Figure 5.4 is the overall robotic setup comprising the robot, the camera attached to the gripper and the scene for the Mikado stick 3D pose estimation.

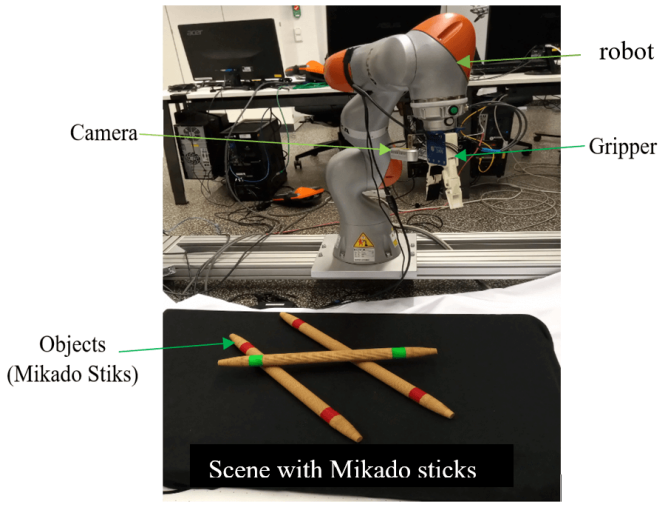

Figure 5.4: Robotic setup for object registration. Comprising the robot, the camera attached to the gripper and the scene for the Mikado stick 3D pose estimation.

5.2 Evaluation of Adopted Methods and Graphical Results

Using synthetic and real-world data generated as described in Chapter 4 and the aforementioned robotic setup, we tested our various 3D pose estimation algorithms. Proper down-sampling of the point clouds and enhancements to the registration process were necessary to ensure reasonable computation time. Some of these synthetic and real-world data are presented in Figure 5.6 (for evaluation of variants of the ICP algorithm) and Figure 5.19 (for evaluation of variants of the PPF algorithm).
The need to come up with a method for accurate evaluation of the registration error was equally necessary.

5.2.1 Error Evaluation Approach

- Rotation error

The special geometry of the objects to be registered makes it possible that rotation around the axis of maximum data variation leaves the object invariant. This can lead to a wrong evaluation of the registration error.
To limit the erroneous increase in the evaluation of the rotation error, it was calculated as the angle between the axis of maximum data variation of the model object and the corresponding scene object, see Figure 5.5. This, therefore, suggests that a maximum rotation error of $\pi/2$ rads could be attained. An increase in the rotation error due to unnecessary rotations about this axis could be observed once the model object had already achieved the right orientation w.r.t the corresponding scene object.

- Translation error

The translation error, on the other hand, was quantified as the Euclidean distance between the centroid of the model object whose pose is to be determined and the centroid of the corresponding scene object, see Figure 5.5.

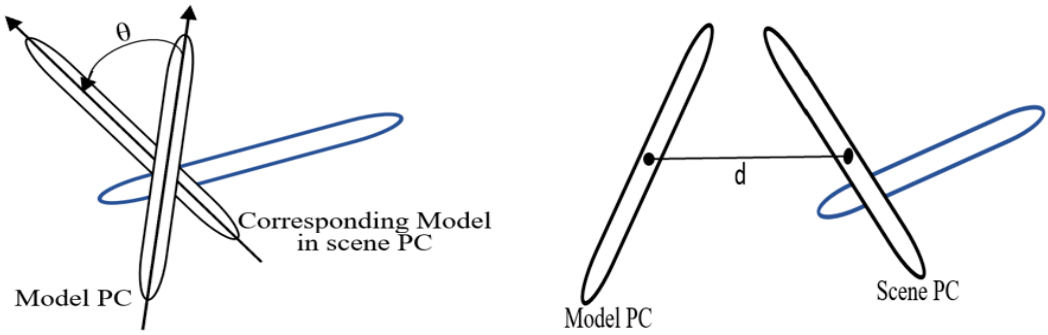

Figure 5.5: Schematic representation of the registration error evaluation. On the left is depicted the rotation error $E(R) = \theta$ while on the right is the translation error $E(t) = d$

For all experiments in this chapter, the registration error is evaluated as described in section 5.2.1 (See Figure 5.5)

5.2.2 Variants of ICP Evaluation and Graphical Results

Dealing with point clouds having approximately a million points each (Figure 5.6), direct implementation of a k-d tree to enhance computation time was mandatory. An attempt to run the algorithms without such enhancement even with a downsampled point cloud took several minutes. All reported experiments on variants of the ICP algorithm (non-colored and colored ICP) were performed directly on scenes with multiple sticks (see Figure 5.6 for example scenes), since we are particularly interested in these scene. Nevertheless, as starting point, these algorithms were tested on scene with single Mikado sticks

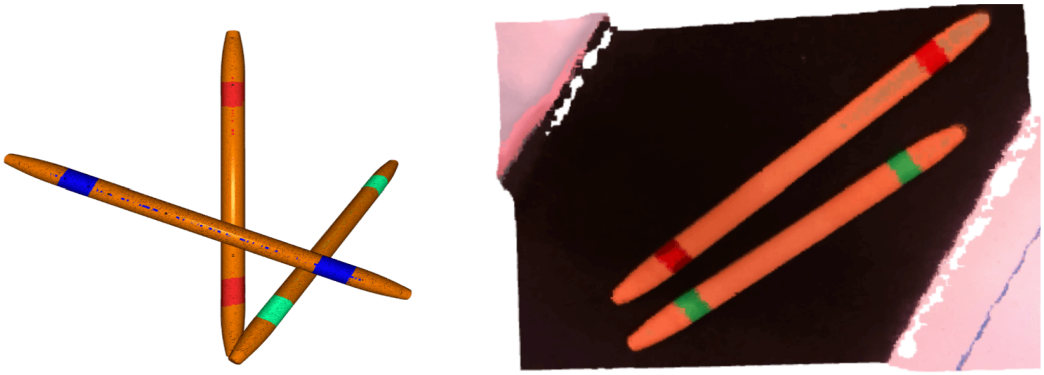

Figure 5.6: Synthetic (left) and real (right) point clouds used for 3D pose estimation.

Throughout the experiments in this section, both the synthetic and real-world data point clouds were uniformly downsampled with voxel sizes of 0.02mm and 0.03mm, respectively, such that all points had the same minimum distance equal to the voxel sizes between them. This further reduced computation time as the point clouds were less dense.
Two variants of the implemented ICP algorithm (non-colored ICP: ICP and colored ICP: C-ICP) were evaluated using the synthetic and real data. To evaluate the robustness and convergence of the

implemented approaches, we ran each algorithm 50 rounds, with each round having a maximum of 20 iterations. For each round, using the same seed for the synthetic and real data, a transformation matrix was generated with random rotations $\theta_{xyz} \in [-2\pi rads, 2\pi rads]$ about the 3 axes (x, y, and z-axis) and random translations $t_{xyz} \in [-500mm, 500mm]$ along the 3 axes. The resulting rotation and translation errors were averaged and plotted for the 50 rounds.
Dealing with Mikado sticks of the same geometry, the color information plays a fundamental role, since without it the model object can easily get registered to the wrong scene object.

Basic ICP Assessment

Using both synthetic and real data, evaluation of the implemented basic ICP (i.e, without exploiting information on color) gave the results reported in Figure 5.7.

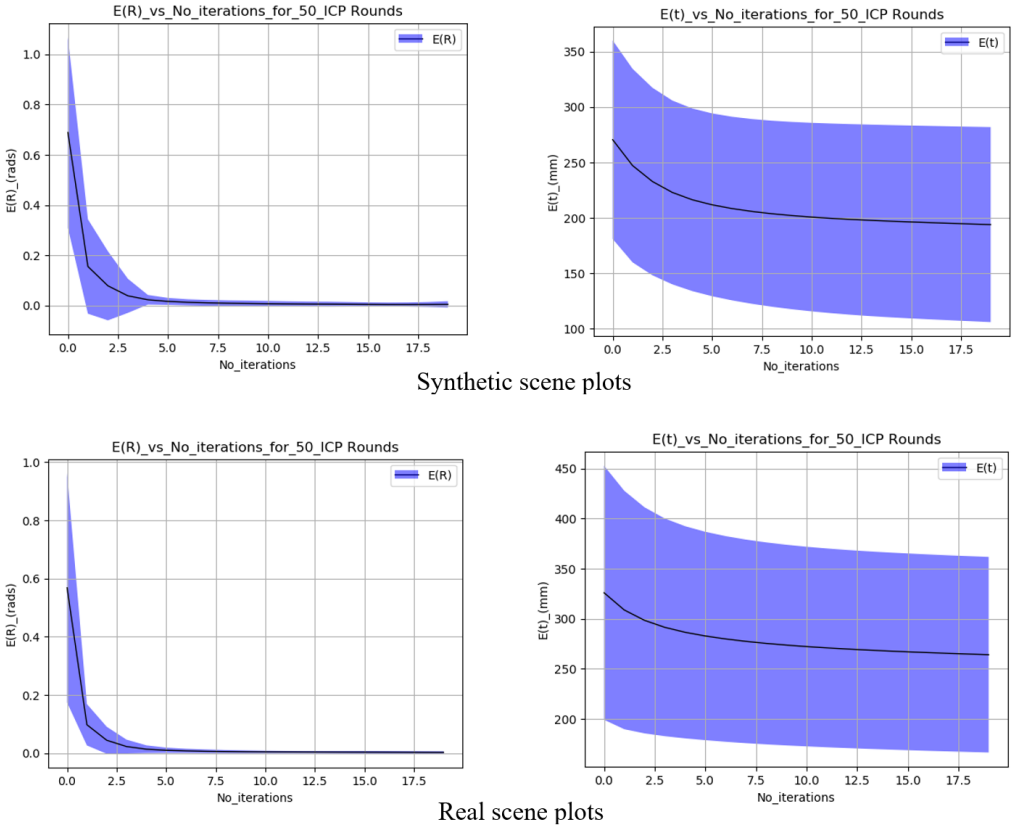

Figure 5.7: Error plots of average rotation and translation vs number of iterations of the basic ICP assessment. Synthetic scene average rotation error (top left) and translation error (top right), and real scene average rotation error (bottom left) and translation error (bottom right) for 20 Iterations per ICP round for 50 ICP rounds. Their confidence intervals are represented by the shaded portions in blue.

After downsampling the model, the synthetic and the real-scene point clouds as mentioned earlier, ICP was run 50 times and the characteristic error plots and their confidence intervals were obtained (See Figure 5.7). Each ICP round was composed of a maximum of 20 iterations. The rotation and translation errors were averaged over the 50 ICP rounds for the 20 iterations and plotted for the

synthetic and real scenes. The confidence intervals were calculated as the standard deviation from the returned average errors.

$$E(R) = \overline{E(R)} \pm \sigma_R \tag{5.1}$$

$$E(t) = \overline{E(t)} \pm \sigma_t \tag{5.2}$$

where $\overline{E(R)}$ and $\overline{E(t)}$ are the mean rotation and translation errors, respectively, with their corresponding standard deviations σ_R and σ_t.
It can be observed from Figure 5.7 that the rotation error abruptly decreases during the initial iterations with a steeper gradient in the synthetic and real data scene scenarios, and rapidly gets very close to the expected value of 0rads. The translation error slowly converges from the initial to final iterations and does not reach the expected value of 0mm in both real and synthetic data cases. The synthetic scene ends up registering a translation error of 186mm, which is better compared to that of the real scene 263mm. The translation error, therefore, accounts for the greater overall registration error for both synthetic and real scenarios with a very wide confidence interval compared to that of the rotation error.

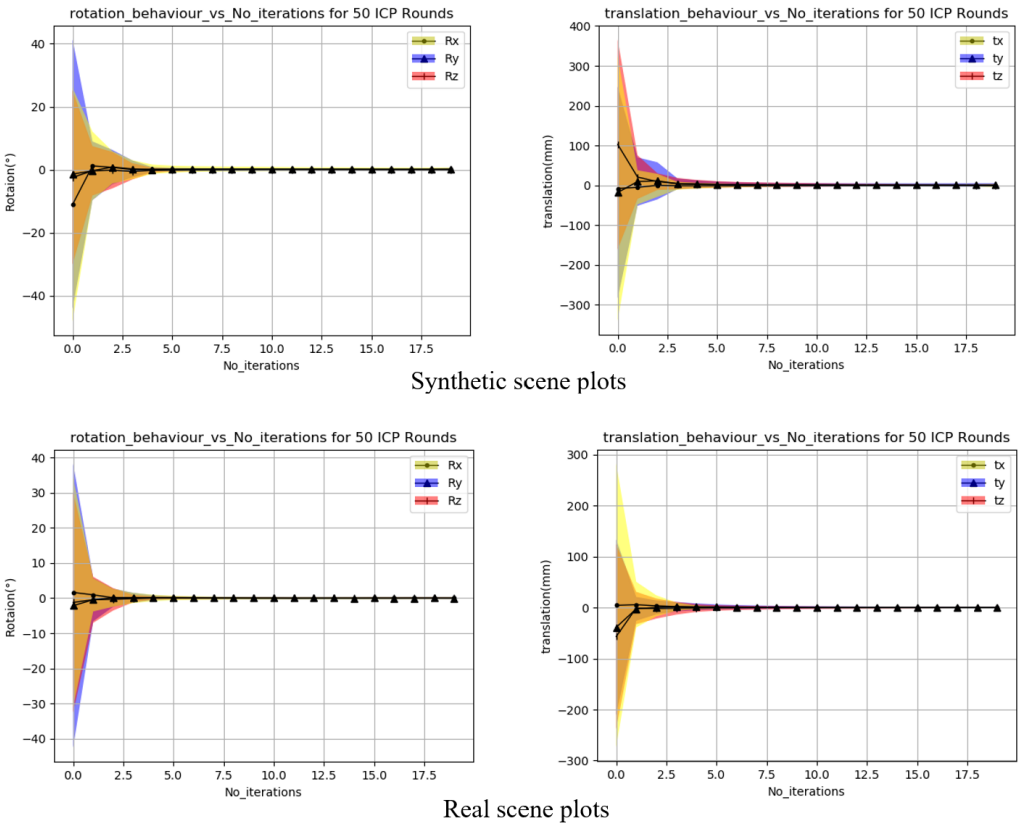

Figure 5.8: Average rotation and translation behaviors of the basic ICP Algorithm. Synthetic scene average rotation behavior (top left) and translation behavior (top right), and real scene average rotation behavior (bottom left) and translation behavior (bottom right) for 20 Iterations per ICP round for 50 ICP rounds.Their corresponding confidence intervals given by the std from the averaged plotted values are represented by the different colored regions of the graphs, as described by their respective legends.

As convergence check of the adopted method, the rotation and translation behaviors relative to the x, y, and z-axes for both synthetic and real scenarios were averaged over the 50 ICP rounds with 20 iterations per round and plotted (Figure 5.8). It can be observed that despite the huge initial variations observed in the confidence intervals of the rotation and translation behaviors for both scenarios, they readily converge to the expected minimum value along all the 3 axes.

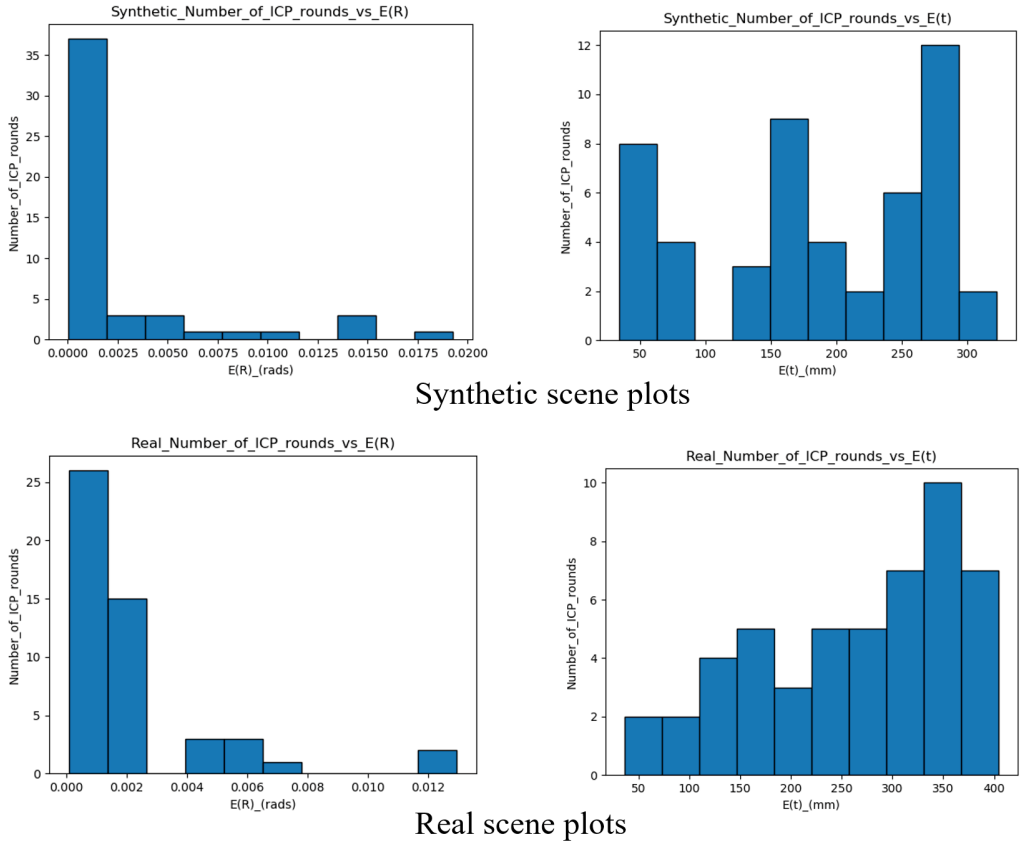

Figure 5.9:
Plots of number of ICP rounds vs rotation and translation errors of the basic ICP algorithm. Synthetic scene Rotation error (top left) and translation error (top right), and real scene Rotation error (bottom left) and translation error (bottom right), for 50 ICP rounds. In each case (synthetic and real scene cases), the rotation and translation errors for all 50 ICP rounds were binned into 10 equal intervals and represented on the horizontal axis. On the vertical axis is the number ICP rounds that fell into various bins.

From the Histograms of the rotation and translation errors w.r.t the 50 ICP rounds (see Figure 5.9), we can observe for the synthetic data scenario that, more than 35 ICP rounds register a rotation error close to $0rads$ (less than $0.0013rads$), while the number of ICP rounds within the minimum translation error interval ($[25mm, 100mm]$) is 14. In the real data case, the rotation behavior is more or less the same as in the synthetic data case, with more than 35 ICP rounds registering a rotation error less than $0.002rads$. However, the number of ICP rounds within the smallest translation error interval ($[25mm, 100mm]$) is less than 5 in the real data case. The misalignment in the registration of most ICP rounds is caused by the translation error in both synthetic and real data scenarios, with more than 15 ICP rounds in the maximum translation error interval $[250mm, 330mm]$ for the synthetic data case, and more than 18 ICP rounds in the

maximum translation error interval $[350mm, 410mm]$ for the real data scenario. Hence, the ICP rounds converge to different values and some rounds do not converge to the global minimum. The fact that not all ICP rounds will converge to the global minimum was, however, expected since the ICP, in general, is subject to the problem of local minima and greatly depends on the goodness of the initial transformation, which was randomly generated. Also, having scene objects of the same geometry and length does not ease the registrations task for ICP, since without any other distinctive features, registration to the wrong scene correspondence or even partial registration to several scene objects are possible (See Figure 5.10).

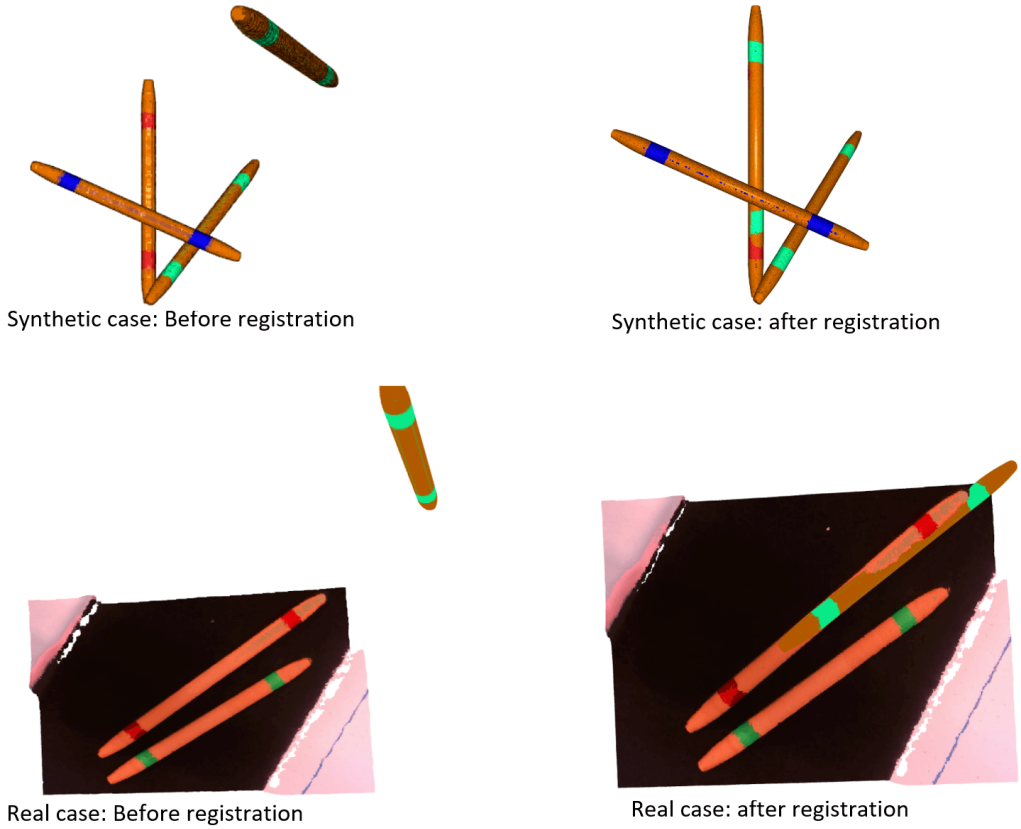

Figure 5.10: ICP wrong registration using a good initial pose for the model object. Heuristic initial poses that position the green stranded object to be registered closer to the scenes as seen in the top and bottom left side of this figure register the model object to the wrong scene object.

Heuristic initial poses that place the green stranded object to be registered closer to the scenes as seen in Figure 5.10 can achieve correct registration with C-ICP (See the registration scenario of the red stranded object in Figure 5.14). This suggests that information on the color further guides the registration process to the correct corresponding scene object.

Colored ICP (C-ICP) Assessment

Following the same baselines as in basic ICP, assessment of C-ICP outputted the following results using the synthetic and real data.

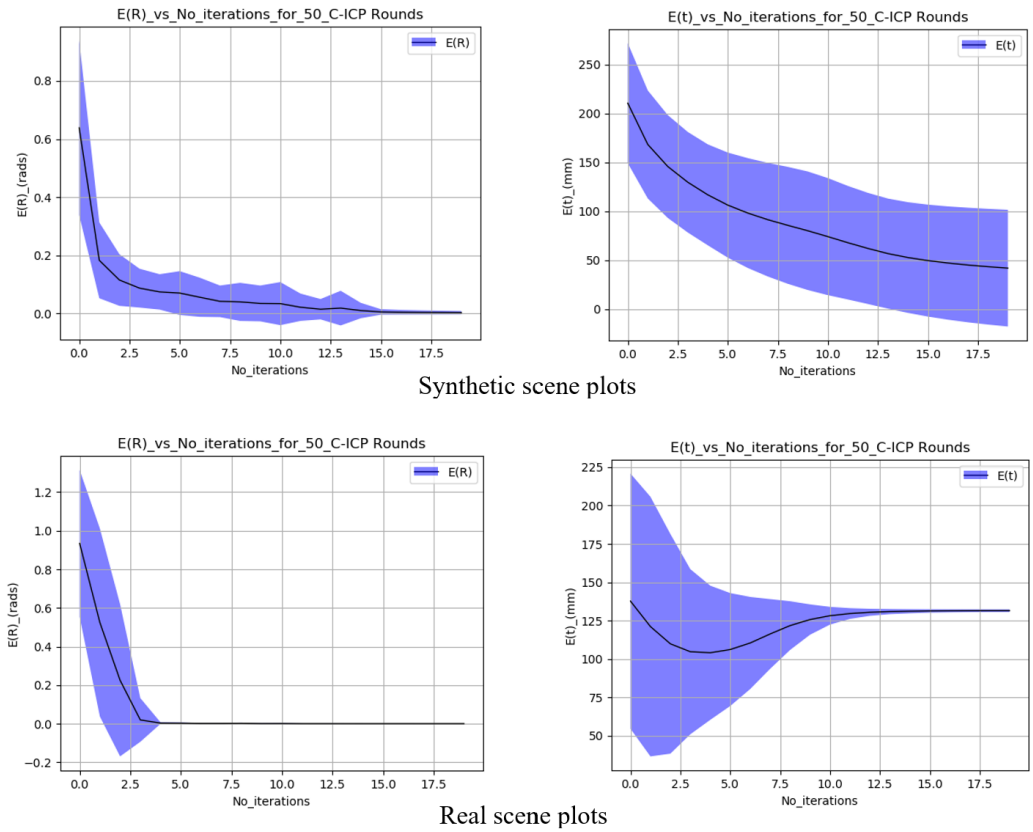

Figure 5.11: Error plots of average rotation and translation vs number of iterations of C-ICP assessment. Synthetic scene average rotation error (top left) and translation error (top right), and real scene average rotation error (bottom left) and translation error (bottom right) for 20 Iterations per C-ICP round for 50 C-ICP rounds. Their confidence intervals are represented by the shaded portions in blue.

After downsampling the model, the synthetic and the real-scene point clouds as mentioned earlier, C-ICP was run 50 times and the characteristic error plots of Figure 5.11 were obtained. Each C-ICP round was composed of a maximum of 20 iterations. The rotation and translation errors were averaged over the 50 C-ICP rounds for the 20 iterations and plotted for the synthetic and real scenes. The confidence intervals were calculated as the standard deviation of the returned average errors, as in (5.1)-(5.2).

It can be observed that the rotation error abruptly decreases during the initial iterations with a steeper gradient in the case of the synthetic scene than in the real scene. However the rotation error curve of the synthetic case, later on, converges slowly to zero towards the final iterations, while the real scene rotation error curve quickly reaches the zero value within the 7^{th} iteration. While the translation error slowly converges from the initial to final iterations and does not reach the expected value of zero for the synthetic scene, in the real scene it slowly decreases to a minimum of 105mm around the 4^{th} iteration, later on rises and settles at a constant value of 108mm toward the final iterations.

The translation error, therefore, accounts for the greater overall registration error for both synthetic and real scenarios with a wider confidence interval compared to that of the rotation error.

As convergence proof of the adopted method, the rotation and translation behaviors relative to the x, y, and z-axes for both synthetic and real scenarios were averaged over the 50 C-ICP rounds with 20 iterations per round and plotted. See Figure 5.12.

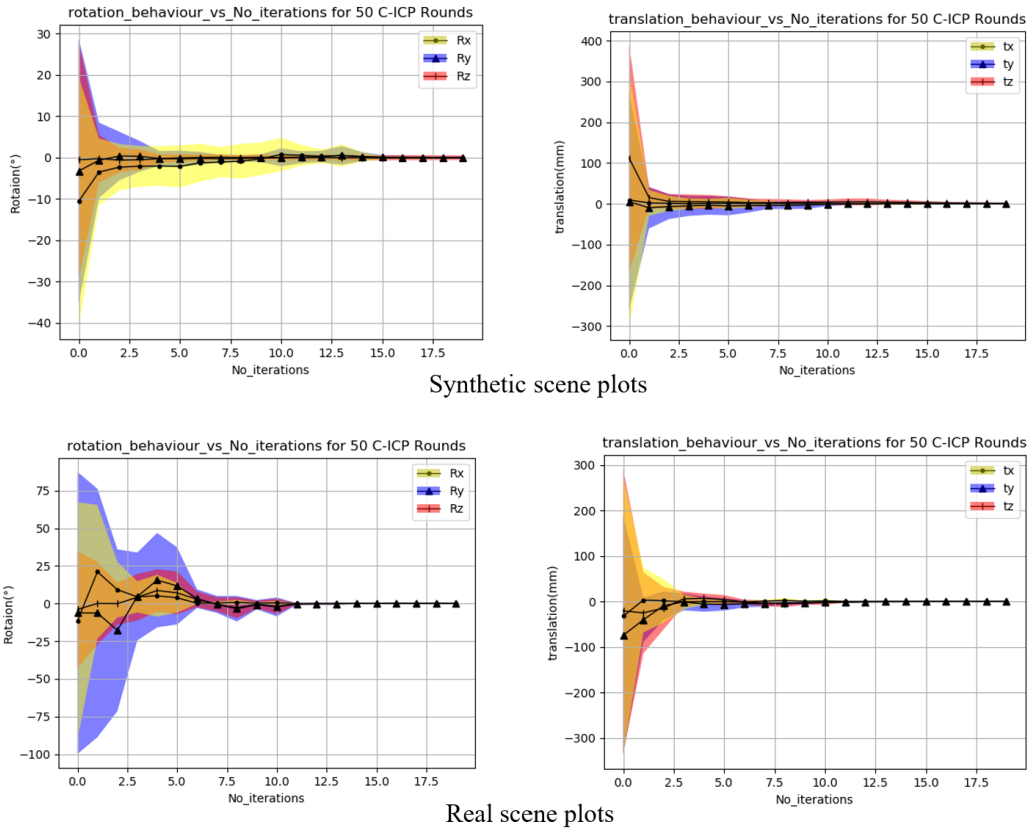

Figure 5.12: Average rotation and translation behaviors of the C-ICP Algorithm. Synthetic scene average rotation behavior (top left) and translation behavior (top right), and real scene average rotation behavior (bottom left) and translation behavior (bottom right) for 20 Iterations per C-ICP round for 50 C-ICP rounds. Their corresponding confidence intervals given by the std from the average values are represented by the different colored regions of the graphs, as described by their respective legends.

It can be observed that despite the huge initial variations observed in the confidence intervals of the rotation and translation behaviors for both scenarios, they nevertheless converge to the expected minimum value along all the 3 axes.

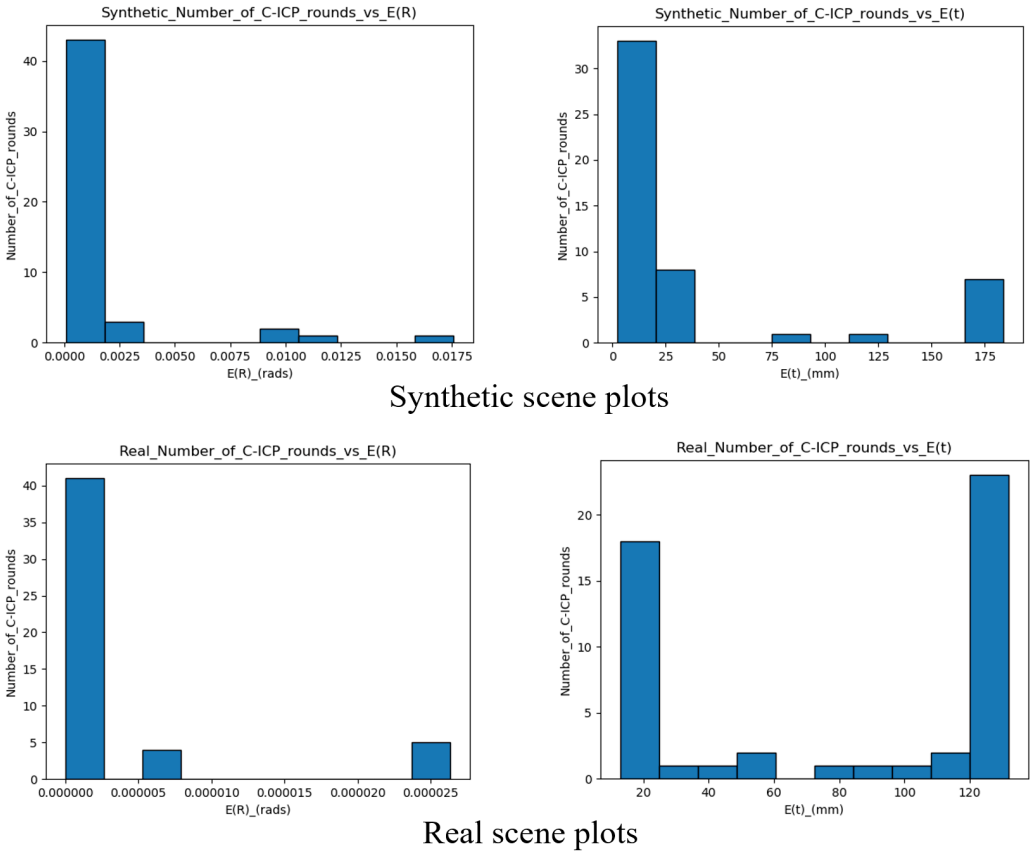

Figure 5.13: Plots of number of C-ICP rounds vs rotation and translation errors of the C-ICP algorithm. Synthetic scene Rotation error (top left) and translation error (top right), and real scene Rotation error (bottom left) and translation error (bottom right), for 50 C-ICP rounds. In each case (synthetic and real scene cases), the rotation and translation errors for all 50 C-ICP rounds were binned into 10 equal intervals and represented on the horizontal axis. On the vertical axis is the number C-ICP rounds that fell into various bins.

From the Histograms of the rotation and translation errors w.r.t the 50 C-ICP rounds (see Figure 5.13), it can be observed that, while the majority of rounds (more than 35 rounds) outputs the correct object registration as seen by their low rotation and translation errors, in the smallest rotation and translation error intervals $[0rads, 0.0013rads]$ and $[0mm, 25mm]$, respectively, a few number of rounds (less than 5 rounds) perform not very good, with rotation and translation errors in the maximum rotation and translation error intervals $[0.015rads, 0.018rads]$ and $[150mm, 188mm]$ respectively.

In the real data case, the behavior is more or less the same, but with almost 23 C-ICP rounds falling withing the maximum rotation and translation error intervals $[0.00002rads, 0.00003rads]$ and $[110mm, 140mm]$, respectively. Here also, the translation error is the main cause of registration misalignment for those C-ICP rounds which did not perform very well.

Obviously, the synthetic scenario outperforms the real scenario due to the noise present in the real world data, such as outliers resulting from the tabletop and variations in the light intensity which affect the colors. The effect of the noise impacts mainly the translations error. Not all C-ICP rounds where expected to converge to the global minimum as C-ICP is merely an extension of

the ICP algorithm. However, some heuristically determined initial transformations proved to give quite good results for both real and synthetic data scenarios, see Figure 5.14.

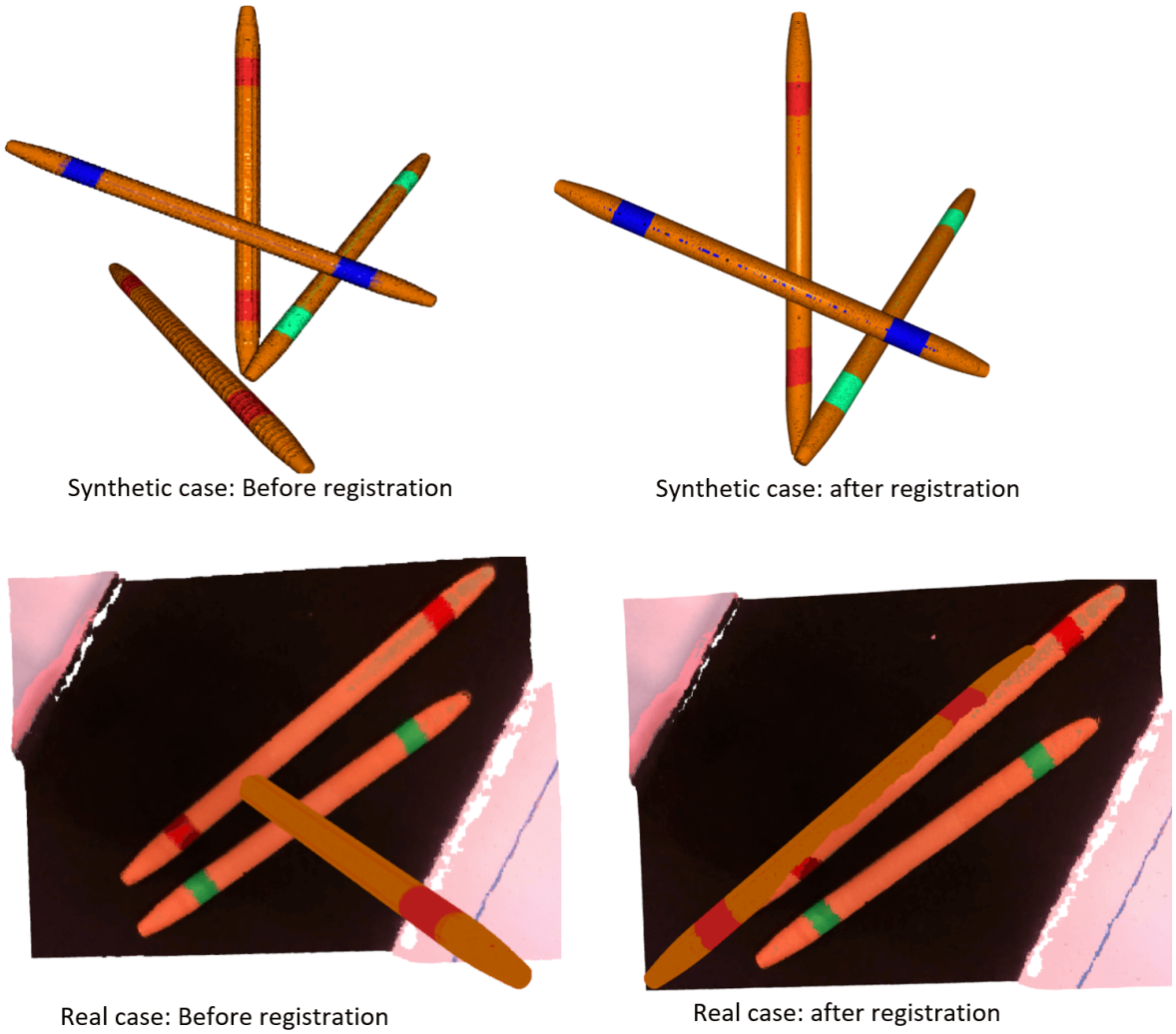

Figure 5.14: C-ICP correct registration using some heuristic initial poses for the model object. Heuristic initial poses that position the red stranded object to be registered closer to the scenes (top and bottom left side) in this figure guarantee correct 3D estimation of the final pose for the synthetic scenario. A slight misalignment in translation error is noticed in the real scenario.

It is important to note here how crucial the color information is, as it directs the object to be registered towards the correct correspondence in the scenes despite the similarity in the geometric shape of all objects present in the scenes.

Comparative Analysis of ICP and C-ICP Algorithms Based on Registration Results

From the behavior of the results presented on ICP and C-ICP, it is clear that the C-ICP algorithm performed better than the ICP algorithm. Nevertheless, a closer view of the gap between the two registration algorithms is presented below for both synthetic and real data scenarios. The confidence intervals were evaluated as the std from mean value as mentioned earlier.

• Synthetic data scenario

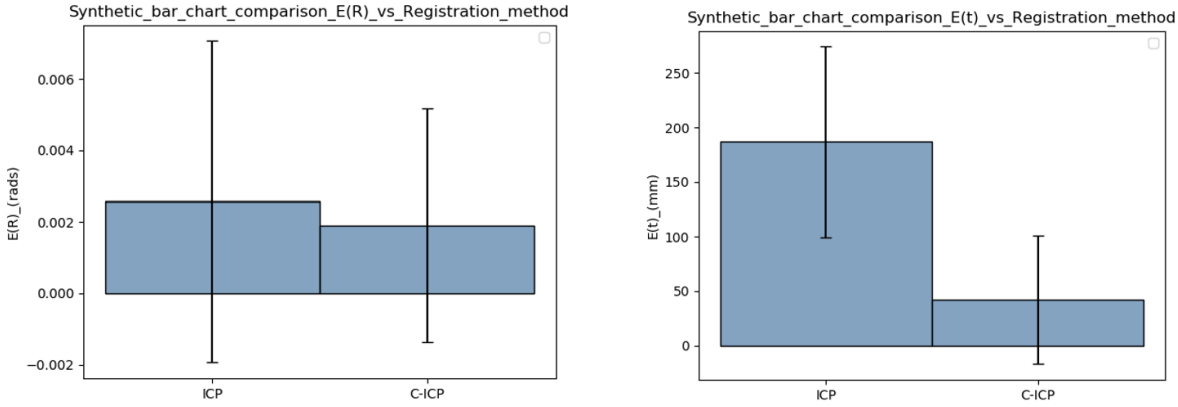

Figure 5.15: Comparative bar chart of ICP and C-ICP using synthetic data. The bar heights represent the average rotation and translation errors, while the vertical lines passing through the center of the bars represent the confidence intervals given by the std from the average errors.

Figure 5.15 reveals that C-ICP performs better on synthetic data than ICP, having both rotation and translation errors less than that obtained in the latter case. In fact, it can be observed that while the gap in rotation error is not significant, the gap in translation error is huge with $E(t)$ for ICP being about 4 times $E(t)$ for C-ICP. The confidence intervals are equally wider for rotation and translation errors for ICP than for C-ICP.

• Real data scenario

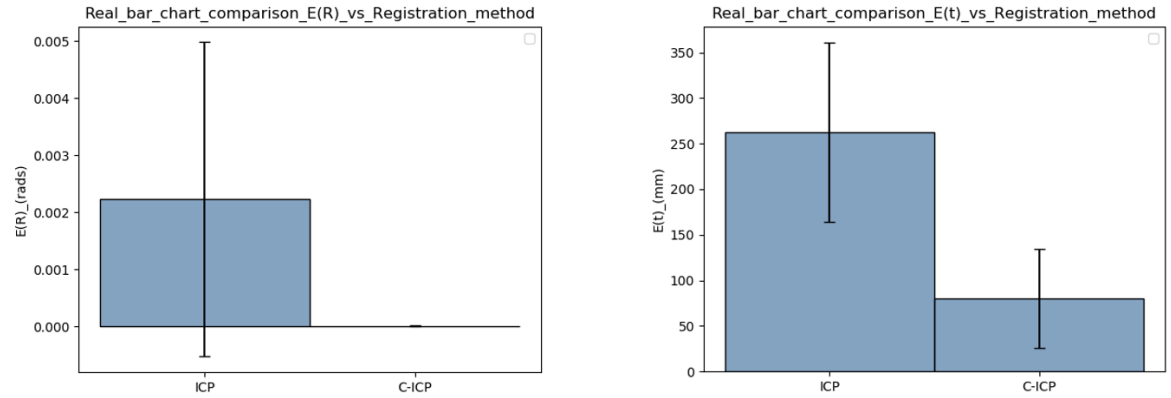

Figure 5.16: Comparative bar chart of ICP and C-ICP using real data. The bar heights represent the average rotation and translation errors, while the vertical lines passing through the center of the bars represent the confidence intervals given by the std from the average errors.

The same trend seen in the synthetic data case is observed using real data. C-ICP outperforms ICP in both rotation and translation errors, registering a rotation error almost equal to 0rads and a translation error almost 3 times less than that of ICP, as observed in the comparative bar chart, see Figure 5.16.

Therefore, independent of the data type, C-ICP gives better pose estimation results than ICP for our specific set-up.

5.2.3 Variants of PPF Evaluation and Graphical Results

Both variants of the point pair feature algorithm were evaluated against multiple synthetic and real datasets. These datasets were generated as described in chapter 4. For all experiments in this section, the model and scene point clouds were uniformly downsampled with voxel sizes of 0.08mm and 0.12mm, respectively, for both synthetic and real-world data datasets. This makes the points in the model and scene point clouds to have a minimum distance of 0.08mm and 0.12mm, respectively, between them. The denser downsampled model point cloud is used in the offline phase to generate the global model description while the dense downsampled scene point cloud is used in the online phase of the registration process. In this way, the computation time is reduced.
The discrete feature vector space was sampled setting the step distance $d_{dist} = 3mm$. The anlges associated to normal orientation were sampled in steps of 12° and the color values for C-PPF were sampled in steps of $u_{color} = 0.001$. This gives room for a variation of 3mm, 12° and 0.001 w.r.t the correct value for distance, normal orientation and color value, respectively.
For every single scene used in the experiments, a sampled fraction of points ranging from $\frac{1}{12}$ to $\frac{1}{4}$ of the corresponding downsampled scene point cloud was used as reference points. After collecting the sampled fraction of reference points, we have to ensure that the normals to these points correspond to the sampling levels. This is achieved by fitting a plane through the four nearest neighbouring points of each point and computing the normal to that plane. This normal is then assigned to the point in consideration.
The rotation and translation errors for all experiments were computed as described in Section 5.2.1, where the rotation error is the angle between the axis of maximum data variation of the model object and the corresponding scene object while, the translation error is the euclidian distance between the centroid of the model object whose pose is to be determined and the centroid of the corresponding scene object. Both error are depicted in Figure 5.5.

5.2.4 PPF and C-PPF Assessment with Single Object Synthetic Scenes

The first set of experiments was carried out using synthetic scenes with a single Mikado stick and the registration errors were plotted. The Mikado stick was assigned different poses in all the five scenes used in the experiments. Figure 5.17 shows some of these scenes and registration scenarios.

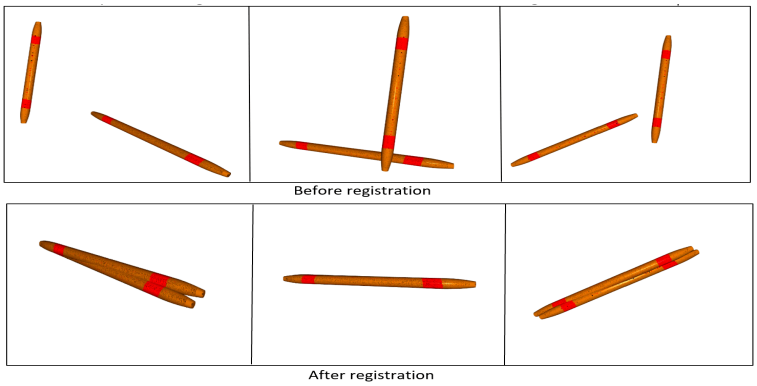

Figure 5.17: Registration scenarios for scenes with single Mikado stick. On the top of this figure are three different scenes before registration. On the bottom are the registration outcomes.

Plots of the registration error can be seen in Figure 5.18.

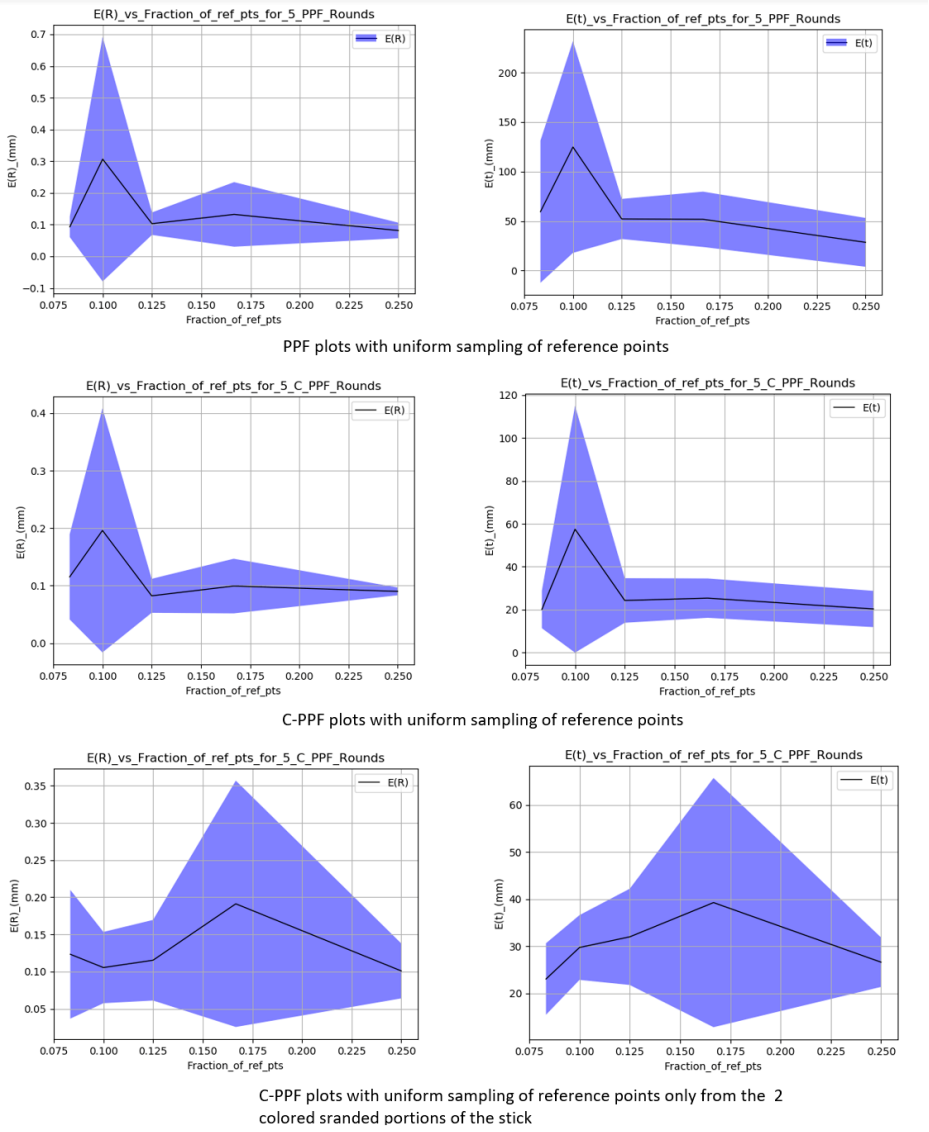

Figure 5.18: PPF and C-PPF average registration error plots for synthetic scenes with only one Mikado stick. Behaviours of the average rotation and translation errors for both PPF and C-PPF with the confidence intervals represented by the blue shaded portions of the plots. The reference points were sampled uniformly from different portions of the synthetic scene object. The top left and right plots represent the rotation and translation errors of the PPF algorithm, for which, the fractions reference points used in the online phase of the registration were sampled uniformly from all portion of the stick. The middle left and right plots represent the average registration (translation and rotation) error of the C-PPF algorithm, for which, the fractions reference points for registration were sampled uniformly from all portion of the stick. In the bottom left and right are the average registration error of the C-PPF algorithm, for which, the fractions of reference points for registration were sampled uniformly only from the two colored stranded portions of the stick.

Figure 5.18 shows the plots of the registrations results obtained after performing the first set of experiments on scenes with single Mikado stick. Three experiments were carried out, and for each of the three experiments, five different scenes were used. For each of the scenes, the algorithms were run five times with five different fractions of points from the downsampled scene point clouds used as reference points. The fraction of scene points varied as r^{-1} with $r \in [4, 12]$. The registration error for all five scenes where averaged and plotted with their confidence intervals given by the standard deviation from the mean error as mentioned earlier in Section 5.2.2 ((5.1) – (5.2)).
From Figure 5.18, it can be noticed that, besides having a slight variation in confidence intervals of the rotation errors of PPF and C-PPF when sampling is done uniformly from all portions of the stick, the overall average registration error is the same. On the other hand, C-PPF shows a lower registration error when sampling of points used for registration is done only from the two red-colored stranded portions of the scene object. Hence in the next set of experiments with C-PPF, the reference points are sampled only from the colored stranded portions of the Mikado sticks.
Despite the random initial behavior of the error plots of Figure 5.18, they turn to follow a decreasing trend towards the end, as the fraction of scene points used as reference points increase. This suggests that using more reference points leads to better registration results. This is because the use of more reference points increases the probability of finding several corresponding point pairs of the scene object that have the same feature vectors as those of the model object.

PPF and C-PPF Assessment with Multiple Objects in the Scenes

The second set of experiments was carried on both synthetic and real scenes with multiple Mikado sticks and the registration errors were plotted.
In the synthetic data case, PPF and C-PPF were evaluated using five different scenes, in which, the red stranded Mikado stick to be registered had different poses. Some of these synthetic scenes and registration outcomes can be seen in Figure 5.19.

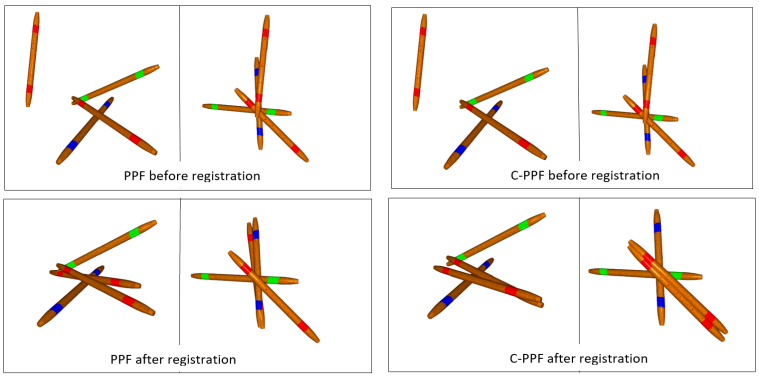

Figure 5.19: PPF and C-PPF registration scenarios for synthetic scenes with multiple Mikado sticks On the top of this figure are two different scenes before registration for PPF and C-PPF. the registration outcomes for both methods are shown under.

After experimenting, the registration error plots of Figure 5.20 were obtained.

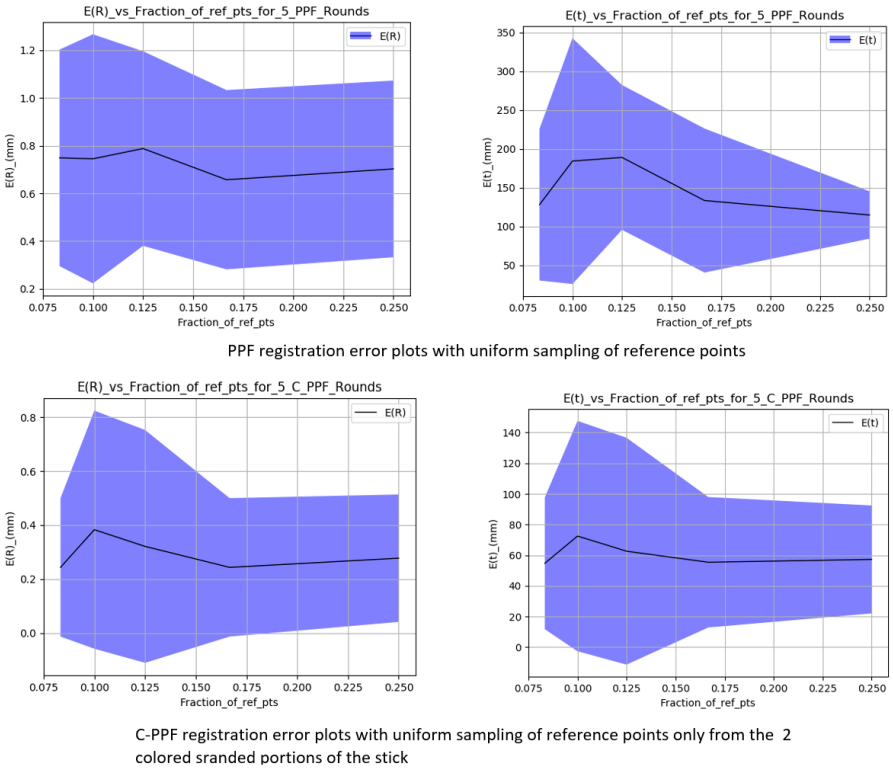

Figure 5.20: PPF and C-PPF average registration error plots for synthetic scenes with multiple Mikado sticks. Behaviors of the average rotation and translation errors for both PPF and C-PPF with the confidence intervals represented by the blue shaded portions of the plots. The top left and right plots represent the average rotation and translation errors of the PPF algorithm, for which, the fractions of reference points used in the online phase of the registration were sampled uniformly from the scene point cloud (i.e, from all the sticks in the scene). In the bottom left and right are the average registration error of the C-PPF algorithm, for which, the fractions reference points for registration were sampled uniformly only from the two red-colored stranded portions of the stick, whose pose is to be determined.

Two Experiments were conducted, and for each of the two experiments, five different scenes were used. For each of the scenes, the algorithms were run five times with increasing fraction of points from the downsampled scene point clouds used as reference points. The fraction of reference points varied exactly like in the fist set of experiments. The registration errors for all five scenes were then averaged and plotted with their confidence intervals given by the standard deviation from the mean error (See Figure 5.20)
All the error plots initially rise a bit to later on fall and remain almost steady towards the end as the fraction of scene points used as reference points increases. Better registration results could be obtained using more reference points for the same reason discussed in the first set of experiments. A clear comparison between both methods on synthetic data can be seen in Figure 5.21.

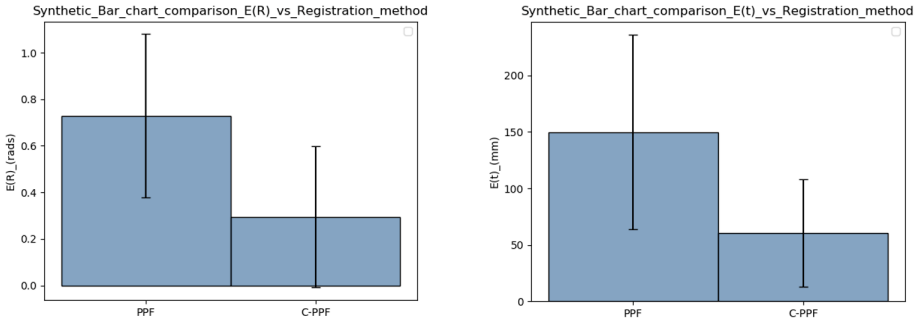

Figure 5.21: Bar chart comparison of PPF and C-PPF on synthetic data. The average rotation and translation errors are represented by the bar heights, while the confidence intervals given by the standard deviation from the average errors are represented by the vertical lines passing through the center of the bars.

It is evident from the bar chart comparison of Figure 5.21 that, C-PPF performs better than PPF in pose estimation of the red Mikado stick in the scene. This could be predicted since PPF samples reference points from all sticks in the scene. Multiple scene point pairs belonging to the wrong scene object, but with feature vectors similar to those of the model object can be retrieved. If the local poses associated with these point pair correspondences obtain a maximum vote in the voting scheme, they can easily lead to wrong pose estimation after pose clustering. C-PPF on the other hand samples reference points only from the right scene object as it is guided by the color information.

The addition of the color values as another parameter in the feature vector therefore plays an important role in orienting the Mikado stick to be registered toward the correct corresponding scene object.

It is equally important to point out that the evaluation of C-PPF sampling reference points only from the 2 stranded portions of the stick gives better results with the single object scenes than with the multiple object scenes. We noticed from experiments that with multiple objects in the scene, the re-computation of the normals by fitting a plane in the neighbourhood of points reduces the accuracy of normal orientation.

In the real data case, PPF and C-PPF were evaluated using only one real scene with two objects. The real scene and registration outcomes can be seen in Figure 5.22.

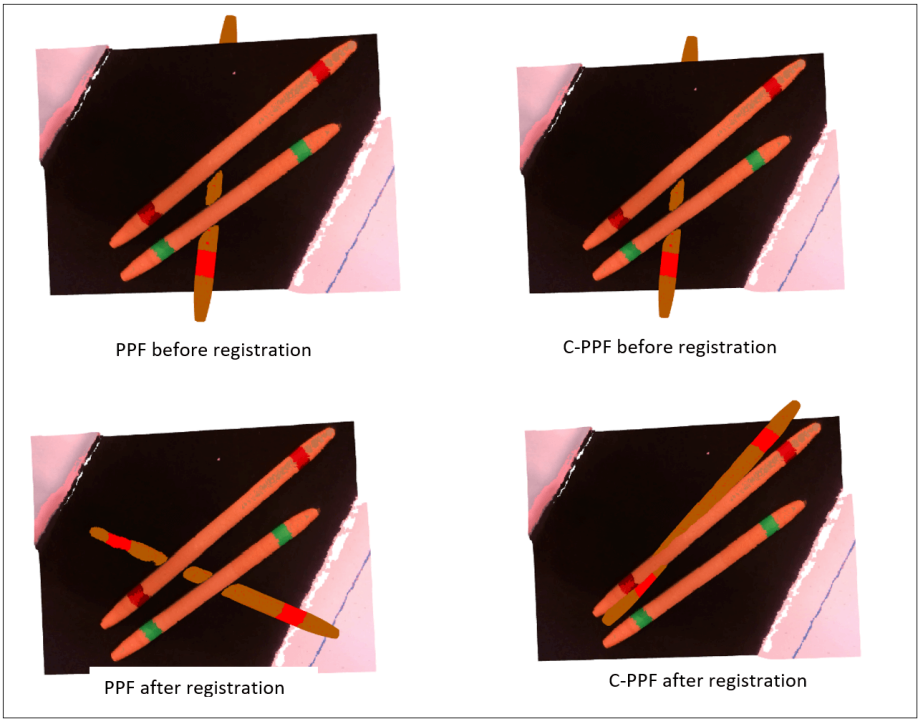

Figure 5.22: PPF and C-PPF registration scenarios for a real scene with two Mikado sticks. On the top of this figure are two real scenes before registration for PPF and C-PPF. the registration outcomes for both methods are shown under.

Registration error plots obtained after evaluating both methods on the real scene are shown in Figure 5.23.

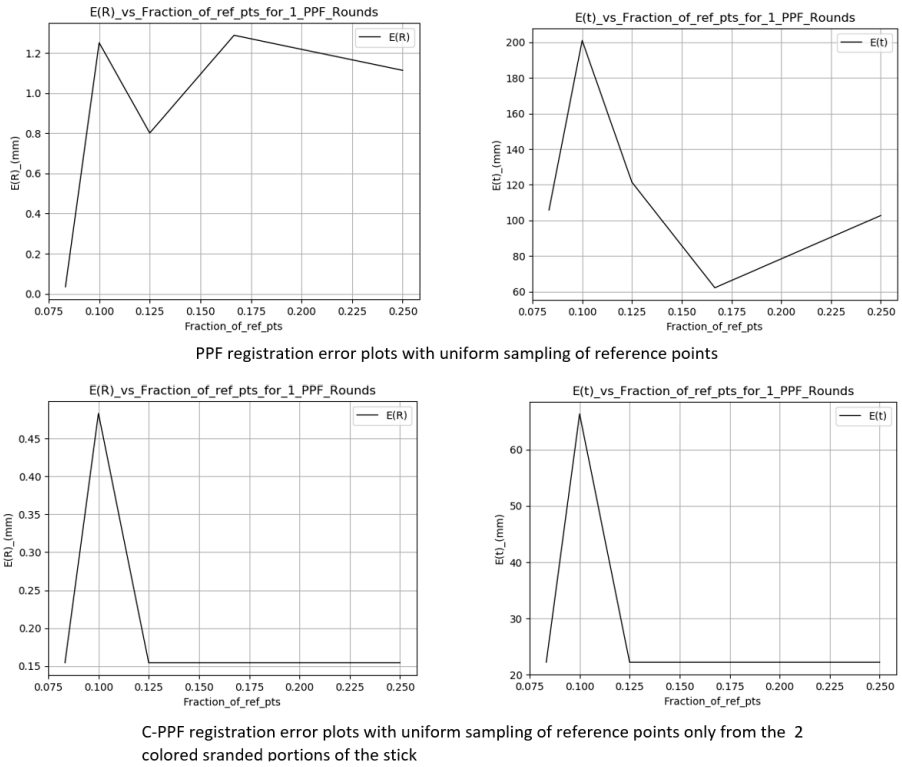

Figure 5.23: PPF and C-PPF average registration error plots for a real scene with two Mikado sticks. The top left and right plots represent the average rotation and translation errors of the PPF algorithm, for which, the fractions of reference points used for registration were sampled uniformly from the scene point cloud (i.e, from both the sticks the black tabletop in the scene). In the bottom left and right are the average registration error of the C-PPF algorithm, for which the fractions reference points used for registration were sampled uniformly only from the two red-coloured stranded portions of the stick to be registered. While both the translation and rotation errors rise and fall continuously throughout the five PPF rounds, C-PPF shows the same trend in both translation and rotation error behavior; it rises during the initial iterations and falls abruptly to maintain a constant value from the 3^{rd} round onwards.

For the experiments using the real scene, the algorithms were run five times with an increasing fraction of points from the downsampled scene point cloud used as reference points. The fraction of reference points again varied like in the previous experiments ($r^{-1},\ r \in [4, 12]$). The registration errors for all five PPF and C-PPF rounds were averaged and plotted in Figure 5.23.
Figure 5.24 compares both approaches on real data.

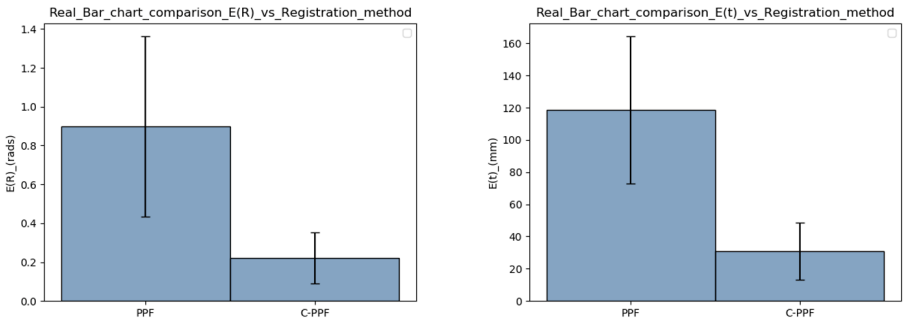

Figure 5.24: Bar chart comparison of PPF and C-PPF on real data. The average rotation and translation errors are represented by the bar heights while the confidence intervals given by the standard deviation from the average errors are represented by the vertical lines passing through the centre of the bars.

Results from Figure 5.24 confirms that, C-PPF performs more than twice better than PPF in pose estimation of the red Mikado stick in the scene. This happens for the same reasons discussed in the synthetic data scenario. Furthermore, the sampling of reference points from the scene without filtering out the points of the tabletop (outlires) further makes things difficult for PPF.

Comparative Results between Variants of ICP and PPF

While PPF searches for point pairs correspondences between model and scene point cloud, ICP searches for point-to-point correspondences between model and scene point clouds. This correspondence search, which unites both approaches makes it worth examining ICP on the same sets of data used for PPF assessment under the same conditions to have an idea of how the former performs w.r.t the latter.

- ICP and C-ICP Assessment with Multiple Objects in the Scenes.

We carried out the third set of experiments on ICP and C-ICP using both synthetic and real scenes with multiple Mikado sticks and the registration errors were plotted. Real and synthetic point clouds were downsampled in the same way as mentioned earlier with the model point cloud being slightly denser than the scene point cloud.
In the synthetic data case, ICP and C-ICP were evaluated using the five different scenes used in the PPF and C-PPF cases. The goal was again that of estimating the pose of the red stranded Mikado stick in the scene. Some of these synthetic scenes and registration outcomes can be seen in Figure 5.25.

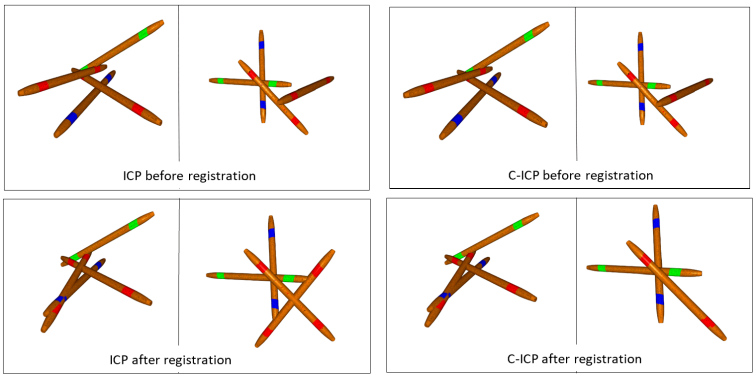

Figure 5.25: ICP and C-ICP registration scenarios for synthetic scenes with multiple Mikado sticks. On the top of this figure are two different scenes before registration for ICP and C-ICP. The registration outcomes for both methods are shown under.

ICP and C-ICP were experimented and for each of the two experiments, five different synthetic scenes were used. For each of the scenes, the algorithms were run five times with random initialization of the model object pose. We used the same seed such that both algorithms could have the same random initialization of the model object in the five scenes. A plot of the overall average errors of both approaches, which equally shows the performance of one w.r.t the other is shown in the comparative bar chart of Figure 5.27. This plot confirms that, despite the fact that both ICP and C-ICP suffer the effects of the random initialization and local minima, C-ICP end up preforming better than ICP.

In the real data case, ICP and C- ICP were evaluated using only one real scene with two objects. The real scene and registration outcomes can be seen in Figure 5.26.

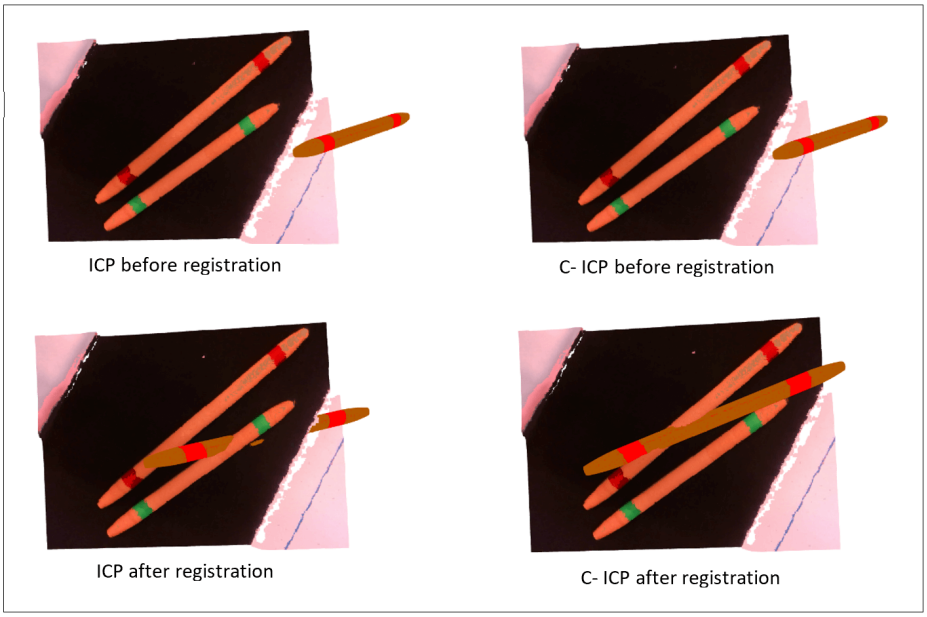

Figure 5.26: ICP and C-ICP registration scenarios for a real scene with two Mikado sticks. On the top of this figure are two real scenes before registration for ICP and C-ICP. The registration outcomes for both methods are shown under.

In these experiments using the real scene, the algorithms were run five times with random initialization of the model object to be registered. We used the same seed such that both algorithms could have the same random initialization of the model object. An overall average registration error plot, which equally shows the comparison between both methods on real data is depicted in the bar chart comparison of all registration methods used in this thesis work, see Figure 5.28.

It is however important to point out that, in the real data case, the effect of outliers (points of the black tabletop) that are not filtered out further affect the registration performace.

- Graphical Comparison of PPF, C-PPF, ICP and C-ICP.

After the detailed in-depth on the individual registration methods, a broad view of the results obtained from all methods using various data types is necessary to clearly understand what method performed best using what data type. To this end, all the registration methods used in this thesis work are hereafter compared plotting their average error on a single charts.

A comparative bar chart of all four methods using synthetic data is depicted in Figure 5.27.

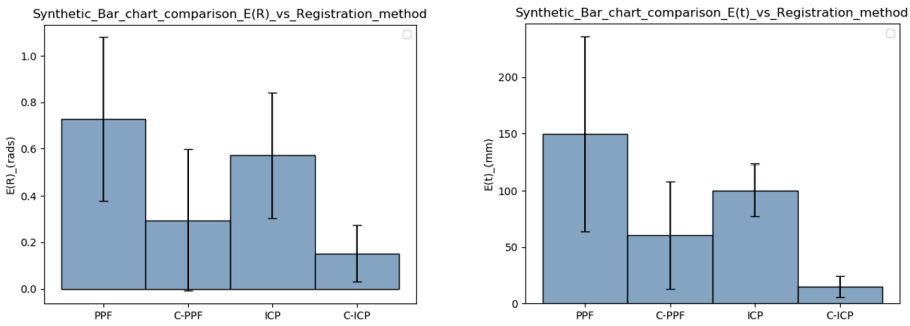

Figure 5.27: Comparative bar chart of PPF, C-PPF, ICP, C-ICP using synthetic data. This figure compares PPF, C-PPF, ICP, and C-ICP on the same downsampled sets of synthetic data. The average rotation and translation errors are represented by the bar heights while the confidence intervals given by the standard deviation from the average errors are represented by the vertical lines passing through the centre of the bars.

From Figure 5.27, which show the average registration error (rotation and translation errors), one can conclude that the overall best approach for determining the pose of Mikado stick of interest in the scene for our set-up is C-ICP. C-ICP performs twice better than all the other approaches on the synthetic data sets used for this evaluation. C-PPF turns to perform better than ICP, and PPF while ICP produces better pose estimation result than PPF.

Figure 5.28 compares all four approaches on a real data pose estimation scenario.

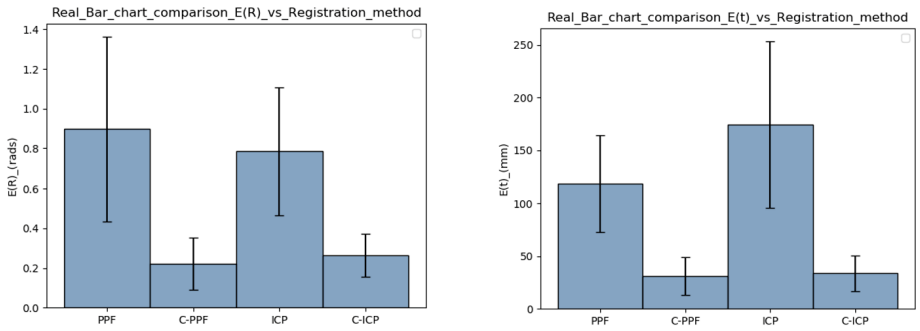

Figure 5.28: Comparative bar chart of PPF, C-PPF, ICP, C-ICP using real data. This figure compares PPF, C-PPF, ICP, and C-ICP on the same downsampled real data-set. The average rotation and translation errors are represented by the bar heights while the confidence intervals are represented by the vertical lines passing through the centre of the bars.

From the bar chart comparison of all the four registration approaches on real data shown in Figure 5.28, the overall average pose estimation error (rotation and translation errors) of the scene object of interest is minimum for C-PPF. C-ICP performs better than PPF and ICP but not as good as C-PPF. It equally turn out that PPF performs better than ICP on the real data used, since the average translation error gap between the two methods is quite huge compared to the very small rotation error gap, See Figure 5.28.
The results obtained after experimentation using real data suggest that in the presence of outliers, the use of normal orientation of the points can better guide the registration process (PPF). Using both the normal orientation of the points and color information provide even better guidance to the correct scene object (C-PPF). C-PPF overcomes the subjective nature of ICP and its variants to the problems local minima and dependency on good initialization. It can thus produce even better registration results in the presence of accurate normal computation and color information.

6 Conclusion

The main objective of this thesis work was to implement and evaluate methods for free-form object registration from 3D point clouds. This goal has been successfully achieved by implementing, assessing, and comparing four different registration algorithms (ICP, C-ICP, PPF, C-PPF) that can compute the pose of the object of interest in the scene. C-ICP and C-PPF were merely upgraded versions of ICP and PPF respectively. Implementation of the former two methods was done to improve pose estimation performances.
All four registration approaches were tested on both synthetic and real-world data. Upon evaluation using both synthetic and real data, C-ICP and C-PPF proved to register the scene object of interest with better accuracy than their respective basic approaches ICP and PPF. C-ICP produced better pose estimation results on synthetic data than C-PPF, but did not perform as well as C-PPF on real data.
Shortcomings encountered by both variants of ICP in certain registration scenarios could be overcome by C-PPF. Nevertheless, the nature of the color information and the normal orientation of the points proved to play a vital role in C-PPF registration performances. This fact leads us to the future work.
As future work, the following improvement could be made to the existing C-PPF registration approach:

- Finding a better way to encode the color information such that the large distance in color between the model object and the real scene object caused by variations in light intensity is minimized.
- Investigating and implementing more accurate methods for computing the normals to the points in the point clouds. This will give better pose estimation results than those obtained till now by plane fitting in the neighborhood points.

Bibliography

[1] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese, "ObjectNet3D: A Large Scale Database for 3D Object Recognition", in Computer Vision – ECCV 2016, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 160–176, ISBN: 978-3-319-46484-8.

[2] M. Vasileiadis, C.-S. Bouganis, and D. Tzovaras, "Multi-person 3D pose estimation from 3D cloud data using 3D convolutional neural networks", Comput. Vis. Image Underst., vol. 185, pp. 12–23, 2019.

[3] M. Gualtieri, A. ten Pas, and R. Platt, "Pick and Place Without Geometric Object Models", 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 7433–7440, 2017.

[4] G. Du, K. Wang, and S. Lian, "Vision-based Robotic Grasping from Object Localization, Pose Estimation, Grasp Detection to Motion Planning: A Review", ArXiv, vol. abs/1905.06658, 2019.

[5] P. J. Sanz, A. Requena, J. M. Inesta, and A. P. Del Pobil, "Grasping the not-so-obvious: vision-based object handling for industrial applications", IEEE Robotics Automation Magazine, vol. 12, no. 3, pp. 44–52, Sep. 2005, ISSN: 1558-223X. DOI: 10.1109/MRA.2005.1511868.

[6] A. Úbeda, B. S. Zapata-Impata, S. T. Puente Méndez, P. Gil, F. A. Candelas-Herías, and F. Torres, A Vision-Driven Collaborative Robotic Grasping System Tele-Operated by Surface Electromyography, 2018-07-20.

[7] J. Yu, K. Weng, G. Liang, and G. Xie, "A vision-based robotic grasping system using deep learning for 3D object recognition and pose estimation", in 2013 IEEE International Conference on Robotics and Biomimetics (ROBIO), Dec. 2013, pp. 1175–1180. DOI: 10.1109/ROBIO.2013.6739623.

[8] M. Ito and M. Shibata, "Non-delayed visual tracking of hand-eye robot for a moving target object", in 2009 ICCAS-SICE, Aug. 2009, pp. 4035–4040.

[9] C.-L. Shih and Y. Lee, "A Simple Robotic Eye-In-Hand Camera Positioning and Alignment Control Method Based on Parallelogram Features", vol. 7, Jun. 2018. [Online]. Available: https://doi.org/10.3390/robotics7020031.

[10] A. J. Ishak and S. N. Mahmood, "Eye in Hand Robot Arm Based Automated Object Grasping System", 2, vol. 7, 2018, pp. 555–566. [Online]. Available: http://pen.ius.edu.ba/index.php/pen/article/view/528/323.

[11] G. Taylor and L. Kleeman, "Visual Perception and Robotic Manipulation", in, 3rd ed. Springer Berlin Heidelberg NewYork, 2006, vol. 26, ch. 2, pp. 11–24.

[12] R. I. Hartley, "Theory and Practice of Projective Rectification", International Journal of Computer Vision, vol. 35, pp. 115–127, Nov. 1999.

[13] Aihua Chen and Bingwei He, "A camera calibration technique based on planar geometry feature", in 2007 14th International Conference on Mechatronics and Machine Vision in Practice, Dec. 2007, pp. 165–169. DOI: 10.1109/MMVIP.2007.4430736.

[14] A. Collet, D. Berenson, S. S. Srinivasa, and D. Ferguson, "Object recognition and full pose registration from a single image for robotic manipulation.", in ICRA, IEEE, Jan. 28, 2010, pp. 48–55. [Online]. Available: http://dblp.uni-trier.de/db/conf/icra/icra2009.html#ColletBSF09.

[15] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition", in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Jun. 2010, pp. 998–1005. DOI: 10.1109/CVPR.2010.5540108.

[16] L. G. Brown, "A Survey of Image Registration Techniques", ACM Computing Surveys, vol. 24, pp. 325–376, 1992.

[17] P. Fua and Y. G. Leclerc, "Registration without correspondences", in Conference on Computer Vision and Pattern Recognition, CVPR 1994, 21-23 June, 1994, Seattle, WA, USA, IEEE, 1994, pp. 121–128. DOI: 10.1109/CVPR.1994.323818. [Online]. Available: https://doi.org/10.1109/CVPR.1994.323818.

[18] D. G. Lowe, "Fitting parameterized three-dimensional models to images", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 5, pp. 441–450, 1991.

[19] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-Squares Fitting of Two 3-D Point Sets", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[20] J. Procházková and D. Martišek, "Notes on Iterative Closest Point Algorithm", Apr. 2018.

[21] O. Faugeras and M. Hebert, "The Representation, Recognition, and Locating of 3-D Objects", The International Journal of Robotics Research, vol. 5, no. 3, pp. 27–52, 1986. DOI: 10.1177/027836498600500302. eprint: https://doi.org/10.1177/027836498600500302. [Online]. Available: https://doi.org/10.1177/027836498600500302.

[22] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes", in Sensor Fusion IV: Control Paradigms and Data Structures, P. S. Schenker, Ed., International Society for Optics and Photonics, vol. 1611, SPIE, 1992, pp. 586–606. DOI: 10.1117/12.57955. [Online]. Available: https://doi.org/10.1117/12.57955.

[23] E. Cuchet, J. Knoplioch, D. Dormont, and C. Marsault, "Registration in Neurosurgery and Neuroradiotherapy Applications", Journal of Image Guided Surgery, vol. 1, no. 4, pp. 198–207, 1995, PMID: 9079446. DOI: 10.3109/10929089509106325. eprint: https://www.tandfonline.com/doi/pdf/10.3109/10929089509106325. [Online]. Available: https://www.tandfonline.com/doi/abs/10.3109/10929089509106325.

[24] Xinhua Zhuang and Yan Huang, "Robust 3-D-3-D pose estimation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 16, no. 8, pp. 818–824, 1994.

[25] R. M. Haralick, H. Joo, C. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose estimation from corresponding point data", IEEE Transactions on Systems, Man, and Cybernetics, vol. 19, no. 6, pp. 1426–1446, 1989.

[26] G. Izatt, H. Dai, and R. Tedrake, "Globally Optimal Object Pose Estimation in Point Clouds with Mixed-Integer Programming", in Robotics Research, N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, Eds., Cham: Springer International Publishing, 2020, pp. 695–710, ISBN: 978-3-030-28619-4.

[27] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm", in Proceedings Third International Conference on 3-D Digital Imaging and Modeling, May 2001, pp. 145–152. DOI: 10.1109/IM.2001.924423.

[28] T. Jost and H. Hugli, "A multi-resolution ICP with heuristic closest point search for fast and robust 3D registration of range images", in Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings., Oct. 2003, pp. 427–433. DOI: 10.1109/IM.2003.1240278.

[29] J. Park, Q.-Y. Zhou, and V. Koltun, "Colored Point Cloud Registration Revisited", in The IEEE International Conference on Computer Vision (ICCV), Oct. 2017, pp. 143–152.

[30] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D Registration Efficiently and Globally Optimally", in The IEEE International Conference on Computer Vision (ICCV), Dec. 2013.

[31] E. Wahl, U. Hillenbrand, and G. Hirzinger, "Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification", in Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings., Oct. 2003, pp. 474–481. DOI: 10.1109/IM.2003.1240284.