

UNIVERSITY OF BIRMINGHAM

ROBOTIC GRASPING STRATEGIES IN UNSTRUCTURED ENVIRONMENTS WITH VISION AND TACTILE SENSING

by

Cristiana Miranda de Farias

A thesis submitted to the University of Birmingham for the degree of
DOCTOR OF PHILOSOPHY

Extreme Robotics Lab
School of Metallurgy and Materials
College of Engineering and Physical Sciences
University of Birmingham
Supervisors: Naresh Marturi and Rustam Stolkin
August 2024

UNIVERSITY OF BIRMINGHAM

University of Birmingham Research Archive

e-theses repository



This unpublished thesis/dissertation is under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence.

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:



Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

Abstract

Robotics has increasingly transitioned from controlled laboratory and industrial settings, where environments and tasks are highly predictable and repetitive, to unstructured environments characterised by variable object shapes, movements, and incomplete sensory information. In such contexts, it is important for robots to adapt and generalise their skills to manage a diverse array of objects, which may be deformed or in motion. This thesis investigates the generalisation of grasping capabilities across various scenarios, including when objects have suffered deformations, when sensory information is missing or in dynamic scenes.

Firstly, this thesis proposes methodologies for extending grasping tasks to different object instances, including those subjected to significant isometric deformations. Initially, grasp transfer algorithms are developed to enable robots to transfer grasps to deformed object instances. Building upon this, the approach is expanded to achieve category-level generalisation, allowing not only the transfer of grasps but also the execution of entire tasks, e.g., mixing bottle contents, applying glue to surfaces, or pressing buttons, across different object categories. Furthermore, the method is further expanded to allow task-aware grasp functions in partially observed object instances by integrating shape estimation techniques with shape uncertainty measurements to ensure grasp stability.

However, given that shape estimation methods can miss surface features when prior information is limited, this thesis proposes to enhance and refine grasp planning across an initially estimated shape through multimodal sensor feedback. For that a novel tactile exploration technique is introduced to jointly enhance shape estimation and grasp stability using a probabilistic metric for force closure.

Finally, to address the dynamic aspects of real-world environments, this work develops grasping capabilities for moving objects. A visual servoing method is introduced, which accounts for challenges such as objects approaching the edges of the workspace and encountering representation singularities, thus enabling effective deployment in unstructured environments.

Acknowledgements

I want to express my gratitude to my supervisors, Prof Rustam Stolkin and Dr Naresh Marturi, for their incredible support and guidance throughout my PhD. Thanks to everyone at the Extreme Robotics Lab for making it such a great place to work—especially Max and Ayush, who I had the pleasure of working closely with.

A special thank you to Dr Yasemin Bekiroglu, who offered guidance during the tough times of the pandemic and beyond, particularly with the grasp exploration work and to Dr Brahim Tamadazte for introducing me to functional maps framework and for all your insights along the way—they. And Luis Figueredo, it was a pleasure collaborating with you on the skill generation work.

Finally, a thank you to my friends and family for always being there. And of course, thanks to Rustam for the great structure of the lab and to the University of Birmingham for providing everything I needed to make this research happen.

Contents

Abstract i

Acknowledgements iii

Contents v

List of Figures xi

List of Tables xix

List of Acronyms xxi

I Motivation 1

1 Introduction 3

1.1 Contributions . . . 6

1.2 Thesis outline . . . 7

1.3 List of publications . . . 8

1.3.1 Published works (peer-reviewed) . . . 8

1.3.2 Unpublished works . . . 9

1.3.3 Other publications . . . 10

2 Related works 11

2.1 Grasp planning . . . 11

2.2 Object understanding . . . 13

2.2.1 Grasping deformable objects . . . 13

2.2.2 Shape correspondence for robotic manipulation . . . 15

2.2.3 Part-based grasping and affordance learning . . . 16

2.2.4 Robot skill acquisition and transfer . . . 17

2.3 Grasping under uncertainty . . . 18
2.3.1 Grasping partially observed objects . . . 19
2.4 Interaction with moving objects . . . 21
2.4.1 Closed loop grasping . . . 22

II Object understanding 25

3 Grasp transfer for deformable objects 27
3.1 Introduction . . . 27
3.2 Problem . . . 29
3.3 Background . . . 31
3.3.1 Shape correspondence . . . 31
3.3.1.1 Coherent Point Drift . . . 32
3.3.1.2 Functional Map correspondence . . . 33
3.3.2 LoCoMo-based grasping . . . 35
3.4 Methodology . . . 37
3.4.1 Method overview . . . 37
3.4.2 Data acquisition . . . 38
3.4.3 Grasp model generation . . . 38
3.4.4 Grasp transfer algorithm . . . 39
3.5 Experiments . . . 41
3.5.1 Object deformation dataset for validations . . . 41
3.5.2 Experimental setup . . . 42
3.5.3 Grasp Transfer Evaluation . . . 42
3.5.4 Discussion . . . 45
3.6 Summary . . . 46

4 One-shot skill transfer 47
4.1 Introduction . . . 47
4.2 Problem overview . . . 48
4.2.1 Conceptual example . . . 49
4.2.1.1 Demonstration phase . . . 49
4.2.1.2 Execution phase . . . 49
4.2.2 Problem Statement . . . 50
4.3 Background . . . 51
4.3.1 User-guided trajectory generation . . . 51
4.3.2 ZoomOut Refinement . . . 52
4.4 Task execution transfer . . . 53
4.4.1 Pipeline . . . 53

4.4.2 Interaction functions . . . 55
4.4.2.1 Robot interaction function (RIF) . . . 56
4.4.2.2 Environment interaction function (EIF) . . . 56
4.4.3 Object selection . . . 57
4.4.4 Task execution in novel scenes . . . 58
4.5 Experiments . . . 60
4.5.1 Robotic setup . . . 60
4.5.2 Virtual experiments . . . 61
4.5.2.1 Single operation tasks . . . 61
4.5.2.2 Multiple-operations tasks . . . 62
4.5.3 Multiple object scenes . . . 65
4.5.4 Real-world transfer . . . 65
4.6 Summary . . . 67

5 Task-aware grasping of partially observed objects 71
5.1 Introduction . . . 71
5.2 Problem statement . . . 72
5.3 Background . . . 72
5.3.1 Gaussian process regression . . . 72
5.3.1.1 Shape estimation . . . 73
5.4 Methodology . . . 74
5.4.1 Pipeline . . . 74
5.4.2 Database construction . . . 75
5.4.3 Task-informed grasping . . . 76
5.4.3.1 Grasping in uncertain regions . . . 77
5.5 Experimental Validations . . . 78
5.5.1 Simulation setup . . . 78
5.5.2 Real-world setup . . . 79
5.5.3 Metrics . . . 80
5.5.4 Simulation Experiments . . . 81
5.5.4.1 Effect of uncertainty function . . . 81
5.5.4.2 Analysis of data partiality . . . 82
5.5.4.3 Comparison of Transfer Error . . . 83
5.5.4.4 Grasp success evaluation . . . 86
5.5.5 Real Robot Experiments . . . 86
5.6 Discussion . . . 90
5.6.1 Comparison With Existing Methods . . . 92
5.7 Summary . . . 93

III Exploratory grasping **95**

6 Tactile-guided grasp refinement **97**
6.1 Introduction . . . 97
6.2 Problem Statement . . . 99
6.3 Background . . . 100
6.3.1 Bayesian optimisation . . . 100
6.3.1.1 Surrogate function . . . 101
6.3.1.2 Acquisition function . . . 102
6.3.2 Grasp stability analysis . . . 103
6.3.2.1 Grasp contact . . . 103
6.3.2.2 Form and force closure . . . 104
6.3.2.3 Grasp quality . . . 105
6.4 Surface exploration and exploitation . . . 106
6.4.1 Method overview . . . 106
6.4.2 Probabilistic grasp stability . . . 108
6.4.3 Target function . . . 109
6.4.4 Input space description . . . 110
6.4.5 Simultaneous grasp exploration and object reconstruction algorithm . . . 111
6.5 Experimental Validations . . . 114
6.5.1 Simulation Setup . . . 114
6.5.2 Shape evolution evaluation . . . 115
6.5.3 Grasp refinement evaluation . . . 116
6.5.4 Grasp stability analysis . . . 120
6.5.5 Baseline comparisons . . . 121
6.6 Summary . . . 123

IV Dynamic environments **125**

7 Grasping moving objects **127**
7.1 Introduction . . . 127
7.2 Problem statement . . . 129
7.3 Background . . . 130
7.3.1 Visual servoing . . . 130
7.3.1.1 Position-based visual servoing (PBVS) . . . 131
7.3.2 Vantage-point tree . . . 132
7.4 Methodology . . . 133
7.4.1 Overview . . . 133
7.4.2 Controller design . . . 135

7.4.2.1 Visual servo velocity control . . . 135
7.4.2.2 Robot controller . . . 136
7.4.2.3 Joint limit avoidance . . . 137
7.4.2.4 Stability analysis . . . 138
7.4.3 Grasp planning and re-ranking . . . 139
7.4.3.1 VP-tree Search for Grasp Selection . . . 139
7.5 Experimental validations . . . 141
7.5.1 Experimental setup . . . 141
7.5.2 Controller convergence analysis . . . 142
7.5.2.1 Stationary objects . . . 142
7.5.2.2 Moving object tracking . . . 143
7.5.2.3 Re-ranking strategy . . . 144
7.5.3 Grasping Arbitrarily Moving Objects . . . 145
7.6 Conclusion . . . 151

V Closing remarks & supplementary materials 155

8 Conclusion and future works 157
8.1 Limitations . . . 158
8.2 Future Directions . . . 159

References 161

A Expanded Tables for PFM 179
A.1 Method Comparison Tables . . . 179
A.2 Expanded Full Results Table . . . 179

B Dual Quaternions 193
B.1 Quaternions . . . 193
B.1.1 Quaternion Products . . . 194
B.2 Dual Numbers . . . 194
B.3 Dual Quaternions . . . 195
B.3.1 Rigid Body Displacement and Twist . . . 195
B.3.2 Frame Transformations and Inertial Twist . . . 196
B.3.3 Pure Dual Quaternion Products . . . 196
B.3.4 Vector Maps . . . 197
B.3.5 Logarithm and Exponential of Dual Quaternions . . . 198

List of Figures

1.1 Different manipulation tasks a human can perform. 3

2.1 Examples of robotics in unstructured environments. From left to right warehouse at Ocado technologies, nuclear industry [133] and dynamic setting with handover [132] and in the bottom row robotic arms used for nuclear decommissioning [141]. 12

3.1 Examples of power cables in different configurations. 27

3.2 Example of a rigid and non-rigid deformation. In (a) all points of the object translate and rotate together, preserving Euclidean distances (d_E) on the grid. In (b) the object shape deforms, yet the geodesic distances (d_g) are preserved. . 29

3.3 LoCoMo grasps planned over specific parts of different objects. From left to right, bear arm, cable plug, plier handle and pipe wheel. The highest ranking grasp has the highest opacity. 36

3.4 Pipeline for non-rigid grasp transfer based on FM. It consists of three steps: grasp model generation (top – green); finding shape correspondences (middle – blue); and, grasp transfer (last – red). 37

3.5 Transformation of a segmented region (in red) of the source object (power cable). (a) Shows the transformation when all the object points are considered and (b) shows the transformation of the points inside the segmentation region only (applying local region transformations). 38

3.6 (a) The object is divided into segments using *k-means* clustering, with each segment represented by a different color. The user selects the segment for robotic grasping. (b) Correspondence of points between two objects is shown, with matching colors denoting identical regions. 40

3.7 Dataset of objects and their deformed versions created for this work. 42

3.8 Simulation results for objects in the dataset. In the first column the point to point map between shapes is shown, where similar colors represent similar regions of the shape. Following, the transformed part from the database shape (in blue) over the target shape. Then grasp candidates and finally screenshots of the grasp execution. 44

4.1 Demonstration of the bottle shaking task, where the robot picks the bottle by the cap, performs a circular motion and subsequently places it back up again. In the last column the generated trajectories are shown. 50

4.2 Pipeline of the proposed method consisting of demonstration and execution stage. Demonstrations are performed kinesthetically, after which the robot is immediately available for task execution. During the demonstration stage, various functions are created based on the placement of the robot's gripper and the motions it performs with the object. When the robot encounters a new scene, these functions can be transferred to similar objects, allowing the robot to extract analogous gripper poses. With a new set of waypoints and the demonstrated trajectory, tasks can be generalized to entirely new scenes. 54

4.3 Illustration of both (i) RIF over the shape of an object, representing the areas where the robot gripper will make contact with the object and (ii) EIF, where the object is in contact with a pre-defined environment region (e.g. the table plane). 56

4.4 Robotic setup used for experiments. 61

4.5 Dataset of objects used in the experiments. (**D1–D8**) user for task demonstrations; and, (**T1–T9**) used for task transfer. 62

4.6 Button press task with the demonstrated object **D4** and the transferred object **T4** refer to Figure 4.5).The top row shows the demonstrated scene, while the bottom row shows the transferred execution.. The first column displays the observed point clouds of the scenes. The second column highlights the function over the object, which, for single-operation tasks, describes the region where the robot interacts with it. For both experiments, the gripper was fully closed throughout. The third column (**c**) shows the demonstrated and imitated pose of the end-effector, while column **(d)** depicts the gripper executing the trajectory. Finally, the last column presents the two paths: the demonstrated trajectory (in the blue dashed line) and the imitated trajectory (in orange) 63

4.7 The robot pushing a can where the demonstrated object is **D2** and the transferred object is **T2**. The top row shows the demonstrated scene, while the bottom row shows the transferred execution. Button press task with the demonstrated object **D4** and the transferred object **T4** refer to Figure 4.5). The first column displays the observed point clouds of the scenes. The second column highlights the function over the object, which, for single-operation tasks, describes the region where the robot interacts with it. For both experiments, the gripper was fully closed throughout. The third column (**c**) shows the demonstrated and imitated pose of the end-effector, while column **(d)** depicts the gripper executing the trajectory. Finally, the last column presents the two paths: the demonstrated trajectory (in the blue dashed line) and the imitated trajectory (in orange) 64

4.8 Repositioning task from "bottle" object **D6** in Figure 4.5 to **T5**. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side. 65

4.9 Drawing task with a cylindrical object (**D8** for the the demonstration and **T9** for the transfer. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side. 66

4.10 Application task with "tube container" objects (**D3** to **T3** in Figure 4.5). In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side. 67

4.11 Wiping task with "whiteboard eraser" objects (**D1** to **T1** in Figure 4.5). In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side. 68

4.12 Painting task with the "paintbrush" in **D7** for the demonstration and **T8 for transfer** in Figure 4.5. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side. 68

4.13 Repositioning task with multiple objects in the scene. Demonstration is done with bottle **D6** and transfer scene is created with object **T7**. In (a) The scene with multiple objects, (b) Colour-coded point-to-point map and functional map with metric R, (c) Obtained end-effector poses, (d) Executed trajectory. The diagonal map between two bottles results in better object selection compared to the bottle and cup. 69

4.14 Real-world bottle shaking task execution by the robot. In the top row the demonstrated functions and trajectories are shown. In the trajectory plot the dashed blue line displays the demonstrated one and the others the imitated trajectory, in orange for the mustard bottle (object **D6** in Figure 4.5), green for the crushed water bottle (**T6**) and in red the small water bottle (**T7**). In the bottom row the transferred task operations are displayed, including RIF and EIF functions and snapshots of the robot executing the task with three different bottles. 70

5.1 Pipeline of the proposed task-informed grasping method. The two modules of the pipeline are shown. A pouring task is shown as an example. In 1, a partial observation of the object is made and the shape is estimated through GPIS. In **2**, the correspondence framework is applied to transfer the function to the reconstructed object. In **3**, grasps are planned in the region defined by the task function, the best grasp is selected, and finally, in **4**, the task-informed grasp is executed. 75

5.2 The individual terms for the task cost function $g(\mathbf{x})$ and the uncertainty function $\tilde{v}(\mathbf{x})$ are defined on the same manifold. Using (5.10), these terms are combined to form $u_{\text{task}}(\mathbf{x})$, which also remains within the same manifold. 78

5.3 Dataset of simulation objects. The reference object is shown in green. Here the **FX** objects are the “Fruits”, the **MX** objects represent various “Mugs”, and the **BX** the “Bottles”. 79

5.4 Objects used for real-world experiments. In addition to the Fruit, Bottle and Mug classes, the Drill object has been added. 80

5.5 Effect of the Gain K_u from (5.10) on the Average Grasp Success Rate and Number of Grasps Found. The overall grasp success rate is shown in red (i.e., the average number of stable grasps after the stability tests – lift, shake and rotate – are performed) , and the average number of grasps found is shown in blue. The experiment was conducted on object **B1** for the functions `Open` and `Pour`. . 82
5.6 Cuts Made Along the X, Y, and Z Axes of Bottle **B1**. The cuts are shown in Figure 5.3, with numbers in circles denoting the cut positions. 83
5.7 Six Different Experiments in Simulation. These experiments demonstrate the transfer, grasp generation (with selected grasps highlighted), and grasp execution. 88
5.8 Transfer Between Two Real-World Objects. Two different functions are defined and demonstrated, showing the transfer from "Drill 1" to a partial version of itself and to a completely different drill. 89
5.9 Results for Transferring Various Task Functions to Real-World Objects. 90
5.10 Function transfer between objects of varying sizes (top row) and shapes (bottom row) within the same class. 90
5.11 Analysis of the impact of shape variation on transfer success. The `Stack` function from the "Pepper" was transferred to three objects: "Banana," "Sponge," and "Wooden Block." 91
5.12 Failure Cases. The top row depicts a transfer between the "Cup" and a take-away coffee cup. The bottom row shows a transfer between "Pepper" and "Banana." An artifact in the reconstruction is visible in the middle image, where the grasps are planned. 91

6.1 To grasp an object with hidden regions, relying on visual cues, humans first try to construct a perceptual model. They then attempt to grasp the object based on this model. Upon contact, if the actual object differs, they refine their perceptual model using tactile feedback, adjusting their fingers until a stable grasp is achieved. 98
6.2 Illustration of contact models. (i) Represents point contact without friction; (ii) depicts the friction cone characteristic of hard finger contacts; and (iii) illustrates soft finger contact. 104
6.3 BO-based exploration pipeline for shape reconstruction and grasp search. . . . 107
6.4 SDH with the each of the fingers highlighted. The offset $\mathbf{x}_{wo}$ and the vector of Euler angles $\mathbf{x}_{we}$ are applied to the approach vector depicted in the image. Image modified from [177]. 110
6.5 Objects used for simulation experiments. 114
6.6 Experimental results for the shape evolution based on GPIS. The figure shows how the object shape estimation evolves as the algorithm suggests new query points. 115

6.7 Evolution of the probability of force closure (PFC) optimum over 30 experiments and example grasps with increasing PFC values from local explorations of every object. 118
6.8 Grasp experiments highlighting the exploratory aspects of our algorithm. The last column shows object models overlaid with a density map of all tactile points found throughout all 30 experiments. Blue areas indicate higher point density, while red areas indicate lower density. 119
6.9 Exploration and test examples (including lifting and shaking) using the full arm. 120
6.10 (Left) Evaluation of the number of stable grasps compared to the baseline; (Right) Evaluation of the number of iterations required to obtain 10 grasps with $Pr^{FC} > 0.8$. 123

7.1 Control diagram illustrating the dual quaternion-based visual servoing process. It shows how the system updates the manipulator's pose online based on the object's visually tracked pose. $\mathcal{G}(\cdot)$ selects the input to the velocity regulator. . 130
7.2 VP-tree when a new query is made and tree structure. 132
7.3 Control diagram of the visual servoing system for grasping dynamic objects. The process is organised into five stages, as marked by circled numbers. Concurrently, the object's position is estimated by tracking attached markers. Initially, a scene-mounted depth camera captures a point cloud of the target object, which the LoCoMo grasp planner uses to generate preliminary grasp options. As the object moves, the desired pose for visual servoing—specifically, the pre-grasp pose provided by the grasp planner—is continuously updated through a re-ranking process. Once visual servoing aligns and the object is successfully grasped, a stability check is performed. 134
7.4 Example of grasp switching during grasp tracking. 135
7.5 Tracking plots for grasping a stationary object with the proposed controller. Here (a) is the 3D trajectory of the end-effector towards the object (grey frame) and some of the the partial poses of the robot motion are represented by the blue frame plotted over its trajectory. (b) shows the position tracking over time, and (c) the rotation tracking. 144
7.6 Error convergence plots for grasping a stationary object. In (a) is the translational error and (b) the rotational error during the grasping process. 145
7.7 Tracking plots for grasping an object following an ellipsoid trajectory. The controller used corresponds to the one presented in (7.16), i.e., without null-space integration. (a) shows the 3D trajectory of the end-effector towards the object, with grey frames representing object motion, and blue ones showing the end-effector pose during various moments of the trajectory. In (b) the position tracking over time, is shown and (c) the rotation tracking. 146

7.8 Error convergence plots for grasping an object following an ellipsoid trajectory using the control law from (7.16), i.e., without null-space integration. (a) The translational error is shown, and (b) the rotational error during the grasping process is shown. Interest regions of the plots are zoomed in. 147
7.9 Tracking plots for grasping an object following an ellipsoid trajectory using the null-space controller from (7.19). (a) depicts the 3D trajectory of the end-effector towards the object, with grey frames representing object motion at some instants and blue frames showing the end-effector pose during various moments of the trajectory. In (b) the position tracking over time is shown, and in (c) the rotation tracking. 151
7.10 Error convergence plots for grasping an object following an ellipsoid trajectory using the null-space control law from (7.19). (a)is the translational error, and (b) the rotational error during the grasping process. 152
7.11 Tracking plots for grasping an object along a diagonal trajectory using the full controller from (7.19), with grasp re-ranking enabled. The moment of the grasp configuration switch is highlighted. (a) shows the 3D trajectory of the end-effector towards the object, with grey frames representing object motion and blue frames showing the end-effector pose at various points along the trajectory. (b) shows the position tracking over time. (c) The rotation tracking. 152
7.12 Error convergence plots for grasping an object along a diagonal trajectory using the full control law from (7.19), with grasp re-ranking enabled. The figure highlights the increase in error when the grasp re-ranking algorithm is activated. (a) depicts the translational error during the grasping process and (b) the rotational error during the grasping process. 153
7.13 Error convergence plots using the full controller for (a) the "Mustard Bottle" object following an ellipsoidal trajectory with object rotation (trajectory 4 from the list in Section 7.5.1), and (b) the "Bleach Cleanser" object following a horizontal trajectory (1 from the list in 7.5.1) with rotation. In both cases, the moments of grasp re-ranking, when switching occurs, are highlighted. 153
7.14 Screenshots depicting the robotic grasping of moving objects. The first three columns show various positions during the trajectory, the fourth column displays the object being grasped, and the final column shows the grasp stability test. . . 154

List of Tables

3.1 Performance evaluation for grasp transfer based on Functional Maps (FM), Coherent Point Drift (CPD), and Iterative Closest Point (ICP). 43

5.1 Analysis of the impact of data partiality on method's success. 84
5.2 Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates. 85
5.3 Overall performance analysis of the proposed approach in simulation. 87

6.1 Experimental evaluation with multiple ojects for both the proposed approach and baseline. 122

7.1 Performance of dynamic grasping with the "Woodblock" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking. 148
7.2 Performance of dynamic grasping with the "Bleach Cleanser" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking. 149
7.3 Performance of dynamic grasping with the "Mustard Bottle" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking. 150

A.1 Pose 1 Open Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates. 180
A.2 Pose 1 Pour Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates. 181
A.3 Pose 2 Open Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates. 182
A.4 Pose 2 Pour Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates. 183

A.5 Bottle Pour Function 184
A.6 Data for the bottle experiments with the pour function 184
A.7 Bottle Open Function 185
A.8 Data for the bottle experiments with the open function 185
A.9 Bottle Handover Function 186
A.10 Data for the bottle experiments with the handover function 186
A.11 Mug Store Function 187
A.12 Data for the mug experiments with the store function 187
A.13 Mug Drink Function 188
A.14 Data for the mug experiments with the drink function 188
A.15 Mug Stack Function 189
A.16 Data for the mug experiments with the stack function 189
A.17 Mug Handover Function 190
A.18 Data for the mug experiments with the handover function 190
A.19 Fruit Handover Function 191
A.20 Data for the fruit experiments with the handover function 191
A.21 Fruit Stack Function 192
A.22 Data for the fruit experiments with the stack function 192

List of Acronyms

BCICP Bijective and Continuous Iterative Closest Point. 35

BO Bayesian Optimisation. 100

CDF Cumulative distribution function. 102

CP Correct part. 86

CPD Coherent Point Drift. 14

DBSCAN density-based spatial clustering of applications with noise. 54

DMP Dynamic Motion Primitives. 17

DO Deformable Objects. 13

DoF Degrees of Freedom. 12

EF end-effector. 54

EI Expected Improvement. 102

EIF Environment interaction functions. 55

FM Functional Maps. 6

GMM Gaussian mixture model. 32

GPIS Gaussian Process Implicit Surfaces. 6

GPR Gaussian process regression. 72

GPs Gaussian process. 72

GWS Grasp Wrench Space. 105

IBVS Image-Based Visual Servoing. 23

ICO Iterative Closest Point. 31

IS Implicit surfaces. 73

LBO Laplace-Beltrami operator. 34

LoCoMo Local Contact Moments. 28

MAE Mean average error. 81

MAX Maximum error. 81

PBVS Position-Based Visual Servoing. 24

PDF Probability density function. 102

PFC Probablility of Force Closure. 107

ProMP Probabilistic Movement Primitives. 17

RIF Robot interaction functions. 55

RMSE Root mean square error. 81

ScLERP Screw linear interpolation. 52

SE Squared exponential. 74

STD Standard deviation of error. 81

TISA Task-Space Imitation Algorithm. 48

TP Thin plate. 74

VP-tree Vantage-Point tree. 130

VS Visual Servoing. 23

WC Wrench Cone. 105

WKS Wave kernel signature. 34

Part I

Motivation

Chapter 1

Introduction

It is morning, and I am about to start my day. I brew some coffee, pour it into a mug, and place it on my desk while checking emails. Without needing to look, I reach for the mug, knowing exactly where it is. Later, I shift the mug to the other side, still aware of its position. As I read papers, I unconsciously pick up a pen, jot down notes, and reflect on the challenges of robot grasping and manipulation.

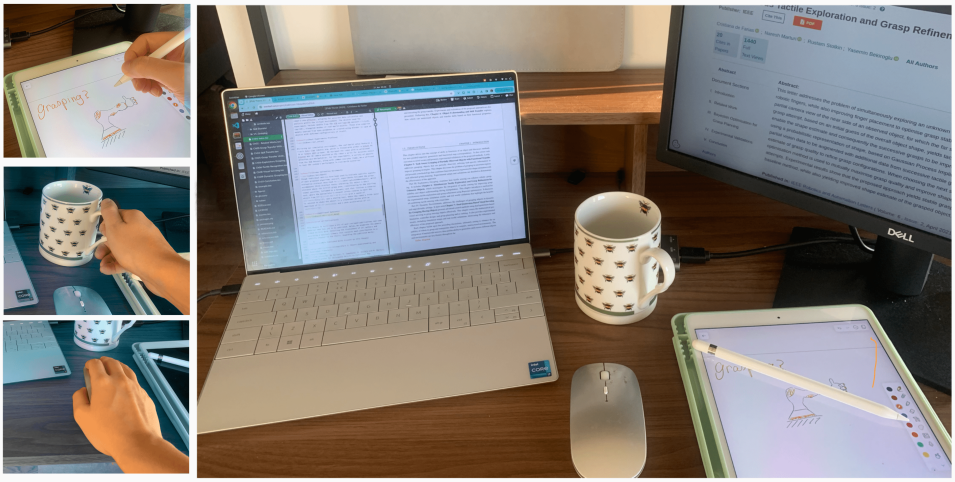


Figure 1.1: Different manipulation tasks a human can perform.

These simple, automatic actions underscore the complexity and dexterity of human motor skills. Indeed, performing these tasks seamlessly, as illustrated in Figure 1.1 highlights the incredible complexity and dexterity of human movement, a feat that robotic systems to this day struggle to replicate.

Consider, for instance, the task of reaching for the mug without looking. For a human, this is an effortless action guided by proprioception: the body's ability to sense its own position and movement. However, for a robot, this requires advanced spatial awareness and precise coordination of its actuators. The robot must rely on its sensors to locate the mug, calculate the

trajectory, and execute the motion without colliding with other objects in the scene. This process is computationally intensive and prone to errors, especially in a dynamic environment where objects might not be in their expected locations. Furthermore, humans can also instinctively map and adapt to the dynamics of the environment. For example, when the mug is placed on the other side of the table, its location is easily known by the human using their mapping and memory skills. A robot, on the other hand, would need continuous updates to its internal model of the environment to track the new position of the mug. This involves integrating data from multiple sensors, maintaining an accurate representation of the environment, and updating its strategies for interaction in real-time. Sensor noise and the lack of such sophisticated and reliable cognitive abilities in robots often result in reduced efficiency and increased likelihood of errors in task execution.

Indeed, while humans perform complex/challenging tasks effortlessly and often without conscious thought, the same actions present technical challenges for robots. The complex nature of these movements, driven by advanced sensory and motor systems, highlights the significant gap that exists in achieving seamless integration of robotic systems in human environments, and that is particularly true for manipulation skills.

According to [134] manipulation can be defined as follows:

Definition 1 (Manipulation)**.** "*Manipulation is an agent's control of its environment through selective contact, whether the agent is a human, animal, or robot*".[1]

This definition is particularly important when considering the challenges of robotic manipulation in *unstructured environments*, which is one of the greater challenges of robotics research today. While picking objects using suction grippers or welding in industrial settings have been common practice in the industry for decades [70] , robots today still cannot sort through a sink filled with dishes. Indeed, the operation of robots in unstructured settings is connected to the notion of variation. Unlike controlled (or structured) environments, the real world can change drastically. For instance, returning to the initial example, while humans can easily adjust to a coffee mug being moved or a pen being in a different spot, robots require extensive recalibration and reprogramming to handle these changes [41], even more if the mug or the pen are replaced by a different versions of these objects.

Definition 2 (Structured environments)**.** "*A task environment that has been engineered to simplify a task is called a structured environment*" [134].

Definition 3 (Unstructured environments)**.** "*Unstructured environments are those that are not engineered for the specific tasks the robot is performing and are often unpredictable and variable*" [134].

[1]There is no consensus of a single definition for manipulation, be it in biological systems or robotics. The survey presented in [134] offers a discussion of different possible definitions before arriving in the one used in this chapter.

Furthermore, while robots can be designed to perform specific tasks well in controlled environments, developing a more general-purpose robot capable of adapting to significant changes in tasks or environments continues to be a major issue. Adaptability, which comes naturally to humans, is something robots are still striving to achieve. This ongoing challenge is aggravated by the fact that grasping an object is not simply about getting fingers around it; the purpose and context of the grasp are crucial. Researchers have long worked to establish theories for stable grasps (e.g., grasps that remain stably in a robot's gripper throughout task execution), but the broader goal of grasping is to enable further manipulation. Understanding why a grasp is executed is as important as the mechanics of the grasp itself [23].

To address these challenges, robots that can generalise and transfer their knowledge across different scenarios are needed. This means developing robots that can adapt to variations in embodiment, object types, and environments:

i *Embodiment*: Robots should be able to transfer skills learned on one platform to another, accounting for differences in sensors, actuators, and physical form.

ii *Object*: Robots need to generalise their manipulation skills to different objects within the same category and even across categories. A robot trained to grasp a ceramic mug should be able to apply similar techniques to a plastic cup or even a differently shaped container.

iii *Environment*: The ability to operate effectively in various environments is crucial. Robots should handle different lighting conditions, the presence of distractors, and dynamic elements. A robot capable of performing tasks in a controlled lab should also function in a cluttered household or an outdoor setting.

Thus, in order for robots to operate in unstructured environments, a twofold challenge must be solved. Firstly, **enhancing robots' understanding of objects by leveraging semantic and physical insights** to improve grasping in various settings, from households to industrial landscapes. This involves recognising and manipulating objects that are unfamiliar or in disarray, reflecting a deeper comprehension of their form and function. Secondly, **addressing the dynamic nature of these environments**, where objects may move unpredictably, and conditions can change rapidly. This requires approaches to planning and control that allow robots to adjust their actions in real-time to achieve successful grasping outcomes, even when faced with uncertainty and movement. The current limitations of robotic manipulation skills in such settings are a significant barrier to their wider use in both commercial and domestic applications. Despite progress from traditional engineering methods, robots still struggle with everyday tasks, highlighting the necessity for intelligent and machine learning-based approaches. These new methods, still in early development, aim to provide robots with the adaptability and flexibility that humans naturally possess, enabling them to operate effectively in dynamic and unpredictable environments [41].

1.1 Contributions

This thesis focuses on the area of robot grasping and manipulation, specifically on how robots can manage to reason about aspects of an unstructured world even when only partial observations are available. Following are the main challenges addressed in this thesis:

1. **Grasp and skill transfer for meaningful object interaction**: This implies understanding and leveraging task and object-centric grasp goals for deployment in new scenes. Endowing robots with such skill transfer capabilities enable robots to grasp objects more effectively and emulate human-like skills for task execution. This involves category-level generalisation, where template matching approaches enable transferring information defined with respect to one specific object to others within the same category. Particularly, a framework based on Functional Maps (FM) correspondence is designed to expand template matching algorithms to objects which have suffered large deformations or objects within a same category. The developed algorithms leverage the fact that functions defined over the surface of an object can encode not only grasp locations but also the cost of performing tasks and the relationships between the object, the robot, and its environment. This capability enables grasp transfer, skill transfer, and even the handling of partially observed objects by using the uncertainty function associated with the object.

2. **Exploratory grasping through integration of tactile sensing**: Addresses the challenge of exploring an unknown object's shape through touch when only partial vision information is available. Particularly, the proposed method uses Gaussian Process Implicit Surfaces (GPIS) to model shapes that get iteratively refined as tactile new data is acquired. The robot is guided to touch object by employing a bi-objective optimisation strategy based on Bayesian optimisation that simultaneously maximises a probabilistic measure of grasp quality and refines the object's shape model. This ensures not only the understanding of an object shape is improved but that an initial grasp plan is refined through the exploration, leading to more stability during grasp execution.

3. **Grasping in dynamic environments**: Addresses the challenge of following/tracking and grasping moving objects in the robot's task-space. To this extent, a new method based on visual servoing using dual quaternions has been proposed and integrated with an online grasp-ranking scheme and reactive planning. The online visual feedback allows robots to adjust their movements and interactions continuously, which are deployed through reactive planning strategies. To ensure feasible reactive motion in unforeseen dynamic and constrained workspace, the proposed approach integrates singularity-handling.

By focusing on the integration of multimodal sensory data and the ability to generalise skills across different objects and environments, this research aims to contribute in developing robots

capable of effective and adaptive operation in the complex, dynamic, and often unpredictable scenarios that define unstructured environments.

1.2 Thesis outline

This thesis is structured into five main parts as detailed in the following.

PART I: Motivation

Provides the foundational background and context for the research. It begins with this chapter, **Chapter 1: Introduction**, which outlines the challenges and significance of robotic grasping in unstructured environments. This chapter discusses the objectives and scope of the thesis, highlighting the importance of developing advanced grasping strategies that integrate vision and tactile sensing. Additionally, it presents an overview of the methodology and the structure of the thesis. In **Chapter 2: Related works** a literature review covering the main aspects of robotic grasping and manipulation, and the current state of the art in related themes, such as one-shot skill learning, grasp affordances, visual servoing, etc. are covered.

PART II: Object understanding and functional transfer

focuses on how robots can understand objects and transfer skills between instances of the same object. It starts with **Chapter 3: Grasp transfer for deformable Objects**, which discusses methods for transferring grasp information between deformable objects using functional map correspondence. It introduces the concept of functional maps and their application in grasp transfer, detailing the methodology for grasp model generation, finding shape correspondences, and executing the grasp transfer. **Chapter 4: One-shot skill transfer** explores how robots can understand objects and transfer skills based on functions describing their poses and trajectories obtained from a single kinesthetic demonstration. This chapter also covers task execution in novel scenes and presents experimental both simulation and real world validations of the proposed methods. Lastly, **Chapter 5: Task-aware grasping of partially observed objects with functional transfer** addresses grasping objects that are partially observed, using task-specific information to improve grasping strategies. This chapter discusses the problem of grasping in uncertain regions and presents a methodology that combines Gaussian process regression for shape estimation with task-informed grasp planning. Experimental setups and validations are detailed to demonstrate the effectiveness of the approach.

PART III: Exploratory grasping

Examines how tactile sensing can enhance robotic grasping, particularly for unknown partially observed objects and in setting with high uncertainty. It includes **Chapter 6: Simultaneous tactile exploration and grasp refinement for unknown objects**, which

investigates the integration of tactile sensing for improving grasp stability and object understanding during manipulation.

Part IV: Dynamic environments Addresses the challenges of grasping objects in dynamic and unpredictable settings. It includes **Chapter 7: Dual quaternion-based visual servoing for grasping moving objects** which presents advanced control methods using dual quaternion-based visual servoing to grasp moving objects effectively. This chapter covers controller design and grasp planning and re-ranking. It also provides experimental results, including simulation setups and real-world validations, showcasing the robustness and efficiency of the proposed approach.

Part V: Closing remarks & supplementary materials Finally, **Chapter 8: Conclusion** presents the main conclusions drawn from this works and future directions and plans to fully integrate each part of this thesis.

Additional results and mathematical details on dual quaternion algebra are presented in appendix.

1.3 List of publications

This thesis is built upon a mix of peer-reviewed publications and some material that has yet to be published. Below, we enumerate the papers linked to this work, detailing the involvement of all authors and providing a succinct summary that connects each contribution to a specific chapter in the thesis. Regarding the published works:

1.3.1 Published works (peer-reviewed)

Part II: Object understanding and functional transfer

1. **C. de Farias**, B. Tamadazte, R. Stolkin and N. Marturi. Grasp Transfer for Deformable Objects by Functional Map Correspondence . In 2022 IEEE International Conference on Robotics and Automation (ICRA), 2022. doi: 10.1109/ICRA46639.2022.9812141.

2. **C. de Farias**, B. Tamadazte, R. Stolkin and N. Marturi. Grasp Transfer for Deformable Objects by Functional Map Correspondence. In 2nd Workshop on Representing and Manipulating Deformable Objects 2022., ICRA 2022.

   - CF was responsible for formalising the theory, the transfer algorithms, code implementation and results. BT, RS and NM provided advice on the theory. Chapter 3 describes the mathematical formulation and the experimental validations of grasp transfer algorithm.

3. **C. de Farias,** B. Tamadazte, M. Adjigble, R. Stolkin and N. Marturi, "Task-Informed Grasping of Partially Observed Objects," in IEEE Robotics and Automation Letters, vol. 9, no. 10, pp. 8394-8401, Oct. 2024, doi: 10.1109/LRA.2024.3445633.***(also in accepted for ICRA 2025)***

   - CF was responsible for formalising the theory, the transfer algorithms, code implementation and results. BT, MA, RS and NM provided advice on the theory. Chapter 5 describes the mathematical formulation and the experiments for grasping partially observed objects.

Part III: Exploratory grasping

4. **C. de Farias**, N. Marturi, R. Stolkin and Y. Bekiroglu. Simultaneous Tactile Exploration and Grasp Refinement for Unknown Objects. In IEEE Robotics and Automation Letters, 2021. doi: 10.1109/LRA.2021.3063074 ***(also accepted for ICRA)***

5. **C. de Farias**, N. Marturi, R. Stolkin and Y. Bekiroglu. Simultaneous Tactile Exploration and Grasp Refinement In Bridging the Gap between Data-driven and Analytical Physics-based Grasping and Manipulation II, ICRA 2021. ***(Best Short Talk Award)***

   - CF was responsible for formalising the theory, the exploration algorithms, code implementation and results. NM, RS, YB provided advice on the theory. YB also provided insights on GPIS and BO, including theory and implementation. 6 presents the algorithm, mathematical details and experiments results on the method for simultaneous grasping and shape exploration.

Part IV: Dynamic environments

6. **C. de Farias,** M. Adjigble, B. Tamadazte, R. Stolkin and N. Marturi. Dual Quaternion-Based Visual Servoing for Grasping Moving Objects. In 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), 2021. doi: 10.1109/CASE49439.2021.9551631

   - CF was responsible for formalising the theory, designing the controller, code implementation and results. MA, BT, RS and NM provided advice on the theory. MA has also provided advice on the use of VP-trees. Chapter 7

1.3.2 Unpublished works

- **C. de Farias,** B. Tamadazte, R. Stolkin, L. F. C. Figueredo and N. Marturi. Geometrically-Aware One-Shot Skill Transfer of Category-Level Objects (Under Preparation)

- CF was responsible for formalising the theory, designing the controller, code implementation and results. LFCF, MA, BT, RS and NM provided advice on the theory with NM and MA also assisting with the implementation on the real system. LFCF provided the interpolation algorithm. Chapter 4 presents the one-shot skill transfer method, its mathematical formulation and experimental analysis.

- Part IV: A. Aggarwal, **C. de Farias**, R. Stolkin, N. Marturi Learning-based Direct Visual Servoing with 3D Edges. (Under preparation RAL)

  - CF provided advice on controller design.

1.3.3 Other publications

- M. Adjigble, **C. de Farias**, R. Stolkin and N. Marturi. SpectGRASP: Robotic Grasping by Spectral Correlation. In EEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021. doi: 10.1109/IROS51168.2021.9636235.

  - CF provided assistance with generating experimental results.

- Adjigble, M., Tamadazte, B., **de Farias, C.**, Stolkin, R., Marturi, N. "3d spectral domain registration-based visual servoing." 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023.

  - CF provided assistance with generating experimental results.

- Adjigble, M., Tamadazte, B., **de Farias, C.**, Stolkin, R., Marturi, N. "Asservissement visuel 3D direct dans le domaine spectral." ORASIS 2023 (journées francophones des jeunes chercheurs en vision par ordinateur). 2023.

  - CF provided assistance with generating experimental results.

- Leblanc, L., Vialle, R., **de Farias, C.**, Saghbini, E., Marturi, N., and Tamadazte, B. "Pedicle Drilling Planning Transfer for Spine Surgery Using Functional Map Correspondences." 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2024). 2024.

  - CF provided insights on the FM formulation.

Chapter 2

Related works

The field of robotics research has evolved significantly since its inception. While earlier generations of robotics primarily focused on industrial applications within highly structured factory environments, contemporary research emphasises unstructured settings. In these contexts, robots must operate in uncertain and dynamic conditions, such as those encountered within modern factory floors, households, warehouses, and other human-inhabited environments [209]. Beyond those, robots are also increasingly being deployed in highly unstructured and hazardous conditions such as those in the nuclear industry [133], space exploration [149] and underwater settings [10]. Furthermore, more diverse types of industry, e.g., agricultural, also presents a challenge for robot grasping and manipulations, where robots need to perform pick and place tasks of highly irregular, often soft, objects in dynamic settings [51, 39]. Figure 2.1 shows some examples of robotic applications in those settings.

To address each of the objectives outlined in Section 1.1, this work deals with understanding object shapes, especially through dense shape correspondence, enabling the generalisation of skills such as grasping and task execution across category-level objects and exploring object shapes, especially through the integration of tactile sensing with uncertainty-aware and probabilistic methods for grasp optimisation. Additionally, while aiming to enhance robotic applications in unstructured settings, this thesis deals with the complexities of grasping moving objects within dynamic scenes. In this chapter, an overview of the relevant literature for each part is presented. Current advancements and the state-of-the-art in each area are analysed, providing a comprehensive understanding of the field.

2.1 Grasp planning

Grasp planning (or grasp synthesis) refers to the problem of determining the optimal configuration for a robot's end-effector in relation to an object, with the aim of achieving stable grasping of the object and/or performing some task [213]. This problem is central in robotic manipulation, as it ensures that a robot can effectively and securely interact with objects in its environment.



Figure 2.1: Examples of robotics in unstructured environments. From left to right warehouse at Ocado technologies, nuclear industry [133] and dynamic setting with handover [132] and in the bottom row robotic arms used for nuclear decommissioning [141].

Since the early days of robotic manipulation, various methods have been developed to address grasp synthesis, and numerous surveys have reviewed different aspects of these approaches [25, 21, 142, 50, 213]. Recently, [50] surveyed deep learning-based grasp generation models for point clouds, and [142] systematically analysed over eighty learning-based papers on deep learning based approaches for 6 Degrees of Freedom (DoF) grasp synthesis.

Early grasp generation algorithms primarily focused on the mechanical principles of grasping [213, 21, 142]. These methods were usually concerned with balancing contact forces, external forces and torques to achieve stability. Such approaches typically involved detailed calculations to ensure that the grasp was physically stable and capable of withstanding external disturbances. By focusing on the fundamental physics of grasping, these early methods laid the groundwork for more advanced techniques.

In recent years, the focus of grasp synthesis has shifted towards incorporating perceptual data into the process. Modern approaches often leverage vision and tactile sensing to compute grasps, allowing robots to adapt to a wide range of objects and environments dynamically. Vision-based methods use cameras and image processing algorithms to identify suitable grasp points on an object, enabling the robot to plan its actions based on visual input [213, 50]. Tactile sensing methods, on the other hand, rely on sensors embedded in the robot's end-effector to gather information about the object's surface properties and adjust the grasp accordingly [58,

129, 47, 146]. These perceptual methods enhance the robot's ability to interact with objects in a more flexible and adaptive manner.

The methods employed in grasp synthesis can be broadly categorized into analytical and data-driven approaches. Though more focus has been given to data-driven grasping methodologies in recent years, [126, 50, 142, 127, 128], some analytical approaches have also been proposed [3, 6, 103, 104], with [3] taking advantage of correlations between fingertip surfaces and point clouds, [6] transforming the problem of grasp searching to the spectral domain and searching for correlations in the space of spherical harmonics, and [103, 104] using probability densities over point clouds to generalise to novel objects.

Aside from methods focused solely on generating stable grasps, recent research has increasingly emphasised on object-centric grasping representations for performing tasks [186, 138, 180], or affordance learning (i.e., identifying tasks which can be performed within an object) [179, 125, 84, 14] and objects characteristics [201, 49]. These approaches involve a deeper understanding of the object for informed grasp decisions. For example, when grasping a knife, a robot might choose to grasp the handle for cutting tasks, while for a handover, where safety is paramount, it might opt to grasp the blade. This shift towards object-aware representations allows for more contextually appropriate and safe interactions with various objects.

Additionally, some studies address practical aspects of grasping, such as handling partial information about an object [58] and managing uncertainty [103, 104]. These factors are critical in real-world applications, where a robot may not have complete information about the objects it needs to manipulate. Techniques that account for these uncertainties can significantly improve the robustness and reliability of robotic grasping in diverse environments.

In this thesis, grasping is explored within the context of unstructured settings, taking into account elements such as task awareness and skill generation. The rest of this chapter will delve into these aspects in greater detail, examining how task-specific requirements and adaptive skills can enhance robotic grasping capabilities in dynamic and unpredictable scenarios.

2.2 Object understanding

In this section a review of recent methods for object-centric grasping is presented. Particularly, methods for grasping complex deformable object, category-level generalisation and task-aware grasping.

2.2.1 Grasping deformable objects

Although the field of robotic manipulation and sensing has advanced significantly over the past decades in handling rigid objects, fewer works have addressed, the manipulation of Deformable Objects (DO). Recent surveys highlight this as an underexplored area despite its importance in

various applications such as industrial manufacturing, medical surgery, food processing, and elderly care [23, 76, 216, 206, 167, 93].

Generally, deformable objects can undergo various types of deformations. Plastic deformation results in a permanent change in shape, whereas elastic deformation allows the object to return to its original form once the deforming force is removed. Elasto-plastic deformation is a combination of both, where the object does not fully revert to its original shape but also does not entirely retain the deformation[167]. In robotic manipulation for deformable objects, studies are also often categorised by the geometry of the manipulated object. Works that deal with "Linear Objects" include manipulation of ropes, cables, and strings [203, 123, 113, 102, 140, 200]; "Planar Objects" encompass papers, cards, and foam sheets, with cloth and garment manipulation being a particularly complex subset of this problem [206, 135, 110, 167]; and "Volumetric Objects" comprise items such as household and food products plush toys, and sponges [87, 111, 2, 173].

One of the major challenges in the manipulation of DOs is the accurate perception of deformations. This task is difficult due to the infinite-dimensional state space of DOs and the prevalence of occlusion and noise in unstructured environments. Indeed while rigid objects can be easily represented by a few parameters describing their configuration, DOs possess infinite dimensionality and complex dynamics, which poses significant challenges for robotic manipulation [76, 216]. Usually, methods for modelling, tracking and registration of DOs use mass-spring systems [185, 190] and finite element methods [52, 102] to represent their physical properties. However, these methods are not sufficient for capturing the full range of deformations encountered in practical scenarios. Many approaches have focused on 2D vision [140, 191, 37, 73], while 3D perception, which captures the state of deformable objects more accurately, poses additional difficulties (occlusion, noise data, etc) [216, 2, 200].

As effective state estimation for DOs requires high robustness and advanced sensing capabilities, a variety of works explored object understanding not only through vision systems but also using of force and tactile sensors, which offer local details about shape and contact points [166, 96]. Integrating these modalities to handle the deformation complexities remains an ongoing challenge for both rigid objects and for deformed objects [42, 100, 80].

Finally, DO manipulation disrupts several conventional assumptions in robotics, such as rigidity, predefined dynamic models, and simplified state spaces. Thus, current data-driven methods for robotic manipulation requiring extensive re-training to adapt learned skills to new instances of the deformable object are impractical for DOs due to their high variability in shape and deformation [76, 216]. Some studies tried to address the generalisation problem through Neural Spacial Encoding [161], contrastive geodesic learning [173] or by exploiting template based methods, in which correspondences based on Coherent Point Drift (CPD) [182, 119, 119, 182] or Functional Maps (FM) [45] attempt to address generalisation for different instances of a deformable object [206].

2.2.2 Shape correspondence for robotic manipulation

The challenge of adapting skills for instances of deformed objects without extensive retraining is addressed by exploring shape correspondences[1]. This can be approached by (i) examining the shape correspondences between different instances of a deformable object [45, 182, 119, 173, 161, 73]; (ii) considering category-level objects as warped versions of one another and seeking geometrical correspondences between their shapes [189, 195, 130, 66]; or (iii) or matching parts of an object (with or without deformation) to its full shape [186, 183].

For general grasping tasks, concepts such as those in [126] and [103] have been applied to minimise grasp uncertainties and to facilitate the transfer of dexterous grasp capabilities to novel objects. Particularly, [126] focuses on robust grasp planning through deep learning and synthetic training data, enhancing the robot's ability to predict successful grasps even with noisy sensory inputs. [103] uses a probabilistic approach to generate grasp models from a single demonstration, effectively transferring these models to unfamiliar objects through the use of contact and hand configuration models, thus achieving high success rates even with partial object views. However, these works focus on local similarities to ensure grasp and neither explicitly focus on grasping a specific part of an object. Alternatively, works such as [186, 162, 163, 83, 181, 125, 189, 45] exploit information transfer by identifying similarities between global object shapes. From such correspondences, grasps, affordances, or similar functions are used to identify a contact region for "functional grasp" and task execution.

Regarding the "shape correspondence" aspect of the problem, particularly between category level objects or non-rigid shapes, recent approaches have employed methods based on foundational models, data-intensive learning, or application-specific methods [46, 195, 189]. Probabilistic generative models of object geometry, such as those described by [73] offer a solution by inferring two-dimensional boundaries of partially occluded, deformable objects and recognising and retrieving complex objects from a pile for grasping tasks. Particularly, [73] leverages the "Probabilistic Procrustean Shape" framework to capture non-rigid deformations and object variability within a class, however it only deals with 2D images.

A few studies have explored learning-free solutions to dense shape correspondence in 3D. A notable method is the non-rigid registration based on CPD. For instance, CPD has been used by [182] to refine grasps within the same class of objects and in [119] to transfer manipulation skills between objects. Additionally, CPD is the also main choice for non-rigid shape matching in studies such as [162, 163]. Despite its efficiency, CPD requires a good initial pose to find good alignment and, in settings where there are larger deformations, is prone to becoming stuck in local minima, leading to poor results.

Recent alternatives to non-rigid shape matching have also been proposed [164, 19]. Functional Maps (FM) correspondence, first formulated by Ovsjanikov et al. [148], proved itself to

[1] Using shape correspondence for manipulation tasks and information transfer is closely related to the concept of template based methods in the last section where information is transferred from a template to a target object–this concept will be thoroughly explored throughout the remainder of this thesis.

be a robust alternative to most state-of-the-art methods [53]. The FM framework treats shape matching as the search for correspondences between functions on a reduced basis, which provides greater flexibility to incorporate linear constraints and simplifies the problem into a convex least-square optimisation. FM has demonstrated robustness in various computer graphics applications, including deformation and symmetry analysis, partial matching and the exploration of shape collections [147, 164, 19] as well as in applications ranging from medical to robotics [172].

2.2.3 Part-based grasping and affordance learning

The concept of affordances, first introduced by Gibson in 1979, refers to the ability of an agent to perform actions (or tasks) with an object in a given environment [72]. Although this concept encountered controversy within the field of psychology, it has gained popularity in various robotic applications, particularly in manipulation. In these contexts, the robotic agent perceives potential actions it can undertake on an object [14]. Thus, in robotic grasping and manipulation, affordances are closely related to "part-based grasping," "functional grasping,"[2] and tool use. Often, the robot must grasp part of an object (or tool) to accomplish a specific function or task.

The literature on this topic is extensive, covering both analytical and data-driven methods [125, 84, 179, 186, 138, 180], with several surveys having been published over the last few years [14, 81, 202]. These studies often focus on learning features based on the global shape of an object [114, 71, 88], graspable parts of category-level objects [138, 186, 121, 73], relations between hand taxonomy and object part [118, 192] or even cross-category similarities [183, 198]. Closely related works in this area explore grasp transfer between similar parts of objects or objects within the same category in a learning free or low data manner [182, 162, 163, 119, 181, 83, 184, 186]. For instance, [186] employed a method for transferring grasping information to novel objects within the same category by decomposing object models based on shape and local volumetric information and labelled semantic information. Other studies, such as those by [181, 83] explored transferring functional grasps through rigid alignment, contact warping, and local replanning. These methods consider category level objects as deformable instances of one another, allowing robots to perform analogous actions on new objects without specific prior grasps. This approach ensures grasp functionality across different objects within the same category by adjusting contact points and ensuring stability through local replanning. [184] proposed bijective contact mapping and grasp replanning to facilitate information transfer between parts of an object. They found rigid alignment by sampling the surface of two objects and minimising the deviation between points.

Furthermore, as discussed in the previous section, [182] and [119] employed CPD methods for non-rigid registration to transfer manipulation skills between objects. Although these methods are effective for shape matching, they can be prone to local minima, requiring good

[2]Functional grasps here refers to a robot's ability to grasp objects based on functionality.

initial poses to function well. [163, 162] proposed learning-based CPD method to account for shape deviations when transferring manipulation skills. Particularly for pose invariant grasping of meaningful object parts with larger variation, a FM based grasping transfer method had been shown to perform well [45].

2.2.4 Robot skill acquisition and transfer

Skill learning[3] and transfer are a natural extension of part-based grasp and affordance learning in robotics. Once the robot knows grasping locations, shape correspondences, such as the ones discussed in 2.2.2, can be employed to facilitate generalisation across category level objects. Furthermore, by leveraging human demonstrations (for a comprehensive review on the topic see [89, 211, 158]), robots can acquire complex skills and generalise these abilities to novel environments in a natural manner and with minimal subsequent interaction once the skill is defined.

Indeed, the ability to readily perform tasks in novel and unknown scenarios is essential for the deployment of robots in various industries, such as manufacturing, healthcare, and domestic environments. Numerous studies have addressed aspects of this problem. Learning-based approaches can establish relationships between objects in a scene and inform robots of the necessary actions for specific tasks [214, 97, 152]. Reinforcement learning has been presented as a solution for end-to-end skill learning [143]. Alternatively, manipulation capabilities derived from expert kinesthetic demonstrations have also shown success [86, 105]. Despite significant advancements, generalisation remains problematic, as learning is often tied to specific objects or pre-trained object categories, with little variation and/or requiring substantial amounts of training data.

A significant approach to task learning involves acquiring modular representations from human demonstrations, enabling users to instruct robots in performing new tasks. However, many existing solutions demand multiple demonstrations, which are often impractical or undesirable in real-world scenarios. A common strategy in this context is the encoding of trajectories, such as through the Dynamic Motion Primitives (DMP) approach [90, 91, 30]. Adapting DMPs to reach waypoints, a critical task for object-centric complex interactions, proves to be complex [150] and challenging for rotational tasks. Probabilistic Movement Primitives (ProMP), enhance generalisation [150], but their application is often limited to the demonstration region, constraining their utility in unseen scenarios [107, 188].

Dynamic systems offer convergence properties, yet they are intricate to design and often yield results confined to joint space. Zero- or one-shot learning approaches [199, 212, 207] assume an optimisation criterion for task definition [15] or depend on strong distributional assumptions about task motion generation [207]. These approaches decouple rotation and translation com-

[3]Here skill is defined as a robot's ability to execute motions that accomplish a task. E.g., pouring water from a bottle, mixing with a spoon, cutting with a knife, etc.

ponents, resulting in reference-dependent trajectories. Even interpolating between keypoints would depend on the choice of reference frame within an object [12], making them impractical for real-world scenarios. Conversely, previous work has proposed a user-guided approach for single-shot demonstrations, encoding relative coordinate-invariant geometric constraints within the path, enabling real-time deployment with object-centric constraint satisfaction [108, 107]. Nevertheless, these approaches are confined to the object used during the demonstration.

Recent research has addressed these limitations by focusing on category-level task acquisition. Manipulation tasks have been described with respect to semantic keypoints extracted from object point clouds [130, 66]. Neural Descriptor Fields have been employed to learn category-level implicit object models, describing interactions based on task-informed relationships between these descriptors [175, 176]. Despite their success, these approaches rely on resource-intensive (re-)learning steps and involve time-consuming data labelling processes. Less data intensive methods have been proposed, include single human demonstrations for peg-in-hole tasks with batteries and gears [195]. Despite its deployment efficiency, this method necessitates an offline training step, hindering the acquisition of task abilities for novel categories. Few-shot learning methods acquire object models and apply non-rigid deformations to their surfaces, augmenting the data available for category-level learning and enabling the immediate definition of novel tasks [189].

2.3 Grasping under uncertainty

Grasping, despite being one of the most crucial components of the manipulation process, is inherently uncertain. Measurements from a robot's sensors can be noisy, incomplete, or subject to calibration errors[4]. Cameras, which provide the robot with a sense of perception, are often prone to occlusion, while tactile sensors typically offer only local measurements. As a result, methods for planning grasps often need to be probabilistic to account for these uncertainties. Most studies divide grasping into two sub-problems: grasp planning and execution [117, 120, 25]. Grasp execution is frequently treated as a control problem, with research often addressing uncertainty from a control perspective [120] while the planning sub-problem deals with challenges related to incomplete or corrupted perception or difficulties that arise when physical properties of objects are often unknown or difficult to estimate.

To address these perception challenges, a variety of strategies have been proposed to manage uncertainties related to object properties that are not fully observable, such as pose, shape, mass distribution, inertia, and friction [151, 129, 146], with shape being shown to be one of the most significant properties for effective grasp execution [61, 55]. Consequently, an optimal shape representation should encode the uncertainty that varies across different surface regions and integrate information from both tactile and visual modalities. This representation should

[4]Work from [45, 58, 44, 59] have been adapted in this section.

also permit the gradual refinement of the shape model with continuous sensory data, facilitate probabilistic models for grasp success, and enable the generation of smooth reaching motions.

Moreover, planning reaching-to-grasp motions that consistently result in stable grasps is also challenging. Effective motion plans can be affected by minor errors and uncertainties that accumulate during open-loop execution. These uncertainties include various factors: the shape and 6-DoF pose of the object being grasped, the dynamic and kinematic parameters, and stiffness of high-DoF arms and hands, material properties that influence friction and contact forces, calibration discrepancies between the robot and vision system, and noise in sensory data reconstruction. In addition, environmental factors such as lighting conditions, occlusions, and the presence of multiple objects can further complicate the perception and manipulation tasks. Without sensory feedback and reactive plan adjustments during task execution, the initial reach-to-grasp plan frequently fails.

With advancements in computational capabilities and improved sensors, many works have begun addressing the inherent uncertainty in perception. This has been achieved either by developing strategies to mitigate it, such as employing alternative sensor capabilities to enhance world models [13, 65, 58], or by incorporating sensory feedback directly into planning strategies [88, 99]. For instance, studies such as [204, 99, 48, 47, 146] have introduced methods for addressing uncertainty through active exploration, and in [65, 24, 58], tactile sensing was combined with vision to compensate for occlusions and noise.

2.3.1 Grasping partially observed objects

In recent years, the challenge of enabling autonomous robots to grasp unfamiliar objects without relying on model matching has gained significant attention [114]. Grasp planning algorithms for unknown objects are typically divided into global methods, which evaluate the entire object (e.g, point cloud) to determine the optimal grasp [71], and local methods, which use object features such as local contact moments [3], spectral features [6], or edge and texture information from visible parts of the object [155] to select stable finger placements. Global approaches are often preferred for multi-finger grasping because they provide a comprehensive evaluation of the object as a whole. While local methods capture detailed surface features that may be lost during the surface estimation needed for global methods, global methods consider the overall shape, size, and surface characteristics of the object. These global algorithms allow for the incorporation of stability metrics, such as form or force closure, ensuring balanced force distribution and reducing the likelihood of slippage or object damage [114].

When designing global planning algorithms that consider the full object, object representation becomes crucial, as the efficiency and performance of the planner are directly influenced by the chosen representation. Some studies have simplified grasp generation through coarse approximations of the object's true shape [71] or by leveraging symmetry properties of common objects to capture shape features and generate grasp candidates [26]. However, to achieve

optimal results, a full observation of the objects is often necessary. Approaches such as those presented in [153] and [118] show the importance of detailed shape representation in grasp synthesis. Specifically, [153] employs smooth differentiable functions derived from point cloud data via spectral analysis, enabling continuous deformation and optimisation of grasps in a joint space, thus ensuring adaptable grasp configurations for various objects. Nonetheless, this parametrisation can degrade with partial point cloud data, highlighting the need for a shape space capable of handling missing data without resorting to unrealistic or oversimplified reconstructions. To deal with the problem without explicitly modelling shapes, [118] uses a database of human grasps and shape matching techniques to maximise the contact surface and select grasps based on the object's full shape and task-specific requirements. While effective for power grasps, this method does not focus on precision grasping needed for dexterous manipulation.

Deep learning has become a widely used approach for grasp synthesis on partially observed objects [215, 169, 122]. These methods rely on grasp data obtained from a specific set of training objects. As a result, they often struggle to generalise effectively to new objects with different properties. To address this limitation, alternative approaches have been developed, such as exploratory actions to find stable grasp configurations for novel objects [47, 48]. Unlike deep learning, exploration-based grasping focuses on identifying grasp configurations without necessarily relying on previous training. In [217] successive reach-to-grasp trajectories that maximise tactile information gain during iterative attempts, refining uncertain pose estimations for objects with known a priori size and shape is proposed. [48] introduces an uncertainty-aware control law that augments a grasp controller by preferring known or unseen regions of an object, but follow an exhaustive exploration strategy. In [178] a continuous exploration and grasping control method using the null-space of a multi-fingered hand has been proposed. Other promising methods involve using more refined grasp searching strategies, such as Bayesian Optimisation-based search. In [144, 33] a method which relies on complete object models uses Bayesian optimisation search for optimising wrist poses for grasping.

One alternative to improve understanding of an object's shape is by using tactile sensing. For instance, [22] developed a dynamic potential field approach for tactile exploration with a multi-fingered robot gripper. This method focuses on identifying grasp affordances by extracting geometric features and exploring regions of interest and then validating candidate grasps through a filtering process that eliminates unfeasible grasps. Another approach focuses on tactile-servoing for surface exploration, with features such as edges taken into account [115]. Additionally, 3D modelling from a single view from monocular vision has been achieved through the use of shape priors from large-scale repositories and active refinement through tactile sensing at uncertain regions [193]. Moreover, probabilistic approaches to shape modelling and reconstruction have also been explored. In [73] a probabilistic model of object geometry had been trained to deal with deformable and partially occluded objects, allowing recognition and grasping, including in cluttered environments. This approach however uses two-dimensional boundary information and adapting it to three-dimensional objects poses significant challenges.

Another promising approach combines active object exploration with tactile sensing for shape modelling. Particularly, Gaussian Processes Implicit Surface (GPIS), first proposed in [196], have been widely used in many studies addressing this problem. [204] proposes a single-finger active tactile exploration method inspired by Bayesian Optimisation, guiding the robot to explore uncertain regions of unknown objects. This method uses a Gaussian process to model the object's surface and associated uncertainty. In a similar manner, [99] introduces an online dynamic exploration strategy using a multi-fingered robotic hand. This method actively explores GPIS-reconstructed objects through touch, maximising the entropy of the acquired tactile data while dynamically balancing global knowledge and local complexity. Furthermore, [38] presents a sliding touch-based continuous exploration approach for GPIS-based 3D shape reconstruction using multi-fingered hands. By leveraging Bayesian Optimisation and a single-leader-multi-follower strategy, this method reduces shape uncertainty and enhances the accuracy of the reconstructed models.

Specifically, for grasp planning under uncertain conditions, GPIS has been used in various ways [129, 47, 146, 58, 59]. One approach involves grasp planning in 2D using visual data alone [129], while another employs blind grasping by following surface contours for shape estimation [47]. Additionally, GPIS has been used for grasping based on pre-trained systems using wrist poses [146]. Despite these advances, grasp planning under uncertainty remains an open problem. Efforts to address this include using GPIS with Bayesian Optimisation to simultaneously find grasps and reconstruct shapes [58], as well as reconstructing partial shapes for task-aware grasping [59].

2.4 Interaction with moving objects

With the increased use of collaborative robots in human-occupied environments, manipulators are increasingly tasked with grasping objects in unstructured settings where dynamic conditions are difficult to predict *a priori*[5]. Involving humans necessitates a high degree of reactiveness to varying conditions. Robots can encounter tasks such as handovers [145], catching objects in flight objects [101], or picking items from conveyor belts [9], and even throwing items at moving targets [27].

Grasping solutions for dynamic environments often focus on either the environment's dynamics or the object's dynamics. Solutions focusing on environmental dynamics address issues like obstacle avoidance and reactiveness to changing conditions. In contrast, object-centric approaches concentrate on planning for grasping and manipulation tasks involving objects with pre trained trajectories or models, tracking, or employing end-to-end methods that integrate vision and object perception as feedback to a control loop.

This problem has been explored since the 1990s. Early works by Houshangi [85] and Allen

[5]Work from [45, 58, 44, 59] have been adapted in this section.

et al. [11] proposed vision-guided schemes for executing grasps on moving objects. These early methods introduced strong assumptions, such as known trajectories and constrained workspaces and grasping spaces (e.g., only top-down grasps). More recently, several studies have sought more suitable solutions for dynamic grasping by relaxing some of these earlier constraints [101, 132, 75, 44].

[101] proposed an approach for catching objects in flight. Their prediction-based system calculates the trajectory of the thrown object and searches the space of reachable grasps for a motion that will intercept it. [132] presented a visual tracking-based method that dynamically replans the robot's trajectories and selects the best candidate from a set of generated hypotheses to grasp an object moved by a human. Similarly, [9] modelled reachability constraints and object motion to quickly screen a large grasp database as the object's state evolves. And more recently [121] proposed a method for part based reactive grasping.

One particular class of solutions to grasping moving objects involves closing the loop with sensing information. This approach continuously updates the robot's actions based on real-time feedback from various sensors, allowing for adjustments to be made on-the-fly. By integrating sensory inputs, such as vision, tactile, and proprioceptive data, robots can more effectively respond to the dynamic nature of the environment and the objects they are attempting to grasp. This feedback loop helps in compensating for uncertainties and variabilities in object motion and positioning, ultimately improving the success rate of dynamic grasping tasks.

2.4.1 Closed loop grasping

Closed-loop control systems are essential for enhancing the capability of robots to grasp moving objects in dynamic environments. Unlike open-loop systems, which execute pre-planned sequences without feedback, closed-loop systems continuously receive sensory information, enabling them to adapt to real-time changes. This adaptability is crucial for tasks involving unpredictable or fast-moving objects, where initial conditions can change rapidly. Particularly, closed-loop control ensures that robots can dynamically and stably adjust their strategies to achieve successful grasps, thus increasing efficiency and reliability in various applications.

A key aspect of closed-loop control is the integration of multiple sensory inputs to inform the robot's actions. Visual, tactile, and proprioceptive information can be combined to create a comprehensive understanding of the robot's environment and the objects within it. This multisensory approach allows robots to not only detect and track objects but also to adapt to unexpected changes in object trajectories or environmental conditions. For instance, tactile sensors can provide feedback on the grip force applied, preventing slippage or damage to delicate objects. Similarly, proprioceptive sensors (e.g., joint encoders) offer data on the robot's joint positions and movements, ensuring precise control and coordination.

Although tactile control for dynamic contour following on moving objects had been proposed [13], vision is still the preferred sensory modality to represent an object's state for closed-loop

control systems for robotic grasping. Through processing visual data, robots can identify target objects, track their movements, and adjust grasping strategies accordingly. Furthermore, the larger fields of view provided by cameras can even allow for motion prediction under the presence of obstacles, clutter, or multiple targets. This capability is particularly useful in complex tasks such as sorting, assembly, or packaging, where the robot needs to handle various objects with different shapes, sizes, and motion patterns.

Despite these advancements, most systems still focus on static grasping and the few works that tackle dynamic grasping propose simplifications to the problem, such as constraining the space to top-down grasps. Handling arbitrarily moving objects in a 3D space remains an open challenge, primarily due to the difficulties associated with planning arm and hand motions while simultaneously tracking the object's trajectory.

To this extent, Visual Servoing (VS) is one alternative to close the loop on robot grasping and manipulation. It involves using real-time image data to guide the robot's movements while regulating the error between some visual feature. The primary advantage of VS is its ability to provide precise and timely adjustments to the robot's movements. By continuously analysing the visual feedback and updating the control commands, VS systems can achieve high accuracy in grasping and manipulation tasks. This capability is especially beneficial in scenarios where objects are moving unpredictably or where precise alignment is required for successful grasping. Furthermore, VS also facilitates smooth and coordinated movements, as it allows for incremental adjustments based on continuous feedback. This smoothness is critical in applications such as robotic surgery or delicate assembly tasks, where abrupt movements could cause damage or failure. Additionally, VS systems can be enhanced with machine learning techniques to improve their adaptability and performance over time, making them more effective in a wide range of dynamic environments.

Effective grasping of moving objects also relies heavily on several factors, including the positioning of cameras and the availability of inverse kinematic solutions. Two common camera configurations are discussed in the literature: *eye-in-hand*, where the camera is attached to the robot's end-effector, and *eye-to-hand*, where the camera is placed within the environment to observe both the robot and the object. The *eye-to-hand* setup is often preferred for dynamic tasks as it eliminates "blind spots," but it demands precise simultaneous tracking and pose estimation, which is challenging due to the lack of robust online tracking methods. Techniques such as the integration of Image-Based Visual Servoing (IBVS) with shared control grasping, as presented in [75], and a novel method for direct visual servoing in the spectral domain from [4], have shown success in grasping moving objects through VS. However, their efficiency is dependent on maintaining inverse kinematic solutions throughout the trajectory. Moreover, manipulators with redundant joints are particularly susceptible to singular configurations, adding another layer of complexity to the grasping process.

In addition to camera placement, the representation of the robot's and the object's poses is essential. One possible representation is dual quaternion algebra. Unit dual quaternions

offer a compact and computationally efficient alternative for representing full 6-DoF, similar to homogeneous transformation matrices or angle-axis representations. Unlike traditional methods, however, unit dual quaternions are singularity-free, making them especially advantageous for VS tasks. Due to the coupling between translation and rotations, they have also proven efficient and useful in many vision-related topics, such as hand-eye calibration [43], simultaneous localisation and mapping [116], and pose estimation and tracking [165, 208, 29] and trajectory interpolation. Although dual quaternions have gained popularity in both robotic control and vision applications, their combined use remains underexplored in the literature. It wasn't until recently that [165] introduced a method employing dual quaternions for VS, where they proposed a Position-Based Visual Servoing (PBVS) schema that regulates camera velocity, with pose tracking achieved through an Extended Kalman Filter. Additionally, de Farias et al. [44] proposed a PBVS schema based on dual quaternions, coupled with a re-ranking and singularity avoidance strategy, which allows for grasping moving objects more effectively.

Part II

Object understanding

Chapter 3

Grasp transfer for deformable objects

Preliminary remarks

The research in this chapter was published in the following papers:

- **C. de Farias**, B. Tamadazte, R. Stolkin and N. Marturi. Grasp Transfer for Deformable Objects by Functional Map Correspondence . In 2022 IEEE International Conference on Robotics and Automation (ICRA), 2022. doi: 10.1109/ICRA46639.2022.9812141.
- **C. de Farias**, B. Tamadazte, R. Stolkin and N. Marturi. Grasp Transfer for Deformable Objects by Functional Map Correspondence. In 2nd Workshop on Representing and Manipulating Deformable Objects 2022., ICRA 2022.

Note: Certain sections of this chapter, such as problem descriptions, specific equations, variable descriptions, interpretations of results, and figure/table captions, have been adapted from [45, 56].

3.1 Introduction

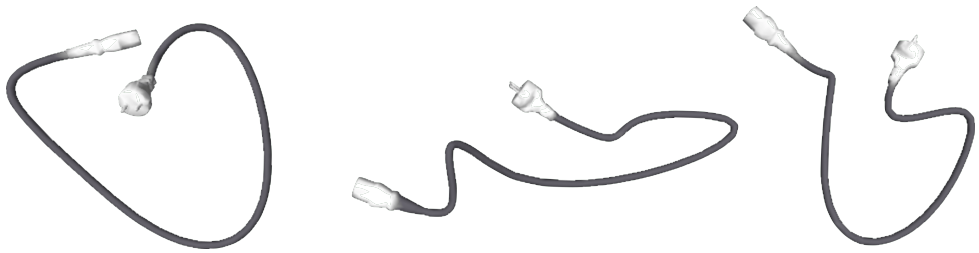


Figure 3.1: Examples of power cables in different configurations.

Robotic grasping poses significant challenges in perception, planning, control, and object manipulation. The complexity of these challenges increases when dealing with deformable

objects, which present additional difficulties due to their tendency to change configurations during manipulation. This includes variations in the centre of mass, soft contacts, and limited understanding of how functional integrity is maintained during deformation. This chapter particularly focuses on the impact of non-rigid deformations on perception and the optimisation of robotic systems to handle such deformations while preserving grasp stability. An example of this problem can be illustrated by the task of grasping a power cable, such as the one in Figure 3.1 and plugging it into a wall socket. Not only can the shape of the plug vary, but the cable itself may adopt numerous configurations, deforming with an infinite number of degrees of freedom.

Machine learning-based methods may be trained on a limited set of deformation scenarios for grasping tasks, but in low-data or learning-free contexts, template methods can be used. However, deformations complicate the situation significantly, making conventional approaches (used for rigid objects) less effective. Although defining a physical model of the object is a potential solution, it is often computationally intensive and dependent on parameters such as the object's stiffness. An alternative approach, proposed in [162, 163] involves the using of non-rigid shape matching algorithms, such as those based on Coherent Point Drift (CPD) method. While powerful, they rely heavily on the initial alignment of points which can limit their applicability.

In this chapter, the Functional Maps (FM) framework [148] is used, leveraging isometric invariant descriptors to establish shape correspondences between near-isometric shapes under non-rigid deformation. This approach is applied to the problem of grasp transfer where a grasp planned for an object in a certain configuration can remain effective for the object in another configuration, provided that the transformed region is *locally rigid* (e.g., the arm of a "teddy bear" may move to different positions, but the arm itself will not undergo significant deformations). Based on this concept, a learning-free method is proposed, enabling direct grasp transfer along deformed objects without the necessity for additional training. It incorporates user input and Local Contact Moments (LoCoMo) based grasp planning [3] to generate a ranked list of grasps focused on the region of interest of an object. It then applies the FM-based matching to accurately map planned grasps onto the appropriate part of the object, which has experienced arbitrary deformation.

The primary contributions of this chapter can be summarised as follows:

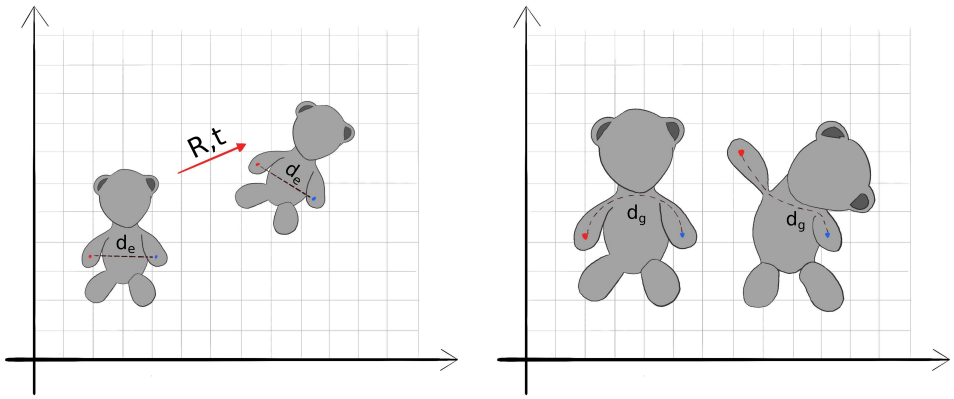
- A novel, robust, learning-free solution is introduced, leveraging the FM framework for transferring grasps between objects undergoing non-rigid deformation. To the best of our knowledge, this is the first work to explore the application of Functional Maps in robotics.

- The method incorporates user input and Local Contact Moments (LoCoMo) based grasp planning to generate a ranked list of grasps focused on the region of interest of an object. This algorithm applies the FM-based method to accurately map planned grasps onto objects that have experienced arbitrary levels of deformation, and subsequently executes the grasp on the appropriate part of the object.

- Comprehensive validation experiments are presented, using a simulated 7-axis robot equipped with a 2-finger gripper across various objects and deformation scenarios. ThThese experiments demonstrate the method's reliability and effectiveness. Furthermore, comparison studies are performed with the existing state-of-the-art techniques.

3.2 Problem

To contextualise the problem more effectively, let the concept of *"object"* or *"object shape"* first be defined as a manifold. That is, the surface of an object, whether it is a bottle, a cup, a teddy bear, or a screwdriver, can be mathematically characterised as a topological space that is locally Euclidean around each point on the surface. Furthermore, here, if this object undergoes a **rigid transformation**, it implies that the object moves while preserving the Euclidean distances between all of its points. This preservation can be observed both on the manifold itself and within the three-dimensional Euclidean space in which the object is embedded. The notion that an object, such as the teddy bear in Figure 3.2(a), maintains all distances on its surface during rotational or translational motions is intuitive; for example, the Euclidean distance between the tips of its arms (or any other pair of points) remains unchanged regardless of how the object is moved. This mathematical property of distance preservation is associated with the concept of **isometry**.



(a) Rigid body motion **(b)** Non-Rigid deformation.

Figure 3.2: Example of a rigid and non-rigid deformation. In (a) all points of the object translate and rotate together, preserving Euclidean distances (d_E) on the grid. In (b) the object shape deforms, yet the geodesic distances (d_g) are preserved.

Definition 4 (Isometric Surfaces)**.** Given the shapes $\mathcal{S}$, $\mathcal{S}'$ and the pairs of points $\mathbf{p}_1, \mathbf{p}_2 \in \mathcal{S}$ and the equivalent $\mathbf{p}'_1, \mathbf{p}'_2 \in \mathcal{S}'$ such that the function $\mathbf{f} \in \mathbb{R}$ is a transformation $\mathcal{S}' = \mathbf{f} \cdot \mathcal{S}$, then $\mathcal{S}$ and $\mathcal{S}'$ are isometric surfaces if a distance metric is preserved (i.e., $d(\mathbf{p}_1, \mathbf{p}_2) = d(\mathbf{p}'_1, \mathbf{p}'_2)$).

The concept of an object can be further refined by equipping it with a Riemannian metric, thereby making it a Riemannian manifold. Given that an object is a smooth manifold $\mathcal{S}$, the

tangent plane T_pS at a point $\mathbf{p} \in \mathcal{S}$ is equipped with a positive-definite inner product $g_{\mathcal{R}}(\cdot,\cdot)$. In connected topological spaces, such as Riemannian manifolds, the distance between two points can be defined as **Geodesic Distances**.

Definition 5 (Geodesic Distances)**.** In a manifold $\mathcal{S}$ equipped with a Riemannian metric, $g_{\mathcal{R}}$, the geodesic distance between two points $\mathbf{p_1}, \mathbf{p_2} \in \mathcal{S}$ is the shortest unique path that lies entirely on $\mathcal{S}$. For a more detailed mathematical definition, refer to [112].

From Definitions 4 and 5, the notion of isometric transformations can be extended to Riemannian Manifolds and more generally to deformable objects. Consider the "teddy bear" in Figure 3.2(b). Unlike the scenario in Figure 3.2(a), this object undergoes non-rigid motions (e.g., its arm may move upwards). However, there remains a connectivity[1] in the shape, which results in the preservation of geodesic distances. Intuitively, if the "teddy bear" moves its arms in space, it would appear to deform from the perspective of an external observer (i.e., it would change its shape). However, the geodesic distance between the tip of the arm and the ear would remain consistent regardless of its configuration [2] .

Thus, the problem of how perception is influenced by non-rigid deformations of an object's shape is introduced—specifically, how the concept of isometric non-rigid transformations can be used to preserve "function" and grasp stability. For humans, it is intuitive that a "teddy bear" that has moved its arms and legs remains the same object. Indeed, correspondences can often be recognised between shapes that have undergone significant deformations, even when there is no obvious relation. However, for a robotic system, this type of association typically demands extensive training data of instances of the object in various configurations, or results in a low success rate when dealing with objects that have undergone more substantial deformations.

In this context, FM is used as a method for transferring grasps between two object shapes. Let $\mathcal{S}_{db}$ represent the shape of an object in a database and $\mathcal{S}_{\text{target}}$ the shape of an object in an observed scene. If $\mathcal{G}$ denotes a set of grasp poses associated with the object $\mathcal{S}_{db}$, then the problem this chapter aims to solve is

$$\mathcal{G}_{\text{target}} = \mathcal{F}(\mathcal{G}, \mathcal{S}_{db}, \mathcal{S}_{\text{target}}), \tag{3.1}$$

where $\mathcal{F}(\cdot)$ is the function that transfers the grasps from the database to the newly deformed object $\mathcal{S}_{\text{target}}$.

[1] For a formal definition of connectivity in Riemannian manifolds, refer to [112].

[2] Some texts refer to rigid transformations as those that preserve distances, using the term rigid-motion interchangeably with isometric transformations. However, as non-rigid objects can undergo transformations that result in shape changes (thus not qualifying as rigid motions, as Euclidean distances in space are not preserved) while still preserving geodesic distances, a distinction is made between the terms in this work.

3.3 Background

This section presents the background concepts relevant to the proposed method. The theory underlying the mapping and analysis of geometrical properties in non-rigid shapes is first discussed, with a focus on the FM framework introduced in [148, 147], which is used for establishing correspondences between non-rigid shapes for grasp transfer. This is followed by grasp planning.

3.3.1 Shape correspondence

Shape matching is a fundamental concept in computer graphics and image processing, involving the identification of similarities between shapes. The objective of shape matching is to determine the similarity between two given shapes, despite variations in size, orientation, or even deformations [164]. Unlike rigid shapes, which typically undergo transformations limited to translations and rotations, deformable shapes present a higher complexity in shape correspondence. For rigid shapes, correspondence is often simplified to finding a translation and rotation that align both forms.

In the example shown in Figure 3.2(a), the aim is to match the shape on the left, $\mathcal{S}_x^{\text{RIGID}}$, to the one on the right, $\mathcal{S}_y^{\text{RIGID}}$, assuming it is rigid. By discretising these shapes (or assuming they are point clouds or meshes), they can be represented as

$$\mathbf{S}_x^{\text{RIGID}} = \{\{\mathbf{x}_i\}, i = 0, 1, 2, \ldots, N\} \tag{3.2}$$

and

$$\mathbf{S}_y^{\text{RIGID}} = \{\{\mathbf{y}_i\}, i = 1, 2, \ldots, N\} \tag{3.3}$$

where N is the number of points (vertices) of each shape and $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{R}^3$. Assuming each $\mathbf{x}_i$ corresponds to $\mathbf{y}_i$, and that the distances between each pair of points are invariant, it is necessary to search the space of possible rigid transformations to align the shapes. That is, to find the optimal transformation, the energy function

$$E(\mathbf{R}, \mathbf{t}) = \sum_{i=0}^{N} \| \mathbf{x}_i - (\mathbf{R}\mathbf{y}_i + \mathbf{t}) \|^2 \tag{3.4}$$

where $\mathbf{R}, \mathbf{t}$ represent the rotation and translation of the object is minimised.

Although this is a simplified case, with the assumption that points are perfectly ordered (i.e., $\mathbf{x}_i$ matches to $\mathbf{y}_i$), minimising (3.4) forms the basis for more complex shape matching problems. Even with rigid shapes, individual points in the object may not be ordered equally, or the number of points in $\mathbf{S}_x^{\text{RIGID}}$ and $\mathbf{S}_y^{\text{RIGID}}$ may differ. A popular method for such registration challenges problems is the Iterative Closest Point (ICO) algorithm [18], which iteratively minimises the difference between two point clouds by finding the best fit between them. In summary, ICP

follows these steps [156]:

1. **Initialisation**: The two sets of points, $\mathbf{S}_x^{\text{RIGID}}$ and $\mathbf{S}_y^{\text{RIGID}}$, are considered. For each pair of points $\{\mathbf{x}_i, \mathbf{y}_j\}$, the Euclidean distance $d_{i,j}(\mathbf{x}_i, \mathbf{y}_j)$ is calculated.

2. **Correspondence finding**: Correspondences are identified based on the nearest neighbour using $d_{i,j}$.

3. **Transformation estimation**: The optimal transformation between the corresponding points identified in the previous step is determined by minimising (3.4).

4. **Application of transformation**: The computed transformation is applied to the source point set, $\mathbf{S}_x^{\text{RIGID}}$, to better align it with the target $\mathbf{S}_y^{\text{RIGID}}$.

5. **Iteration**: The error $e = \sum_i^{N_x} \sum_j^{N_j} d_{i,j}$ is calculated, and if $e >$ threshold, steps 2 through 4 are repeated.

6. **Termination**: The algorithm terminates when $e <$ threshold, indicating that $\mathbf{S}_x^{\text{RIGID}}$ has been sufficiently aligned with $\mathbf{S}_y^{\text{RIGID}}$.

Note that for rigid shapes, the matching problem is simplified due to the low dimensionality of the transformation space (e.g., rotations and translations). In contrast, non-rigid shape matching is not limited to searching over a low-dimensional space of transformations (even when isometry-preserving transformations are considered); rather, it involves establishing point-to-point correspondences between two shapes. When a shape is discretised into N points, any point in the source shape could, in principle, be mapped to any point in the target shape. As a result, the number of possible mappings increases on the order of of $O(N!)$ in the worst case [164].

3.3.1.1 Coherent Point Drift

Another choice for non-rigid point matching and registration, which has shown promising results in robotics applications is Coherent point drift (CPD) algorithm [139, 163, 162, 182, 119]. CPD formulates the registration problem as a probabilistic estimation problem based on Gaussian mixture model (GMM).

Thus, let two point sets be considered, $\mathbf{S}_\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n\}$ (referred to as the floating set) and $\mathbf{S}_\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_j, \ldots, \mathbf{y}_n\}$ (denoted as the fixed set), such that a GMM is constructed with centres at the points in $\mathbf{S}_\mathcal{Y}$. The points in $\mathbf{S}_\mathcal{X}$ are treated as observations of this GMM. The alignment of the distributions is then iteratively optimised by the algorithm through expectation-maximisation optimisation, which consists of two main steps:

E-step the correspondence probabilities between the sets of points are calculated based on a current estimate of the transformation parameters, where each point in $\mathbf{S}_\mathcal{X}$ is matched to the GMM based on $\mathbf{S}_\mathcal{Y}$.

M-step once the probabilities are known, the transformation parameters (only rotation, translation, and scaling in the rigid case, with additional regularisation parameters in the non-rigid CPD) are solved for. This is achieved by maximising the likelihood function to align the two sets of points according to the estimated probabilities. Once solved, the values are updated.

In this work, CPD-based grasp transfer has only been used as a baseline. This choice was due to some of CPD's drawbacks, such as the need for a good initial alignment between point clouds and its inability to deal with large deformation. Instead, Functional Maps Correspondence was chosen as an alternative.

3.3.1.2 Functional Map correspondence

The FM framework is a recent advancement in non-rigid shape matching that has demonstrated effectiveness in a diverse set of applications. In this approach, the map $M : \mathcal{S}_{\text{target}} \rightarrow \mathcal{S}_{db}$ is not determined through a computationally intensive assignment problem (originally $O(N!)$ in complexity), but rather as a more manageable optimisation between real-based functions represented on a low-rank compact base. This distinction makes FMs much more tractable and robust, converting it into a linear least-squares problem. This representation not only simplifies the optimisation process, but also enables the inclusion of linear constraints to guide the mapping process while maintaining its linear nature. Furthermore, depending on the chosen bases and descriptors, the framework can retain invariance to both rigid motions and near-isometric deformations.

Given the shapes $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$, the real-valued functions $f : \mathcal{S}_{db} \rightarrow \mathbb{R}$ and $h : \mathcal{S}_{\text{target}} \rightarrow \mathbb{R}$ are considered. The mapping M transfers f to $\mathcal{S}_{\text{target}}$ via the composition $h = f \circ M$, resulting in $h(p) = f(M(p))$ for any $p \in \mathcal{S}_{\text{target}}$. This transformation between functions is linear, allowing it to be represented in matrix form as $\mathbf{h} = \mathbf{\Pi f}$, with $\mathbf{f}$ and $\mathbf{h}$ as vector forms of the functions and $\mathbf{\Pi}$ as a linear map between these functions (not necessarily a bijection).

In the FM formulation, extra complexity is avoided by focusing on reduced bases for the functions instead of working in the full $\mathbb{R}^{n_{\text{target}}}$ or $\mathbb{R}^{n_{db}}$ spaces. A small set of orthonormal basis functions is computed over $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$, with their coefficients represented in the columns of $\mathbf{\Phi}_{db}$ and $\mathbf{\Phi}_{\text{target}}$. It should be noted that the number of bases, k_{db} and k_{target}, used to describe the correspondence in this formulation is typically less than the number of vertices in the object, *i.e.*, $k_{db} \ll n_{db}$ and $k_{\text{target}} \ll n_{\text{target}}$. Furthermore, for $\mathbf{f}_1, \mathbf{f}_2 \in \mathcal{S}_{db}$, the inner product on the manifold is defined as $\langle \mathbf{f}_1, \mathbf{f}_2 \rangle_{\mathcal{S}_{db}} = \mathbf{f}_1^T \mathbf{A}_{db} \mathbf{f}_2$, where $\mathbf{A}_{db}$ is a diagonal matrix of weights, facilitating calculations on the geometric structure of $\mathcal{S}_{db}$. A similar definition applies to $\mathcal{S}_{\text{target}}$. For $\mathbf{h}_1, \mathbf{h}_2 \in \mathcal{S}_{\text{target}}$, $\langle \mathbf{h}_1, \mathbf{h}_2 \rangle_{\mathcal{S}_{\text{target}}} = \mathbf{h}_1^T \mathbf{A}_{\text{target}} \mathbf{h}_2$. It is important to highlight that if the vectors are orthonormal with respect to the weighted inner product, then the inner product becomes an identity matrix. Thus, for orthonormal bases, $\mathbf{\Phi}_{db}^T \mathbf{A}_{db} \mathbf{\Phi}_{db} = \mathbf{I}$ (or $\mathbf{\Phi}_{\text{target}}^T \mathbf{A}_{\text{target}} \mathbf{\Phi}_{\text{target}} = \mathbf{I}$), facilitating straightforward transformations. The FM matrix $\mathbf{C}$, which represents the transformation

from $\mathcal{S}_{db}$ to $\mathcal{S}_{\text{target}}$, is computed as

$$\mathbf{C} = \mathbf{\Phi}_{\text{target}}^{T}\mathbf{A}_{\text{target}}\mathbf{\Pi}\mathbf{\Phi}_{db}, \tag{3.5}$$

where $\mathbf{\Pi}$ denotes the matrix representation of the map M. This formulation assumes that the initial correspondence between the shapes is known, however, in practise that is not often the case. To estimate the map the FM pipeline aims to find $\mathbf{C}$ and subsequently M. Overall the FM pipeline is summarised as: (i) computation of a small set of orthonormal basis, (ii) computation a set of descriptor vector functions, (iii) estimation an optimal value of $\mathbf{C}$ and finally (iv) obtaining the point-to-point map from $\mathbf{C}$. Each of these steps is described below [147, 148].

(i) Choice of basis The choice of basis is one important aspect of the functional maps, as it can lead to reductions in the complexity of the problem and stability under deformations. A popular choice of basis, proposed in the initial FM pipeline, is the first n Laplace-Beltrami eigen-functions [148]. The Laplace-Beltrami operator (LBO) is the generalisation of the Laplacian to Riemannian manifolds and it admits eigen-decomposition with non-negative eigenvalues λ_i and corresponding orthonormal eigen-functions ϕ_i such that $\Delta\phi_i = -\lambda_i\phi_i$. In this work the cotangent scheme in [137] is used to compute the LBO. The eigen-functions provided by the LBO operator are ordered from low to high frequency, and are often referred to as the harmonic of the manifold. This choice of basis is particularly well-suited for shape matching problems due to its invariance to rigid motions and isometries. It is also computationally efficient and offers a natural multi-scale approach for approximating functions.

(ii) Choice of descriptor In the FM pipeline, a descriptor is a quantity that should be approximately preserved by the mapping. There are many choices of descriptors encoding different properties of shapes. Some examples are the heat kernel signature (HKS)[147], Signature of histogram orientations (SHOT) [147] or some learned descriptor. Particularly, a sub-set of interesting descriptors for the non-rigid shape matching problem are those derived of the LBO, as these are invariant to isometry and robust to some non-isometric deformations. One LBO based descriptor is the Wave kernel signature (WKS) [16], which arises from the Schrodinger equation governing the temporal dissipation of quantum mechanical particles in a surface. For more details on its computation and properties the reader is referred to [16]. Particularly to the shape matching problem, the WKS has been shown to perform well on a variety for datasets [159] and is selected in this work. It is worth noting that shape matching a stable and accurate matching between shapes is crucial for transferring proprieties (e.g., grasp poses) between shapes. Once a choice of descriptors is made, they are computed for the source and target shapes, resulting respectively in $\mathbf{w}_i^{db}$ and $\mathbf{w}_i^{\text{target}}$ which in matrix form are $\mathbf{W}^{db}$ and $\mathbf{W}^{\text{target}}$.

(iii) Computation As noted above, the descriptors should be preserved by the mapping. That is, $\mathbf{C}\mathbf{w}_i^{db} \approx \mathbf{w}_i^{\text{target}}$. Thus, it is expect that the following energy function to be minimal

$$E_{\text{desc}}(\mathbf{C}) = \|\mathbf{C}\mathbf{W}^{db} - \mathbf{W}^{\text{target}}\|^2. \tag{3.6}$$

Additionally to the descriptor preservation, a number of linear regularisation constraints can easily be added so that the resulting map is improved without additional complexity of the optimisation problem. Some of the desirable properties to be induced in the maps are that they should be approximately intrinsically isometric, pointwise function products of descriptors should be preserved to guarantee a better association with point-to-point maps and orientations should be preserved. For details on how each regularisation constrain is defined, the reader is referred to [147]. Here the overall energy function which includes isometry, orientation and product preservation can be expressed as $E_{\text{Reg}}(\mathbf{C})$ and the map $\mathbf{C}$ can then be obtained by solving the optimisation problem

$$\mathbf{C} = \arg\min_{\mathbf{C}} \left(\alpha_1 E_{\text{desc}}(\mathbf{C}) + \alpha_2 E_{\text{Reg}}(\mathbf{C})\right), \tag{3.7}$$

with α_1 and α_2 being scalar weights. Note that, as all energy functions within $E_{\text{Reg}}(\mathbf{C})$ are linear, the optimisation problem is still leads to a simple convex least squares solution.

(iv) Refinement and point-wise map recovery To obtain the transformation between different parts of an object, it is necessary to recover the point-wise map from the functional map $\mathbf{C}$. This recovery can be achieved through iterative alignment and refinement in the bases domain, using the Bijective and Continuous Iterative Closest Point (BCICP) algorithm as proposed in [159]. BCICP improves upon traditional ICP by not only ensuring bijectivity, and thus avoiding overlaps and maintaining invertibility, but also by enforcing continuity across the mappings. BCICP works iteratively by taking the maps $\mathbf{C}_{DT}$ (computed from $\mathcal{S}_{db}$ to $\mathcal{S}_{\text{target}}$) and the inverse map, $\mathbf{C}_{TD}$ (from $\mathcal{S}_{\text{target}}$ to $\mathcal{S}_{db}$). The algorithm alternates between computing point-wise maps using nearest neighbor searches within the spectral domain and updating the FM by projecting it onto the closest orthonormal matrix to minimize mismatches. This projection ensures the map remains within the space of isometries, thus maintaining bijectivity and enhancing continuity. Continuity and bijectivity are promoted by optimising both in the spectral and spatial domains, leading to computationally efficient and higher-quality mappings. In Figure 3.6(b) the point-to-point correspondences between two objects is shown, with similar colours indicating similar regions.

3.3.2 LoCoMo-based grasping

Local Contact Moments (LoCoMo) based grasp planning method has been implemented in this work for the sake of generating grasp hypotheses [3]. It offers two significant benefits: first, it

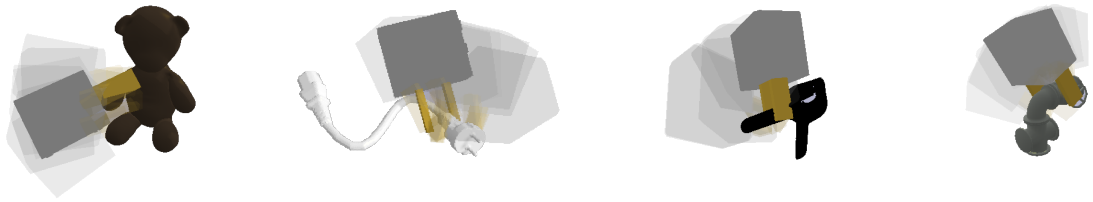


Figure 3.3: LoCoMo grasps planned over specific parts of different objects. From left to right, bear arm, cable plug, plier handle and pipe wheel. The highest ranking grasp has the highest opacity.

does not require pre-existing 3D models of objects or any form of offline training, facilitating its integration with our system. Second, it supports grasp selection through a grasp ranking scheme [5], resulting in an ordered list of grasps with quality metric associated with it.

Given an object point cloud and a point cloud of the finger model, an initial set of grasp and pre-grasp poses[3] are generated using the LoCoMo grasp planner. This is achieved by first computing the zero-moment shift features on both clouds. Here, the zero-moment shift refers to the displacement of the centroid of a subset of the point cloud's points relative to a predefined reference point, thereby capturing characteristics of limited regions of surfaces. The zero-moment shift is particularly good at encoding information about surface curvature. These features of object and gripper are then searched in order to maximise the contact surface. This is done through the LoCoMo probability function given by,

$$\begin{aligned} L_i &= \frac{1}{N_s} \sum_{j=1}^{n} \left((2\pi)^n \left|\Sigma\right|\right)^{\frac{1}{2}} \mathcal{N}(\Psi_j; \vec{0}, \Sigma) \\ \mathcal{R} &= \gamma \prod_{i=1}^{N_f} L_i^{\omega_i} \end{aligned} \tag{3.8}$$

where L_i represents the contact moment for each gripper finger and $\mathcal{R}$ denotes the ranking score. The term N_s acts as a normalisation factor, and n represents the number of point cloud patches sampled around‘ each finger's current position. The function $\mathcal{N}(\Psi_j; \vec{0}, \Sigma)$ is the multivariate Gaussian density centered at $\vec{0}$ with covariance Σ, where Ψ_j captures the difference in zero-moment shift vectors between the object and the gripper surfaces. The parameters ω_i and γ are weighting factors, and N_f indicates the total number of fingers used in the grasp. Figure 3.3 illustrates examples of grasps planned using LoCoMo.

[3] A pre-grasp pose is defined as the grasp pose with an offset along the robot's approach axis, so the robot can reach the object without risking collisions.

3.4 Methodology

In this section, the methodology for grasp transfer across different instances of deformable objects is detailed.

3.4.1 Method overview

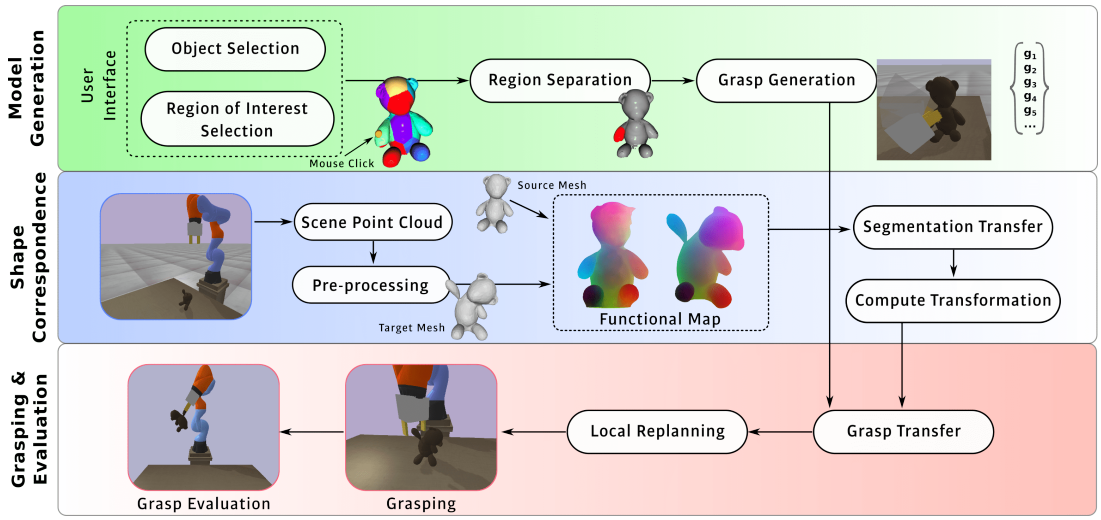


Figure 3.4: Pipeline for non-rigid grasp transfer based on FM. It consists of three steps: grasp model generation (top – green); finding shape correspondences (middle – blue); and, grasp transfer (last – red).

Given a pair of non-rigid shapes, $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$, embedded in $\mathbb{R}^3$, the FM framework finds a correspondences map between their features. As previously stated, a shape is denoted as a Riemannian manifold. In practice, $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$ are stored as meshes with n_{db} and n_{target} vertices, respectively. The point-wise correspondence map $M : \mathcal{S}_{\text{target}} \rightarrow \mathcal{S}_{db}$ is then extracted from the FM, establishing the relation between each segmented part of the source and target objects. From this, the optimal transformation $\mathbf{T} \in \text{SE}(3)$ between these segmented regions can be identified, enabling the effective transfer of a grasp between instances of an object. The pipeline depicted in Figure 3.4 summarises the proposed grasp transfer approach. It includes three main steps:

Grasp model generation: This phase is initialised by user input, involving the generation of grasping models by LoCoMo over a selected segmented region of the object.

Shape correspondence: The FM framework finds the correspondence map M between the source object shape $\mathcal{S}_{db}$ and the target object shape $\mathcal{S}_{\text{target}}$. With M, the corresponding segment of the object can be obtained in $\mathcal{S}_{\text{target}}$ (segmentation transfer in 3.4) and subsequently $\mathbf{T}$ can be obtained.

Grasp transfer, execution, and evaluation: The grasp is then adjusted and effectively transferred to the target, ensuring stability and feasibility. The stability of the grasp is rigorously evaluated as outlined in [17].

3.4.2 Data acquisition

In this work, only fully observed and nearly isometric shapes (i.e., shapes in which the geometric distances within the surface are mostly preserved), are considered. Thus, it is necessary to pay close attention to the data acquisition process in order to ensure consistency between the source and target shapes. Partially observed shapes are handled in Chapters 5 and 6.

Let $\mathbf{S}$ denote the ground plane segmented point cloud of an object. This cloud is obtained from a wrist mounted camera, where multiple point clouds are acquired from various viewpoints of an object and stitched together. The source and target objects are denoted respectively as $\mathbf{S}_{db}$ and $\mathbf{S}_{\text{target}}$.

Once the point clouds are available, they are transformed into meshes using the Poisson surface reconstruction algorithm [98]. This generates watertight meshes from oriented point sets and adeptly preserves the geometric details and can be used as discrete approximations of Riemannian manifolds. Given the typically high density of the resulting meshes, a step for quadric mesh decimation is also included in the process to simplify the meshes. The processed meshes (manifolds) are denoted $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$.

3.4.3 Grasp model generation

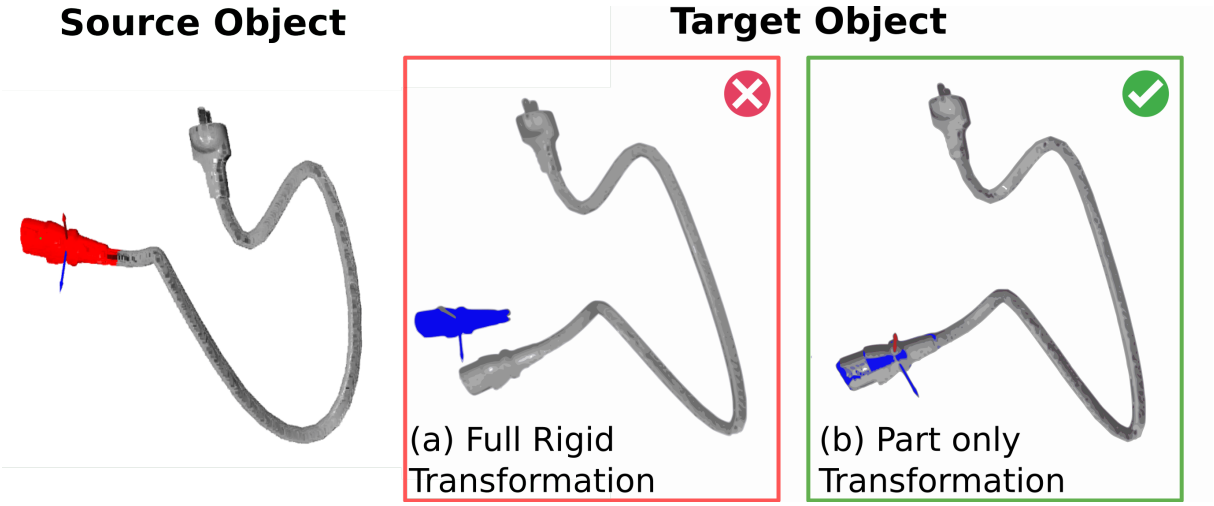


Figure 3.5: Transformation of a segmented region (in red) of the source object (power cable). (a) Shows the transformation when all the object points are considered and (b) shows the transformation of the points inside the segmentation region only (applying local region transformations).

The grasp model is comprised of a set of variables that fully describing grasps relative to a

shape. To better contextualise the problem, first, consider the rigid case. For transferring grasps between the same rigid object in different poses, the information required in the grasp model consists of a list of grasp hypotheses and an object point cloud. In this case, the problem can be reduced to finding an optimal rigid transformation between the source object and the target.

However, transferring grasps between non-rigid shapes presents a more complex challenge. Even if the shapes are isometric, the rigid transformations affecting different parts of the object are likely to be non-uniform and nonlinear. To address this, it is proposed that the object be segmented into multiple parts, and the problem be tackled by identifying rigid transformations for these segmented regions. As illustrated in Figure 3.5, if the full shape is used and an attempt is made to obtain the optimal rigid transformation between points, the alignment will be incorrect. However, by considering only a local region of the object, precise alignment can be achieved. It is assumed here that each segmented part is sufficiently large to encompass all finger contacts for at least one grasp, and that no significant deformations occur within each segmented region.

Moreover, when an object is grasped, the aim is to perform some task with it. Therefore, grasps should be strategically planned to facilitate task accomplishment. In this approach, expert human knowledge is leveraged to inform grasp selection. This is facilitated through a user interface that allows the human expert to select a region of interest on the object (or the region the object should be grasped)—this is depicted as the "mouse click" in Figure 3.4. The interface then displays a segmented point cloud of the object, divided into N_C parts using *k-means* clustering, as shown in Figure 3.6(a). The segmented cloud is denoted as $\mathcal{S}_{\text{part}}$ with $\mathcal{S}_{\text{part}} \subseteq \mathcal{S}_{db}$.

In this work, grasps on $\mathbf{S}_{db}$ are synthesised using the LoCoMo grasp planner, as discussed in Section 3.3.2. The ranked list of N_G grasp poses in the world frame is denoted as the set $\mathcal{G} = \{\mathbf{g}_i \in \text{SE}(3), i = 1...N_G\}$. Considering only part-based grasps, grasp poses that result in finger placements outside the segmented regions are filtered out, leading to the subset $\mathcal{G}_{\text{part}} \subseteq \mathcal{G}$. Finally, after the segmentation and grasp generation steps, the grasp model is defined as the set $\{\mathcal{G}_{\text{part}}, \mathcal{S}_{\text{part}}, \mathcal{S}_{db}\}$.

3.4.4 Grasp transfer algorithm

The next step involves computing the grasp transformation and ensuring its stability. The process begins by calculating the optimal transformation $\mathbf{T}$ between the segmented regions of the two shapes. To achieve this, the n vertices in $\mathcal{S}_{db}$ that correspond to the region defined by $\mathcal{S}_{\text{part}}$ are first computed, and their positions are stored in a $3 \times n$ matrix $\mathbf{V}_{db}$. Using the map M, the corresponding points that define the segmented region of $\mathcal{S}_{\text{target}}$ are identified, and their positions are stored in $\mathbf{V}_{\text{target}}$, ensuring that row i in $\mathbf{V}_{db}$ corresponds to row i in $\mathbf{V}_{\text{target}}$ for all $1 \leq i \leq n$. With $\mathbf{V}_{db}$ and $\mathbf{V}_{\text{target}}$, the optimal rigid rotation and translation, $(\mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3)$, are calculated between the two sets of points. Inspired by (3.4), the transformation can be

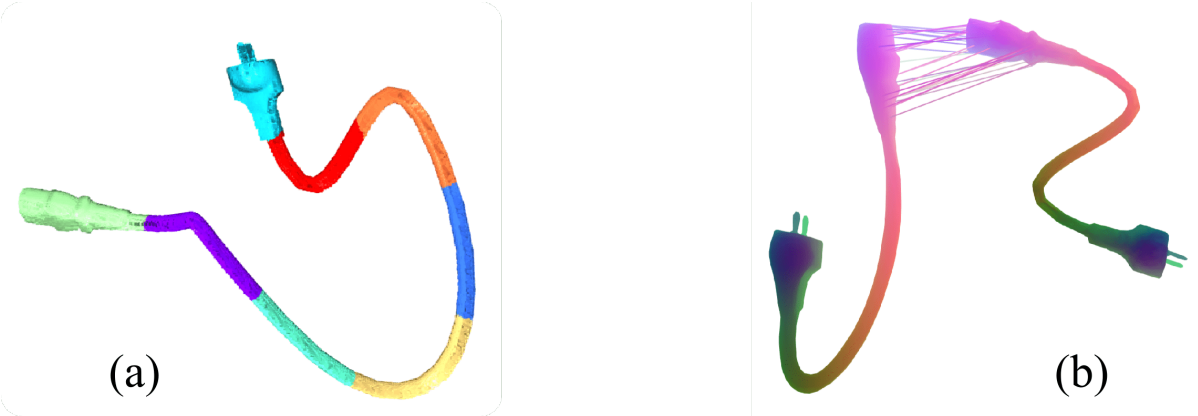


Figure 3.6: (a) The object is divided into segments using *k-means* clustering, with each segment represented by a different color. The user selects the segment for robotic grasping. (b) Correspondence of points between two objects is shown, with matching colors denoting identical regions.

obtained from

$$\mathbf{R}, \mathbf{t} = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{n} \|\mathbf{R}\mathbf{V}_{db}^{(i)} + \mathbf{t} - \mathbf{V}_{\text{target}}^{(i)}\|^2. \tag{3.9}$$

where the full rigid transformation $\mathbf{T} \in \text{SE}(3)$ is determined from $(\mathbf{R}, \mathbf{t})$. $\mathbf{T}$ is then used to compute the transferred grasp $\mathbf{g}_{\text{target}} = \mathbf{T}\mathbf{g}_i$, where $\mathbf{g}_i$ is the top-ranked feasible grasp in $\mathcal{G}_{\text{part}}$. To verify the feasibility of the new grasp, an inverse kinematics (IK) check is performed after applying the transformation.

In ideal scenarios, where the segmentation map is perfect and both shapes correspond precisely without deformations within the segmented regions or noise, $\mathbf{T}$ alone is sufficient to transfer the grasp between objects. However, this is often not the case. To enhance grasp transfer success under less ideal conditions, an additional step of local re-planning inspired by [184] is introduced. That is, for a gripper with N_F fingers, let $\mathbf{p}_{db}^{(j)}$, $j = 1...N_F$ represent the point on the surface of the object closest to the j-th finger when the gripper is at $\mathbf{g}_i$. Using the map M, the equivalent points $\mathbf{p}_{\text{target}}^{(j)}$ in $\mathcal{S}_{\text{target}}$. Assuming $\mathbf{p}_c^{(j)}$ represents the finger positions when the grasp is at $\mathbf{g}'_{\text{target}}$. Then, asm$\mathbf{p}_c^{(j)} \approx \mathbf{p}_{\text{target}}^{(j)}$, the grasp can be re-planned by minimising

$$\mathbf{g}'_{\text{target}} = \arg\min_{\mathbf{g}'_{\text{target}}} \sum_{j=1}^{N_F} (\mu_1 \|\mathbf{p}_c^{(j)} - \mathbf{p}_{\text{target}}^{(j)}\|) + \mu_2 \epsilon + \Psi, \tag{3.10}$$

where μ_1 and μ_2 are scalar weights, ϵ the error between $\mathbf{g}'_{\text{target}}$ and $\mathbf{g}_{\text{target}}$ (indicating how much the grasp has changed), and Ψ is a scalar penalty term for unreachable grasps.

Algorithm 1 summarises the method.

Algorithm 1: Grasp transfer for deformable objects.

Input : $\{\mathcal{G}_{\text{part}}, \mathcal{S}_{\text{part}}, \mathcal{S}_{db}\}$, M
$\mathbf{V}_{db}$ is obtained from $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{part}}$
$\mathbf{V}_{\text{target}}$ is obtained from M and $\mathcal{S}_{\text{target}}$
$\mathbf{R}, \mathbf{t}$ are calculated from (3.9) to obtain $\mathbf{T}$
while *not IK* **do**
    The next $\mathbf{g}_i$ is obtained from $\mathcal{G}_{\text{part}}$
    $\mathbf{g}_{\text{target}} = \mathbf{T}\mathbf{g}_i$ is calculated
    A check is performed to determine if $\mathbf{g}_{\text{target}}$ is IK
end
$\mathbf{p}_{db}^{(j)}$ is obtained for all j from $\mathbf{g}_i$
$\mathbf{p}_{\text{target}}^{(j)}$ is obtained from M and $\mathbf{p}_{db}^{(j)}$
$\mathbf{g}'_{\text{target}}$ is calculated from (3.10)
The object is grasped
Return: $\mathbf{g}'_{\text{target}}$

3.5 Experiments

This section details the simulation experiments that were conducted to assess the effectiveness of the proposed method, which employs FM for transferring grasps between objects subjected to varying degrees of deformation. Initially, the process used to generate deformed object models for validation is described. Subsequently, the performance of the method is examined by comparing it with both the CPD algorithm and a version of the ICP algorithms.

3.5.1 Object deformation dataset for validations

Due to the lack of available datasets containing multiple deformable models for experimental use in simulation, a custom dataset was created, comprising various objects with arbitrary non-rigid deformations. This dataset currently includes four different object models, each with five distinct deformations. By considering each deformed pose as a separate entity, the dataset effectively comprises 20 unique objects. The initial 3D models were sourced from multiple scans or downloaded from online databases (e.g., a power cable model, pliers). These models underwent preprocessing, such as smoothing, and were subsequently manipulated using 3D modelling software, such as Blender, to produce the deformed variants. The resulting deformed models were saved as Wavefront object files, making them readily accessible for experimental use. The generated object models are displayed in Figure 3.7. The dataset is available for download at: https://gitlab.com/cristianafar/deformable-object-dataset.

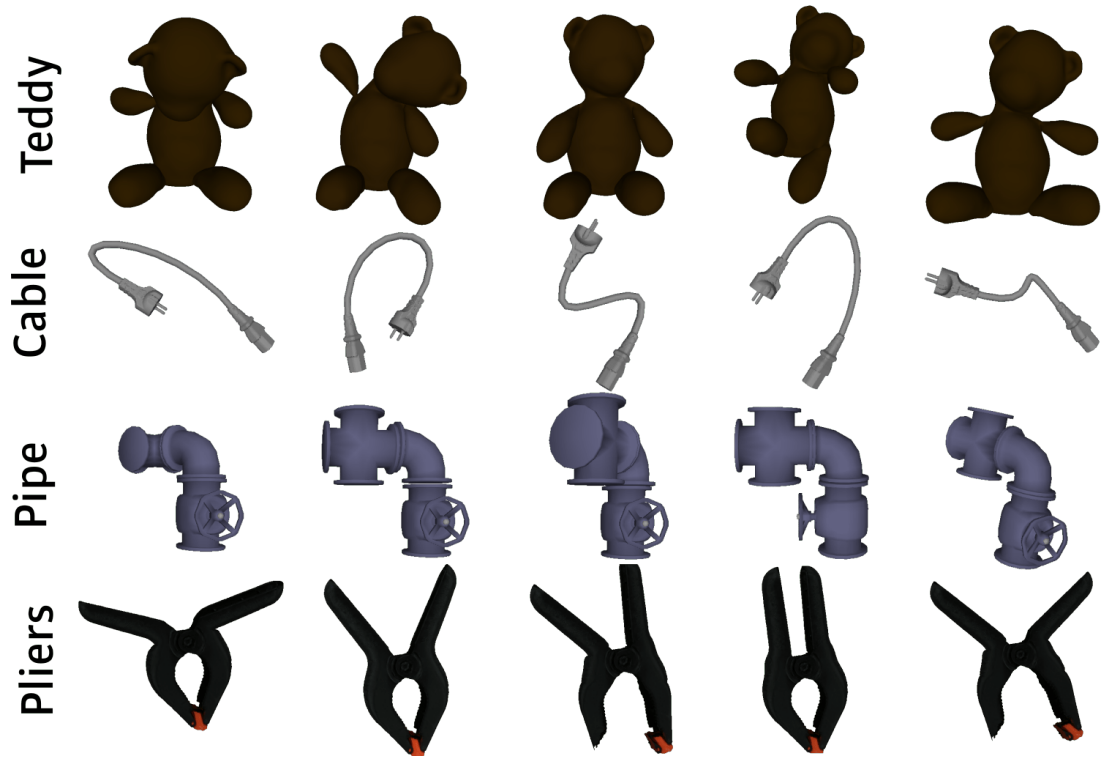


Figure 3.7: Dataset of objects and their deformed versions created for this work.

3.5.2 Experimental setup

All experiments were conducted using the Pybullet simulator, with a 7 DoF robot equipped with a 2-finger gripper. Scene data were captured by mounting a simulated 3D camera on the robot's end-effector. Since the method relies on high-quality meshes, point cloud data were collected by positioning the robot at 15 different locations for each new object (scene). As detailed in Section 3.4.2, these point clouds were then stitched together to create the input source mesh. To balance quality with processing time, the source mesh was limited to 3000 vertices. The FM correspondence was implemented using the software package referenced in [159], using 50 WKS descriptors and 100 eigen-vectors for the LBO basis. In the grasping pipeline, the parameters $\mu_1 = 0.2$ and $\mu_2 = 0.8$ were set empirically.

3.5.3 Grasp Transfer Evaluation

In this section, the evaluation of the proposed grasp transfer algorithm is presented. The efficacy of the algorithm was assessed by attempting to grasp all objects within the dataset. For each object, one of the five meshes was designated as the model mesh, while the remaining four were used in robot experiments. The model mesh was segmented using k-means clustering, with $N_C = 7$ clusters for each object, and displayed on the initial screen for user interaction. A region of interest, denoted as $\mathcal{S}_{\text{part}}$ in the algorithm, was then selected by the user from one of the clusters.

Table 3.1: Performance evaluation for grasp transfer based on Functional Maps (FM), Coherent Point Drift (CPD), and Iterative Closest Point (ICP).

Object	Ours (FM-based method)				CPD				ICP			
Configuration[3]	Lift[1]	Rot.[1]	Shake[1]	Part[2]	Lift[1]	Rot.[1]	Shake[1]	Part[2]	Lift[1]	Rot.[1]	Shake[1]	Part[2]
Power Cable (Selected region: Plug)												
C1	100	100	100	100	100	100	100	100	0	0	0	0
C2	100	100	100	100	0	0	0	0	0	0	0	0
C3	100	100	100	100	100	100	100	100	100	0	0	0
C4	100	100	100	100	100	0	0	0	0	0	0	0
Teddy Bear (Selected region: Right Arm)												
C1	100	100	100	100	100	0	0	0	0	0	0	0
C2	100	100	100	100	0	0	0	0	0	0	0	0
C3	100	100	100	100	0	0	0	0	0	0	0	0
C4	100	100	100	100	100	0	0	0	0	0	0	0
Modular Pipe (Selected region: Wheel Section)												
C1	100	100	0	100	100	100	100	100	100	100	100	100
C2	100	100	100	100	0	0	0	100	100	0	0	100
C3	100	100	0	100	0	0	0	100	0	0	0	0
C4	100	100	100	100	100	0	0	100	0	0	0	100
Pliers (Selected region: Left Handle)												
C1	100	100	100	0	100	100	100	0	100	100	0	0
C2	100	100	100	100	100	100	100	100	0	0	0	100
C3	100	0	0	100	100	100	100	100	100	100	100	100
C4	100	100	100	100	100	100	100	100	0	0	0	0
Total	**100**	**93.75**	**81.25**	**93.75**	68.75	43.75	43.75	56.25	31.25	18.75	12.5	31.25

[1] Percentage of success (%) of lifting, rotational and shaking tests – reported out of 3 trials for each configuration.
[2] Percentage of success in grasping the correct object part – Accuracy test.
[3] Out of five models in our dataset, four deformation configurations of each object are used for robotic experiments.

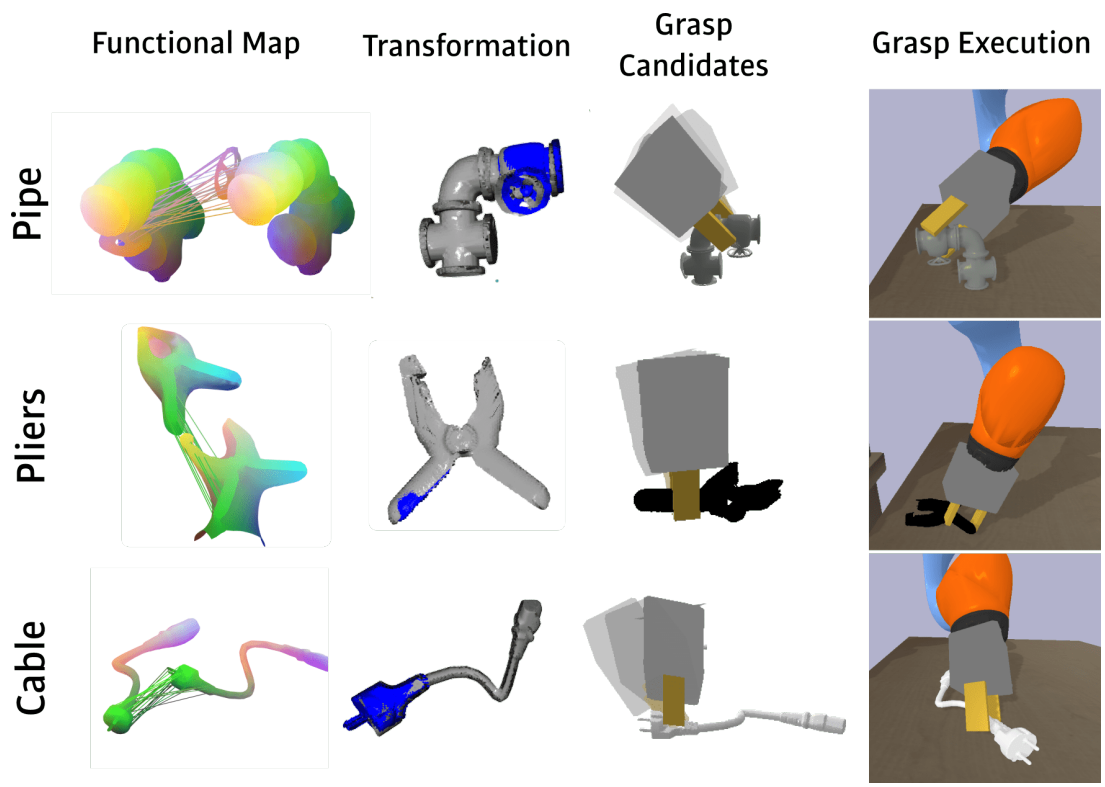


Figure 3.8: Simulation results for objects in the dataset. In the first column the point to point map between shapes is shown, where similar colors represent similar regions of the shape. Following, the transformed part from the database shape (in blue) over the target shape. Then grasp candidates and finally screenshots of the grasp execution.

Subsequently, using $\mathcal{S}_{\text{part}}$, a set of grasps denoted by $\mathcal{G}$ was generated by the LoCoMo grasp planner and stored. The number of grasps vary from object to object, depending on the geometry of the object, surface area and the density of the point cloud. For a full object cloud LoCoMo often generates over 10k grasps. The LoCoMo planner provides a ranked list of these grasps, from which kinematically unfeasible ones were filtered out after the transfer. For each object category, the experiments were repeated three times for each of the remaining four deformations. Grasp execution was carried out with a force of 50 N.

The stability of the grasp was then assessed through three sequential tests, as described in [17]: first a lifting test, then shaking, and rotating the object. If an object slipped at any stage, the results were marked as a failed outcome, and no further tests were performed. The accuracy test was conducted by verifying whether the fingers were positioned within the chosen region of interest. Sample results are shown in Figure 3.8.

Additionally, to compare the performance of the proposed method against other similar algorithms, baseline experiments were conducted by replacing the FM framework within our algorithm with the CPD method implemented via the PyPCD library, and the ICP method using the Open3D Python library. Comparative results are summarised in Table 3.1.

3.5.4 Discussion

From Table 3.1, the effectiveness of the FM-based method in handling deformable objects is clearly demonstrated. A success rate of **93.75**% was achieved in accuracy tests, with an overall success rate of **100**% in lifting objects, and an **81.25**% success rate following rotation and shaking tests. In comparison, the CPD method exhibited lower performance, with a 56% success rate in grasping the correct region, 68.75% in lifting, and only 43.74% success after rotation and shaking tests. The ICP method also performed poorly with deformable objects, successfully grasping the correct region in only 31.24% of attempts, achieving a 31.25% success rate in lifting, and just 12.5% after rotation and shaking tests.

The low success rate of the ICP method was expected, as it is designed for rigid alignments and does not handle deformations. Nevertheless, it managed to grasp some objects with smaller deformations, such as certain configurations of pliers and pipes. Even when the alignment was not perfect, the local re-planning step allowed for some successful grasps. The CPD algorithm, which is better suited for non-rigid transformations, has shown promise in previous studies [182, 119, 163, 162], but struggles with larger deformations due to its dependency on accurate initialisation. Although nearly 100% success was achieved in grasping the pliers, the method failed with most other objects, even when correctly aligning to the intended part. This failure can be attributed to the fact that, even with integrated re-planning, if the initial transformation is poor, the final transformation will be far from the region where the grasp was originally planned, thus compromising stability.

The FM-based method has proven to be a robust alternative, particularly effective in cases with larger deformations where the other methods fail. The primary reason for this success is that the FM algorithm finds correspondences directly on the object's shape manifold, remaining invariant to both rigid motions and near-isometric deformations. This robustness was particularly evident with challenging objects such as the Teddy Bear and power cable, where the FM-based method was the only one to successfully transfer grasps. The failure with the Pliers object was not due to mapping issues but rather a change in object configuration (see Fig. 3.7, last row, fourth configuration), where the closed position of the object prevented proper grasping. This suggests that incorporating additional collision checks during grasp filtering could enhance success rates.

A limitation of this approach is that only the grasps available on the target shape can be transferred. For example, in the case of the Teddy Bear's arm (Figure 3.7, third column), grasps that would collide with the body are not computed. If the arms are raised (as in Figure 3.7, second column), these grasps would become available. However, since they were not initially computed, they are not considered. One way to address this limitation is to frame the problem as affordance transfer. In this approach, the segmented region of the object is transferred first, and grasps are computed afterward. This method ensures that graspability is assessed dynamically rather than being constrained by the initial shape configuration. The concept of affordance

transfer will be explored further in subsequent chapters.

3.6 Summary

In this chapter, a method for transferring grasps between different instances of objects that have undergone arbitrary deformations has been introduced. It has been demonstrated that the proposed approach effectively transfers grasps from one region of an object to the equivalent region on a deformed version of the same object using the FM framework. The robustness of FM to large deformations was validated by the high success rate achieved in grasping across a diverse set of objects with varying levels of deformation. The method not only showed superior performance in terms of grasp success but also in the accuracy of identifying grasp regions when compared with other techniques.

Potential enhancements for the methodology include integration with learning-based approaches for autonomous semantic understanding of objects, incorporation of additional sensory data such as tactile feedback to refine grasp quality, and expansion into task-aware grasping strategies by learning about object affordances.

Chapter 4

One-shot skill transfer

4.1 Introduction

In the previous chapter, the concept of using the Functional Maps (FM) framework to transfer grasps between different shapes (represented as Riemannian manifolds) was introduced. There the grasping problem is addressed by identifying "locally rigid" regions of an object and determining the rigid transformations between corresponding regions across different objects. Although this segmentation-based approach is effective, a local alignment step is required to account for minor deformations, errors arising from to the complexities of real-world shapes, and deviations from the isometry condition.

In this chapter, instead of focusing solely on grasp transfer, a functional view of the problem is proposed. Here, functions are defined over the surface of an object and these quantities (related to physical properties, affordances, or skills) are transferred across object surfaces. Grasp transfer alone is therefore redefined as a subset of this broader problem, with the grasp encoded within a "locally rigid" region. Thus, by defining a function f over an object's manifold and leveraging the FM framework, functions are projected onto the Laplacian space, allowing direct transfer of properties across different shapes. This method generalises properties across object categories while maintaining near-isometry, providing a stable, pose- and isometry-invariant foundation for accurate transfer.

Here, different functions defined over objects and their equivalents on other shapes can be defined. These functions are categorised into "locally rigid functions," encompassing grasp poses, "physical functions" such as friction and stiffness, "task cost functions" (or affordances), and "physical interaction functions" derived from robot-environment interactions. This chapter focuses on how physical interaction functions obtained through a single kinesthetic demonstration can be used by a robot to replicate tasks with different objects and configurations.

Thus, a method for instantly acquiring new object-centric manipulation capabilities through a one-time user demonstration, without prior training, can be described. Correspondences between object geometries are explored, allowing information to be transferred across scenes with objects

of the same category. Unlike in Chapter 3, where FM was used to transfer grasps to deformed configurations, this work leverages FM to simplify the transfer of functions across shapes into a straightforward matrix product. By modeling interactions between the robot, objects, and the environment as functions over object shapes, FM can map information across near-isometric objects, even with significant non-rigid deformations. Once interaction functions are mapped, relative poses are extracted for path generation. The Task-Space Imitation Algorithm (TISA) [108] is then used to generate a new path that transfers the constraints of the demonstrated task in a coordinate-invariant manner. This framework enables the robot to replicate skills learned from a single demonstration in different scenarios both safely and instantaneously.

The key contributions of this work are summarised as follows:

- A novel learning-free framework that generalises object-centric interaction and planning based on a one-time user demonstration is presented. This framework enables the robot to understand and replicate complex manipulation tasks without the need for extensive training or multiple demonstrations. By leveraging the Functional Maps Framework, the demonstrated task is encoded as a set of functions over the object's surface, which can then be transferred to similar objects in different scenarios. The system is designed to be robust and adaptable, allowing for efficient and accurate task replication across various environments and object configurations.

- Online *skill transfer* methodology, consisting of skill function transfer and path planning for robots interacting with category-level objects and satisfying single-demonstration constraints, are introduced. This approach allows robots to instantly apply the learned skills from a single user demonstration to new, unseen objects of the same category.

- An object selection metric based on the functional map between two objects that can identify the best matching object in a novel scene.

In this chapter, the results from Chapter 3 are further generalised, and focus is also placed on the the object proprieties described in [181], where shapes are viewed as warping of one another.

4.2 Problem overview

By leveraging kinesthetic teaching and functional encoding of tasks, robots can learn complex tasks from a single human demonstration and subsequently adapt these skills to new but similar scenarios. Section 4.2.1 details a conceptual example of task-execution to illustrate the goal of the proposed method. Following, in Section 4.2.2 a more general problem statement is presented.

4.2.1 Conceptual example

To illustrate the proposed method for transferring the ability to execute a robotic task in different scenarios and with different objects, consider the task of mixing a liquid in a bottle. Though seemingly simple, the task requires the execution of a sequence of precise steps. Assuming the bottle is upright, and that the robot must grasp the bottle at an optimal location with the correct orientation. Though grasping the middle section of the bottle ensures stability, grasping the cap with an approach axis aligned with the upright z-axis allows for more angular movement, enhancing mixing. After grasping, the robot must perform circular movements and then place the bottle back on the table with its bottom touching the surface for further movement. Here, in order to generalise movements, the robot is first taught how to perform it through a single kinesthetic demonstration and then by capturing a different scene it transfers the information to a new scenario.

4.2.1.1 Demonstration phase

During the demonstration, a user guides the robot through the motions of picking up, shaking, and placing down the bottle. The main elements recorded include:

Gripper location function: Encodes the optimal grasp point—in this case at the cap for angular movement.

Motion path function: Records the mixing motion, ensuring the correct final orientation.

Placement function: Determines the final placement of the bottle with its bottom on the table.

4.2.1.2 Execution phase

In a new phase, the robot uses the learned functions to adapt its actions:

Initial grasp location : Identifies the optimal grasp point for the circling movement by shape correspondence.

Final bottle configuration : Obtains the correct final orientation after mixing is done.

Final placement : Places the bottle with its bottom touching the table.

Figure 4.1 illustrates the task execution as described, showing the robot performing the mixing circular motion and placing the bottle back on the table. This example demonstrates how the task is transferred to a new setup.

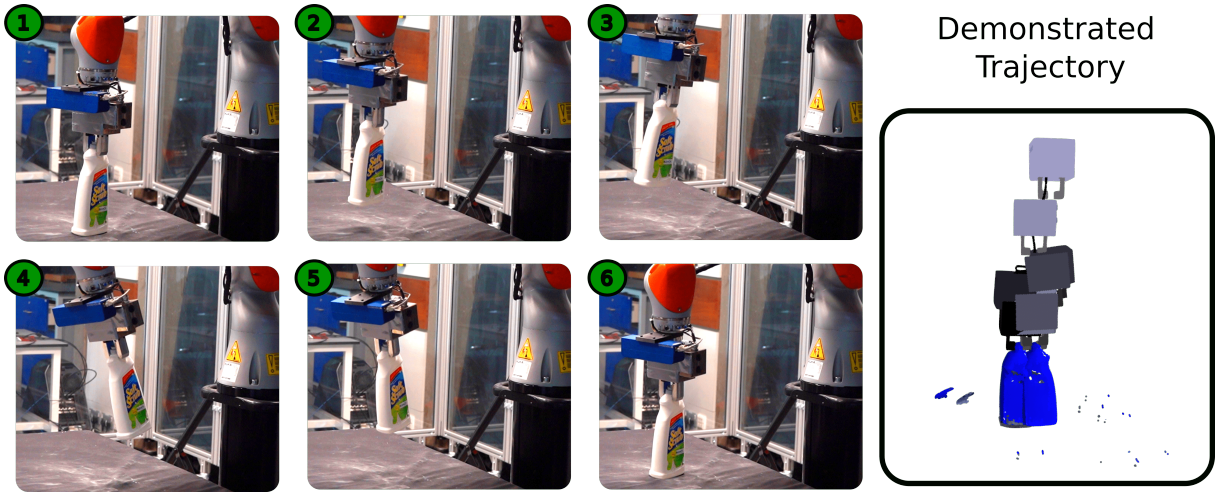


Figure 4.1: Demonstration of the bottle shaking task, where the robot picks the bottle by the cap, performs a circular motion and subsequently places it back up again. In the last column the generated trajectories are shown.

4.2.2 Problem Statement

Referring to the conceptual example, this chapter addresses the problem of representing a demonstrated skill by incorporating geometric constraints and functional representations of contacts and distances relevant to objects in a scene. These representations must fully describe the robot's motions and be generalisable to new instances and objects in different scenes. In that manner, given an object, denoted by the Riemannian manifold $\mathcal{S}$, let a "Task Operation" i.e., the robot's ability to perform a task be defined as follows:

Definition 6 (Task Operation)**.** *A Task Operation, denoted by $\mathcal{T}$, is the set of trajectories and functions encoding the geometric characteristics of the desired task.* It can be expressed as

$$\mathcal{T} = \{\underline{\boldsymbol{X}}, \mathbf{f}_{(\text{task})}\}, \tag{4.1}$$

where $\mathbf{f}_{(\text{task})} \in \mathcal{S}$ is an interaction function defining the task-specific relationships between the robot, environment, or other objects in the scene and $\underline{\boldsymbol{X}} = \{\underline{\boldsymbol{x}}_0, \underline{\boldsymbol{x}}_1, \underline{\boldsymbol{x}}_2 \dots \underline{\boldsymbol{x}}_n\}$ is the robot's end-effector trajectory between the initial robot pose $\underline{\boldsymbol{x}}_0$ and the final robot pose $\underline{\boldsymbol{x}}_n$.

In that manner, let the demonstration scene $\boldsymbol{\Theta}^{\text{dem}}$ composed of N objects, be represented as

$$\boldsymbol{\Theta}^{\text{dem}} = \{\mathcal{S}_1^{\text{dem}}, \mathcal{S}_2^{\text{dem}}, \dots, \mathcal{S}_N^{\text{dem}}\}. \tag{4.2}$$

Then, during the demonstration stage, a user guides the robot through a series of trajectories, resulting in k sequential operations, each characterised by paths and interaction functions:

$$\mathcal{T}_i^{\text{dem}} = \{\underline{\boldsymbol{X}}_i^{\text{dem}}, \mathbf{f}_{(\text{task},i)}\}, \quad i = 1, \dots, k. \tag{4.3}$$

When the robot encounters a new scene, such as $\mathcal{S}'$, with M objects,

$$\boldsymbol{\Theta}' = \{\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_M\}, \tag{4.4}$$

the objective is to derive the corresponding robot operations $\mathcal{T}_i'$ that enable the robot to replicate the demonstrated operation with novel objects and the constraints of the new environment.

The problem can thus be articulated through the following:

1. **Transferability of interaction functions**: The interaction functions that describe spatial and constrained relationships from the demonstrated task $\boldsymbol{\Theta}^{\text{dem}}$ must be transferable to new objects of the same category, accommodating variations in shape and deformation.

2. **Maintenance of geometric constraints**: All implicit geometric constraints observed during the demonstration must be preserved during deployment, regardless of changes in the reference frame.

4.3 Background

This section outlines the background concepts used in the proposed method. It covers the interpolation technique for encoding trajectories in Section 4.3.1 and the FM. While the main algorithm for FM was introduced in Section 3.3.1, Section 4.3.2 presents the *ZoomOut* extension of the FM framework for refining maps on objects that are not fully isometric, as introduced in [136]. Furthermore, elements of trajectory planning using Dual Quaternion Algebra are discussed, with fundamental properties detailed in Appendix B.

4.3.1 User-guided trajectory generation

As mentioned earlier, the TSIA [108] is used to generate a trajectory that implicitly adheres to the geometric constraints from the demonstrated path $\underline{\boldsymbol{X}}^{\text{DEM}}$. Initially, this path is represented as a sequence of unit dual quaternion poses

$$\underline{\boldsymbol{X}}^{\text{dem}} \triangleq \left\{ \begin{array}{cccc} \underline{\boldsymbol{x}}_0 & \underline{\boldsymbol{x}}_1 & \ldots & \underline{\boldsymbol{x}}_n \end{array} \right\}, \tag{4.5}$$

with $\underline{\boldsymbol{x}}_1, \underline{\boldsymbol{x}}_2, \ldots, \underline{\boldsymbol{x}}_n \in \mathbb{H} \otimes \mathbb{D}$. The relative motion with respect to the final end-effector pose $\underline{\boldsymbol{x}}_n$ is then computed as in [108],

$$\underline{\boldsymbol{\delta}}_i = \underline{\boldsymbol{x}}_{i-1}^{*} \underline{\boldsymbol{x}}_i, \quad i = 2, \ldots, n. \tag{4.6}$$

Given the desired final goal pose of the end-effector $\underline{\boldsymbol{x}}'_n$ in the new scene $\boldsymbol{\Theta}'$, the imitated path

$$\underline{\boldsymbol{X}}' \triangleq \left\{ \begin{array}{cccc} \underline{\boldsymbol{x}}'_0 & \underline{\boldsymbol{x}}'_1 & \ldots & \underline{\boldsymbol{x}}'_n \end{array} \right\} \tag{4.7}$$

is obtained from (4.6), as [108]

$$\underline{\boldsymbol{x}}'_{i-1} = \underline{\boldsymbol{x}}'_n \underline{\boldsymbol{\delta}}^*_i, \quad i = 2, \ldots, n. \tag{4.8}$$

This representation ensures that the imitated path is calculated in such a way that the relative transformations computed over the demonstrated path are respected throughout the generation of the imitated path from the initial pose $\underline{\boldsymbol{x}}'_0$ to the new goal $\underline{\boldsymbol{x}}'_n$.

The objective here is to construct a new demonstrated path starting from any new initial configuration $\underline{\boldsymbol{x}}''_0$ in the task space, which does not necessarily coincide with the pose $\underline{\boldsymbol{x}}'_0$. To this end, intermediate points in the task space are generated such that the demonstrated path $\underline{\boldsymbol{\mathcal{X}}}''$ blends into $\underline{\boldsymbol{\mathcal{X}}}''$, using the Screw linear interpolation (ScLERP) method [168]. Given the current end-effector pose $\underline{\boldsymbol{x}}_{\text{eff}}$ at any time step, and $\underline{\boldsymbol{x}}'_i \in \underline{\boldsymbol{\mathcal{X}}}'$ being a guiding pose on the imitated path, the reference end-effector pose is given by [108]

$$\underline{\boldsymbol{x}}_r(\tau) = \underline{\boldsymbol{x}}_{\text{eff}} \left(\underline{\boldsymbol{x}}^*_{\text{eff}} \underline{\boldsymbol{x}}'_i\right)^{\tau}, \tag{4.9}$$

with $\tau \in [0, 1]$, representing the interpolation sampling time that makes a discrete path linearly-scaled along the geodesic between any current pose of the end-effector and the guiding pose. The resulting trajectory, being a C_0[1] path, might not be feasible for the robot, especially when choosing the first guiding pose closer to $\underline{\boldsymbol{x}}'_1$.

4.3.2 ZoomOut Refinement

The *ZoomOut* algorithm, proposed in [136], is an iterative refinement method designed to enhance the accuracy of FM and convert them into precise point-to-point correspondences. The process begins with an initial functional map $\mathbf{C}_0$ obtained through the optimisation process described earlier. $\mathbf{C}_0$ provides a coarse correspondence between the shapes $\mathcal{S}_1$ and $\mathcal{S}_2$, capturing their broad structural similarities. It then processes this initial map and iteratively refines it by progressively increasing its resolution and accuracy. At each iteration, a larger set of basis functions is computed for both shapes, incorporating higher-frequency eigenfunctions of the Laplace-Beltrami operator[2]. Following the main steps of the refinement process are outlined.

- **Initialisation**: Begin with $k = k_0$ and $\mathbf{C}_0$ of size $k_0 \times k_0$.
- **Optimisation and Map Update**: For each iteration:

[1] In the context of path planning and complex analysis, C^0, C^1, and C^2 refer to the continuity of curves. C^0 indicates continuous curves without breaks, C^1 refers to continuous curves with a continuous first derivative, ensuring no sharp corners, and C^2 represents curves with both continuous first and second derivatives, providing smooth transitions and continuous curvature.

[2] The LB basis functions are ordered from low to high frequency, so increasing the map size effectively incorporates higher frequencies on the final result.

1. Compute the optimal point-wise correspondence matrix $\mathbf{\Pi}$ that minimises the Frobenius norm:

$$\mathbf{\Pi} = \arg\min_{\mathbf{\Pi}} \|\Pi\Phi_k^N \mathbf{C}_k^T - \Phi_k^M\|_F^2 \tag{4.10}$$

   Here, Φ_k^N and Φ_k^M are the matrices of the first k eigenfunctions for the source and target shapes, respectively, and $\mathbf{C}_k$ is the current functional map of dimensions $k \times k$.

2. Increase the basis size to $k + 1$ and update the functional map:

$$\mathbf{C}_{k+1} = (\Phi_{k+1}^M)^+ \mathbf{\Pi} \Phi_{k+1}^N \tag{4.11}$$

   where $(\Phi_{k+1}^M)^+$ denotes the pseudoinverse of the expanded target basis matrix.

- **Iteration**: Increment k and repeat the optimisation and map update steps until the basis size reaches $k_{\max}$ or until the changes in the functional map $\mathbf{C}_k$ between iterations fall below a certain threshold.

The *ZoomOut* method offers several advantages. It significantly improves the accuracy of the point-to-point correspondence by refining the functional map iteratively. It is robust to noise and initial misalignments, making it suitable for various applications in shape analysis and comparison. Moreover, *ZoomOut* is scalable, efficiently handling large datasets and complex shapes. For a detailed mathematical formulation and experimental results, refer to [136].

4.4 Task execution transfer

In this section, the method for transferring task operations between two instances of a scene with distinct category-level objects is outlined. An overview of the proposed pipeline is presented in Section 4.4.1. The approach to defining functions over objects during the demonstration stage is described in Section 4.4.2. Lastly, the transfer to new scenes and the method for demonstration are discussed in Section 4.4.4.

4.4.1 Pipeline

As illustrated in Fig. 4.2, the method consists of two stages:

1. **Demonstration stage**, where the demonstration scene $\mathbf{\Theta}^{\text{dem}}$ is acquired; and
2. **Task execution stage**, where the robot is presented with a novel scene and tasked with executing the demonstrated task using new objects.

The **Demonstration stage** begins with acquiring a complete scene $\mathbf{\Theta}^{\text{dem}}$ as described in (4.2). Full scene models are obtained by fusing point clouds from sequential scans of the robot's

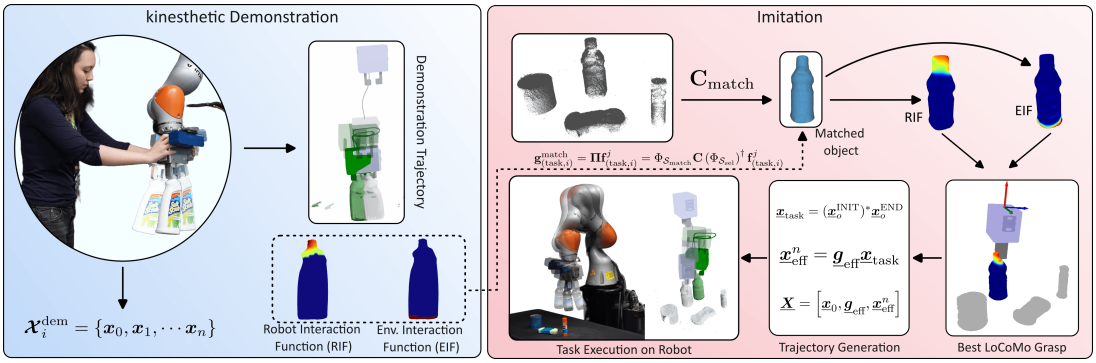


Figure 4.2: Pipeline of the proposed method consisting of demonstration and execution stage. Demonstrations are performed kinesthetically, after which the robot is immediately available for task execution. During the demonstration stage, various functions are created based on the placement of the robot's gripper and the motions it performs with the object. When the robot encounters a new scene, these functions can be transferred to similar objects, allowing the robot to extract analogous gripper poses. With a new set of waypoints and the demonstrated trajectory, tasks can be generalized to entirely new scenes.

workspace using a wrist-mounted 3D camera. These scans are then processed to segment the objects and the ground plane, storing each element $\mathcal{S}_j^{\text{dem}} \in \mathbf{\Theta}^{\text{dem}}$ as a mesh in the robot's memory. It is assumed that objects are sufficiently separated, allowing for density-based spatial clustering of applications with noise (DBSCAN) based clustering to be used for segmentation [54]. Meshes are generated from point clouds through the screened Poisson algorithm [98]. This step ensures that a geometric representation of the scene is captured, which is essential for accurate skill transfer.

A user kinesthetically demonstrates the task by guiding the robot through the required motions. This process involves physically moving the robot's end-effector to perform the desired task, resulting in the set of paths $\{\underline{\mathcal{X}}_i | i = 1, 2...\}$ with each path $\underline{\mathcal{X}}_i^{\text{dem}} = \{\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_n\}$ computed using the end-effector (EF) poses. These trajectories capture the sequence of poses that the end-effector assumes during each part of the task.

From $\mathbf{\Theta}^{\text{dem}}$ and $\underline{\mathcal{X}}_i^{\text{dem}}$, the interaction functions $\mathbf{f}^j_{(\text{task,i})} \in \mathcal{S}_j^{\text{dem}}$ are computed. These functions define the task-specific relationships between the robot, the environment, and the objects in the scene. They are crucial for ensuring that the demonstrated skill can be generalised to different instances of the scene with novel objects. Further details on the various interaction functions are provided in Section 4.4.2. Finally, from each element in the set of demonstrated $\underline{\mathcal{X}}_i^{\text{dem}}$ and functions $\mathbf{f}^j_{(\text{task,i})}$, the task operation representation $\mathcal{T}_i^{\text{dem}}$ is obtained as in (4.3).

The **Task Execution Stage** aims at obtaining the equivalent $\mathcal{T}_i'$ containing a new set of functions and trajectories to those obtained during the demonstration. This step begins with acquiring a complete scene, in this case $\mathbf{\Theta}'$, as described in (4.4). This involves similar processing steps to the demonstration stage, including ground plane segmentation, cluster extraction, and mesh construction, to prepare the object shapes from the scene cloud. The object meshes of the new scene will be stored as $\mathcal{S}_k' \in \mathbf{\Theta}'$. This preprocessing ensures that the geometric information of the novel scene is accurately captured.

Once the shapes are extracted, the FM framework transfers the functions $\mathbf{f}^{j}_{(\text{task},i)}$ such that

$$\mathbf{g}^{k}_{(\text{task},i)} = \mathbf{\Pi f}^{j}_{(\text{task},i)}, \tag{4.12}$$

where $\mathbf{g}^{k}_{(\text{task},i)} \in \mathcal{S}'_k$ is the transferred function in the object $\mathcal{S}'_k$ that corresponds to the best matching $\mathcal{S}^{\text{dem}}_i$ acquired during the demonstration phase and $\mathbf{Pi}$ is obtained from the last step in (4.10). When multiple candidate objects are present in the new scene, a matching score is used to select the most suitable one. This score assesses how well each candidate object matches the geometric structure of the demonstrated objects in which $\mathbf{f}^{j}_{(\text{task},i)}$ is described.

Once all functions are successfully mapped, new end-effector poses are calculated, and the path is determined based on the TSIA algorithm detailed in Section 4.3.1, with the final path $\underline{\boldsymbol{\chi}}'$ given by (4.7) and (4.8). As this is a ScLERP-based method, it ensures smooth and continuous motion trajectories by interpolating between poses using screw theory. This method maintains the geometric constraints observed during the demonstration, ensuring that the robot can replicate the demonstrated task accurately in the new scene.

4.4.2 Interaction functions

In this work, an object-centric approach is proposed for encoding demonstrated tasks, where interactions between objects, the robot, the environment, and other objects are represented as functions on the object's surface. While custom functions can be defined as needed, many tasks can be effectively represented using the following types of functions:

1. Robot interaction functions (RIF): Defines the contact regions between the robot's end-effector and the object.

2. Environment interaction functions (EIF): Defines the interactions between the object and a known part of the environment.

Thus, given a scene $\mathbf{\Theta}^{\text{dem}}$ and the demonstrated trajectory $\underline{\boldsymbol{\chi}}^{\text{dem}}_i$, the remaining element needed to obtain $\mathcal{T}^{\text{dem}}_i$ is the set of interaction functions $\mathbf{f}^{j}_{(\text{task},i)}$. Before defining each of the interaction functions, it is important to note that most scenes contain multiple objects, necessitating the creation of a function to select the object of interest. Let $\mathbf{c}_j(\mathcal{S}_j)$ be the centre of gravity of $\mathcal{S}_j \in \mathbf{\Theta}$ and $\mathbf{p}$ a point in space (i.e., the EF position). Then, the selected object is given by

$$\mathcal{S}_{\text{sel}}\left(\mathbf{\Theta}, \mathbf{p}\right) = \arg\min_{\mathcal{S}_j} \left\| \mathbf{c}_j(\mathcal{S}_j) - \mathbf{p} \right\|. \tag{4.13}$$

Below, it is detailed how each type of interaction function can be obtained.

4.4.2.1 Robot interaction function (RIF)

RIF define areas on the object's surface where the robot is allowed to interact (e.g., to grasp, poke, push, etc.). Let $\underline{\boldsymbol{x}}_n \in \underline{\boldsymbol{\mathcal{X}}}^{\text{dem}}$ be the dual quaternion representing the end position of the demonstrated path, with its translational part in vector form being $\mathbf{x}_n^t \in \mathbb{R}^3$. Using (4.13), the object of interest for robot interaction can be determined as $\mathcal{S}_{RI}^{\text{dem}} = \mathcal{S}_{\text{sel}}(\boldsymbol{\Theta}, \mathbf{x}_n^t)$. Then, using information from the gripper model and configuration, as well as $\mathcal{S}_{RI}^{\text{dem}}$, the N_f contact points between each of the robot's f fingers and the object can be identified. Using these contact points, $\mathbf{Q}^{(\text{contacts},f)} \in (\mathcal{S}_{RI}^{\text{dem}})^{N_f}$, the RIF can be defined as

$$\mathbf{f}_{(\text{task},i)}^{\text{RIF}} = \max\left(1 - \frac{\|\bar{\mathbf{Q}}^f - \mathbf{p}\|}{\lambda_D}, 0\right), \tag{4.14}$$

for all $\mathbf{p} \in \mathcal{S}_{RI}^{\text{dem}}$ and for all $\bar{\mathbf{Q}}^f$, where $\bar{\mathbf{Q}}^f$ is the average contact point for each $\mathbf{Q}^{(\text{contacts},f)}$ and λ_D is a distance threshold. Here, λ_D can be defined to encompass only the actual contact points with the fingers or slightly larger areas. In Fig. 4.3-(i), (4.14) is illustrated for both an object being pushed (left) and for an object being grasped (right).

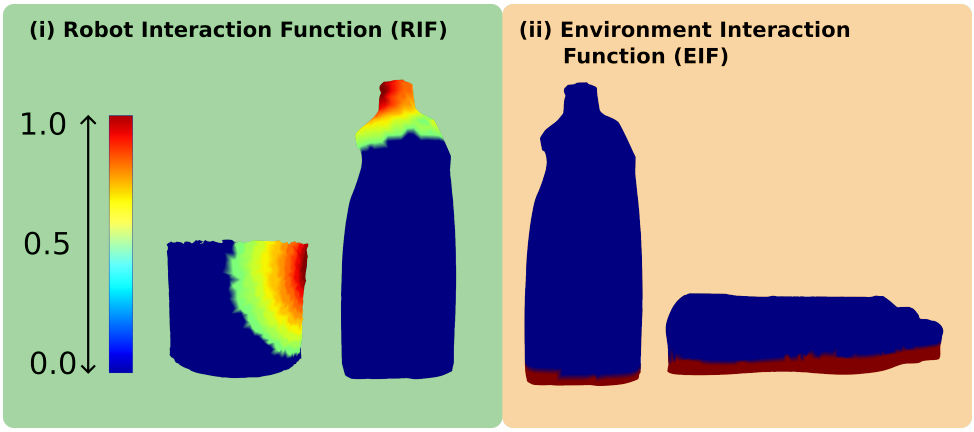


Figure 4.3: Illustration of both (i) RIF over the shape of an object, representing the areas where the robot gripper will make contact with the object and (ii) EIF, where the object is in contact with a pre-defined environment region (e.g. the table plane).

4.4.2.2 Environment interaction function (EIF)

EIF defines the area of an object that interacts with known elements of the environment. For this, environmental elements are simplified as a set of N_p planes $\mathbf{P} = \{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_{N_p}\}$, where each plane $\mathcal{P}_k$ is defined by a point $\mathbf{o}_k \in \mathbb{R}^3$ on the plane and a normal vector $\mathbf{n}_k \in \mathbb{R}^3$ passing through $\mathbf{o}_k$.

Given a previously selected object $\mathcal{S}_{RI}^{\text{dem}}$ and the relative transformation obtained from the demonstration, $\underline{\boldsymbol{x}}_{\text{rel}} = \underline{\boldsymbol{x}}_0^* \underline{\boldsymbol{x}}_n$, where $\underline{\boldsymbol{x}}_0, \underline{\boldsymbol{x}}_n \in \underline{\boldsymbol{\mathcal{X}}}_i^{\text{dem}}$ are the initial and final points of the trajectory,

$\mathcal{S}_{RI}^{\text{dem}}$ can be transformed to a new configuration (updating $\mathbf{\Theta}^{\text{dem}}$ accordingly). Once the pose of $\mathcal{S}_{RI}^{\text{dem}}$ is updated, the EIF can be defined as follows:

$$\mathbf{f}_{(\text{task},i)}^{\text{EIF}} = \begin{cases} 1, & \text{if} \quad \left\| (\mathbf{o}_k + \lambda_p \mathbf{n}_k) - \mathbf{p} \cdot \mathbf{n}_k \right\| \leq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.15}$$

for all $\mathbf{p} \in \mathcal{S}_{RI}^{\text{dem}}$, where λ_p is a distance threshold.

It should be noted that a separate function is defined for each plane the object is in contact with. Usually, three planes are sufficient to determine the full pose of an object. If fewer planes are provided, a goal must be specified. In Fig. 4.3-(ii), (4.15) is illustrated for objects in contact with the ground plane.

4.4.3 Object selection

When the FM correspondence matrix $\mathbf{C}$ is calculated, an energy function is minimised to describe features in the LBO bases $\mathbf{\Phi}_{\mathcal{S}_1}$ and $\mathbf{\Phi}_{\mathcal{S}_2}$. For two perfectly isometric shapes with perfectly aligned bases, each component corresponds directly to one another, making $\mathbf{C}$ a diagonal matrix. However, when the requirement for isometry is relaxed, and misalignments and noise are introduced, $\mathbf{C}$ becomes diagonally dominant rather than strictly diagonal[3]. The degree of diagonal dominance deteriorates as the shapes become less isometric.

Given this behaviour, a selected object, such as $\mathcal{S}_{RI}^{\text{dem}}$, can be matched against all objects in the observed scene, resulting in a set of maps $C = \{\mathbf{C}^1, \mathbf{C}^2, \ldots, \mathbf{C}^j, \ldots, \mathbf{C}_M\}$, where each $\mathbf{C}^j$ represents the map between $\mathcal{S}_{RI}^{\text{dem}}$ and $\mathcal{S}'_j \in \mathbf{\Theta}'$. The most diagonally dominant map in this set is considered the most similar to the target object and often belongs to the same class.

To classify and identify the most similar object, a score is used to determine which $\mathbf{C}_{\text{match}} \in C$ is the most diagonally dominant. For this purpose, a mask $\mathbf{M}$ is defined as a normalised weight matrix, where elements around the diagonal have higher values. The metric $R\left(\mathbf{C}^j\right)$ is then defined as

$$R\left(\mathbf{C}^j\right) = \left(\sum_{i,k} (1 - m_{ik}) |c_{ik}| \right) - \left(\sum_{i,k} m_{ik} |c_{ik}| \right) \tag{4.16}$$

where c_{ik} and m_{ik} are the elements of $\mathbf{C}^j$ and $\mathbf{M}$, respectively. This metric results in a score that indicates the degree of diagonal dominance. The object in the scene with the highest score is selected as the most similar object. The selection is given by

$$\mathcal{S}_{\text{match}}, \mathbf{C}_{\text{match}} = \underset{\mathcal{S}'_j \in \mathbf{\Theta}'}{\arg\max}\, R\left(\mathbf{C}^j\right) \tag{4.17}$$

[3] A strictly diagonal $\mathbf{C}$ matrix means there is a clear, one-to-one mapping between the features of the two shapes. When $\mathbf{C}$ is diagonally dominant, this clear correspondence becomes less distinct, making it more challenging to interpret or rely on the mapping. This ambiguity can lead to errors or less accurate mapping between shapes.

4.4.4 Task execution in novel scenes

Once the demonstrated task operations $\mathcal{T}_i^{\text{dem}}$, containing the demonstrated trajectories and associated functions, has been obtained, the robot can be immediately deployed in new scenarios, i.e, in a new environment with different objects in new configurations. The deployment stage begins when the robot acquires a new scene containing a new set of objects on which the task operations can be executed. After the scene is acquired, objects are segmented using DBSCAN [54], and meshes are generated, creating the new $\boldsymbol{\Theta}'$ as described in (4.4). For each function in each $\mathcal{T}_i^{\text{dem}}$, the associated object is used as the source shape within the FM framework. Specifically, for a given function $\mathbf{f}_{(\text{task},i)} \in \mathcal{S}_{RI}^{\text{dem}}$, the FM is computed for every object, generating the set of maps C.

Following the process outlined in Section 4.4.3, the best matching object $\mathcal{S}_{\text{match}}$ is identified. While this process could be repeated for each function being transferred, for simplicity, it is assumed that all operations executed by the robot pertain to the same object, making $\mathcal{S}_{\text{match}}$ the selected object for all function transfers. Consequently, using the corresponding map $\mathbf{C}_{\text{match}}$, the functions over the best matching shape can be obtained as shown in (4.12). Thus,

$$\mathbf{g}_{(\text{task},i)}^{\text{match}} = \boldsymbol{\Pi}\mathbf{f}_{(\text{task},i)}^{j} = \Phi_{\mathcal{S}_{\text{match}}} \mathbf{C} \left(\Phi_{\mathcal{S}_{RI}^{\text{dem}}}\right)^{\dagger}. \tag{4.18}$$

where $\Phi_{\mathcal{S}_{\text{match}}}$ denotes the Laplacian bases on the matched shape, $\Phi_{\mathcal{S}_{RI}^{\text{dem}}}$ represents the bases on the demonstrated object shape, and † indicates the matrix pseudoinverse. This process is repeated for all functions in the demonstrated set. For the proposed method, typically two different scenarios can occur[4]:

(i) The full demonstration contains one RIF function with one or two trajectories : one for the approach and another for any movement performed after contact.

(ii) The demonstration contains two task operations: one trajectory with an RIF referring to the grasping task and one EIF referring to some specific task execution.

In the first scenario, a simplified version of the algorithm is applied where no grasping is performed. This is useful for tasks such as pushing or pressing buttons. Here, the robot's gripper is closed, and the relative distance between the gripper and the RIF during the demonstration is used to set the desired gripper pose with respect to the new object's RIF. By setting this as the goal configuration for the TSIA algorithm (see Section 4.3.1), the robot can interpolate a new trajectory from its current position and execute the task.

Next, the process for scenario (ii), involving two functions, is outlined. When two functions are demonstrated, typically one RIF, where the gripper's contact regions are set, and one EIF,

[4]In theory, more complex demonstrations could involve tasks with several trajectories, functions, and even interactions between objects. However, as a proof of concept, this work focuses on operations containing one or two functions.

where the object's final pose is defined in relation to the environment, the process begins by selecting the region defined by the RIF and planning a set of grasps over it. To achieve this, a point cloud created from the vertices in the mesh whose values for the function $\mathbf{g}^{\text{RIF}}_{(\text{task},i)}$ are $\mathbf{g}^{\text{RIF}}_{(\text{task},i)} > \delta$ for some threshold δ is defined. With this cloud, a set of grasp poses $\mathcal{G}$ is generated by LoCoMo grasp planner (see Section 3.3.2 for further details). Here, $\mathcal{G}$ is an ordered set in which grasps are ranked by quality. However, this set must be further filtered to enable task execution. First, grasps that result in collisions with the environment or are kinematically unfeasible are removed. Then $\mathcal{G}$ is further filtered according to the approach direction established during the demonstration stage. This is done because task execution often requires certain grasp wrenches to be applied in specific directions.

Specifically, a frame is attached to the centre of the RIF region during the demonstration—this is referred to as the *RIF-object frame*. At the end of the demonstrated trajectory, the approach axis (in this case, the x-axis) from $\underline{\boldsymbol{x}}^{\text{dem}}_n \in \underline{\boldsymbol{X}}^{\text{dem}}$ is extracted and transformed into the RIF-object frame. This direction is represented by the vector $\mathbf{a}^{\text{dem}}_o$. Using the correspondence between the shapes, the equivalent RIF-object frame in the selected shape $\mathcal{S}_{\text{match}}$ is identified, along with the corresponding approach direction $\mathbf{a}_o$. Only grasp poses $\underline{\boldsymbol{g}}^o_{\text{eff}} \in \mathcal{G}_o$, defined in the object's frame, whose approach vector lies within a cone formed by $\mathbf{a}_o$ rotating over a small half-angle θ (empirically determined), are retained. In other words, a grasp is considered valid only if it satisfies the following inequality,

$$\cos(\theta) \leq \mathbf{g}^o_{\text{eff}} \cdot \mathbf{a}_o, \tag{4.19}$$

where $\mathbf{g}^o_{\text{eff}}$ is the approach vector extracted from $\underline{\boldsymbol{g}}^o_{\text{eff}}$ and the $\mathbf{g}^o_{\text{eff}}$, $\mathbf{a}^{\text{dem}}_o$and $\mathbf{a}_o$ vectors are unit length. The top grasp, $\underline{\boldsymbol{g}}^{\text{top}}$ from the remaining grasp candidates, which is both feasible and aligns with the task wrench in the world frame, is selected as the goal for the transferred operation.

The second step in the transferring process involves obtaining the poses related to the EIF. Here, the function represents the regions where the object interacts with the environment. A point cloud is thus generated for the demonstrated object shape $\mathcal{S}^{\text{dem}}_{RI}$ where $\mathbf{S}^{\text{dem}}_{RI} = \{\mathbf{p} | \forall \mathbf{p} : \mathbf{f}^{\text{EIF}}_{(\text{task},i)}(\mathbf{p}) = 1\}$, and similarly, a cloud is generated for the selected shape $\mathcal{S}_{\text{match}}$, such that $\mathbf{S}_{\text{match}} = \{\mathbf{p} | \forall \mathbf{p} : \mathbf{g}^{\text{EIF}}_{(\text{task},i)}(\mathbf{p}) = 1\}$. By recovering a point-to-point map $\mathbf{T}_p$ from the FM representation, the optimal rotation quaternion $\mathbf{x}_R$ that transforms the cloud $\mathbf{S}_{\text{match}}$ to $\mathbf{S}^{\text{dem}}_{RI}$ is obtained. For improved accuracy, ICP is used to refine the transformation after the initial alignment. With $\mathbf{x}_R$, the final object configuration is defined. In this work, the environment functions considered are related to the plane on which the object rests, leaving the degrees of freedom concerning the x and y position on the table unspecified. For simplicity, these are manually defined, and the final object pose is $\underline{\boldsymbol{x}}^{\text{END}}_o$. Given the initial object pose, $\underline{\boldsymbol{x}}^{\text{INIT}}_o$, the transformation the object will undergo from it's initial pose to the final one is

$$\underline{\boldsymbol{x}}_{\text{task}} = (\underline{\boldsymbol{x}}^{\text{INIT}}_o)^* \underline{\boldsymbol{x}}^{\text{END}}_o. \tag{4.20}$$

As the object is firmly held by the gripper, this transformation, applied to the gripper's grasping pose in the world frame, can be used to obtain the robot's final pose. That is,

$$\underline{\boldsymbol{x}}^{n}_{\text{eff}} = \underline{\boldsymbol{g}}_{\text{eff}}\underline{\boldsymbol{x}}_{\text{task}}. \tag{4.21}$$

Finally, with the new set of robot poses derived from the transferred functions, the TSIA algorithm (Section 4.3.1) is used to create new trajectories based on the set of demonstrated trajectories $\underline{\boldsymbol{X}}^{\text{dem}}$. Specifically, a list of new end goals $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_0, \underline{\boldsymbol{g}}_{\text{eff}}, \underline{\boldsymbol{x}}^{n}_{\text{eff}}\right]$ is created, and for each consecutive pair in $\underline{\boldsymbol{X}}$ and the demonstrated trajectory $\mathcal{X}_i^{\text{dem}}$, a novel imitated path is defined, $\underline{\boldsymbol{X}}_i''$ as per (4.9). With the functions and trajectories, the task operations $\mathcal{T}_i'$ for the new scene are obtained. By following these new paths, the robot can execute the task operations in the novel environment. Particularly, grasping motion is activated between the first and second trajectory.

4.5 Experiments

In this section, the proposed approach is evaluated through both virtual and real experiments. A variety of scenes and robot demonstrations were initially collected to assess the method's effectiveness, with different skill operations being transferred to demonstrate its viability. The evaluation included both simple tasks, where single-stage operations were performed, and more complex tasks involving multiple trajectories. Object selection was also validated through tests in scenes with multiple available objects. Finally, a proof-of-concept task, fully demonstrated on the real robot, is presented.

4.5.1 Robotic setup

The robotic setup (shown in Figure 4.4 comprised a 7-axis KUKA iiwa robot equipped with a Schunk PG 70 gripper. A depth camera (Ensenso N35) was mounted on the robot's wrist and calibrated with respect to the robot's base. The point-cloud processing and controller software were developed in C++ and executed on a computer running Windows with an i7 4-core CPU at 2.70 GHz. Object clouds are formed by stitching multiple ground-plane segmented clouds from different views. The task demonstration is carried out by an user who physically guided the robot (under gravity compensation) to perform the task where joint configurations were recorded. From those a list of end effector poses was obtained using the robot's forward kinematics model. It is worth noting that all demonstrations have been made on the physical robotic platform. All objects selected for experiments are shown in Figure 4.5, with the objects marked in green (**D1**-**D8**) being the objects used for demonstrations and the ones in yellow (**T1**-**T9**) used for the transfer. Note here that the challenge in the selected dataset lies in the variability of shape across instances within each category. Even small differences in diameter or height can significantly affect grasp success. Additionally, local geometric differences, such as the presence of ridges,

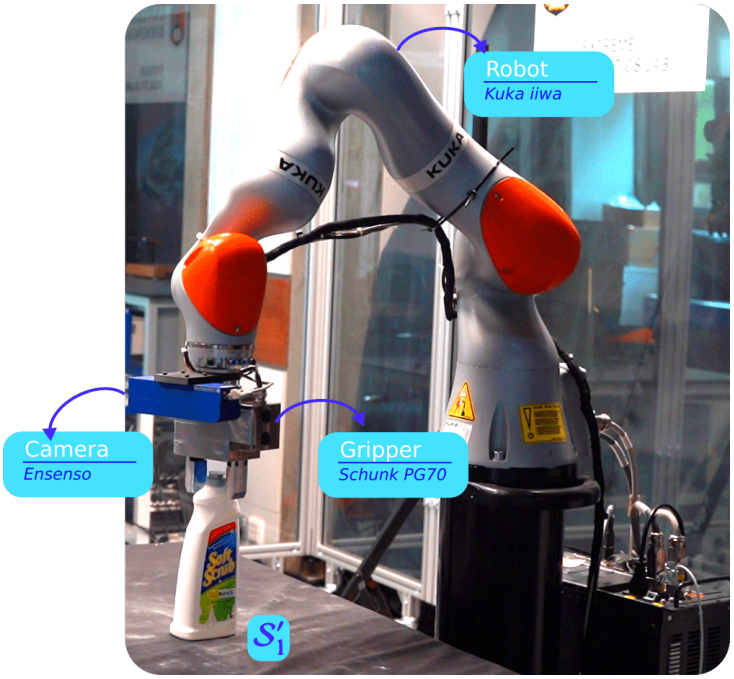


Figure 4.4: Robotic setup used for experiments.

necks, introduce further complexity in achieving generalisation. Lastly, position and orientation variability when the objects are places in the workspace provide additional challenges.

For the proposed algorithm, the FM implementation was adapted from the code provided in [136, 160], and the DQRobotics library [7] is used for dual-quaternion operations. The TSIA algorithm was provided by the authors of [108]. Python code was developed for both virtual and real experiments, with the generated trajectories executed on the robot during real-world experiments. Additionally, mesh creation was performed using the Screened Poisson method [98].

4.5.2 Virtual experiments

Two types of experiments are presented here: (i) single-operation task execution, where the robot performs a one single interaction with the object, such as pressing a button, pushing a mug, pressing the lid of a soap dispenser, etc and (ii) multiple-operation task execution, where the robot executes a task-aware grasp, aligns the task wrench, completes the task, and then returns the object to the desired configuration. As these experiments do not have variability between trials, each was repeated only once for each object. In total, 8 experiments were performed.

4.5.2.1 Single operation tasks

In Figure 4.6, the robot performs the task of pressing a button (**D4** for the demonstration and **T4** for the transfer), while Figure 4.7 depicts the task of pushing a can (**D2** for the demonstration



Figure 4.5: Dataset of objects used in the experiments. (**D1–D8**) user for task demonstrations; and, (**T1–T9**) used for task transfer.

and **T2** for the transfer). In both figures, the top row shows the demonstrated scene, and the bottom row illustrates the transferred execution, where the robot can perform a similar task with different objects in varying configurations.

From these experiments, it can be observed that the robot successfully identifies correspondences between different objects of the same class, and the imitated trajectories maintain the geometric constraints of the demonstrated path, ensuring successful task execution. In the button press task (Figure 4.6), note that the robot moves downward in a similar manner, even though the initial and target goals are not perfectly aligned. Furthermore, as it will be discussed in Section 4.5.3, the method effectively segments different elements from the scene when multiple objects are present. In this case, the cable attached to the button is segmented in both the demonstration and target scenes. The correct object in the demonstration scene is selected using (4.13) while the classification metric from (4.16) is used to select the correct object in the target scene. Similarly, in Figure 4.7, the robot maintains the demonstrated trajectory during the execution of the task.

4.5.2.2 Multiple-operations tasks

For multiple-operation tasks, five experiments were conducted using a diverse set of shapes and tasks to demonstrate the versatility of the proposed approach:

- Figure 4.8 illustrates a **repositioning** task involving "bottle" objects. In the demonstration, the bottle is initially lying on its side, then lifted and repositioned on its side again. In the imitation, the object is upright and repositioned to lie on its side.

- Figure 4.9 shows a small cylindrical object used to **draw** in the air. The task in this case is to draw a square.

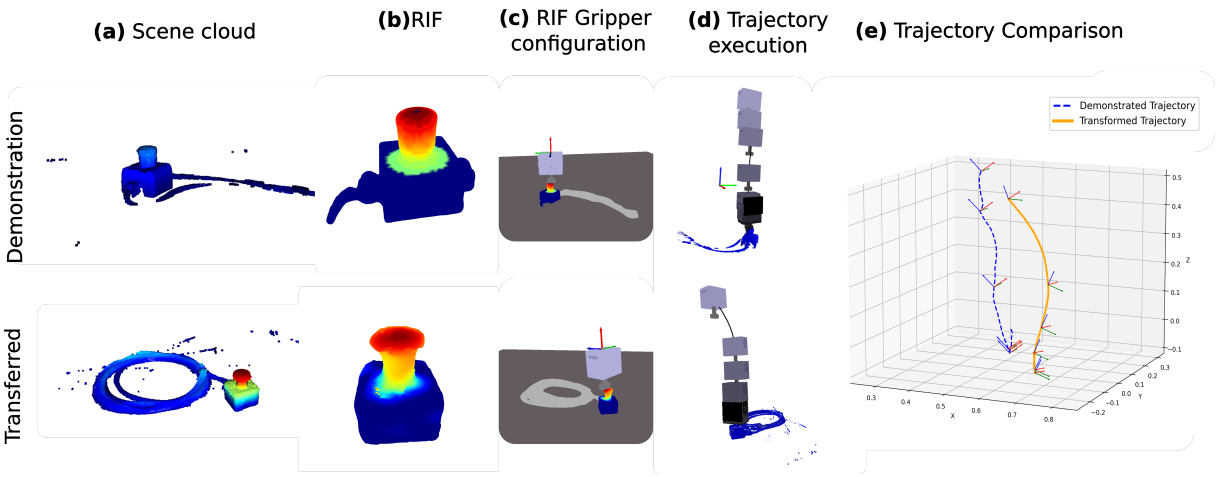


Figure 4.6: Button press task with the demonstrated object **D4** and the transferred object **T4** refer to Figure 4.5).The top row shows the demonstrated scene, while the bottom row shows the transferred execution.. The first column displays the observed point clouds of the scenes. The second column highlights the function over the object, which, for single-operation tasks, describes the region where the robot interacts with it. For both experiments, the gripper was fully closed throughout. The third column (**c**) shows the demonstrated and imitated pose of the end-effector, while column **(d)** depicts the gripper executing the trajectory. Finally, the last column presents the two paths: the demonstrated trajectory (in the blue dashed line) and the imitated trajectory (in orange)

- Figure 4.10 depicts a "tube container" that is grasped on the side and then tilted to apply **glue** onto a surface.
- Figure 4.11 demonstrates a **wiping** motion using a "whiteboard eraser," with the robot moving in uneven circles after it grasps the object.
- Figure 4.12 displays the "paintbrush" objects being grasped, followed by "painting" motions in the air, where the brush moves up and down.

Figures 4.8 through 4.12 depict each experiment similarly to the previous section.

It can be observed that the proposed method enables generalisation across a diverse set of shapes and tasks of varying complexities. The experiments with the "bottles," "tube container," and "whiteboard eraser" demonstrate that the FM approach effectively allows transfer between different shapes of the same class, including for objects that have suffered deformation, such as the tube, and differ more from the template. Additionally, experiments involving repositioning tasks highlight the strength of the method in describing object and end-effector configuration in space using the RIF and EIF functions. Notably, in Figure 4.10, it is evident that by leveraging only the map between shapes and the EIF transfer, an optimal alignment of the tube's cap with the table is achieved, enabling task execution. Similarly, in the bottle repositioning task shown in Figure 4.8, it is clear that even when the initial configuration of the object differs from that in the demonstration, the task can be successfully executed, with the resulting trajectory

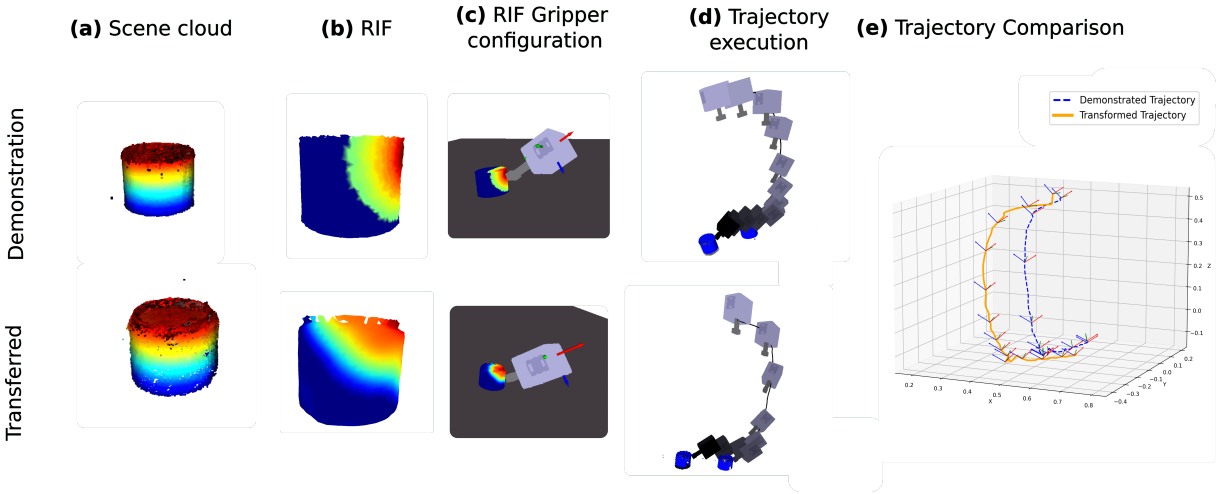


Figure 4.7: The robot pushing a can where the demonstrated object is **D2** and the transferred object is **T2**. The top row shows the demonstrated scene, while the bottom row shows the transferred execution. Button press task with the demonstrated object **D4** and the transferred object **T4** refer to Figure 4.5). The first column displays the observed point clouds of the scenes. The second column highlights the function over the object, which, for single-operation tasks, describes the region where the robot interacts with it. For both experiments, the gripper was fully closed throughout. The third column (**c**) shows the demonstrated and imitated pose of the end-effector, while column **(d)** depicts the gripper executing the trajectory. Finally, the last column presents the two paths: the demonstrated trajectory (in the blue dashed line) and the imitated trajectory (in orange)

maintaining the same geometric constraints as the demonstration—specifically, the sharp edge between grasping and executing.

Furthermore, this method also allows for complex trajectories within a task. This was demonstrated in the **drawing** and **wiping** tasks, where the robot was shown how to draw a square in the air and perform uneven circular movements, respectively. Figures 4.9 (for drawing) and 4.11 (circular motions) illustrate how these tasks are successfully transferred, regardless of complexity.

Additionally, it is notable that the trajectories, and therefore task execution, depend on the alignment of the approach vector with the task wrench and, more importantly, with the trajectory. During grasp computation, only those grasps that satisfy the inequality in (4.19), with a half-angle $\theta = 15°$ are considered. The effects of deviating from the demonstrated wrench can be observed in Figures 4.9 and 4.12. Despite the grasps meeting the criteria outlined in (4.19), the best grasp still had a slight angle relative to the demonstrated gripper pose. In both cases, while the misalignment was within the acceptable bounds and the task was successfully transferred, the imitated trajectory (shown in column **(f)** in the figures) is slightly angled. This misalignment could become more pronounced if approach wrenches were not considered, leading to a loss of the task's intended effectiveness.

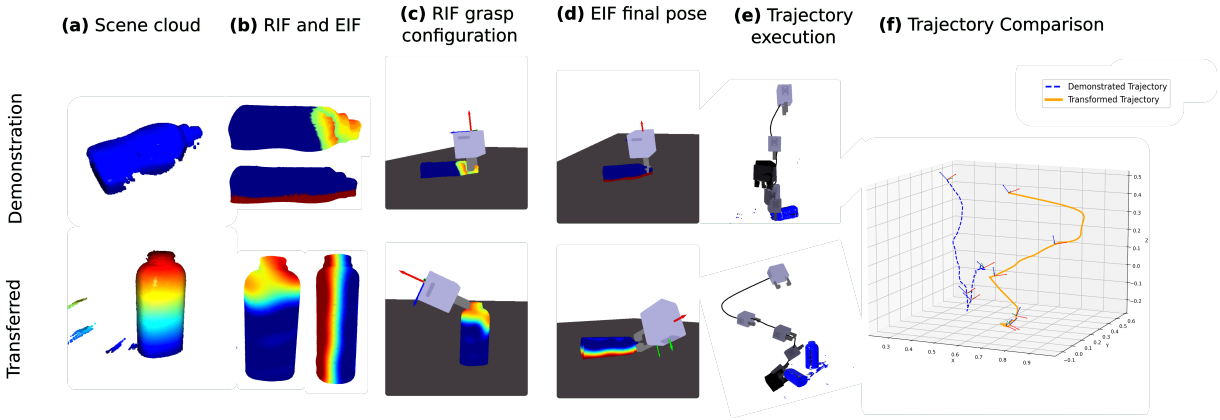


Figure 4.8: Repositioning task from "bottle" object **D6** in Figure 4.5 to **T5**. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side.

4.5.3 Multiple object scenes

This section showcases the selection method for identifying the best object match in scenes where multiple objects are present. Figure 4.13 illustrates the results of the repositioning experiment involving a bottle, conducted in a scene with several available objects. In **(a)**, the scene is depicted. In **(b)**, the colour-coded point-to-point map between each pair of objects is shown, where similar colours indicate corresponding regions on the objects. Additionally, the functional map and the metric R are displayed. In **(c)**, the obtained end-effector poses are presented, and in **(d)**, the executed trajectory is shown.

The main observation here is the mapping between shapes. It is important to note that although a reasonably good quality map can be found between the bottle and the cup (Figure 4.13-b top row), the map between two bottles (Figure 4.13-b bottom row) is more diagonal dominant, leading to a better score being calculated, which effectively results in the correct object being selected. For successful task execution, collision checks were employed. However, for future work, obstacle avoidance strategies will be integrated into the overall pipeline to enable more effective obstacle avoidance during the trajectory interpolation step.

4.5.4 Real-world transfer

In the previous sections, the proposed method was validated by demonstrating the transfer of functions extracted from real-world scenes. These functions were used to generate gripper poses and robot trajectories suitable for task execution. However, these trajectories were only simulated to showcase the method's viability and were not applied to a real robotic system. In this section, the full execution of tasks using a real robot is presented, demonstrating the practical application of the method.

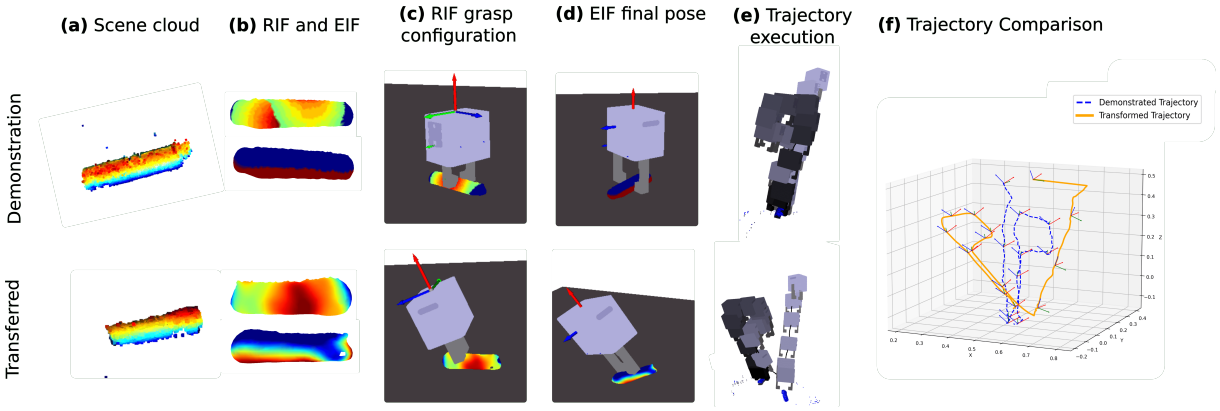


Figure 4.9: Drawing task with a cylindrical object (**D8** for the the demonstration and **T9** for the transfer. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side.

Figure 4.14 illustrates the task execution performed by the robot. The demonstrated task involves picking up a bottle (**D5** in Figure 4.5) by its lid, **mixing** the contents inside by moving in a circular motion (stirring motion), and placing it back in an upright position at an arbitrarily selected location in the table. In the top row of the figure, the demonstrated functions, as well as the demonstrated trajectory (shown as a dashed blue line in the plot), are presented. The bottom row depicts the transferred task operation, displaying the RIF and EIF functions, along with images captured at different moments during the robot's execution for three different types of bottles (**D6, T6** and **T7** in Figure 4.5). The executed trajectories are also plotted alongside the demonstrated trajectory.

An important observation is that all transferred trajectories perform a similar rotating motion, regardless of the goal position. This consistency is achieved through the accurate transfer of the RIF and EIF functions and the alignment of the gripper's approach direction, which ensures that the robot can effectively determine meaningful initial, grasping, and final waypoints for the new trajectory, resulting in successful task execution across different scenarios. Additionally the robot attempted to place the bottles in an upright position at the end of each task. However, due to a angular approach direction when placing the "small water bottle" that one tumbled in the end of the task execution. Moreover, here it's shown the method exhibits strong generalisation capabilities, allowing varying initial and final positions of the object, as well as different objects within the same class but with distinct shapes.

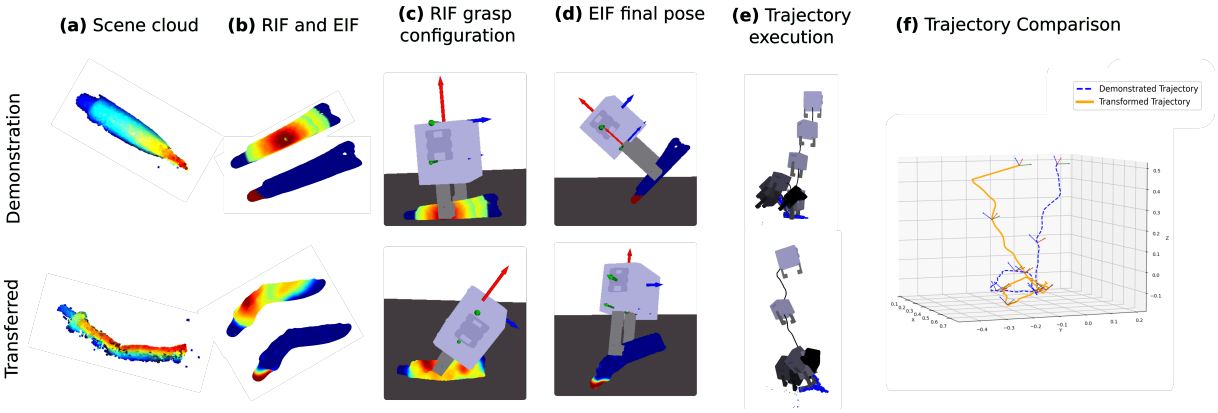


Figure 4.10: Application task with "tube container" objects (**D3** to **T3** in Figure 4.5). In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side.

4.6 Summary

In this chapter, a method was introduced to extend the grasp-transfer algorithm from Chapter 3 to encompass full task-operation transfer. Through a single kinesthetic demonstration, the method defines functions over an object's surface and uses the FM framework to transfer these functions between different instances of an object within the same class. These functions determine the interaction regions between the object and both the robot and its environment. The approach also extracts poses relative to the grasped regions, establishing a template approach vector for the gripper during task execution. By transferring these functions to a new object, the interaction regions are aligned, enabling the identification of gripper configurations that remain consistent with the demonstrated task in a new environment. Moreover, the TSIA method, based on ScLerp interpolation, was employed to generate new trajectories that preserve the geometric characteristics of the demonstration, facilitating successful task execution. The method was validated through virtual experiments, which demonstrated the viability of function and trajectory transfers to new object instances.

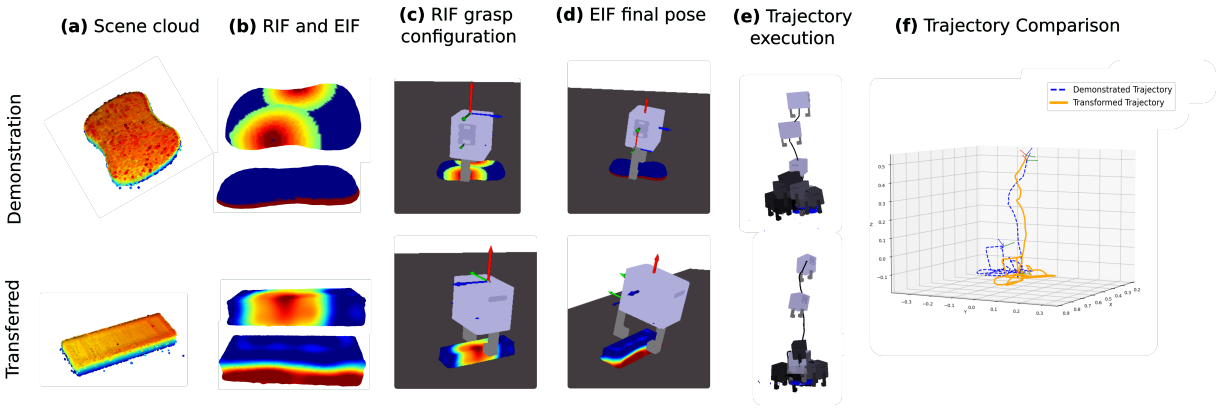


Figure 4.11: Wiping task with "whiteboard eraser" objects (**D1** to **T1** in Figure 4.5). In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side.

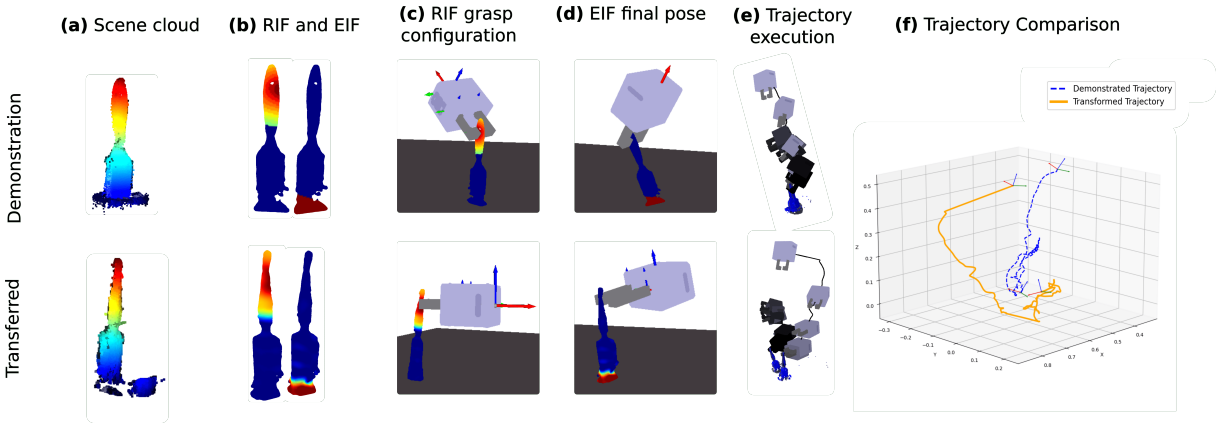


Figure 4.12: Painting task with the "paintbrush" in **D7** for the demonstration and **T8 for transfer** in Figure 4.5. In the first column of the images, the scene point clouds are shown. The second column displays the functions, both the RIF and EIF, obtained from the demonstration and the transfer. Columns **(c)** and **(d)** show the gripper poses extracted from the functions and transformations. Column **(e)** depicts the gripper performing both the demonstrated and imitated trajectories, and in the last column, both trajectories are shown side by side.

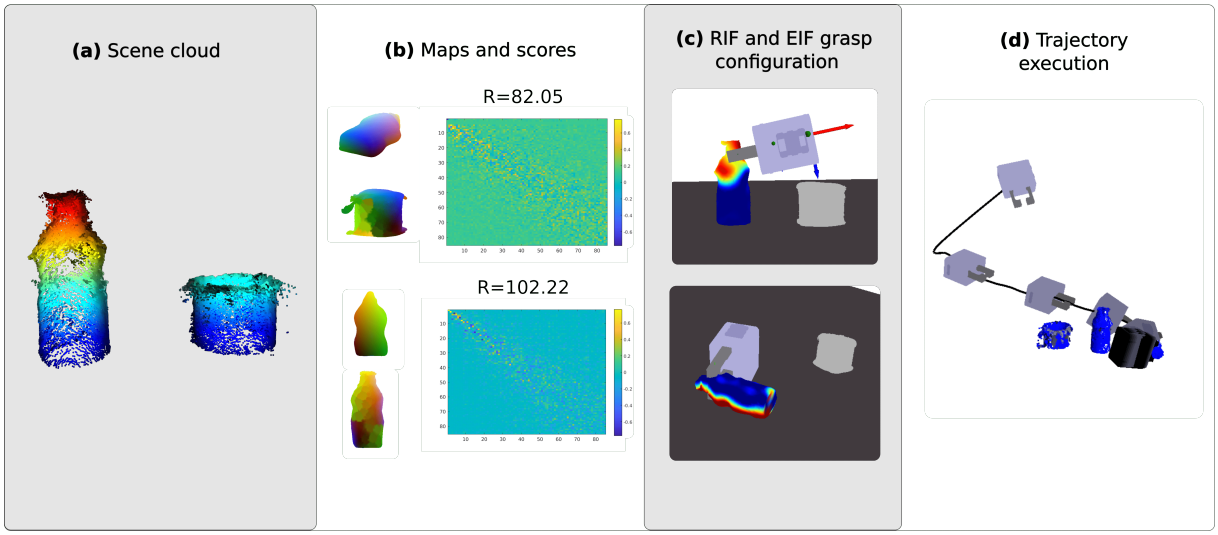


Figure 4.13: Repositioning task with multiple objects in the scene. Demonstration is done with bottle **D6** and transfer scene is created with object **T7**. In (a) The scene with multiple objects, (b) Colour-coded point-to-point map and functional map with metric R, (c) Obtained end-effector poses, (d) Executed trajectory. The diagonal map between two bottles results in better object selection compared to the bottle and cup.

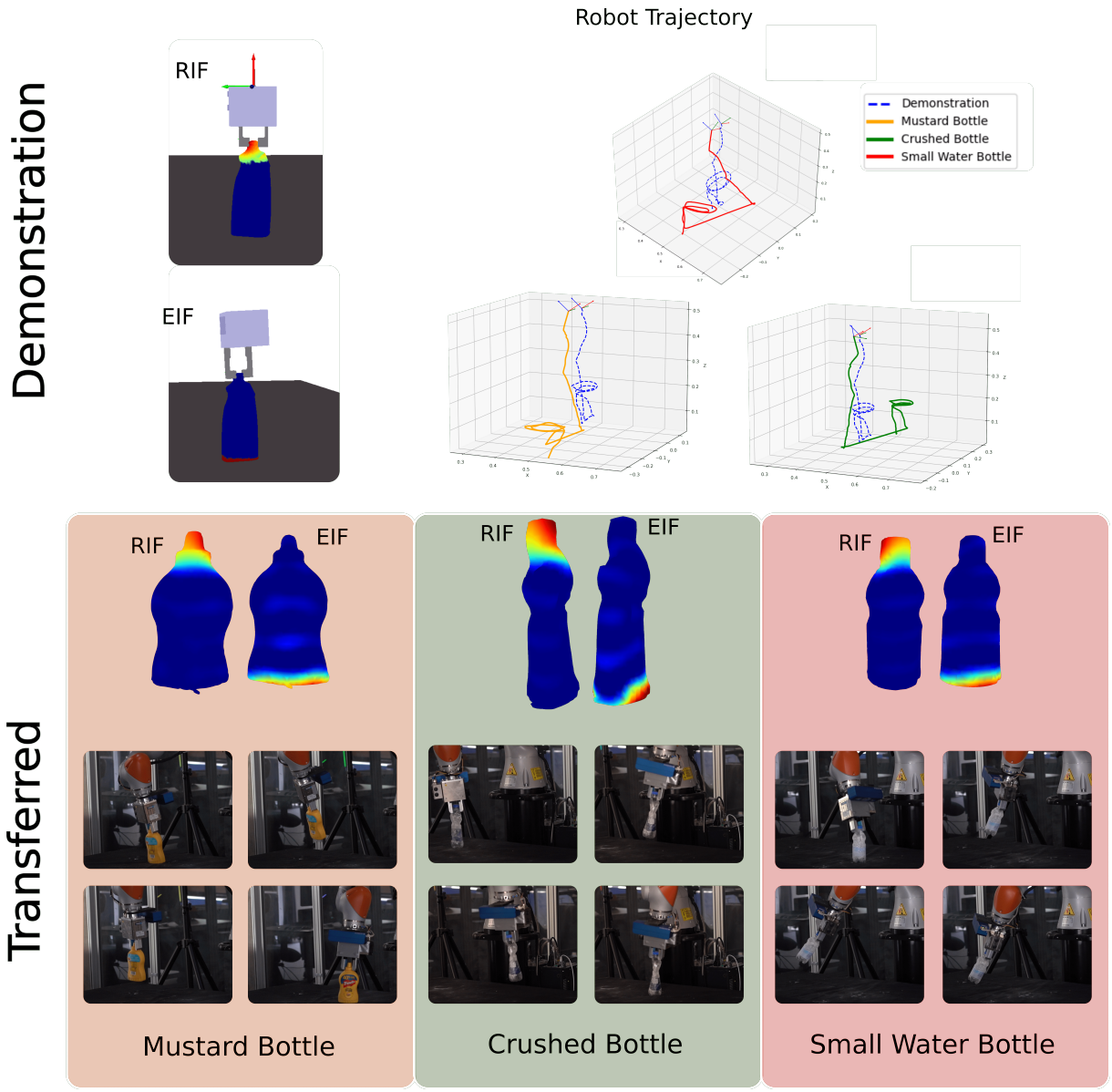


Figure 4.14: Real-world bottle shaking task execution by the robot. In the top row the demonstrated functions and trajectories are shown. In the trajectory plot the dashed blue line displays the demonstrated one and the others the imitated trajectory, in orange for the mustard bottle (object **D6** in Figure 4.5), green for the crushed water bottle (**T6**) and in red the small water bottle (**T7**). In the bottom row the transferred task operations are displayed, including RIF and EIF functions and snapshots of the robot executing the task with three different bottles.

Chapter 5

Task-aware grasping of partially observed objects

Preliminary remarks

The research in this chapter was published in the following paper:

- **C. de Farias,** B. Tamadazte, M. Adjigble, R. Stolkin and N. Marturi, "Task-Informed Grasping of Partially Observed Objects," in IEEE Robotics and Automation Letters, vol. 9, no. 10, pp. 8394-8401, Oct. 2024, doi: 10.1109/LRA.2024.3445633.

Note: Certain sections of this chapter, such as problem descriptions, specific equations, variable descriptions, interpretations of results, and figure/table captions, have been adapted from [59].

5.1 Introduction

In the previous chapters, Functional Maps (FM) have been used to facilitate the transfer of information between objects. Chapter 3, demonstrated the use of FM in grasp transfer. This approach highlighted the ability of FM to handle the complexities of grasp information transfer, even when the objects undergo deformation. Chapter 4, further extended the concept, showing that functions describing various properties, such as grasp regions, motion trajectories etc. could be transferred between fully observed shapes of objects within the same class. This highlighted the versatility of FM in skill transfer scenarios, where understanding of contact regions and motion vectors is crucial.

However, both applications relied on having the full (or near full) geometry of the objects, which presents significant challenges in practical applications. Obtaining full models of objects is often expensive, requiring a setup with numerous cameras or that many observations of the object are made. Additionally, in real-world environments, objects are frequently occluded, making it challenging to achieve full visibility.

To overcome these limitations, this chapter extends the FM framework by integrating shape estimation techniques with a focus on Gaussian process regression (GPR). GPR is used both to incorporate smooth, task-aware functions on object surfaces and to provide uncertainty-aware shape estimates. This approach ensures successful grasping even with partially observed objects, thereby overcoming the challenges posed by incomplete geometric information.

5.2 Problem statement

The primary focus of this chapter is to explore ways to transfer task-specific grasp cost functions between objects belonging to the same object category. Particularly, between a database of common types of objects to partially observed similar objects. Let $\mathbf{P}_{db} = \{\mathbf{p}_i^{db} | i = 1, 2, \dots M\}$ denote a point cloud of size M representing an object in the database and let $\mathcal{S}_{db}$ represent its corresponding Riemannian manifold. Then task cost functions, $\boldsymbol{f}_i : \mathcal{S}_{db} \rightarrow \mathbb{R}$, which depict the graspable regions of the object, can be defined. Let $\mathbf{P}_{\text{target}} = \{\mathbf{p}_i^{\text{target}} | i = 1, 2, \dots N\}$ represent an N-sized partial point cloud representing a similar object with $\mathcal{S}_{\text{target}}$ as its manifold. The problem is then formulated as obtaining the function $\boldsymbol{g} : \mathcal{S}_{\text{target}} \rightarrow \mathbb{R}$ by transferring this function between $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{target}}$. That is,

$$\boldsymbol{g} = \boldsymbol{\Pi} \boldsymbol{f}_i \tag{5.1}$$

where, $\boldsymbol{\Pi}$ is a linear map between points in the manifolds. Once the function is transferred, the set of grasps $\mathcal{G}(\mathbf{P}_{\text{task}})$ over the observed object can be obtained. Here, the cloud $\mathbf{P}_{\text{task}}$ is obtained from the surface $\mathcal{S}_{\text{task}} \subset \mathcal{S}_{\text{target}}$ defined as $\mathcal{S}_{\text{task}} \triangleq \boldsymbol{g}(\mathbf{x}) > \varepsilon_{\text{task}}$ with $\mathbf{x} \in \mathcal{S}_{\text{target}}$, and $\varepsilon_{\text{task}}$ a threshold value. To address the problem in (5.1), it is essential to obtain the map $\boldsymbol{\Pi}$. As stated earlier, the FM framework is employed for this purpose.

5.3 Background

This section outlines the background concepts for the proposed method. Specifically the GPR tools for shape estimation and task function estimation [157, 196]. The reader is referred to those Sections 3.3.1.2 and 4.3.2 for an overview of the FM methods.

5.3.1 Gaussian process regression

Both to define functions on a surface and estimate shapes, this work employs GPR as its backbone. GPR is a non-parametric, probabilistic model widely used for regression tasks due to its flexibility and robustness. It models distributions over functions by leveraging the properties of Gaussian process (GPs). A GP is a collection of random variables, which have a joint

Gaussian distribution. Formally, a GP is defined as

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}') \right), \tag{5.2}$$

with $m(\mathbf{x})$ being the mean function and $k(\mathbf{x}, \mathbf{x}')$ the covariance function (or kernel) which defines the covariance between the values of the function at any two points $\mathbf{x}$ and $\mathbf{x}'$.

Mathematical formulation Consider a training dataset, $[\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}]$ where $\boldsymbol{\mathcal{X}} = [\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_n]$ with $\mathbf{x}_i \in \mathbb{R}^d$ represents the n d-dimensional input points and $\boldsymbol{\mathcal{Y}} = [y_1, y_2, \dots, y_n]$ with $y \in \mathbb{R}$ the correspondent observations. Then, expanding (5.2), the GP can be described by its mean and kernel functions. Considering the GP prior, which assumes that the function values at the training points follow a joint Gaussian distribution and that the mean function $m(\mathbf{x}) = 0$, the joint distribution of observed values $\boldsymbol{\mathcal{Y}}$ and predictions $f_*(\mathbf{x}')$ at a test point $\mathbf{x}'$ are

$$\begin{bmatrix} \boldsymbol{\mathcal{Y}} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{X}}) + \sigma_n^2 \mathbf{I} & k(\boldsymbol{\mathcal{X}}, \mathbf{x}') \\ k(\mathbf{x}', \boldsymbol{\mathcal{X}}) & k(\mathbf{x}', \mathbf{x}') \end{bmatrix} \right) \tag{5.3}$$

with σ_n being the variance of the observation noise.

The posterior distribution over the function values at the test points, given the training data, is also Gaussian, given by

$$Pr(f_* | \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}, \mathbf{x}') \sim \mathcal{N}(\bar{h}(\mathbf{x}'), \mathbb{V}(\mathbf{x}')) \tag{5.4}$$

where the mean and covariance are given by

$$\begin{aligned} \bar{h}(\mathbf{x}') &= k(\boldsymbol{\mathcal{X}}, \mathbf{x}')^T \boldsymbol{\Sigma} \boldsymbol{\mathcal{Y}} \\ \mathbb{V}(\mathbf{x}') &= k(\mathbf{x}', \mathbf{x}') - k(\mathbf{x}', \mathbf{x})^T \boldsymbol{\Sigma} k(\mathbf{x}', \mathbf{x})^T \end{aligned} \tag{5.5}$$

with $\boldsymbol{\Sigma} = \left(k(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{X}}) + \sigma_n^2 \boldsymbol{I} \right)^{-1}$. Here $\bar{h}(\mathbf{x}')$ is the predicted mean at $\mathbf{x}'$ and $\mathbb{V}(\mathbf{x}')$ is the predicted covariance matrix. [157]

5.3.1.1 Shape estimation

One particular application of GPR is shape estimation, as shown in [65, 196, 24]. GPR can be combined with Implicit surfaces (IS) to create a probabilistic estimate of an object's shape. IS are defined by a level set of continuous scalar field is used to define the shape. Mathematically, an IS is described by a function $h(\mathbf{x})$ such that the surface is the set of points $\mathbf{x}$ where $h(\mathbf{x}) = 0$. In this context, the scalar field $h(\mathbf{x})$ is modeled as a GP. The IS is then inferred as the zero level set of this GP. This allows incorporation of uncertainty into the shape estimation, providing not just a single shape estimate but a distribution over possible shapes. The probabilistic nature of GPR combined with IS enables robust shape estimation even in the presence of noisy or

incomplete data.

The function $h : \mathbb{R}^d \rightarrow \mathbb{R}$ can therefore define a $(d-1)$-dimensional manifold $\mathcal{S}$ whenever the function intersects a particular value, such as zero. Consequently, the data pair $[\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}]$ can be defined by the piecewise function as follows

$$h_{\text{IS}}(\mathbf{x}) = \begin{cases} -1, & \text{if } \mathbf{x} \text{ is below the surface} \\ 0, & \text{if } \mathbf{x} \text{ is on the surface} \\ 1, & \text{if } \mathbf{x} \text{ is above the surface.} \end{cases} \tag{5.6}$$

where $\mathbf{x}$ is comes from the object's point cloud along with supplementary synthetic points positioned at the scene's boundaries. These additional points ensures that the shape remains both closed and bounded as it introduces a region in space where it is guaranteed no object will be.

Thus, given $[\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}]$, GPR can be used to estimate the surface. This process, which combines GPR with IS, was first described in [196] and is known as Gaussian Process Implicit Surface (GPIS). The surface is modeled as $h_{\text{IS}} \sim \mathcal{GP}\left(0, k_{\text{tp}}(\mathbf{x}, \mathbf{x}')\right)$, where $k_{\text{tp}}(\mathbf{x}, \mathbf{x}') = 2r^3 - 3Rr^2 + R^3$, is the Thin plate (TP) kernel with $r = \|\mathbf{x} - \mathbf{x}'\|$ and R representing the maximum possible value of r [196]. Note that any kernel function can be used in GPR for shape estimation. Several studies have successfully employed the Squared exponential (SE) kernel to model shapes [47, 48, 129]. In particular, the effects of both the TP and SE kernels on shape estimation with GPIS have been studied [65, 131], with an expanded analysis to include kernels from the Matérn family also provided [65]. Their findings support that the TP kernel generally yields better results. Moreover, an analysis comparing the SE and TP kernels, along with an SE-based variation of the algorithm that incorporates geometric priors for reconstruction, has also been conducted [131]. Although the SE with geometric priors produced superior results, the use of priors and the incorporation of the kernel's second derivative (for normal estimation) is outside the scope of this particular work. In this work, when experimentally comparing the zero-mean TP with the base algorithm for the SE, it was observed that the GPIS reconstruction with the SE kernel tends to extrapolate the shape towards positive regions without data, often resulting in open surfaces. Additionally, to enhance computational efficiency, the sparse version of GPIS is adopted as described in [65].

5.4 Methodology

5.4.1 Pipeline

The pipeline for grasping partially observed objects with shape estimation and FM is illustrated in Figure 5.1. As depicted in the Figure the method is composed of two modules: (i) **Database construction**, where a database of selected objects and their associated task func-

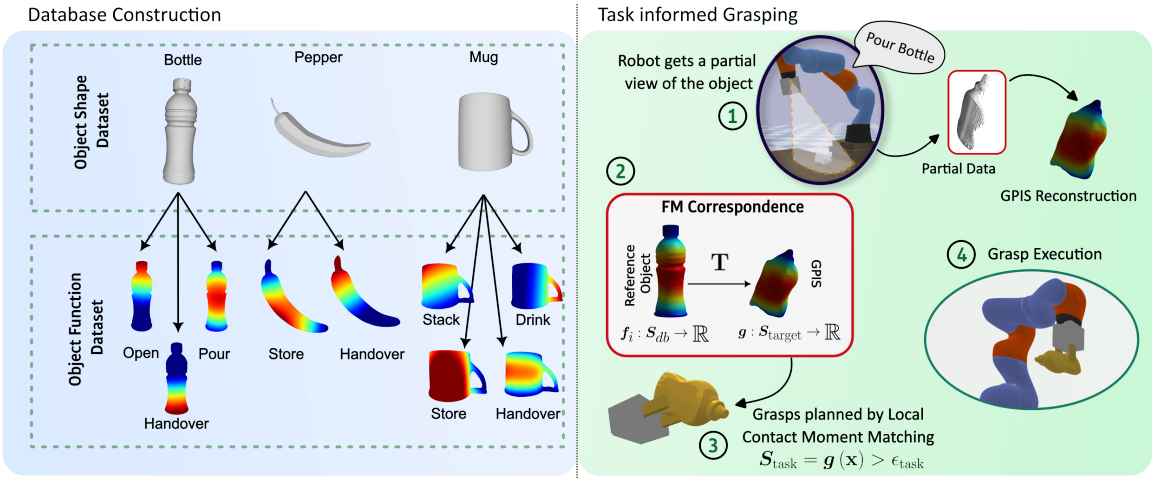


Figure 5.1: Pipeline of the proposed task-informed grasping method. The two modules of the pipeline are shown. A pouring task is shown as an example. In 1, a partial observation of the object is made and the shape is estimated through GPIS. In **2**, the correspondence framework is applied to transfer the function to the reconstructed object. In **3**, grasps are planned in the region defined by the task function, the best grasp is selected, and finally, in **4**, the task-informed grasp is executed.

tions is defined, and (ii) **Task-informed grasping**, where (5.1) is solved and a set of k grasps, $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_k\}$, is calculated in the high probability region of $\mathbf{P}_{\text{target}}$ (indicated by a transferred task-function $\boldsymbol{f}_i$).

During the database construction step, various functions indicating the grasp region are created for each object. For instance, three functions can be defined for a bottle object to cover different tasks: one for pouring, another for opening, and a third for handing over the object. In the task-informed phase, an object and a corresponding task are selected by the user from the database. Subsequently, a partial view of the scene object is captured by the robot, and its GPIS is constructed, assuming that the object belongs to the same class as the reference. The FM then transfers $\boldsymbol{f}_i$ (e.g., the "pour" function in Figure 5.1) to the object's GPIS. Finally, grasps are planned and executed within the preferred region of the current object.

5.4.2 Database construction

During this offline step, a dataset with reference objects and their associated task functions is constructed. To generate the manifolds $\mathcal{S}_{db}$ for each object, their point clouds are captured and their meshes are constructed using the Screened Poisson algorithm [98]. This process is performed by a robot equipped with a wrist-mounted camera, which captures ground plane segmented point clouds of objects by registering clouds from multiple viewpoints [132].

Once the point cloud of an object is obtained, the GPR defined in (5.5) with the squared exponential kernel[1] is used to generate functions $\mathbf{f_i} \in \mathcal{S}_{db}$. The data points $\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}$ are manually

[1] While the Thin Plate Kernel is preferred for shape reconstruction, here a function is being fitted over an already defined surface, and the SE kernel provides smooth, infinite differentiable results, with its hyper parameter, l adding

annotated by the user, with $\mathcal{Y} = [0, 1]$ for a chosen location $\mathcal{X}$. Points with values closer to 1 indicate regions in the manifold more suitable for grasping, while those closer to 0 denote unsuitable grasping regions. This annotation process is quick, typically requiring only a few seconds, as most task functions can be formulated with just 2 to 5 selected points. Alternatively, labeled datasets or video-based annotations or demonstration such as in chapter 4 can be used for this purpose. On the left side of Figure 5.1, different functions for three different objects are illustrated, with red-coloured areas indicating the graspable zones. These functions correspond to different tasks, such as "pour," "open," and "handover," and are generated based on the annotated graspable regions.

5.4.3 Task-informed grasping

Once the database has been created, the grasping can be deployed. The only prerequisite is that a reference object $\mathcal{S}_{db}$ and a task function $\boldsymbol{f}_i$ are selected from the database beforehand. As depicted in Figure 5.1, this phase involves the robot making a partial observation of the object by capturing the scene's point cloud from a single viewpoint. It is worth noting that the extent of the object's surface captured in a single view can vary from one object to another. To analyse that, experiments on how the degree of partiality influences results are presented in Section 5.5.4.2. After capturing the partial cloud, the corresponding IS h_{IS} is created from (5.6), with the points set as the surface. A grid of points $\mathbf{x}_{\text{grid}}$ is then defined over the workspace, and the GPR is applied, defining the manifold $\mathcal{S}_{\text{GPIS}} \triangleq \left\{\text{mesh}(\mathbf{x}_{\text{grid}}) | -\delta \leq \mathbf{x}_{\text{grid}} \leq +\delta\right\}$, where δ is a small threshold value and mesh$(\cdot)$ is the function to create a continuous shape from a set of points. With $\mathcal{S}_{\text{GPIS}}$ available, the FM process outlined in Sections 3.3.1.2 and 4.3.2 is employed to obtain the LBO basis $\Phi_{\mathcal{S}_{db}}$ and $\Phi_{\mathcal{S}_{\text{GPIS}}}$, as well as the functional map $\mathbf{C}$ from (3.7). As previously mentioned, FM is well-suited for transferring information across different configurations of an object. This transfer can be achieved through the equation

$$\boldsymbol{g} = \Phi_{\mathcal{S}_{\text{GPIS}}} \mathbf{C} \left(\Phi_{\mathcal{S}_{db}}\right)^{\dagger} \boldsymbol{f}_i \tag{5.7}$$

where, † denotes the *Moore-Penrose* pseudo-inverse. Using (5.1) and (5.7), $\mathbf{\Pi}$[2] can be then calculated as

$$\mathbf{\Pi} = \Phi_{\mathcal{S}_{\text{GPIS}}} \mathbf{C} \left(\Phi_{\mathcal{S}_{db}}\right)^{\dagger}. \tag{5.8}$$

Following this, the point cloud $\mathbf{P}_{\text{task}}$ is obtained from sampling $\mathcal{S}_{\text{task}} \triangleq \{g(\mathbf{x}) > \varepsilon_{\text{task}}\}$. Then, given the information transfer between $\mathcal{S}_{db}$ and $\mathcal{S}_{\text{GPIS}}$, it follows that $\mathbf{x} \in \mathcal{S}_{\text{GPIS}}$. That is,

some control over it [157]

[2] In equation (5.1), no dimensions have been specified for $\mathbf{\Pi}$ when highlighting the problem. This was intentional, as the problem involves transferring a function over a continuous full Riemannian manifold $\mathcal{S}_{db}$ to a second continuous Riemannian manifold resulting from the partial cloud $\mathbf{P}_{\text{target}}$. In practice, however, the dimensions of $\mathbf{\Pi}$ depend on the discrete shapes obtained from approximating the manifolds to the meshes $\mathcal{S}_{db}$ and the reconstructed $\mathcal{S}_{\text{GPIS}}$. Thus, the dimensions of $\mathbf{\Pi}$ are determined by the number of vertices in each mesh. Specifically, if $\mathcal{S}_{db}$ has N_{db} vertices and $\mathcal{S}_{\text{GPIS}}$ is reconstructed to have N_{GP} vertices, then $\mathbf{\Pi} \in \mathbb{R}^{N_{db} \times N_{GP}}$.

the points extracted are part of the estimated shape. One way to ensure grasp stability in this case is to plan grasps over surfaces where object information is present. This can be achieved by planning directly in a subset corresponding to the partial cloud $\mathbf{P}_{\text{target}}$. Thus, the graspable region is the cloud obtained from the intersection of $\mathbf{P}_{\text{task}}$ and $\mathbf{P}_{\text{target}}$,

$$\mathbf{P}_{\text{task}}^{\text{partial}} = \mathbf{P}_{\text{task}} \cap \mathbf{P}_{\text{target}}. \tag{5.9}$$

To obtain the grasp set $\mathcal{G}(\mathbf{P}_{\text{task}}^{\text{partial}})$ over $\mathbf{P}_{\text{task}}^{\text{partial}}$, the LoCoMo grasp planner is used (see Section 3.3.2 and [3] for more details). This planner generates grasp hypotheses by assessing local similarities between an object's surface and gripper fingers using zero moments shift. Grasps are then presented in a ranked list based on a shape similarity score. The set of k grasps is denoted as $\boldsymbol{\mathcal{G}} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_k\}$, where each grasp is $\mathcal{G}_j = \{\mathbf{q}_j, r_j\}$, with $\mathbf{q}_j \in \text{SE}(3)$ being the gripper pose in the world frame and r_j the LoCoMo ranking score, calculated from (3.8). Once grasp poses are generated, the top-ranked and best feasible pose is selected by the robot for execution.

5.4.3.1 Grasping in uncertain regions

Using (5.9) to define the object's graspable area ensures stability by limiting grasp planning to known regions of the object. However, this approach also requires that the observed partial region is graspable, (*i.e.*, it must fit within the gripper's stroke) which is not always the case. Planning grasps directly on the GPIS-reconstructed surface provides a way to deal with these limitations. Nevertheless, this strategy might lead to sub-optimal planning, especially in areas characterised by high uncertainty.

GPIS-reconstructed surfaces offer another significant advantage in tackling this issue: it encodes uncertainty in the estimated shapes. From (5.5), only data points falling within $\mathcal{S}_{\text{GPIS}}$ are considered, thus allowing $\mathbb{V}$ and $\boldsymbol{g}$ to be both written as functions on the same manifold (that is, $\mathbb{V} : \mathcal{S}_{\text{GPIS}} \rightarrow \mathbb{R}$ and $\boldsymbol{g} : \mathcal{S}_{\text{GPIS}} \rightarrow \mathbb{R}$). Then, if they are in the same space, the functions can be merged and $\mathbb{V}$ can be used in a manner that skews the grasp area defined by $\boldsymbol{g}$ towards regions with lower uncertainty, effectively increasing the graspable region while maintaining the robot it close to regions of higher confidence. In this manner, a new function can be defined,

$$u_{task}(\mathbf{x}) = (1 - K_u \tilde{v}(\mathbf{x}))\, g(\mathbf{x}) \tag{5.10}$$

where K_u is a scaling factor between 0 and 1, and $\tilde{v}(\mathbf{x})$ is the re-scaled variance, calculated as

$$\tilde{v}(\mathbf{x}) = \frac{\mathbb{V}(x) - \mathbb{V}_{min}}{\mathbb{V}_{max} - \mathbb{V}_{min}}. \tag{5.11}$$

By using (5.10) and (5.11), the procedure outlined in Section 5.4.3 for obtaining grasps can be updated by defining $\mathcal{S}_{\text{task}} \triangleq \{u_{task}(\mathbf{x}) > \varepsilon_{\text{task}}\}$ and then obtaining the new $\mathbf{P}_{\text{task}}$ on which grasps

can directly be planned, i.e. $\mathcal{G}(\mathbf{P}_{\text{task}})$. The task cost functions on reconstructed objects after applying (5.10) are illustrated in Figure 5.2.

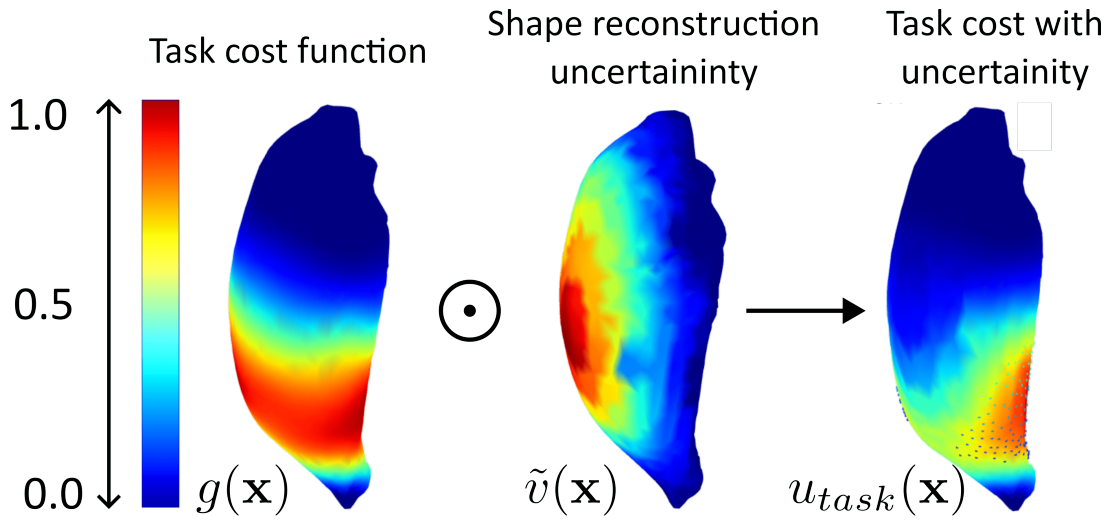


Figure 5.2: The individual terms for the task cost function $g(\mathbf{x})$ and the uncertainty function $\tilde{v}(\mathbf{x})$ are defined on the same manifold. Using (5.10), these terms are combined to form $u_{\text{task}}(\mathbf{x})$, which also remains within the same manifold.

5.5 Experimental Validations

The proposed approach is evaluated through both simulation and real robot experiments. Initially, an exhaustive analysis of various methodological aspects is conducted via simulation. This includes an examination of data partiality, an analysis of the effect of the uncertainty function, and an assessment of overall grasp success. Subsequently, the proposed grasp transfer technique is demonstrated using a real-robot setup.

5.5.1 Simulation setup

As in previous chapters, simulation experiments were conducted using the Pybullet simulator with a 7-DoF robot equipped with the 2-finger gripper. A simulated 3D camera was fixed on the robot's end-effector for image capture. To create reference objects in the database, the robot was moved to ten positions to build a point cloud. The point clouds captured at each location were processed and stitched together to generate full object meshes. For the grasping experiments, the objects were treated as unknown and only partially observable, with one or two views (depending on the object) acquired. To balance quality with processing time, the source mesh was limited to 5000 vertices. For GPIS, the GPy library [74] was used, and for FM, the code provided by [136] was integrated. All experiments were executed on an Ubuntu 20.04 machine with an i7 4-core CPU.

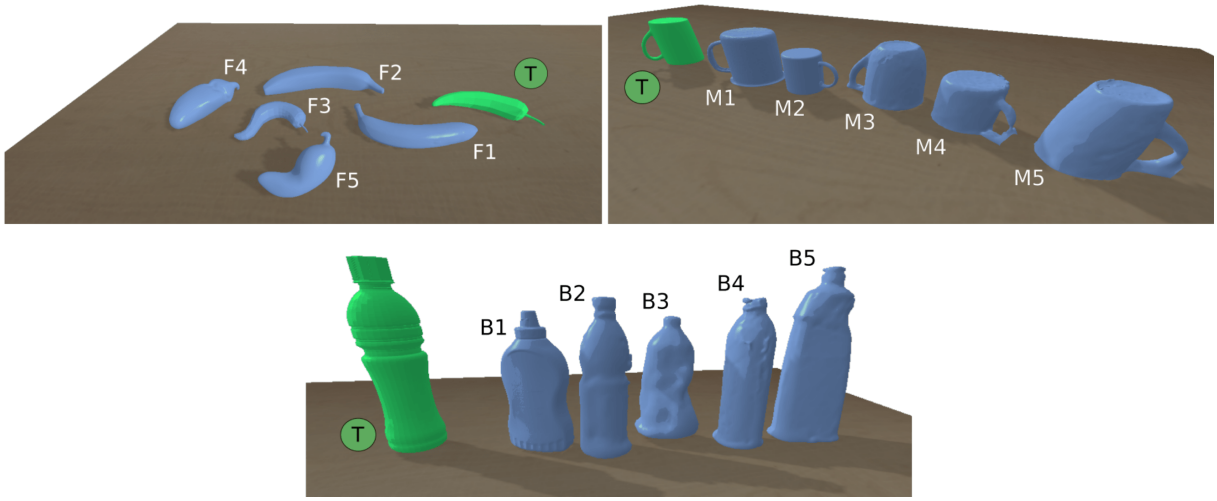


Figure 5.3: Dataset of simulation objects. The reference object is shown in green. Here the **FX** objects are the "Fruits", the **MX** objects represent various "Mugs", and the **BX** the "Bottles".

Three sets of experiments were conducted to validate the approach and gain a deeper understanding of how various elements within the method influence success. First, the impact of the uncertainty gain K_u, as described in (5.10), on the final grasping success rate was analysed (refer to Section 5.5.4.1). Then, the influence of the amount of partial data on the method's success was examined (see Section 5.5.4.2). Finally, the overall grasp success for three different classes of objects—bottles, fruit (e.g., pepper, banana), and mugs—was evaluated. Six different objects were chosen for each class, with one designated as the reference. Experiments involved transferring a task function from the reference object to the remaining five objects within the same class. Figure 5.3 depicts the dataset of objects used for simulation experiments. For both real and simulated experiments, objects were selected based on the following criteria: **a)** they needed to have a sufficiently visible area for the experiments. Objects such as small balls or fruits, where a single camera view could capture most of their shape, were excluded; **b)** objects needed to be compatible with the gripper used, meaning at least one grasp should fit within the gripper's strokes; **c)** the objects must either be deformable or have readily available variations (such as different bottles and mugs) in CAD models for simulations and physically for real experiments; **d)** completely symmetrical objects were not considered in this work, as they pose additional challenges, for instance, self-symmetric maps can lead to significant variations in results, with all being valid.

5.5.2 Real-world setup

Similar to the simulation setup, the real robot setup consists of a 7-axis Kuka iiwa robot equipped with the Schunk PG 70 gripper. A calibrated Ensenso N35 3D camera is affixed to the robot's wrist. Real-world experiments involve either transferring functions between different physical objects of the same class or transferring between the CAD model of a template object, as



Figure 5.4: Objects used for real-world experiments. In addition to the Fruit, Bottle and Mug classes, the Drill object has been added.

shown in Figure 5.3, and a real object of the same class, shown in Figure 5.4. The point-cloud processing and controller software are developed in C++ and executed on a computer running Windows with an i7 4-core CPU at 2.70GHz. The proposed algorithm uses the same Python code from the simulation experiments. Furthermore, an FTP protocol is implemented to transfer point-clouds and grasps between the two systems.

5.5.3 Metrics

All experiments followed to the grasp evaluation protocol outlined in [17], which includes three consecutive tests: lifting the object, shaking it by rotating the robot's fifth joint, and rotating it around the robot's seventh joint. The success score presented in the results reflects the aggregate performance across all three tests. Additionally, experiments with each test object were repeated five times. Due to variability observed in the sparse GPIS reconstruction, a new point cloud was acquired, and the entire pipeline was executed for each iteration. The results presented in this paper represent the average of all runs. Furthermore, the Chamfer distance between the full object point cloud and the reconstructed models (in simulation only) was computed to assess whether the robot successfully grasped the intended part of the object. Additional experiments to compare the proposed method to the state of the art have also been conducted in simulation. In this case, similarly to Chapter 3 the FM-based framework was compared to the method with the CPD algorithm and (see Section 3.3.1.1) and with ICP (see 3.3.1). The choice to compare with these two alternatives was motivated by a gap in the literature for *learning-free* approaches to correspondence-based transfer, where CPD still stands as the state of the art. While comparisons with other learning-based approaches, such as those leveraging DINO features, could be made, these methods are fundamentally different in nature. The proposed method aims to provide a reliable, low-data alternative, whereas DINO-based approaches rely on large-scale feature

learning, making direct comparisons less meaningful within the intended scope of evaluation . For evaluating each algorithm a ground truth of the functional transfer has been generated and a number of error metrics calculated. That is, for the ground truth function $\boldsymbol{g}^*$ and the transferred function $\boldsymbol{g}$, the metrics are:

- **Mean average error (MAE)**:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\boldsymbol{g}_i^* - \boldsymbol{g}_i|$$

- **Root mean square error (RMSE)**:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{g}_i^* - \boldsymbol{g}_i)^2}$$

- **Maximum error (MAX)**:

$$\text{MAX} = \max_{i\in\{1,\dots,N\}}|\boldsymbol{g}_i^* - \boldsymbol{g}_i|$$

- **Standard deviation of error (STD)**:

$$\text{STD} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(|\boldsymbol{g}_i^* - \boldsymbol{g}_i| - \text{MAE}\right)^2}$$

Each of these metrics captures different aspects of error: MAE provides an overall measure of accuracy, while RMSE makes larger deviations more apperant. MAX identifies the worst-case errors STD quantifies error variability.

5.5.4 Simulation Experiments

5.5.4.1 Effect of uncertainty function

To analyse the impact of using shape uncertainty on the success of the proposed method, the variable K_u in (5.10) was changed (with values ranging from $0-1$ used) and the grasp success rate evaluated. Here experiments were conducted using the "Bottle" object labeled **B1** in Figure 5.3 with the functions `Open` and `Pour` used when testing.

From the plots shown in Figure 5.5, it is evident that leveraging uncertainty is crucial for the successful transfer of grasp functions. Shape reconstruction may interpolate occluded parts into non-existent areas, which are often associated with high uncertainty. By utilising the uncertainty information from GPIS reconstruction, areas with more reliable data can be prioritised for grasping.

While using uncertainty to avoid risky areas improves results, as indicated by the red plot in Figure 5.5, where lower values of K_u lead to poor success rates, extreme values of K_u also have drawbacks. When $K_u = 1$, completely avoiding reconstructed parts of the object is attempted. Despite the expectation of increased grasp success, as confidence in the shape is high, the red plot in Figure 5.5 shows a decrease in success. This occurs because, as shown in the blue plot, the number of feasible grasps significantly decreases, potentially leaving no viable options. Empirical results suggest that the optimal values for K_u are between 0.7 and 0.9.

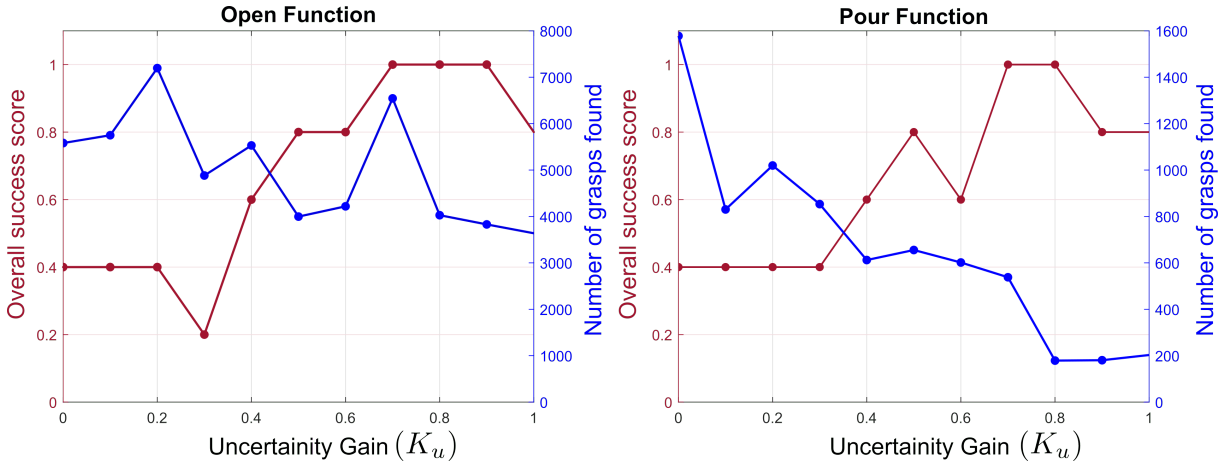


Figure 5.5: Effect of the Gain K_u from (5.10) on the Average Grasp Success Rate and Number of Grasps Found. The overall grasp success rate is shown in red (i.e., the average number of stable grasps after the stability tests – lift, shake and rotate – are performed) , and the average number of grasps found is shown in blue. The experiment was conducted on object **B1** for the functions `Open` and `Pour`.

5.5.4.2 Analysis of data partiality

In this section, the impact of partial observation, *i.e.,* the extent of the object region captured, on the success of grasp transfer is explored. Bottle B1 was used for this analysis, and three different cuts along each of its axes were made, as illustrated in Figure 5.6. The evaluation was performed incrementally, beginning with the segment from the first cut, then extending to include up to the second cut, progressing to the third, and finally considering the entire bottle. The percentage of points in each section depends on the object's symmetry and the axis along which the cuts are made. The functions `Open` and `Pour` were examined. The results are summarized in Table 5.1.

The results indicate that the success rate for achieving stability and accurately grasping the intended part improves as a larger portion of the object is considered, as expected. Since the full-to-full object transfer has a success rate of 100%, the closer the object is to its full shape, the better the results. Particularly the cuts along the X and Y axes (refer to Figure 5.6 for a depiction of each cut) perform better than those along the Z axis. This is due to two main reasons: First, as seen in the figure, between cut-1 and cut-2, the region corresponding to the `Pour` function is outside the cut, making it harder to find nearby stable grasps. Secondly, for this object, removing

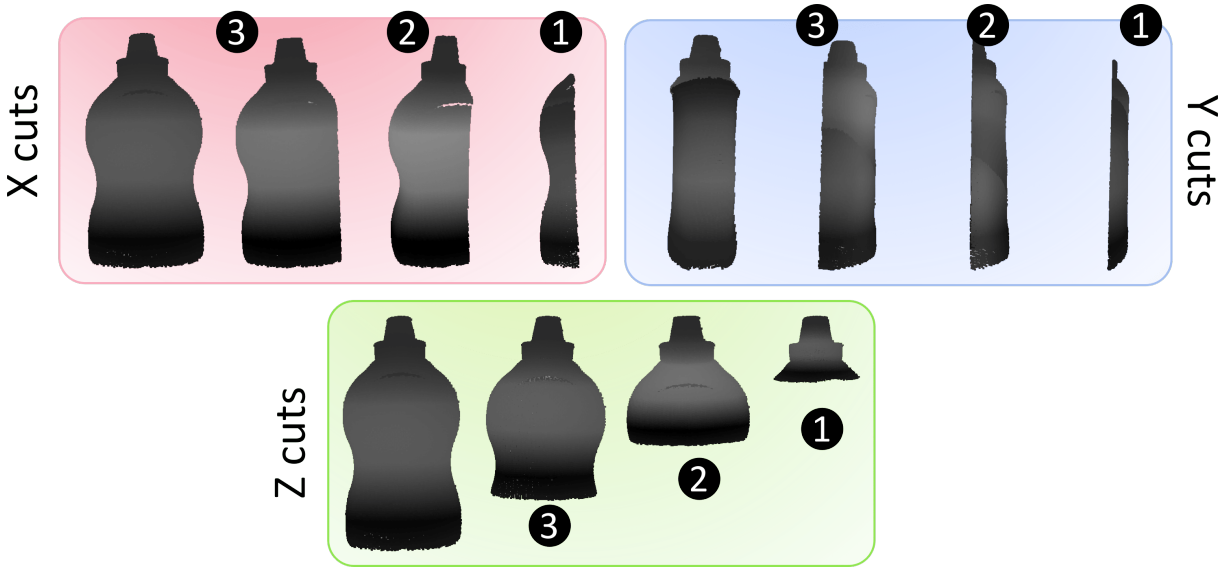


Figure 5.6: Cuts Made Along the X, Y, and Z Axes of Bottle **B1**. The cuts are shown in Figure 5.3, with numbers in circles denoting the cut positions.

larger sections in the Z-axis results in smaller shapes with geometries that further deviate from the template, thus hindering the FM-based matching.

5.5.4.3 Comparison of Transfer Error

Similar to the previous section, this analysis examines the effects of the data partiality. However, the focus here is on comparing the transfer of the "base" FM-based method for task functions and two variations of this method, namely with ICP and CPD. Experiments were conducted for each of the cuts shown in Figure 5.6, using each transfer method. The object was placed in two positions: aligned with the original object and rotated 90 degrees. Each experiment was repeated five times. Table 5.2 presents the overall results, while Appendix A provides detailed results for each function and object configuration.

From the results, it is evident that the FM-based approach achieves the highest success rate for most cuts and metrics. This is attributed to the FM-based approach's ability to successfully and smoothly transfer functions across nearly isometric shapes, invariant to the pose. By analysing the expanded results in Tables A.1 to A.4 in Appendix A, it is notable that while ICP and CPD demonstrate some level of success when objects are aligned (i.e., they are in similar positions, enabling the algorithms to match the points), the results degrade significantly when the object is rotated 90 degrees (as seen in Tables A.3 and A.4). A key insight from the experiments with aligned objects is that ICP and CPD can often represent a general region well, but the details of the transferred functions are lost, as evidenced by larger errors and significantly larger maximum errors.

Furthermore, a detailed analysis of the results reveals another key insight. As also shown in Table 5.1, Z-Cuts exhibit poorer performance with the FM algorithm. This is because, even

Table 5.1: Analysis of the impact of data partiality on method's success.

Cuts	**% Pts.**[1]	**C. D.**[2]	**C.P.**[3]	**Stability**[4]
		X-Cuts		
Cut–1	16.99	2.10	30.0	20.0
Cut–2	40.57	3.10	87.5	87.5
Cut–3	71.38	1.93	90.0	90.0
		Y-Cuts		
Cut–1	41.42	3.58	30.0	30.0
Cut–2	71.58	3.53	40.0	80.0
Cut–3	87.73	2.06	80.0	90.0
		Z-Cuts		
Cut–1	30.22	17.22	10.0	30.0
Cut–2	60.11	6.67	40.0	30.0
Cut–3	80.33	3.01	80.0	90.0
Full model	100.0	4.36	100.0	100.0

[1] Percentage of points compared to full cloud.
[2] Chamfer distance in millimetres between the full and reconstructed models.
[3] Average percentage of success in grasping the correct object part.
[4] Average success of stability tests out of 5 trials.

Table 5.2: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates.

	FM					CPD					ICP				
Cuts	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]
X-Cuts															
C1	**0.2363**	**0.2742**	**0.5645**	0.2463	30.00	0.3173	0.3893	0.9007	**0.2415**	**40.00**	0.3457	0.4232	0.8774	0.2427	25.00
C2	**0.0890**	**0.1135**	**0.2857**	**0.1040**	**90.00**	0.3321	0.4496	0.9588	0.3305	45.00	0.3339	0.4440	0.9596	0.3065	50.00
C3	**0.0766**	**0.0980**	**0.2825**	**0.0796**	**90.00**	0.3452	0.4652	0.9750	0.3454	50.00	0.3377	0.4509	0.9664	0.3160	50.00
Y-Cuts															
C1	**0.1758**	**0.2111**	**0.4552**	**0.1718**	**55.00**	0.3011	0.3974	0.9431	0.2779	40.00	0.3182	0.4039	0.9368	0.2709	45.00
C2	**0.0851**	**0.1123**	**0.2873**	**0.1000**	**80.00**	0.3161	0.4316	0.9652	0.3173	50.00	0.3217	0.4262	0.9592	0.2987	50.00
C3	**0.0800**	**0.1007**	**0.2842**	**0.0951**	**95.00**	0.3199	0.4389	0.9719	0.3287	50.00	0.3291	0.4357	0.9686	0.3049	50.00
Z-Cuts															
C1	0.4792	0.5064	**0.6753**	0.1508	**5.00**	0.4836	0.5141	0.7976	0.1586	0.00	**0.4762**	**0.4964**	0.7209	**0.1292**	0.00
C2	**0.2662**	**0.3057**	**0.6341**	**0.2036**	**60.00**	0.3969	0.4699	0.9331	0.2916	25.00	0.3553	0.4295	0.8925	0.2333	50.00
C3	**0.3087**	**0.3647**	**0.7623**	**0.3076**	**60.00**	0.3603	0.4818	0.9725	0.3522	50.00	0.3346	0.4446	0.9719	0.2977	50.00

[1] MAE: Mean Average Error
[2] RMSE: Root Mean Square Error
[3] MAX: Maximum Error
[4] STD: Standard Deviation of Error
[5] CP: Average percentage of success in grasping the correct object part.

with reconstruction, the geodesic distances are not preserved, leading to degraded results due to the violation of this assumption. However, ICP and CPD do not have this requirement. While objects need to be aligned for some information to be transferred to a nearby region, their shape can undergo greater variation. This fact results in a slightly better performance for Correct part (CP) in the Z-Cuts when the objects are aligned.

When analysing Tables A.3 and A.4 for the rotated objects, the CP results fall to zero for both the CPD and ICP methods. This is because these methods attempt to match points to a nearby region rather than matching features on the surface, as the FM-based approach does. In the case of ICP, the algorithm fails completely, setting all points in the transferred results to zero. Consequently, in some cases, the error in ICP is smaller because the zeroes are matched correctly, but the positive values of the functions are not.

5.5.4.4 Grasp success evaluation

Three classes of objects, each comprising six items, were used for these tests. The objects were individually positioned randomly within the robot's task space, enabling partial observation with the camera. Five trials were performed with each object for every transferred function. Overall results of the experiments are presented bellow Table 5.3, with full results shown in Appendix A. Furthermore, following is an analysis of each object:

- **Mug**: Two poses were selected for all functions. A total of 40 different tests were conducted, each repeated five times for a total of 200 runs, resulting in a **92.8**% success rate in grasping the correct part and an **85.9**% overall grasp success rate.
- **Bottles**: Three poses were selected for the `open` and `pour` functions (except when the bottle was symmetric), and one pose for the `handover` function. A total of 31 experiments were conducted, each repeated five times for a total of 155 runs, resulting in a **96.1**% success rate in grasping the correct part and an **88.8**% overall grasp success rate.
- **Fruit**: Two poses were selected for all functions. A total of 20 different tests were conducted, each repeated five times for a total of 100 runs, resulting in a **90**% success rate in grasping the correct part and an **86.8**% overall grasp success rate.

Figure 5.7 depicts grasping tests for two objects/functions from each class, as well as the transfer from the initial task function to the new object shape.

5.5.5 Real Robot Experiments

The proposed algorithm was tested in two real-world scenarios. First, as shown in Figure 5.8, functions were transferred from CAD models of objects (green templates in Figure 5.3) to real-world objects of the same class (shown in Figure 5.4). Next, the transfer of grasps between two

Table 5.3: Overall performance analysis of the proposed approach in simulation.

Function	C.D.	Part*	Lift‡	Rot.‡	Shake‡
			Bottle		
Pour	4.41	97.3	93.8	93.8	92.4
Open	3.70	91.1	93.8	92.4	89.8
Handover	3.60	100.0	96.0	96.0	84.0
Total	**3.82**	**96.1**	**94.45**	**94.1**	**88.8**
			Mug		
Drink	2.79	100.0	96.0	93.75	85.75
Store	2.83	98.0	94.0	94.0	88.0
Stack	2.70	84.0	92.0	88.0	82.0
Handover	2.96	89.5	96.0	94.0	88.0
Total	**2.82**	**92.8**	**94.5**	**92.4**	**85.9**
			Fruit		
Handover	5.49	86.0	92.0	90.0	87.5
Stack	4.08	94.0	92.0	92.0	86.0
Total	**5.51**	**90.0**	**92.0**	**91.0**	**86.8**

* Average % of success in grasping the correct object part.
‡ Average % of success – reported out of 5 trials.

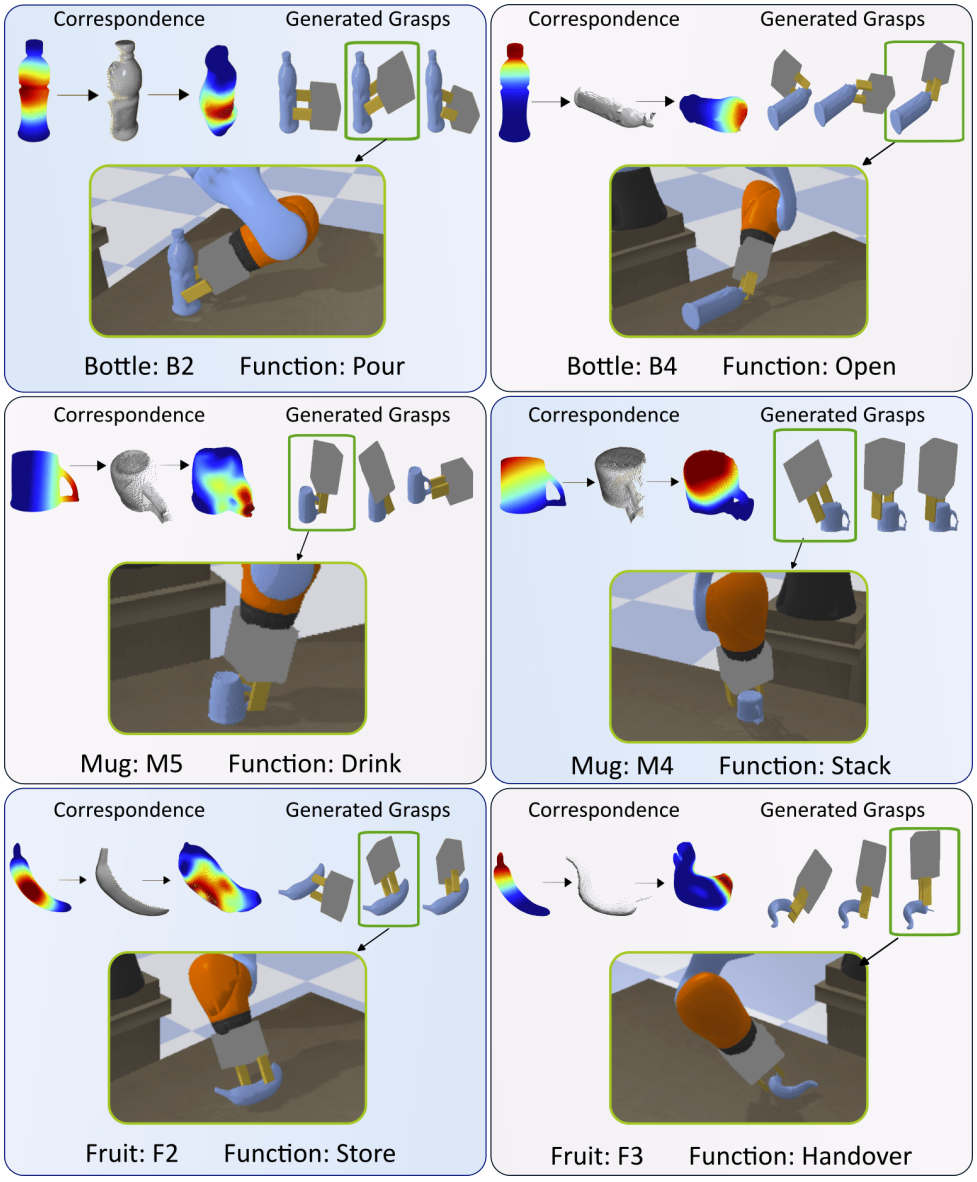


Figure 5.7: Six Different Experiments in Simulation. These experiments demonstrate the transfer, grasp generation (with selected grasps highlighted), and grasp execution.

real-world objects was evaluated, as depicted in Figure 5.9. In this scenario, the model derived from the complete point cloud of one object was used to define a function, which was then transferred to another, similar real-world object.

Two additional experiments were conducted to further evaluate the method's applicability and its tolerance for deviation from the reference template. The first experiment, depicted in Figure 5.10, involved selecting objects from a similar category but with significant shape differences from the reference. Specifically, the top row of the figure examines the transfer of the `drink` function from a "mug" (reference object) to a "pitcher" (target object). Although the functions were successfully transferred to the appropriate regions, some discrepancies were noted. The use of geodesic distances in functional maps, which are linked to the Laplacian basis

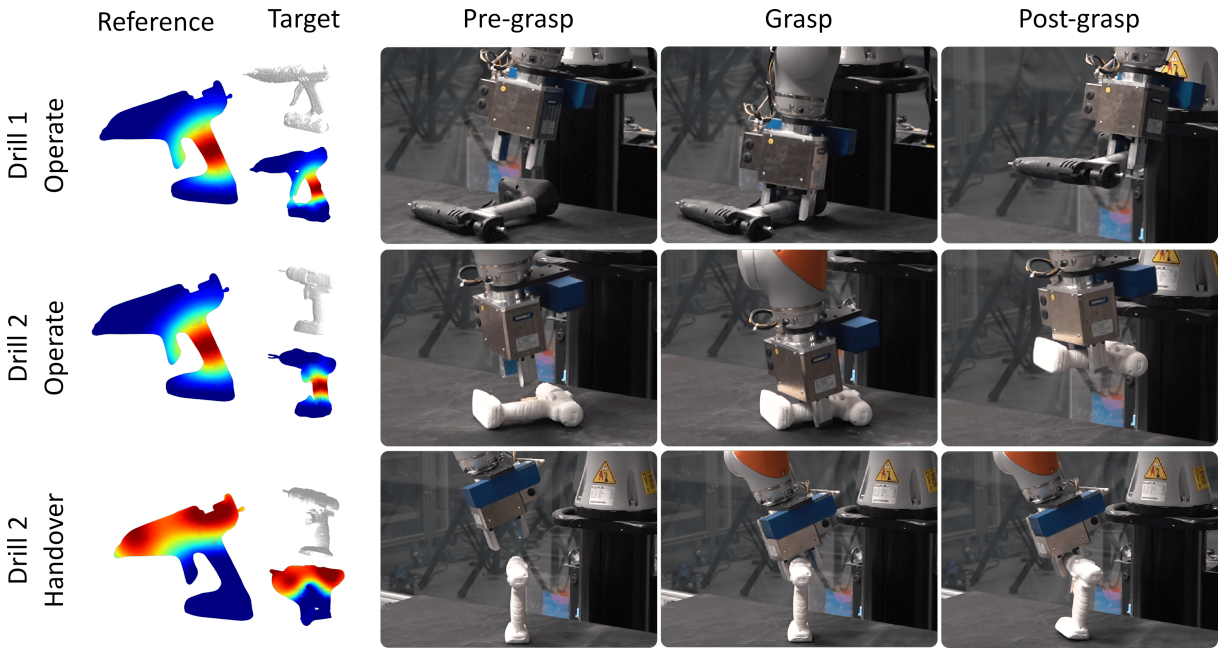


Figure 5.8: Transfer Between Two Real-World Objects. Two different functions are defined and demonstrated, showing the transfer from "Drill 1" to a partial version of itself and to a completely different drill.

and descriptors on the object's surface, resulted in performance degradation when there were considerable size variations. Essentially, as the deviation from isometric conditions increases, the method's effectiveness decreases. Overcoming this limitation is a key objective for future research, where solutions such as normalised scaling or adopting scale-invariant descriptors will be investigated.

In the second experiment, illustrated in the bottom row of Figure 5.10, the `Open` function was transferred from a reference bottle to another bottle with a completely different shape, yet within the same scale. This experiment was successful because the geometry of the target bottle can be considered a near-isometric transformation of the reference, where the isometric distances are largely preserved.

To address the question, "How different can objects be while still allowing for a stable transfer of a grasp function?" an experiment was conducted to transfer the `store` function from the "pepper" object (F4 in Figure 5.3 to three other objects (banana, sponge, and wooden block) of similar scale but varying shapes. The results are shown in Figure 5.11.

As in the simulation experiments, the transfer to the banana was successful. Similarly, the sponge yielded successful results. However, the transfer to the block was unsuccessful, with no viable grasps found. Despite being of similar scale, the block's sharp edges likely interfered with the algorithm's ability to calculate matching features, an issue not encountered with the sponge, which more closely resembles the other shapes.

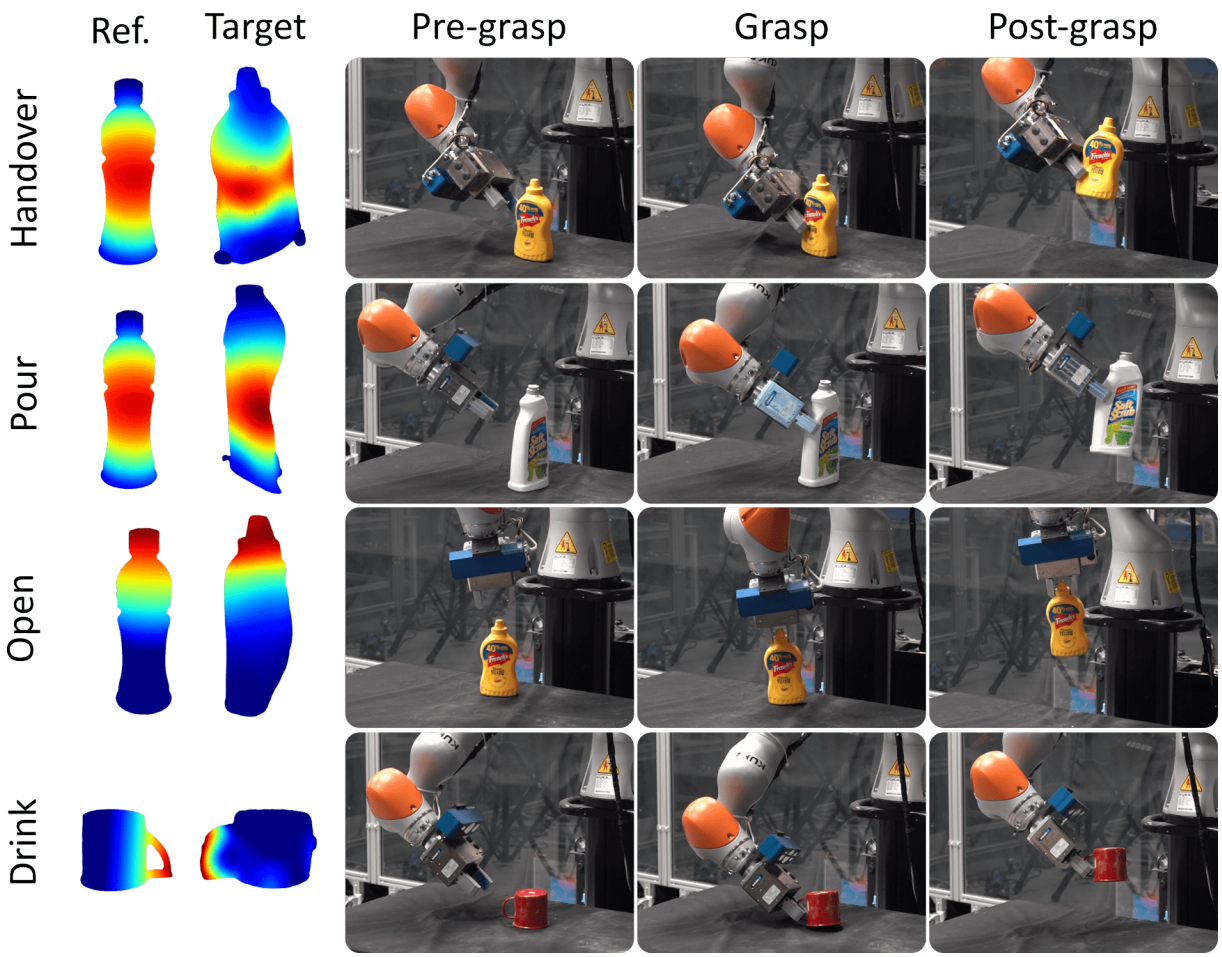


Figure 5.9: Results for Transferring Various Task Functions to Real-World Objects.

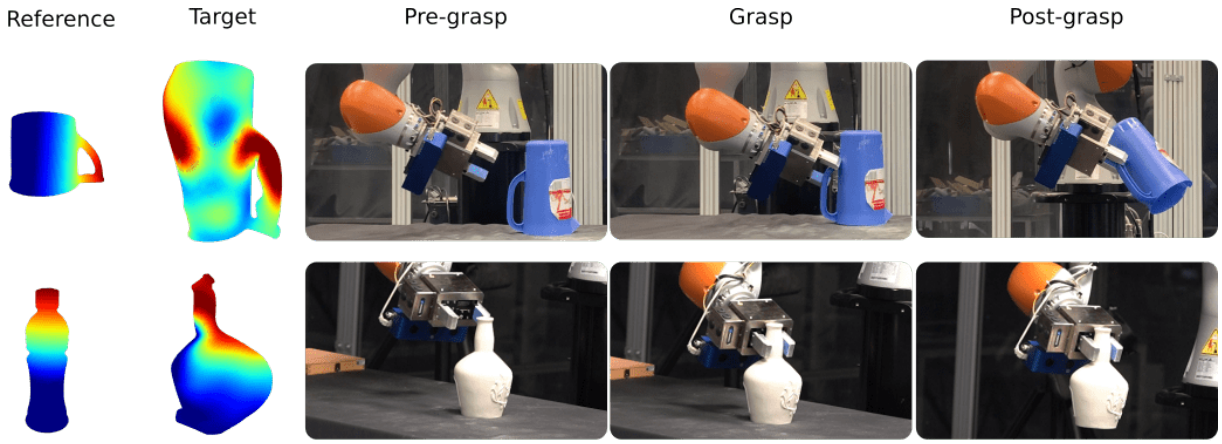


Figure 5.10: Function transfer between objects of varying sizes (top row) and shapes (bottom row) within the same class.

5.6 Discussion

Here a method for transferring functions representing different tasks between objects has been proposed and demonstrated both in simulation and real world experiments. Particularly, as

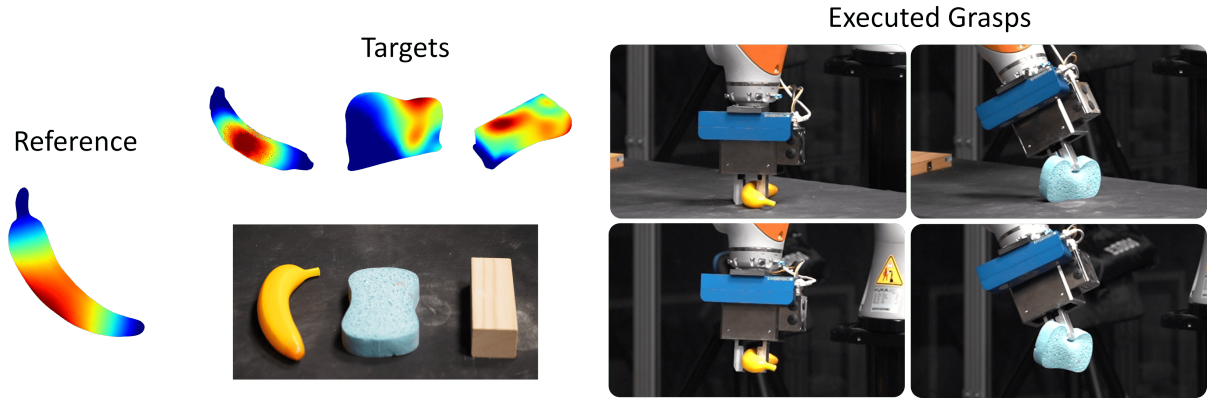


Figure 5.11: Analysis of the impact of shape variation on transfer success. The `Stack` function from the "Pepper" was transferred to three objects: "Banana," "Sponge," and "Wooden Block."

shown in Table 5.3, stable grasps for different function were achieved across various objects. As detailed in Section 5.5.5, this method can be readily adapted to real-world scenarios. Since the proposed pipeline is based on the FM framework, it ensures pose invariance, facilitating transfers between generic CAD models and real objects, as illustrated in Figure 5.8. Notably, this approach does not require a learning stage, and satisfactory results can be obtained as long as a reference template is available. Moreover, an analysis of the effects of object partiality and the measurement of uncertainty was conducted. Although partiality is highly dependent on how the object is occluded rather than the number of points, a correlation between uncertainty and stability was found.

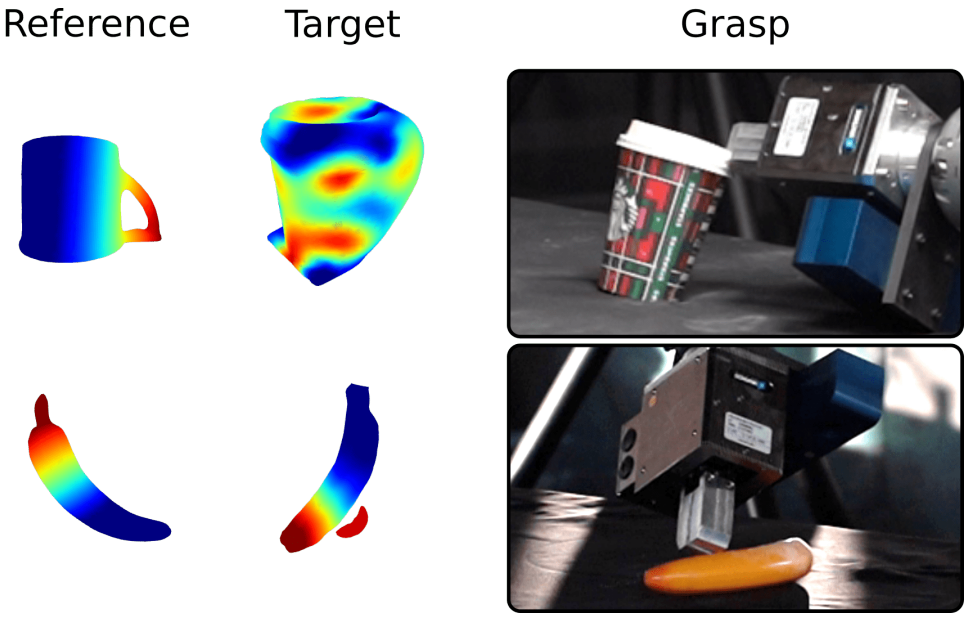


Figure 5.12: Failure Cases. The top row depicts a transfer between the "Cup" and a take-away coffee cup. The bottom row shows a transfer between "Pepper" and "Banana." An artifact in the reconstruction is visible in the middle image, where the grasps are planned.

However, the current method does have some limitations. If too many artifacts occur during

the GPIS or mesh construction, the results will deteriorate. Some common failure cases are shown in Fig. 5.12. In the top example, the transfer between the "Mug" template and a take-away coffee cup failed due to a mismatch between the reference and reconstructed shapes. In the bottom example, despite a successful transfer from "Pepper" to "Banana," an artifact in the reconstruction led to grasp calculations on a non-existent part of the object, even when incorporating uncertainty into the analysis.

The second limitation is that the employed FM pipeline is not scale-invariant. As shown in Fig. 5.10, while certain results are achievable with scale variations, the outcomes are unreliable for objects of different sizes but within the same class. Addressing this issue is planned for future work.

Finally, challenges related to symmetry have been noted, particularly with objects like the "Mug." The `Stack` function assumed that the grasping area would be on the top. However, when the mug was placed upside-down and the opening was not visible, the method failed to differentiate between the two symmetric sides. Future work will explore self-symmetric solution spaces to overcome this issue. Despite some limitations, the high success rate observed in the experiments show the viability of the proposed method.

5.6.1 Comparison With Existing Methods

The proposed method demonstrates several advantages over learning-based methods and shows promise for integration with Large Language Models (LLMs) and other learning-based architectures in future work. Notably, the method requires only one labelled piece of data, as described in the database creation process in Section 5.4.2. This enables high generalisation for category-level objects, even under deformation, without the need for an embedded training process, making the method both unique and widely usable. In contrast, existing methods such as those proposed in [130, 66, 189, 195] often require significant amounts of training data and time at some step of the pipeline. Furthermore, while some of these methods support one-shot learning, they often fail to effectively handle partially observed or complex-shaped objects.

Additionally, recent research has explored integrating foundational models into one-shot learning-by-demonstration pipelines, such as the work by [46]. However, no methods to date have focused on local details that can be exploited using 3D data. Building on the works of [46, 197, 1], future research directions should incorporate these models to enhance grasping and skill-related functions with semantic understanding, particularly in handling partial and full 3D data.

One aspect of the proposed approach involves the manual definition of functions. This is a simplified approach requiring minimal effort from a user to define functions over objects, however creating large databases manually can still be tedious. To further automate the process one possibility is leveraging advanced techniques to extract functions from robot-object interactions, either through reinforcement learning or expert user demonstrations. Alternatively, this could be

achieved by analysing video data of robots or humans interacting with objects and extrapolating these functions to a 3D model of the object. Although this approach is beyond the scope of the current work, the process of defining points of interest would remain the same. Finaly, it is important to note that this experimental section does not include direct comparisons with other methods from the literature. Instead, a discussion focused on the unique aspects of the proposed approach is presented. This is because no other method in the literature combines being learning-free, one-shot, and task-aware for grasp transfer with partial 3D data.

5.7 Summary

In this chapter, a method for transferring meaningful functions that depict the grasping region associated with a task between different objects of the same class has been demonstrated. Various experiments in both simulation and on the real robot were conducted to analyse different aspects of the method. Here, the FM framework was applied to achieve functional transfer between two objects. However, unlike in the Grasp Transfer (Chapter 3) and Skill Transfer (Chapter 4) methods, the focus here was on improving grasp success and achieving meaningful transfer over partially observed objects. This was accomplished using a shape reconstruction algorithm based on GPR, specifically GPIS and the uncertainty metric associated with it. Despite some limitations, the high success rate observed in the experiments underscores the potential of the proposed method to effectively transfer grasp regions across diverse scenarios.

Part III

Exploratory grasping

Chapter 6

Tactile-guided grasp refinement

Preliminary remarks

The research in this chapter was published in the following papers:

- **C. de Farias**, N. Marturi, R. Stolkin and Y. Bekiroglu. Simultaneous Tactile Exploration and Grasp Refinement for Unknown Objects. In IEEE Robotics and Automation Letters, 2021. doi: 10.1109/LRA.2021.3063074
- **C. de Farias**, N. Marturi, R. Stolkin and Y. Bekiroglu. Simultaneous Tactile Exploration and Grasp Refinement In Bridging the Gap between Data-driven and Analytical Physics-based Grasping and Manipulation II, ICRA 2021.

Note: Certain sections of this chapter, such as problem descriptions, specific equations, variable descriptions, interpretations of results, and figure/table captions, have been adapted from [58, 57].

6.1 Introduction

In Part I of this work, the concept of defining functions or grasps over objects and transferring them to either new but deformed instances of the same object or to objects within the same class was explored. Specifically, in Chapter 5, a method for transferring grasps to partially observed objects was presented. In this method, the shape of the objects was estimated using GPIS, and its underlying uncertainty metric was leveraged to increase the graspable area before a task-aware function was employed. While this approach proved successful, it has one key limitation within the context of shape estimation: the shape is extrapolated from the observable region of the object, which does not always accurately reflect its full structure. For instance, consider a mug with an occluded handle. When the handle is not visible, the GPIS often estimates the shape to be closer to a symmetric cup, which, while plausible, limits the robot's ability to find optimal

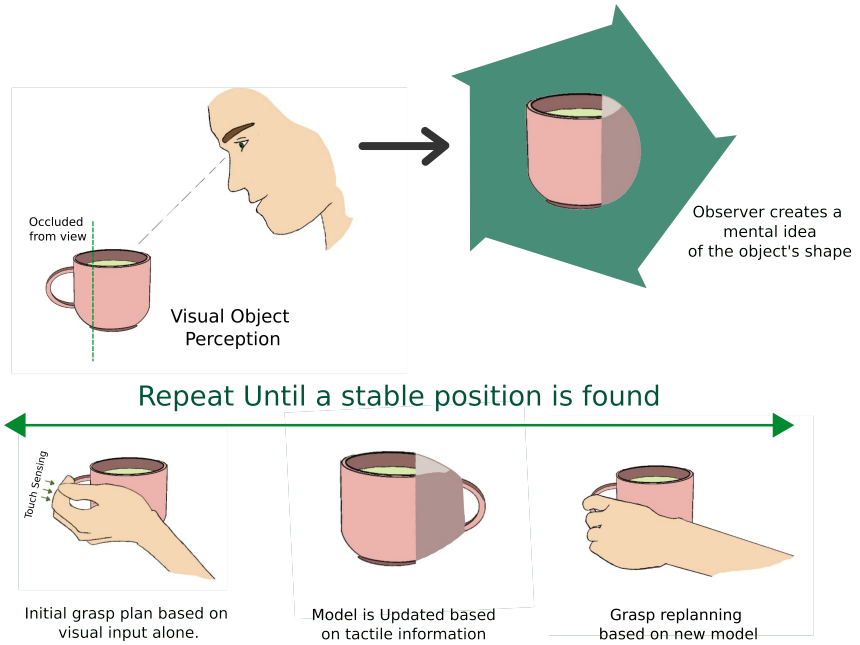


Figure 6.1: To grasp an object with hidden regions, relying on visual cues, humans first try to construct a perceptual model. They then attempt to grasp the object based on this model. Upon contact, if the actual object differs, they refine their perceptual model using tactile feedback, adjusting their fingers until a stable grasp is achieved.

grasps on the surface. For example, if the mug is too large for the gripper, the only feasible grasps are on the handle, causing grasp planning to fail under occlusion.

To address this, inspiration was drawn from biological models of human grasping, which highlight the crucial role of tactile information in achieving successful grasps [82, 95, 94]. In Figure 6.1, a human is shown encountering a similar mug. Initially, the mug may be perceived as symmetrical or smooth. However, upon touching the handle, tactile feedback updates the mental model, identifying the occluded handle. The fingers then adjust to find an optimal grasp, continuously refining the shape model based on tactile feedback.

From this observation, the key steps of an autonomous grasping algorithm for partially observed objects, enhanced by tactile feedback, can be identified as follows: *visual observation of the object*, *initial shape estimation*, *grasp planning*, and *exploration and re-planning*. Considering these steps, this chapter proposes an optimization framework that relies on Bayesian estimation techniques. With these steps in mind, a Bayesian Estimation based optimisation framework is proposed in this chapter. In this framework, successive finger placements are planned to *mutually improve* both grasp stability and shape representation. This approach enables tactile exploration to iteratively refine grasps and perceived shapes, ultimately resulting in successful grasping of partially observed objects based on combined visual and tactile information. Furthermore, this method is designed to overcome the limitations of approaches proposed in the previous chapters by incorporating tactile feedback, thereby allowing the robot to adjust its grasp in real-time as new shape information is acquired. By doing so, the proposed framework

not only enhances the robot's ability to handle partially observed objects but also improves the overall robustness and adaptability of the grasping process.

To this extent, the main contributions to the literature are:

- **Grasp refinement and shape exploration via Bayesian optimisation**: A novel method for simultaneous grasp refinements based on *Bayesian Optimisation* and shape exploration is proposed. This method enables a multi-finger robot equipped with tactile sensors to iteratively explore different finger placements, identifying areas with a high probability of successful grasps.

- **Integration of visual and tactile data for enhanced grasping:** The proposed approach integrates visual and tactile data to enhance the robot's ability to grasp partially observed objects. Initial shape estimations are made using visual data, which are then iteratively refined through tactile feedback obtained during successive grasp attempts. This integration allows for more accurate shape modeling and improved grasp stability.

- **Iterative probabilistic modeling for force closure and shape refinement:** An iterative probabilistic model for force closure is integrated with shape refinement, leveraging GPR. This model accounts for shape uncertainty within the overall grasping exploration process, enhancing the accuracy and reliability of grasp planning.

- **Comprehensive simulation analysis:** A thorough analysis of the method's performance is conducted through simulations. The proposed method is benchmarked against a naive random exploration approach, demonstrating significant improvements. Additionally, specific aspects of the model are analysed for various objects within the YCB dataset, providing detailed insights into its efficacy and robustness.

6.2 Problem Statement

Let $\mathbf{P}(0)$ represent the initial, partially observed point cloud of an object, obtained from a single depth camera view. This initial point cloud provides a limited and incomplete representation of the object's shape, as only the visible surfaces are captured. Here, the goal is to enhance the robot's understanding of the object's geometry as well as to find the best possible grasp by iteratively refining the shape estimate through tactile exploration. Let $\mathcal{S}_{\text{GPIS}}(k)$ be the reconstructed shape at iteration k, modeled using GPIS. This probabilistic representation enables the integration of both visual and tactile data, capturing the uncertainties associated with each modality. At the initial step ($k = 0$), the shape estimate $\mathcal{S}_{\text{GPIS}}(0)$ is constructed solely from $\mathbf{P}(0)$. As the robot performs an initial grasp attempt, it collects additional tactile data, $\mathbf{P}_{\text{tactile}}(k)$, from the object's occluded surfaces. These tactile glances are then incorporated into the initial cloud,

$$\mathbf{P}(k+1) = \mathbf{P}(0) \cup \bigcup_{i=1}^{k} \mathbf{P}_{\text{tactile}}(i), \tag{6.1}$$

which is then used to update the shape model $\mathcal{S}_{\text{GPIS}}(k+1)$. Mathematically, the process can be described as,

$$h_{\text{GPIS}}^{(k+1)}(\mathbf{x}) \sim \mathcal{GP}\Big(m(\mathbf{x}|\mathbf{P}(k+1)), k(\mathbf{x}, \mathbf{x}'|\mathbf{P}(k+1))\Big), \tag{6.2}$$

where $m(\cdot)$ is a mean function and $k(\cdot)$ the kernel function.

The goal then is to iteratively improve both the shape estimate simultaneous to the probability of success for a certain grasp configuration. If $\mathcal{G}$ is set of all possible grasps on the surface of an object and $\mathbf{g} \in \mathcal{G}$ the set of variables in the robot's configuration space describing one single grasp, then the problem becomes,

$$\mathbf{g}(k+1) = \arg\max_{\mathbf{x}^{BO}} \left[f\left(Pr(\mathcal{G}, \mathbf{x}^{BO}, k), h_{\text{GPIS}}^{k}(\mathbf{x}^{BO}) \right) \right] \tag{6.3}$$

with $Pr(\mathcal{G}, k)$ being a probability function indicating grasp success at moment k, $\mathbf{x}^{BO}$ a variable representing the configuration and location of the gripper and $f(\cdot)$ a function linking grasp success with surface uncertainty. This process is repeated iteratively until the desired grasp stability is achieved or a predefined number of iterations $k_{\max}$ is reached. As more information is gained in each step, it ensures refinement.

6.3 Background

In this section, the fundamental concepts employed in this method are outlined. Specifically, the fundamental details of Bayesian Optimisation (BO) and the concepts of *Force Closure* and its associated quality metrics are explained, including how these metrics can be extended to a probability function. GPR, which are essential for BO have been presented in Section 5.3.1 and GPIS used for shape reconstruction have been presented in Section 5.3.1.1.

6.3.1 Bayesian optimisation

Optimisation problems have been studied since the early days of mathematics. Classically, given a search space $\Omega \in \mathbb{R}^d$, these problems take the form,

$$x = \arg\max_{x \in \Omega} f(x) \quad \text{or} \quad x = \arg\min_{x \in \Omega} f(x), \tag{6.4}$$

where derivatives and gradients of the function $f(x)$ are often calculated to find optimal solutions. Typically, assumptions are made that $f(x)$ is easy to evaluate, its gradient is easy to find, and the function is known or convex. However, these assumptions do not hold for all functions. For

Algorithm 2: Bayesian Optimisation [67, 64]

Input : initial set of observations $\mathcal{D} = [\boldsymbol{\mathcal{X}}^{BO}(0), \boldsymbol{\mathcal{Y}}^{BO}(0)]$ (*obs. can be empty*)

1 **while** *termination condition not met* **do**
2 Update GP surrogate model (from (5.2)) model with $\mathcal{D}$
3 $\mathbf{x}' \leftarrow \arg\max_x$ Acquisition Function$(\mathbf{x}, \mathcal{D})$ ▶ Select next query point
4 $y' \leftarrow f(\mathbf{x}')$ ▶ Evaluate objective function
5 $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x', y')\}$ ▶ Update dataset with new observation
6 **end**

Return: Maximum observation in $x' = \arg\max_{x \in \Omega} \boldsymbol{\mathcal{Y}}_{BO}(k)$ (for maximisation)

instance, in robotics, when grasping an unknown object, the underlying function describing the task may be difficult to infer from data alone, making gradient-based methods impractical.

BO offers an alternative for tackling such "black-box" optimisation problems. Rooted in Bayesian Inference, BO treats the optimisation process as iterative, where each new observation informs the choice of subsequent observations. Beliefs about their behaviour can be modelled and exploited using stochastic (or more specifically, Bayesian) principles. In this scenario, the "black-box" function to be optimised is called the *Objective Function*, which, in this work, is assumed to have additive Gaussian error. Since the function is not explicitly known, a *Surrogate Function* is used to model the observed random variables. Here, a GP, as described in Section 5.3.1, is employed. The Bayesian framework then iteratively updates the surrogate model based on new function evaluations, refining the model's accuracy and uncertainty estimates over time [67, 32, 64, 172].

Particularly, BO typically requires two main components: the statistical model (or surrogate function) mentioned above and a method to choose the next location where the function will be evaluated. In BO, *Acquisition Functions* are used to make this choice, often balancing the exploration of the search space and exploitation of the surrogate model's beliefs about where the optimal value lies. There are a number of different Acquisition Functions in the literature, and the trade-off between exploration and exploitation is a well-studied problem [67]. Overall, the BO algorithm is summarised by Algorithm 2. Following, a more detailed discussion on the surrogate and acquisition functions used in this work are presented.

6.3.1.1 Surrogate function

BO provides a framework to reason about an uncertain objective function, where success is not based on beliefs about the data but on the outcomes of observations. Let the objective function of the process, i.e., the real unknown function that describes the observed values, be denoted as ϕ. Let $\mathcal{D} = [\boldsymbol{\mathcal{X}}_{BO}, \boldsymbol{\mathcal{Y}}_{BO}]$ represent the dataset of observed data[1]. Real-world processes are

[1]Here, $\boldsymbol{\mathcal{X}}_{BO}$ and $\boldsymbol{\mathcal{Y}}_{BO}$ differ from $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$ in Section 5.3.1.1 as the GPIS models the shape, whereas BO is applied to explore its surface.

inherently noisy; hence, assuming Gaussian noise $\epsilon \sim \mathcal{N}(0, (\sigma^{BO})^2)$,

$$y_i^{BO} = \phi(\mathbf{x}_i^{BO}) + \epsilon. \tag{6.5}$$

Given the GP from (5.2) (and the extended formulation in (5.5)), along with the chosen SE Kernel

$$k_{\text{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_{\text{SE}}^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2l^2}\right). \tag{6.6}$$

where $\sigma_{\text{SE}} = 0.001$ and $l = 1$ were the parameters used in the experiments, a surrogate function can be modelled as $f_{\text{surr.}} \sim \mathcal{GP}(0, k_{SE}(\mathbf{x}, \mathbf{x}'))$. This yields $\overline{f}_{\text{surr.}}(\mathbf{x}')$ and $\mathbb{V}_{\text{surr.}}(\mathbf{x}')$ at the query point $\mathbf{x}$ (as per (5.5)). This surrogate function is then explored using the acquisition function to efficiently search the space.

6.3.1.2 Acquisition function

The acquisition function $\alpha(\mathbf{x})$ quantifies the utility of sampling a new point $\mathbf{x}$. It is maximised to determine the next point to sample. A common and effective acquisition function is the Expected Improvement (EI), defined as

$$\alpha_{\text{EI}}(\mathbf{x}) = (\overline{f}_{\text{surr.}}(\mathbf{x}) - y^{\text{best}})\Phi(\Lambda) + \mathbb{V}_{\text{surr.}}(\mathbf{x})\varphi(\Lambda), \tag{6.7}$$

where $\Lambda = \frac{\overline{f}_{\text{surr.}}(\mathbf{x}) - y^{\text{best}}}{\mathbb{V}_{\text{surr.}}(\mathbf{x})}$, y^{best} is the best observed value so far, $\Phi(\cdot)$ is the Cumulative distribution function (CDF) of the standard normal distribution, and $\varphi(\cdot)$ is the Probability density function (PDF) of the standard normal distribution [64].

The EI function presented in (6.7) provides a way to balance exploitation and exploration components in the search strategy:

I - Exploitation: The term $(\overline{f}_{\text{surr.}}(\mathbf{x}) - y^{\text{best}})\Phi(\Lambda)$ focuses on exploiting current knowledge by sampling points that are expected to yield high values. When $\overline{f}_{\text{surr.}}(\mathbf{x})$ is significantly higher than y^{best}, Λ is large, making the CDF, $\Phi(\Lambda)$, approach 1. This increases the influence of this term, encouraging sampling in regions with high expected values.

II - Exploration: The term $\mathbb{V}_{\text{surr.}}(\mathbf{x})\varphi(\Lambda)$ encourages exploration by sampling points where the model's uncertainty is high. When $\mathbb{V}_{\text{surr.}}(\mathbf{x})$ is large, the exploration term becomes significant as $\varphi(\alpha)$ is maximised when Λ is around 0 (the higher probability in the PDF), indicating regions where the predicted mean is close to the best known value but with high uncertainty. This ensures that the algorithm explores less understood areas, refining the surrogate model and improving the understanding of the objective function.

By combining these two terms, the EI function balances the trade-off between exploration and exploitation. This balance ensures that the optimisation process is thorough, progressively

improving the model and efficiently navigating towards the global optimum.

6.3.2 Grasp stability analysis

In previous chapters, grasp planning has been simplified to finding regions where the finger (surface) regions of a two-finger parallel gripper have a high correlation with the object's surface [3]. However, in this case, a three-finger gripper is used. Multi-finger grippers are particularly advantageous for achieving finer manipulation and greater stability, as they provide multiple points of contact. Therefore, instead of relying on the metrics, e.g., LoCoMo score, in this chapter classical grasping dynamics are explored to identify regions on the object where the contact vectors are likely to create balanced forces.

Unlike heuristic approaches that rely on empirical data or simplified models, classical grasping dynamics provide a framework for analysing and optimising the interactions between the gripper and the object. This involves examining the mechanics of contact, force distribution, and the resulting stability of the grasp.

6.3.2.1 Grasp contact

In robotics grasping dynamics, understanding the interactions between the gripper and the object at the contact points is one of the most crucial elements. Typically, three primary contact models are used for grasp analysis, as illustrated in Figure 6.2. Despite the existence of more complex models, these three simplify the equations that govern the transmission of twists and wrenches between the gripper and the object. Following [174] these models are:

(i) **Point contact without friction**: This model is used when the contact area between the gripper and object is small, and the hand's surfaces are slippery. In such cases, only the normal component of linear velocity is transferred, while tangential and angular velocities, along with frictional forces, are considered negligible.

(ii) **Hard finger contact**: This model is used when a small contact patch with sufficient friction to avoid the object to slip is defined by a friction cone. It allows all three linear velocity components at the contact point, along with all three components of the contact force, to be transmitted. However, angular velocity components and moments are not transmitted through the contact.

(iii) **Soft finger contact**: This model is used when the contact patch and surface friction are sufficiently large to produce significant friction forces and a friction moment around the contact normal. Under these conditions, not only the linear force and velocity components can be transmitted, but also torques around the normal can also be exerted at the contact point.

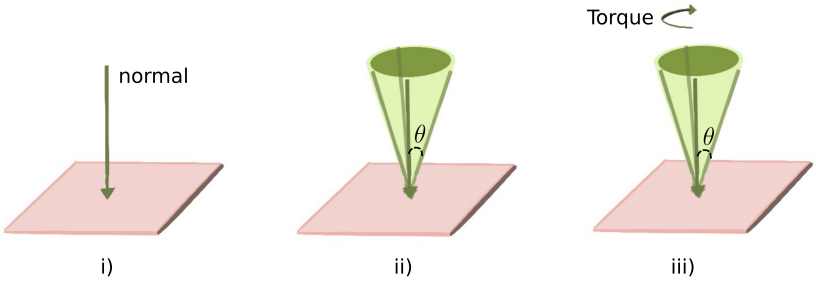


Figure 6.2: Illustration of contact models. (i) Represents point contact without friction; (ii) depicts the friction cone characteristic of hard finger contacts; and (iii) illustrates soft finger contact.

This work considers hard finger contacts, commonly linked to precision grasps where only the fingertips are in contact with the object. The effects of the forces applied at each fingertip contact can therefore be described by the ensuing wrench

$$\omega = \begin{pmatrix} \mathbf{w} \\ \mathbf{f} \end{pmatrix} \tag{6.8}$$

where $\mathbf{w}$ and $\mathbf{f}$ represent the torques and forces, respectively. For an object to be grasped stably, the equilibrium condition can be written as

$$\sum_{i=1}^{N} \omega_i + \omega_{\text{ext}} = 0 \tag{6.9}$$

where ω_i represents each of the N_f contacts (in the case of a three-finger gripper, $N_f = 3$) and ω_{ext} are the external wrenches being applied to the object [28].

Finally, to conclude the discussion on contacts, the forces applied at a hard-contact with friction can be modelled by the friction cone shown in Figure 6.2-(ii) and are expressed as

$$\mathbf{f}_i = \begin{cases} \mu \mathbf{n}_i \cos(\theta) & \text{if } \theta \leq \tan^{-1}(\mu) \\ 0 & \text{otherwise} \end{cases} \tag{6.10}$$

where μ is the friction coefficient and $\mathbf{n}_i$ is the normal force at the contact point. The friction cone angle is given by $\theta = \tan^{-1}(\mu)$ [124, 174].

6.3.2.2 Form and force closure

The concept of closure in grasp analysis refers to the ability of a gripper to hold an object against external disturbances, ensuring that (6.9) is satisfied. Two primary types of closure are usually considered: form closure and force closure [174].

Form Closure: Form closure is defined as a condition where the object is completely constrained by the contacts, without relying on friction. In this case, the hand wraps around the object, preventing all movement regardless of external forces.

Force Closure: Force closure, on the other hand, relies on friction to achieve stability. A grasp is considered force closure if the contact forces can generate a resultant force and moment that counteract any external wrench ω_{ext}.

Although form closure often provides maximum stability for a grasp, achieving it is more challenging as it requires a higher number of contact points, making it more complex to implement. Conversely, force closure can be easily achieved in the context of precision grasps, where frictional contact is sufficient to ensure stability. In this work, grasp success analysis is provided in the context of force closure.

6.3.2.3 Grasp quality

A number of grasp quality metrics have been proposed in the literature, most focusing on analytic approaches [28, 20, 106]. Here, the forces that lie within the friction cone are considered for each contact. The friction cone, often simplified to a polyhedral convex cone to facilitate linear analysis, can be defined by the positive span of the basis vectors forming the edges of the cone. That is, for a wrench ω_i, which can be decomposed into the basis vectors $\{\mathbf{v}_{i1}, \mathbf{v}_{i2}, \ldots, \mathbf{v}_{in}\}$, applied to the contact point $\mathbf{x}_{Fi}$, the Wrench Cone (WC) can be defined as

$$(WC)_i = span(\mathbf{v}_{in}) = \left\{ \sum_{j=1}^{n} \lambda_j \mathbf{v}_{ij} \mid \lambda_j \geq 0 \right\} \tag{6.11}$$

where λ_j are non-negative scalars gains.

Furthermore, the overall Grasp Wrench Space (GWS), which represents the set of all possible wrenches that can be applied to an object by the gripper, is defined by

$$\text{GWS} = span(\text{WC}_1, \text{WC}_2, \ldots, \text{WC}_{Nf}) \tag{6.12}$$

where WC_i denotes the wrench cone at the i-th contact point, and N is the total number of contact points. As the GWS encompasses all combinations of wrenches that can be exerted by the gripper, it provides a representation of the gripper's capability to manipulate the object. By ensuring that the resultant wrench from the gripper lies within the GWS, it is possible to establish whether the gripper can effectively apply the necessary forces and torques to maintain a stable grasp on the object.

In the context of force closure, this is achieved by testing whether the GWS includes the origin of the system. The origin in this context represents a state where all force and torque components are zero, indicating balance. If the origin is within the GWS, it implies that the

gripper can produce internal forces to counteract any external disturbance, ensuring stability. Practically, this can be assessed by approximating the GWS to a convex hull over the spanning vectors and checking if the origin is inside the hull [210]. Here note that if the convex hull approximation of the GWS is given by $\overline{\text{GWS}}$, then by definition $\overline{\text{GWS}} \subseteq \text{GWS}$ and therefore the origin of $\overline{\text{GWS}}$ is also contained in GWS.

While ensuring force closure is crucial for determining the feasibility of a grasp, it is also important to evaluate the quality of the grasp. Several grasp quality metrics are employed for this purpose, providing a quantifiable measure of how well a grasp can perform under various conditions. These metrics are essential as they allow for comparing different grasp configurations, optimising for the best grasps, and understanding how well the grasp can resist external wrenches for a given bound on the contact forces [210].

One widely used grasp quality metric is the Ferrari-Canny metric [62, 28]. This metric quantifies the quality of a grasp by evaluating the smallest wrench that can be resisted in any direction when unit strength forces are applied at the contact points. The Ferrari-Canny metric is calculated as the radius of the largest ball that can be inscribed within the convex hull of the GWS, centred at the origin (denoted by ε_ω). This provides a measure of the grasp's robustness, as it represents the maximum disturbance the grasp can resist uniformly in all directions.

6.4 Surface exploration and exploitation

In this section, the method for simultaneous tactile exploration and grasp stability exploitation is introduced. First, an overview of the proposed algorithm is presented. This is followed by a detailed explanation of the probabilistic model used for grasping. Finally, an in-depth analysis of the proposed algorithm is conducted.

6.4.1 Method overview

To implement the system inspired by Figure 6.1, several key components are required: (i) a method for acquiring data and estimating the shape, (ii) a *target function* to interpret the observed data and guide optimisation goals, and (iii) a procedure for searching for optimal grasps while simultaneously optimising the shape.

Let $\mathbf{P}(0)$ is the initially observed partial point cloud of an object at moment $k = 0$, defined as

$$\mathbf{P}(0) = \{\mathbf{x}_i(0) \mid \mathbf{x}_i(0) \in \mathbb{R}^3,\ i \in \{1, 2, \ldots, N(0)\}\} \tag{6.13}$$

with $N(0)$ being the number of points in the cloud and $N(k)$ the points in subsequent iteration once (6.1) is updated. While grasps could be planned directly on the partial cloud using LoCoMo or the partial grasping method proposed in the previous chapter, the main interest here is in iteratively refining those initial estimates through directed tactile exploration. From (6.2),

(5.2) and (5.6), the initial shape $\mathcal{S}_{\text{GPIS}}(0)$ of the object can be estimated from the data provided by $\mathbf{P}(0)$. With each new tactile exploration, the cloud is updated as in (6.1), and the GPIS process is reapplied, generating a new $\mathcal{S}_{\text{GPIS}}(k)$ for each $\mathbf{P}(k)$ at a given moment k.

With the shape estimate obtained, the next steps involve analysing the data from each updated shape and the new tactile data to develop a target function that can guide the optimisation process. The target function, which is given as $f : \mathbf{x}^{BO}(k) \rightarrow \mathbb{R}, \mathbf{x}^{BO}(k) \in \mathbb{R}^M$, where M is the dimensionality of the input space, must take into account both the probability of grasp success based on the metrics described in Section 6.3.2 as well as the confidence in the shape. The concepts of Probablility of Force Closure (PFC) and how it is used together with the GPIS model to generate the target function are further explored in the following sections.

Finally, the exploration and search procedure aiming to optimise (6.3) is defined. The best gripper configuration is searched through a process based on BO that aims to optimise the target function $f_t\left(Pr(\mathcal{G}, \mathbf{x}^{BO}, k), h^k_{\text{GPIS}}(\mathbf{x}^{BO})\right)$. That is, given the estimated shape GPIS function $h^k_{\text{GPIS}}(\cdot)$ and the probability of grasp success $Pr(\mathcal{G}, \cdot, k)$ the BO algorithm suggests a query point $\mathbf{x}^{BO}_i(k)$ that should maximise f. The robot is then moved to the suggested configuration to evaluate the target function at the queried location based on the new information from the robot's tactile sensors placed at its fingertips. The evaluation value is then given by

$$y^{BO}_i(k) = f_t\left(Pr(\mathcal{G}, \mathbf{x}^{BO}, k), h^k_{\text{GPIS}}(\mathbf{x}^{BO})\right). \tag{6.14}$$

With the new data pair $[\mathbf{x}^{BO}_i(k), y^{BO}_i(k)]$, the GPIS and BO models are updated and refined, and a new data point is suggested. Here the set of BO observations is denoted $\mathcal{D}^{BO}(k+1) = [\boldsymbol{\mathcal{X}}^{BO}(k+1), \boldsymbol{\mathcal{Y}}^{BO}(k+1)]$. This algorithm is summarised in Figure 6.3 and is iteratively repeated until a stop criterion is met.

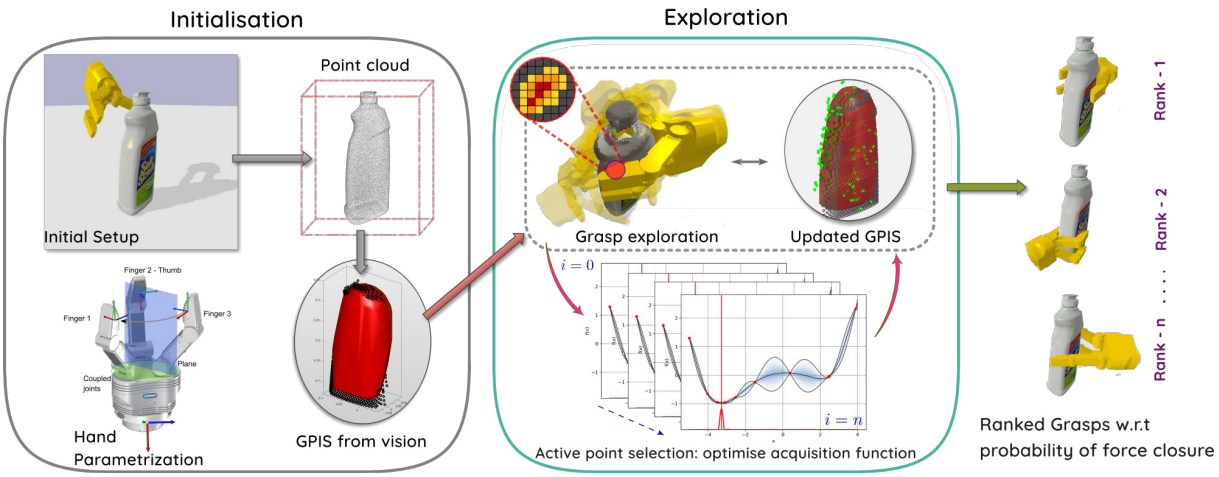


Figure 6.3: BO-based exploration pipeline for shape reconstruction and grasp search.

6.4.2 Probabilistic grasp stability

When refining grasps, various aspects need to be considered. As highlighted in Section 6.3.2, factors such as force and form closure can be taken into account when evaluating grasps. Additionally, grasp quality metrics like the Ferrari-Canny measure, which relates to force closure [62], are employed in this work. Specifically, the quality of a grasp is computed as the smallest wrench (ε_ω) that can break the hold over an object (see Section 6.3.2.3 for more details). Although this, along with other grasp quality metrics, provides a framework for analysing and comparing different grasp configurations, precise parameters are often required. However, such parameters are usually uncertain in realistic settings [129, 36, 109, 194].

To obtain precise grasp quality values, the finger placement must be known exactly. The value for surface friction, which yields the angle of the friction cone as $\theta = \tan^{-1}(\mu)$ for a friction coefficient μ, must be available. Additionally, it must be possible to calculate the actual centre of mass of the object so that the force closure analysis can account for correct the balancing of resulting moments from finger contacts. This scenario, of course, is an ideal one. The robotic fingers can have uncertainties in their encoders, or the kinematic models can lack precision, resulting in inaccurate estimations of the contact pose. Even a few millimetres of deviation can significantly deteriorate grasp quality. Estimating friction is also challenging, even with state-of-the-art learning algorithms, as objects within the same category can be made of different materials and are susceptible to environmental changes such as humidity and dust. Furthermore, the nature of the problem itself introduces uncertainty in the shape and centre of mass. The geometric centre of the object is estimated through the object's shape, which is uncertain as the shape stems from the current GPIS and is constantly refined as the robot interacts with the object. Consequently, the calculation of the centre of mass can vary widely as the algorithm progresses. Even if the shape was precisely known, factors such as different materials and mass distribution within the object are difficult to estimate accurately a priori without physically lifting and observing the object's dynamics, thus making the centre of mass estimation uncertain even when the geometric centre of the object is known.

Given these challenges, an approach similar to [129] is adopted in this work, where instead of force closure, the probability of force closure is considered. Let $\boldsymbol{n}^i$ be the contact normals resulting from the robot's i-th finger contacting the surface at point $\boldsymbol{c}^i$, μ be the friction coefficient, and $\boldsymbol{p}_{\text{com}}$ the centre of mass. Rather than using imprecise measurements for these variables, they are modelled as Gaussian distributions: $\boldsymbol{n}^i \sim \mathcal{N}(\hat{\boldsymbol{n}}^i, \sigma^2_{\boldsymbol{n}^i})$, $\boldsymbol{c}^i \sim \mathcal{N}(\hat{\boldsymbol{c}}^i, \sigma^2_{\boldsymbol{c}^i})$, $\mu \sim \mathcal{N}(\hat{\mu}, \sigma^2_\mu)$, and $\boldsymbol{p}_{\text{com}} \sim \mathcal{N}(\hat{\boldsymbol{p}}_{\text{com}}, \sigma^2_{\text{com}})$, where $\hat{\boldsymbol{n}}^i$, $\hat{\boldsymbol{c}}^i$, $\hat{\mu}$, and $\hat{\boldsymbol{p}}_{\text{com}}$ are the estimated means, and $\sigma^2_{\boldsymbol{n}^i}$, $\sigma^2_{\boldsymbol{c}^i}$, σ^2_μ, and σ^2_{com} are the variances of the normal distributions.

Considering these sources of uncertainty, the probability of force closure (PFC) is defined as

$$Pr^{FC} = Pr(\varepsilon_\omega > \delta \mid \boldsymbol{n}^i, \boldsymbol{c}^i, \mu, \boldsymbol{p}_{\text{com}}) \quad \forall i \in \{1, 2, \ldots, N_f\}, \tag{6.15}$$

where $\delta > 0$ is an empirically set threshold.

6.4.3 Target function

BO, as described in (6.5), is used to optimise "black-box" functions by making observations and guiding the exploration process. The target function $f_t(\cdot)$ in (6.14), is responsible for taking the current state of the system and converting it into a meaningful value. But how should this function be defined? One might assume that considering only the PFC would be sufficient to find stable grasps. However, this assumption can be misleading. Consider the following scenarios:

(i) **Data derived from finger configuration alone**: In the simplest case, normals and contact points are obtained solely from finger placements. Without information about actual contacts, the BO algorithm might prioritise a configuration where each of the fingers generates vectors that are balanced but are actually outside of the surface of the object, thus resulting in unsuccessfully grasps.

(ii) **Data derived from tactile patches**: In this scenario, the PFC is calculated from tactile information. Here, the contact point of the finger comes from the centre of the contact patch, and normal information from the average plane perpendicular to it. This yields a function $f_t(\cdot)$ of the form

$$f_t\Big(Pr(\mathcal{G}, \mathbf{x}^{BO}, k)\Big) = Pr^{FC}, \tag{6.16}$$

where the PFC could be saturated to zero whenever a finger does not find a contact point. Although this approach yields some level of success, using PFC alone has a significant drawback. Over values of $\mathbf{x}^{BO}$ that fall on the surface of the object, the function yields probabilities. However, around the rest of the workspace, which is often much larger than the object, $f_t(\cdot) = 0$. As the shape is unknown, and part of the BO algorithm enforces exploration of areas with little information but potential for improvement, having the space outside the object as a constant often results in slower convergence rates, as the robot is encouraged to explore more of the empty space without gradients guiding it to any other region.

(iii) **Data derived from finger configuration and GPIS**: In this final case, the GPIS is used alongside the PFC to provide more information about the space. From (5.6), it is known that the space outside the object is set as $h_{\text{IS}} = 1$, while the surface is $h_{\text{IS}} = 0$. Thus, $h^{(k+1)}_{\text{GPIS}}(\mathbf{x})$ (see (6.2)) tends to grow as the fingers move away from the surface. This can be used as a penalty term to further guide the robot towards the surface, resulting in the function

$$f_t\Big(Pr(\mathcal{G}, \mathbf{x}^{BO}, k), h^k_{\text{GPIS}}(\mathbf{x}^{BO})\Big) = \lambda Pr^{FC} - \sum_{i=1}^{N_F} \Big[\overline{h_{\text{GPIS}}}^k(\mathbf{x}_{F_i})\Big]^2, \tag{6.17}$$

where N_F is the number of fingers, $\mathbf{x}_{F_i}$ the fingertip locations reached after the suggested configuration $\mathbf{x}^{BO}$, and λ is a scalar gain set empirically.

In this work, the third approach, which combines PFC from tactile information with GPIS, is used. This integrated approach not only improves the robot's ability to identify stable grasps but also accelerates the learning process by effectively guiding the exploration.

6.4.4 Input space description

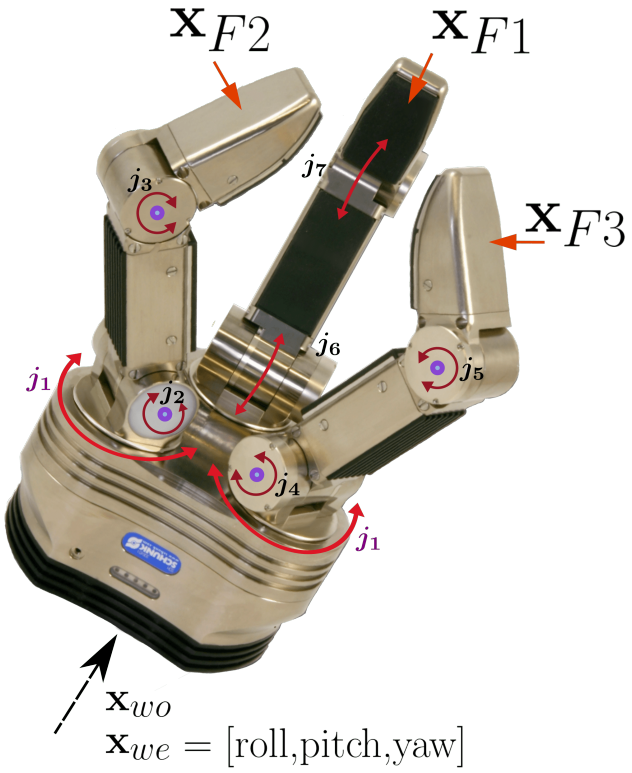


Figure 6.4: SDH with the each of the fingers highlighted. The offset $\mathbf{x}_{wo}$ and the vector of Euler angles $\mathbf{x}_{we}$ are applied to the approach vector depicted in the image. Image modified from [177].

For BO based algorithms, defining the input space on which optimal values will be searched is crucial. While parametrisation can be achieved in different ways, the approach here leverages the degrees of freedom of the used Schunk SDH gripper (shown in Figure 6.4) for efficient exploration and grasp planning. Unlike other works [144, 33], which have shown successful optimisation using wrist position and a power grasp, the proposed method achieves more precise grasping by taking advantage of the full configuration space of the robot rather than just the wrist position within the 6D task-space. Though this parametrisation is specific to the gripper used, similar results can be achieved with other grippers merely replacing the kinematic description.

With the three-finger Schunk SDH hand, the query point, $\mathbf{x}^{BO}$, is defined as the desired 3D global position for the thumb and the first finger. Due to the constraints imposed by the coupled SDH joint, the last finger's position is restricted to a 2D plane relative to its knuckle. The wrist

position is constrained by the chosen fingertip locations; thus, once its position is determined, the orientation is defined by Euler angles around the approach vector, plus an offset on the position. Overall, the input space is composed of $M = 12$ dimensions, which include:

- The 3D positions for the thumb and the first finger, denoted as $\mathbf{x}_{F1}, \mathbf{x}_{F2} \in \mathbf{x}^{BO}$,
- The 2D position of the last finger, $\mathbf{x}_{F3} \in \mathbf{x}^{BO}$, in the plane relative to the knuckle[2],
- The Euler angles for the wrist orientation, $\mathbf{x}_{we} \in \mathbf{x}^{BO}$,
- An offset distance along the approach vector for the wrist, allowing it to move closer to the object. $\mathbf{x}_{wo} \in \mathbf{x}^{BO}$.

The hand is positioned accordingly for each query point, and the model is updated with the resulting observations after the fingers are closed. It is worth noting that the model is not updated with the suggested configurations, but with the ones actually achieved by the robot. This is done because there are instances where a surface might be encountered before the finger reaches its desired position, or, when no surface is found, the robot is encouraged to keep closing the grasp in case the surface lies below the suggested point. Furthermore, two additional variables are defined here, the resulting wrist pose at iteration k, $\mathbf{q}(k) \in SE(3)$ and the resulting joint configuration vector $\mathbf{j}(k)$ once the final pose is reached.

6.4.5 Simultaneous grasp exploration and object reconstruction algorithm

In this section, the complete algorithm for simultaneously performing tactile exploration to reconstruct object surface models while also searching for stable grasp configurations using a three-fingered dexterous hand is presented. The algorithm is summarised in Algorithm 3.

The overall pipeline consists of three main components: initialisation, BO-based tactile exploration, and post-exploration evaluation. During the pre-exploration phase, the system is initialised with the hand placed in its initial home configuration, positioned above the object with fingers open. Using a 3D camera mounted in the scene, the initial point cloud $\mathbf{P}(0)$ of the object is captured, providing a partial view of the object visible to the camera. A bounding box is placed around the object point cloud to define the boundaries of the domain. Then the initial GPIS based surface $\mathcal{S}_{\text{GPIS}}(0)$ is constructed using this visual information.

Once initialised, the BO algorithm starts, the GPIS model will be updated every k^{th} iteration with new tactile observations $\mathbf{P}_{\text{tactile}}(k)$. Values for PFC and the target function (namely Pr^{FC} from (6.15) and $f_t(\cdot)$ from (6.17)) calculated in the process of searching the best configuration. The algorithm follows these steps:

[2]the 3D position for the last finger can be easily extracted from the other variables, so for simplicity $\mathbf{x}_{F3}$ will be used interchangeably for the 2D and 3D variables

I **Initialisation:** The GPIS model $\mathcal{S}_{\text{GPIS}}(0)$ is generated from the initial point cloud $\mathbf{P}(0)$, and the initial centre of mass, $\boldsymbol{p}_{\text{com}}(0)$, is estimated. Mean and variance values for the friction coefficient ($\hat{\mu}$, σ_{μ}^2), variance for the normals ($\sigma_{\boldsymbol{n}^i}^2$), and variance for the contact points ($\sigma_{\boldsymbol{c}^i}^2$) are selected based on prior knowledge. Additionally, to avoid initialising the $\mathcal{D}$ empty (such as in Algorithm 2) a few random explorations are done prior to the experiments and fed to the initialisation[3].

II **New query point suggestion:** The acquisition function (6.7) suggests a new point to explore. For the three-finger Schunk SDH hand, the query point, $\mathbf{x}^{BO}$, is described as in Section 6.4.4.

III **Tactile exploration:** The robot moves to the suggested configuration[4]. If not reachable the robot explores the closest reachable option instead. From sensory feedback, the contact points $\hat{\boldsymbol{c}}^i$ and normals $\hat{\boldsymbol{n}}^i$ for each finger are obtained. If all fingers reach a contact area, N_S samples for each variable ($\boldsymbol{n}^i$, $\boldsymbol{c}^i$, μ, $\boldsymbol{p}_{\text{com}}$) are drawn from their respective Gaussian distributions. The new points from $\mathbf{P}_{\text{tactile}}(k)$ are then saved.

IV **PFC calculation:** The PFC is computed as described in Sections 6.3.2.3 and 6.4.2, and particularly (6.15). If any finger does not detect contact, its position is recorded, and Pr^{FC} is set to zero for that configuration.

V **Model update:** At iteration k, using (6.17) and (6.14), the target function and $y^{BO}(k)$ are updated, respectively. Here the GPIS and surrogate function $f_{\text{surr.}}$ are updated with the new observations. Namely, the GPIS is updated by estimating a new shape from the new cloud $\mathbf{P}(k+1)$ obtained as in (6.1) with the additional data points stemming from tactile data, $\mathbf{P}_{\text{tactile}}(k)$. Then $\mathbf{x}^{BO}$ is updated with data from the actual configuration reached by the gripper and the $f_{\text{surr.}}$ is updated by adding the new data pair $[\mathbf{x}_i^{BO}(k), y_i^{BO}(k)]$ to the dataset $\mathcal{D}^{BO}(k+1) = [\boldsymbol{\mathcal{X}}^{BO}(k+1), \boldsymbol{\mathcal{Y}}^{BO}(k+1)]$. Additionally, the centre of mass $\boldsymbol{p}_{\text{com}}$ is refined from the mean and variance values from the GPIS estimate.

VI **Iteration:** The process is repeated until the number of desired iterations $k_{\max}$ is reached or a stop criterion is met. The algorithm returns a list $\mathbf{Q}$ of stable wrist poses and the associated joint configurations, $\mathbf{J}$, with high PFC values for further evaluation.

During post-exploration evaluation, grasps with the highest PFC values are executed and tested for stability through lifting and shaking tests to verify their stability[17].

[3]Note that initialising the exploration process with an empty dataset is just as valid. However, BO literature, including [67], often discuss the algorithm in terms of budget. Some of the budget available for experiments can be used for random (or structured) exploration and then the BO will have a starting point to start optimisation. When initialising $\mathcal{D} = \varnothing$ not much evaluation of the proposed algorithm was possible, as there was not much to guide exploration, so as a design choice in this work a "random exploration" prior was used.

[4]An out of the box controller was used for the arm and the fingers joints were obtained from an inverse kinematics based model

Algorithm 3: Exploration and grasp refinement.

input : GPIS, priors, $\hat{\mu}$, σ^2_{μ}, $\sigma^2_{\boldsymbol{n}^i}$, $\sigma^2_{\boldsymbol{c}^i}$, $\boldsymbol{p}_{\text{com}}(0)$, N_S, $k_{\max}$

output : $\{\mathbf{Q}, \mathbf{J}\}$

while *TRUE* **do**
- $\mathbf{x}^{BO} \leftarrow$ from EI in (6.7);
- $\mathbf{x}_{Fi} \forall i = \{1, ...N_F\}, \mathbf{x}_{we}, \mathbf{x}_{wo} \leftarrow \mathbf{x}^{BO}$
- $\mathbf{q}(k), \mathbf{j}(k) \leftarrow \mathbf{x}_{F_i}, \mathbf{x}_{we}, \mathbf{x}_{wo}$
- Move robot to $\mathbf{q}(k)$ and $\mathbf{j}(k)$ configuration
- $\hat{\boldsymbol{c}}^i, \hat{\boldsymbol{n}}^i \leftarrow$ sensory feedback, $\quad \forall i \in \{1, ...N_F\}$
- **if** $\hat{\boldsymbol{c}}^i \neq NULL, \forall \hat{\boldsymbol{c}}^i$ **then**
  - $\mathbf{x}_{F_i} \leftarrow \hat{\boldsymbol{c}}^i \quad \forall i \in \{1, ...N_F\}$
  - $P_{\text{FC}} \leftarrow \{\boldsymbol{a}_1 \dots \boldsymbol{a}_{N_S}\} \sim \mathcal{N}(\hat{\boldsymbol{a}}, \sigma^2_{\boldsymbol{a}}), \forall \boldsymbol{a} \in \{\boldsymbol{n}^i, \boldsymbol{c}^i, \mu, \boldsymbol{p}_{\text{com}}\}$
- **else**
  - **for** $i \in \{1, ...N_F\}$ **do**
    - **if** $\hat{\boldsymbol{c}}^i = NULL$ **then**
      - $\mathbf{x}_{F_i} \leftarrow \text{FKM}(\mathbf{j}(\mathbf{k}))$[a]
      - $P_{\text{FC}} \leftarrow 0$
    - **else**
      - $\mathbf{x}_{F_i} \leftarrow \hat{\boldsymbol{c}}^i_i$
    - **end**
  - **end**
- **end**
- $f_t\left(Pr(\mathcal{G}, \mathbf{x}^{BO}, k), h^k_{\text{GPIS}}(\mathbf{x}^{BO})\right) \leftarrow$ (6.17)
- Update $\mathbf{x}^{BO}$ with $\mathbf{x}_{F_i}$ and reached $\mathbf{x}_{we}, \mathbf{x}_{wo}$
- Update $f_{\text{surr.}}$ with $\mathcal{D}^{BO}(k+1) = [\boldsymbol{\mathcal{X}}^{BO}(k+1), \boldsymbol{\mathcal{Y}}^{BO}(k+1)]$ with $[\mathbf{x}^{BO}_i(k), y^{BO}_i(k)]$
- Update GPIS with $\mathbf{P}_{\text{tactile}}(k)$
- Update $\boldsymbol{p}_{\text{com}}$ from GPIS
- $k = k + 1$
- **if** $Pr^{FC} > 0.5$ **then**
  - $\mathbf{Q}, \mathbf{J} \leftarrow \mathbf{q}(k), \mathbf{j}(k)$
- **end**
- **if** $k = k_{\max}$ **then**
  - **break**
- **end**

end

[a] FKM indicates the Forward Kinematics Model, which obtains the positions $\mathbf{x}_{F_i}$ from joints.

6.5 Experimental Validations

In this section, the experimental validations for the proposed method are presented. All experiments are conducted in simulation, including an analysis of how the shape evolves during the refinement process and its impact on grasp success. Additionally, these experiments are compared to a baseline based on random grasp exploration.

6.5.1 Simulation Setup



Figure 6.5: Objects used for simulation experiments.

Simulation experiments were conducted using the Pybullet simulator with a 7-DoF robot, equipped with a three-finger Schunk SDH2 hand featuring tactile pads on the fingertips (see Figure 6.4). This setup, though simulated, was designed to mimic real-life conditions and facilitate an easy transfer to a physical setup. Initially, a proof of concept was demonstrated with a free-floating hand to avoid constraints due to reachability. Next, experiments were performed with the full robot arm to evaluate the approach in a more realistic setting. Grasp evaluations were validated against a baseline method based on random exploration. Seven different object models (shown in Figure 6.5) from two benchmark sets were used in the experiments: five from the YCB object set [31] and two from the CapriDB database [154]. All experiments were executed on an Ubuntu 20.04 machine with an i7 4-core CPU. Furthermore, to calculate (6.17), the following was used: $\sigma^2_{\boldsymbol{n}^i} = \pi/8$, $\sigma^2_{\boldsymbol{c}^i} = 0.0025$, $\sigma^2_{\mu} = 0.1250$, σ_{com} which is obtained from the object model, $N_S = 10$.

The grasp evaluation protocol employed is outlined in [17]. It involves lifting the object 20 cm above the table and shaking it by rotating the robot's fifth joint along a sinusoidal trajectory (frequency 0.4 Hz at 0.59rad/s). Success scores were computed based on the aggregate performance across both lifting and shaking tests. For GPIS reconstruction, the rate of improvement of the Hausdorff distance from the ground truth was used to evaluate the final shape after all explorations. Additionally, experiments testing the number of iterations until convergence to a good grasp are presented.

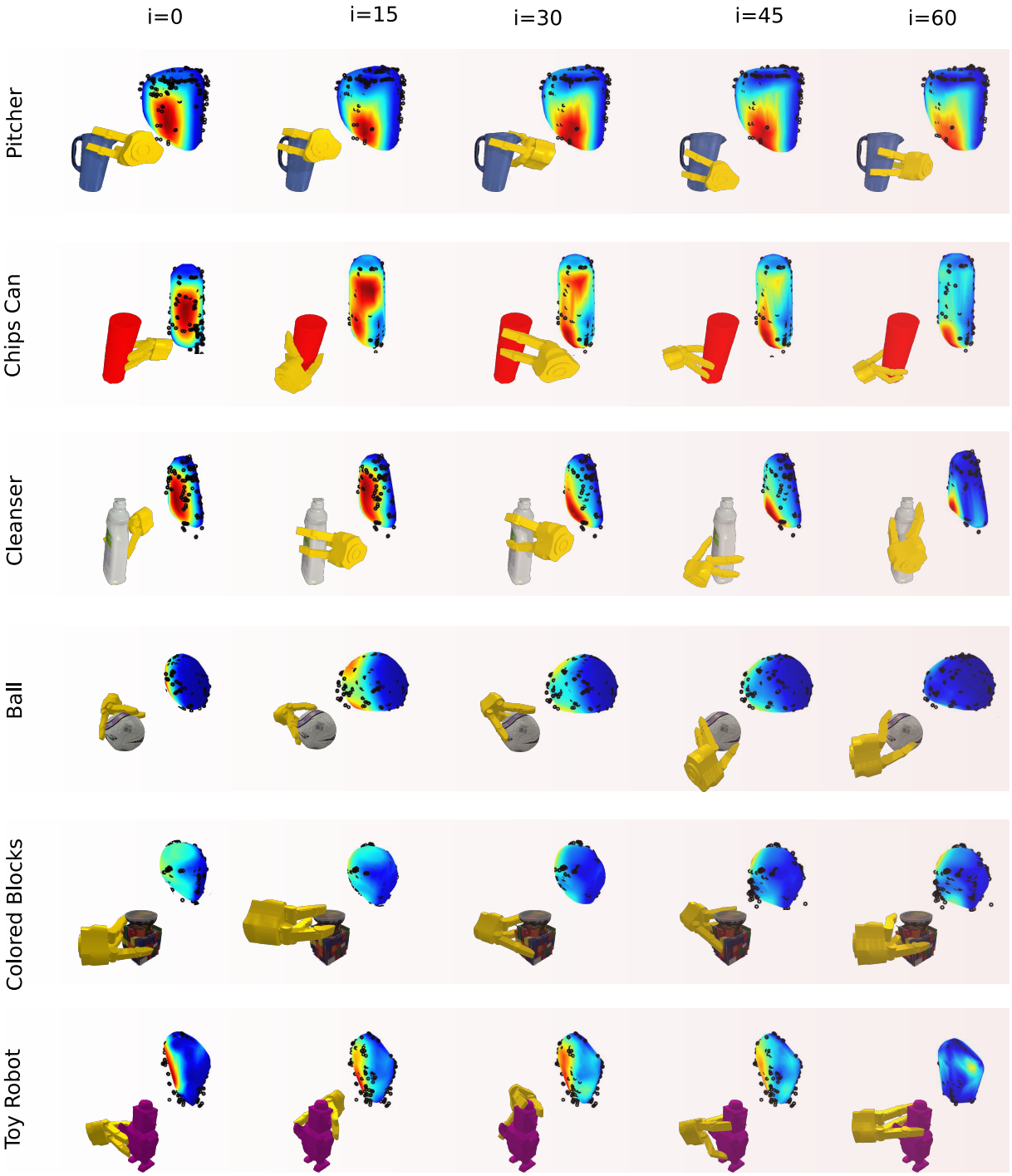


Figure 6.6: Experimental results for the shape evolution based on GPIS. The figure shows how the object shape estimation evolves as the algorithm suggests new query points.

6.5.2 Shape evolution evaluation

The shape evolution of objects during the tactile exploration process is depicted in Figure 6.6 where the iterative improvement of the GPIS models at various stages of the exploration is shown. Specifically the state of the objects at iterations $k = 0, 15, 30, 45, 60$ is shown. The

initial GPIS, constructed solely from visual data, is shown in the first column, while the final GPIS, incorporating both visual and tactile data, is presented in the last column. Tactile points obtained from contacts are marked in black. As these observations are acquired, the variance of the GPIS in those areas decreases, and the constructed surface adjusts to follow the observation points.

During the exploration, it was observed that the uncertainty in the shape model visibly decreases in regions where more tactile exploration occurs. However, it does not entirely disappear across the entire object. This phenomenon is due to the BO's trade-off between exploration and exploitation. At a certain point, the optimisation algorithm deems further exploration in some areas is unnecessary, resulting in those regions remaining unexplored. Specifically, this happens due to (6.7), where the variance across the search space falls bellow the term related to the the best value observed so far. At this point, when evaluating the function, the selected values will then be closer to the best observations.

Note that the GPIS reconstructed models are estimates; henceforth, the final shapes may not perfectly resemble the actual objects. This is particularly evident in the case of the "Ball" object, where the lack of reachability in the bottom half resulted in significant portions being missed. Despite substantial improvements from the initial severely underestimated shape to the final iteration, it still deviates from reality. Achieving perfectly accurate shapes was not the primary objective of the method. The goal was to obtain better estimates that facilitate grasping and encourage exploration aiming grasp refinement in areas not visible through vision alone, and perfect reconstruction are not always necessary in this case. One particular aspect that can be noted is that finer details of the object shapes remain coarse. For example, with the "Toy Robot," although visual data could capture some points on the arms, a significant portion remained occluded. The GPIS tends to fit a smoother surface in such cases. While improvements could be made to capture more details, such as tuning the noise parameters or enhancing the method, the limitations are also due to the tactile sensors' inability to pick up these details. Future work aims to investigate how fine-grained information impacts grasping in this algorithm.

Furthermore, Table 6.1 presents the evolution of the GPIS mean and variance for each object during tactile exploration. It highlights the reduction in the distance to the ground truth from the start to the end of the exploration, as well as the decrease in GPIS variance. These results quantitatively show that both the average distance to the ground truth and the surface uncertainty reduced over the course of the exploration, demonstrating that the proposed method effectively enhances the accuracy of object models through ongoing tactile exploration.

6.5.3 Grasp refinement evaluation

Figure 6.7 illustrates the evolution of the optimum PFC (Pr^{FC}) over 30 experiments, along with corresponding grasp examples showing improved Pr^{FC} values. Local adjustments in both hand positions and joint configurations lead to more stable grasps, which pass lifting and

shaking tests. For most objects, grasps with $Pr^{FC} > 0.8$ are found within approximately 20 iterations. The initial grasps evolve quickly from a random-exploration-based prior, but after the initial improvement, a slower increase rate is observed. This indicates the algorithm's focus on refining on its prior knowledge of the best candidate, often leading to successful results. Some objects, like the "Ball", present more challenges due to their geometry, resulting in a slower increase in success rates. In contrast, simpler objects such as the cylindrical "Chips Can" exhibit high initial success rates due to the ease of finding stable grasps, but even in those cases refinement is observed.

The "Book" is a notable example in the experiments. Initially, the success rates for the it remained near zero, with the algorithm struggling to find stable grasps. This was due to the larger sides of the book being searched for, which are unsuitable for grasping, as they did not fit within the hand's maximum stroke. As the exploration continued, the algorithm eventually identified the smaller sides as optimal grasp points. This discovery led to a dramatic improvement in success rates, evidenced by a sharp increase in Pr^{FC} values in Figure 6.7. Once these positions were identified, the variance in success rates increased. This suggests that, despite the initial difficulties faced in the experiments, a wide range of possibilities emerged over time. This result highlights the algorithm's ability to adapt and refine its strategy based on the physical constraints and characteristics of the objects.

Figure 6.8 presents stable grasp configurations that succeed in lifting tests, along with the resulting contact point density overlaid on the ground truth model of the object. Here it can be seen that certain contact areas are favoured, these are the regions leading to more stable grasps, i.e. *convergence regions*. By increasing the exploration duration, the number of stable grasps and Pr^{FC} values generated by the system can be further increased.

Furthermore, the trade-off between exploration and exploitation is clearly illustrated in Figure 6.8. This is achieved through the EI acquisition functions, described by (6.7). Through it, the robot is encouraged to explore regions near known good grasps that exhibit high variance. This approach ensures that the algorithm capitalises on promising areas while still gathering valuable information. As the variance in these regions decreases, indicating that the algorithm has gained sufficient confidence in the quality of grasps, it begins to suggest points further away. This strategy allows the robot to refine or expand the search area based on the positive results obtained. The result is a dynamic and adaptive exploration process where the robot systematically improves its grasping capability by balancing the immediate benefits of exploitation with the long-term gains of exploration.

In Figure 6.8, the effects of the trade off between exploration and exploitation in other objects can also be seen. In the case of the "Chips Can," a high density of blue points is observed along a large section on the side, expanding from a particular cluster. This indicates that stable grasps were discovered throughout the region, leading to extensive exploration. Conversely, for the "Book" object, initial exploration results in sparse contacts along the smaller face, and as the algorithm identifies the bigger sides as optimal touch points, a denser cluster of tactile points

emerges in these regions, demonstrating successful exploitation.

Finally, Figure 6.9 shows the algorithm can successfully operate with a full arm. Here, unreachable grasps were simply discarded, but alternative strategies based on BO have been proven positive in works such as [33] and are planned to be explored in future works.

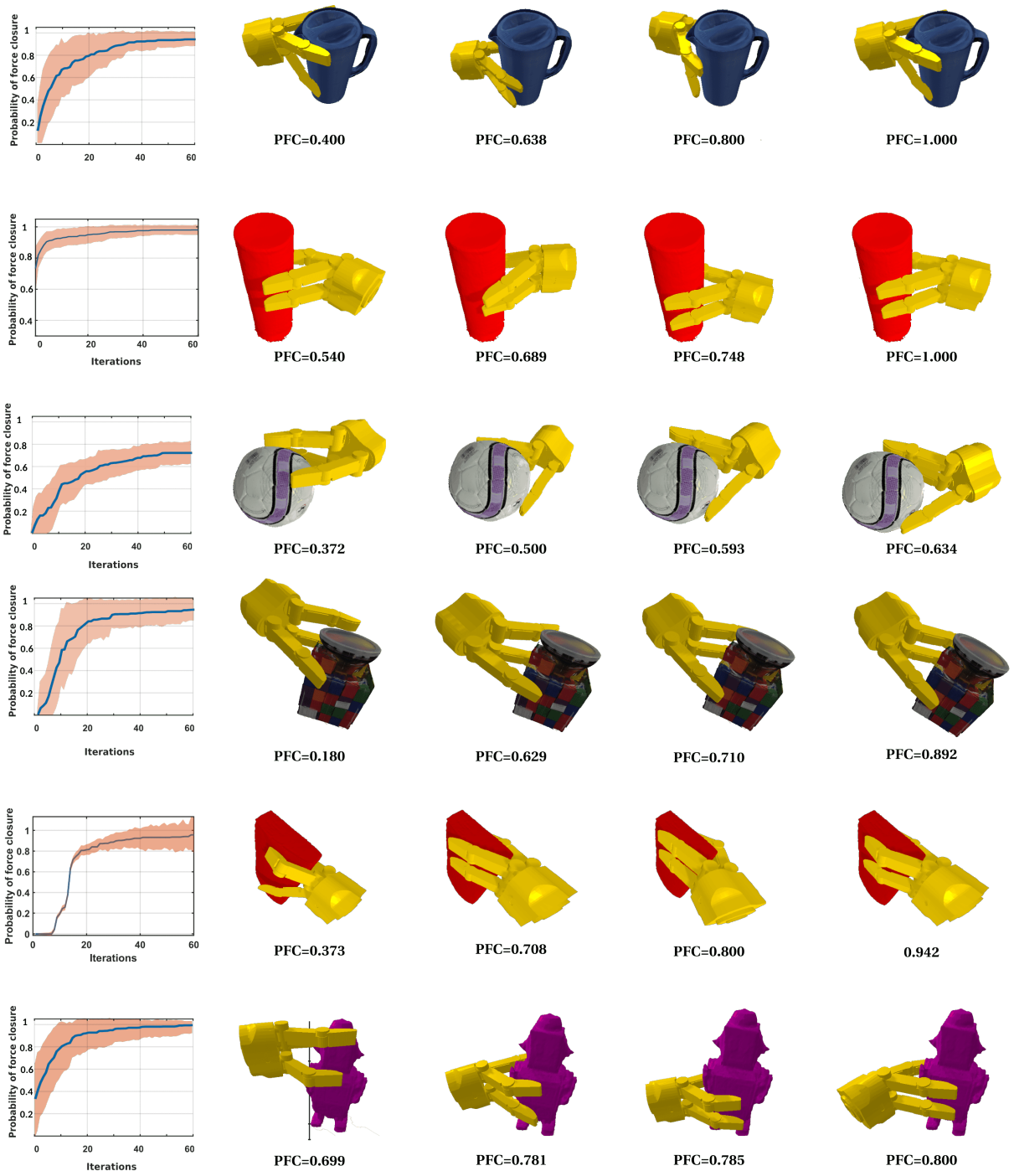


Figure 6.7: Evolution of the probability of force closure (PFC) optimum over 30 experiments and example grasps with increasing PFC values from local explorations of every object.

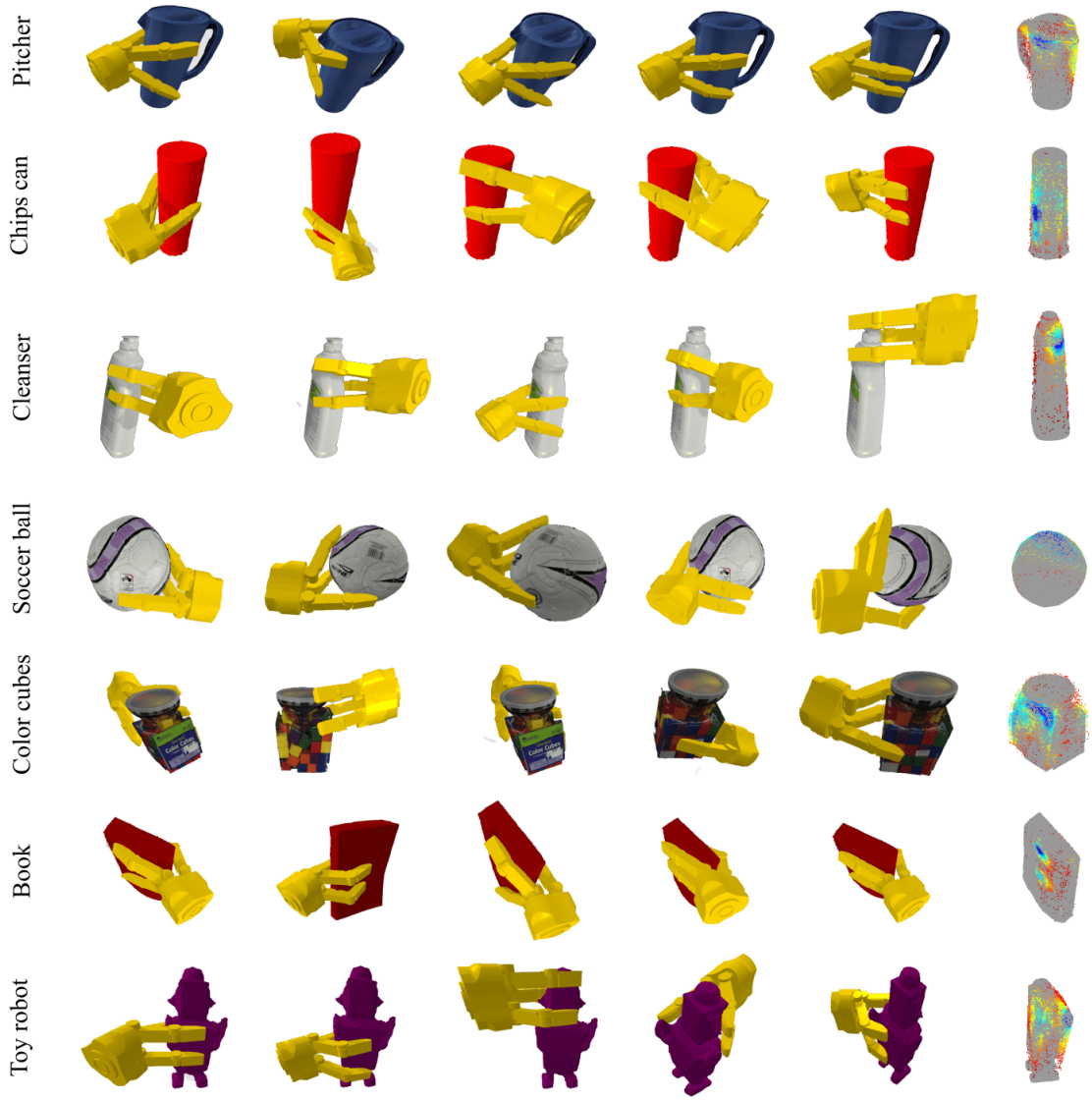


Figure 6.8: Grasp experiments highlighting the exploratory aspects of our algorithm. The last column shows object models overlaid with a density map of all tactile points found throughout all 30 experiments. Blue areas indicate higher point density, while red areas indicate lower density.

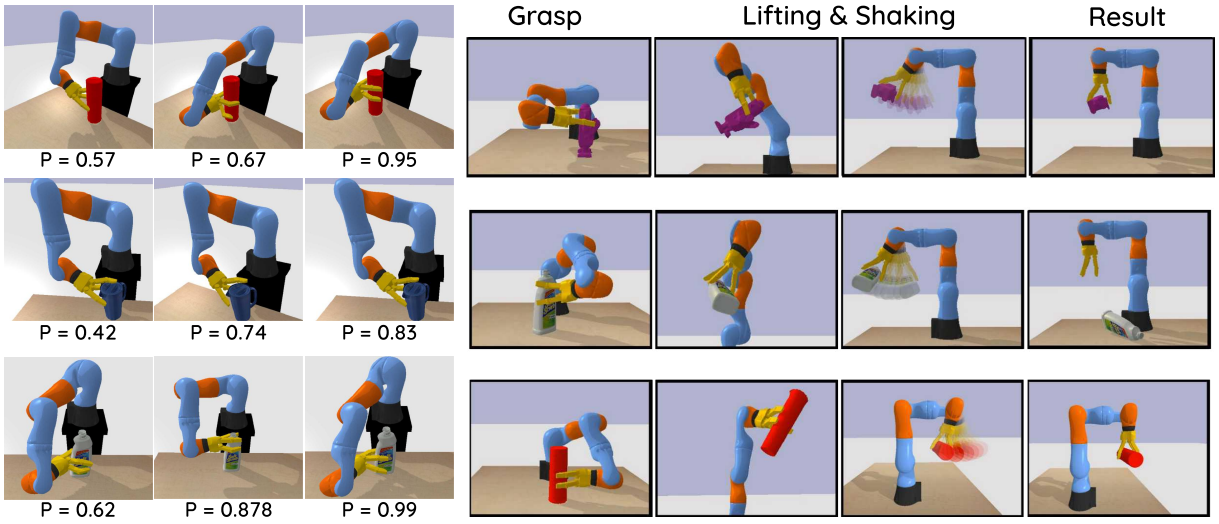


Figure 6.9: Exploration and test examples (including lifting and shaking) using the full arm.

6.5.4 Grasp stability analysis

In Table 6.1, the results obtained for a variety of objects using the proposed approach are presented. For these experiments, the optimisation loop was initialised with a prior obtained from Gaussian samples corresponding to finger positions and the evaluation of them based on (6.17). The number of these prior grasps depended on the complexity of the object, with priors built on at least 3 grasps having Pr^{FC} values greater than 0.4. The exploration experiments, detailed in Table 6.1, were conducted with $k_{\max} = 60$ and repeated 30 times for each object. The average of the best Pr^{FC} values, the average number of stable grasps with Pr^{FC} greater than 0.5, and their average success rates from each trial were reported. Success rates were computed based on lifting and shaking experiments described in Section 6.5.1 where grasp configurations leading to successful lifting and shaking tests were labelled successful.

The results demonstrate that the proposed approach yields stable grasp configurations with high Pr^{FC} values and lifting success rates. However, a decrease in success rates after the shaking test was observed for some objects, likely due to uncertainties in the physics models of the objects and fingers, such as imprecise collision models and physics engine inaccuracies. The conditions for the shaking test were intentionally harsh, involving high accelerations.

The results show that the "Pitcher," "Chips," and "Toy Robot" objects all achieved a perfect PFC of 1.0, with lifting success rates of 80.19%, 86.99%, and 86.54% and shaking success rates of 51.78%, 78.82%, and 77.14%, respectively. The "Cleanser" object had a slightly lower PFC of 0.99, with lifting and shaking success rates of 86.48% and 71.72%. The "Ball" object, however, had a PFC of 0.93, the lowest in the experiments, with success rates of 72.46% for lifting and 60.4% for shaking, reflecting the difficulty of grasping it due to its size and spherical shape, which did not always lead to properly aligned normals. The "Woodblocks" object achieved a PFC of 0.95, with a high lifting success rate of 78.74%, but a lower shaking success rate of

32.23%. As seen in Figure 6.8, grasps were located in a transition region between the cubic overall shape and the lid of the box, and the lack of detailed information in the reconstruction (evident from Figure 6.6 and a consistently high reconstruction error of 15.02 ± 1.06) could have contributed to the instability seen during the shaking tests. Finally, the "Book" object displayed a high PFC of 0.98, with lifting and shaking success rates of 92.76% and 80.36%, respectively.

Overall, the results highlight the effectiveness of the proposed approach in finding and refining stable grasps across various object shapes and complexities.

6.5.5 Baseline comparisons

The proposed approach was compared with a point cloud-based heuristic exploration baseline, where exploration trials were also conducted for 60 iterations. In this baseline, instead of BO guiding the robot, samples were drawn from a Gaussian distribution within the same domain as the BO. To facilitate this heuristic, the thumb position was restricted to the neighbourhood of a randomly sampled location from the point cloud with $\sigma_{th} = 0.05$ being the variance used. Table 6.1 also presents the results of the baseline experiments applied to all test objects.

It can be observed that the total number of stable grasps and final PFC values are lower in the baseline when compared to the proposed BO-based approach. However, as the BO-based method outputs a larger number of grasps satisfying both lifting and shaking on all tests becomes harder. While most of the success rates for lifting and shaking are higher with BO-based methods method, the baseline outperforms it in some experiments, notably the for the "Woodblocks". In this case the baseline finds very few grasps with $Pr^{FC} > 0.5$ (only 25 in total, compared to 440 grasps found by the BO method). Because the baseline method identifies so few successful grasps, randomly achieving a perfect lifting score becomes much easier in this scenario. In contrast, the BO method evaluates a much larger set of grasps, making the comparison more comprehensive, and results more meaningful.. Also notable is the lifting test for the "Pitcher", which, despite not having such a difference in number of grasps found (117 for the baseline against 469 for the proposed approach), still corresponds to a larger number of successful cases overall. These results clearly demonstrate the efficiency of the proposed method.

Furthermore, Figure 6.10 (left) shows the evolution of the average number of stable grasps for two objects: the book and the cleanser and compares it between the BO-method and the baseline. For the "Book", which, as discussed, has fewer good grasp regions, the baseline method finds almost no grasps, whereas the proposed approach identifies optimal regions showing a high improvement by the end of the 60 iterations. The "Cleanser", with more feasible grasp regions, allows the baseline to find some stable grasps, however, the BO method converges to the optimal regions more efficiently and finds more stable grasps rapidly.

Finally, Figure 6.10 (right) illustrates how the proposed approach outperforms the baseline in finding stable grasps rapidly. There, the number of iterations needed for each method to find 10 grasps with $Pr^{FC} > 0.8$ is shown. These experiments were conducted five times for

Table 6.1: Experimental evaluation with multiple ojects for both the proposed approach and baseline.

Object[1]	Best Prob. (Std.)		GPIS Evolution[2] [%]		Total Grasps (Avg., Std.)		Lifting %		Shaking %	
	BO	Baseline	BO		BO	Baseline	BO	Baseline	BO	Baseline
Pitcher	1.0(0.0)	0.82(0.14)	18.56, 1.59		469(15.6, 8.23)	117(3.90, 2.06)	80.19	86.06	51.78	44.70
Chips	1.0(0.0)	0.92(0.08)	7.78, 3.9		792(26.4, 10.6)	242(8.07, 2.53)	86.99	84.77	78.82	60.54
Cleanser	0.99(0.04)	0.81(0.19)	20.40, 13.9		501(16.7, 7.4)	101(3.4, 1.90)	86.48	84.25	71.72	68.62
Ball	0.93(0.14)	0.71(0.19)	2.02, 10.02		148(4.9, 2.6)	62(2.07, 1.36)	72.46	68.07	60.4	46.73
Woodblocks	0.95(0.14)	0.55(0.25)	15.02, 1.06		440(14.7, 11.2)	25(0.83, 1.02)	78.74	100	32.23	44.87
Book	0.98(0.07)	0.48(0.38)	27.3, 15.53		266(8.87, 6.0)	19(0.63, 0.72)	92.76	38.89	80.36	22.22
Toy Robot	1.0(0.01)	0.83(0.18)	24.69, 1.07		375(12.5, 5.4)	87(2.90, 1.79)	86.54	61.63	77.14	59.30

[1] Objects are shown in Figure 6.5.

[2] Rate of improvement of the Hausdorff distance from ground truth and the associated rate of change for the variance.

each object, capped to a maximum of 300 iterations. If fewer than 10 grasps were found, it was considered that no convergence was possible. For the three objects shown in Figure 6.10 (right), the baseline was only successful with the "Chips Can", requiring an average of 120 iterations (and a minimum 51 iterations) to complete the experiment. In comparison, the BO method required an average of only 37.8 iterations (minimum 15) for the same object. For a harder objects, such as the "Pitcher", the BO-based approach completed the experiments in an average of 103.6 iterations (minimum 63), and for the "Book", the average was 25.4 iterations (minimum 20).

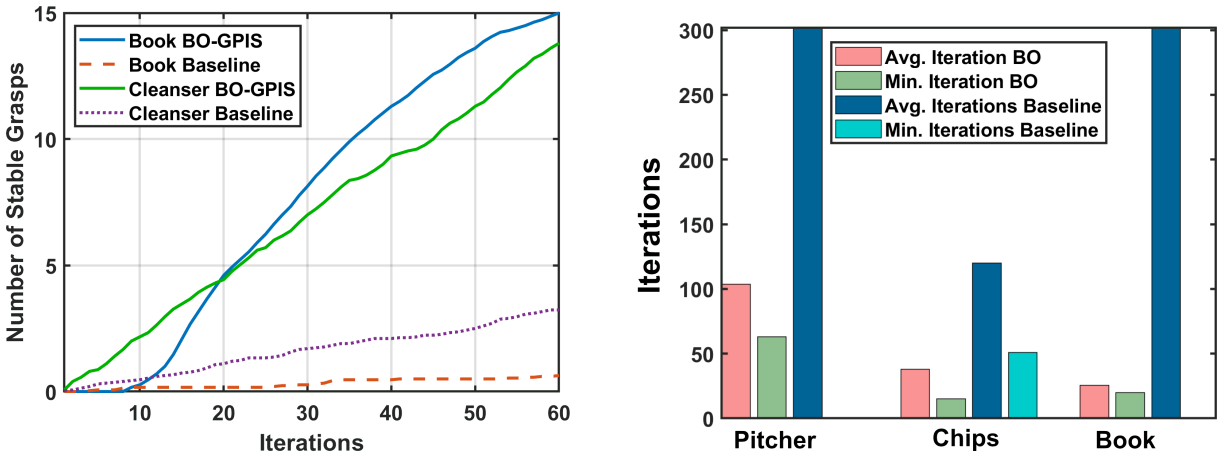


Figure 6.10: (Left) Evaluation of the number of stable grasps compared to the baseline; (Right) Evaluation of the number of iterations required to obtain 10 grasps with $Pr^{FC} > 0.8$.

6.6 Summary

This chapter has presented an exploration-based grasping method for unknown objects using a multi-finger hand, leveraging both visual and tactile data. The approach is takes advantage of BO, which is employed to systematically search for parameters that maximise grasp stability while also enhancing the perception of object shape. The exploration process begins with visual data and progressively incorporates tactile sensing to refine grasp configurations. A probabilistic force closure-based evaluation is employed to address uncertainties in key parameters such as shape, friction, contact points, and normals. The experimental results demonstrate that the proposed method achieves stable grasp configurations in fewer iterations compared to a baseline approach based on random exploration of a partial point-cloud. The effectiveness of the method was demonstrated through simulations, and since the algorithm is designed to handle perception uncertainties in real-world applications, replicating the results on a real setup should be straightforward, given a few assumptions. First, the proposed approach does not account for object movement, which can be a challenge in real-world scenarios where objects may slide or tumble upon contact. Second, while the full-arm experiments consider kinematics, planning remains a challenge, particularly in handling collisions and ensuring smooth motion execution in

real-world environments. Despite these factors, the strong performance in simulation suggests that the method has the potential to transfer well to real-world applications, provided these limitations are addressed.

Part IV

Dynamic environments

Chapter 7

Grasping moving objects

Preliminary remarks

The research in this chapter was published in the following paper:

- **C. de Farias,** M. Adjigble, B. Tamadazte, R. Stolkin and N. Marturi. Dual Quaternion-Based Visual Servoing for Grasping Moving Objects. In 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), 2021.
  doi: 10.1109/CASE49439.2021.9551631

Note: Certain sections of this chapter, such as problem descriptions, specific equations, variable descriptions, interpretations of results, and figure/table captions, have been adapted from [44].

7.1 Introduction

In the previous parts of this thesis, various approaches were explored to address the different roles of object understanding and uncertainties perception during the grasping process. These aspects are a crucial for effectively handling unstructured environments. However, it is often naive to assume that the scene remains static. In real-world scenarios, both domestic environments and industrial settings are highly dynamic. Industries often use conveyors to transport objects, while human handovers are essential in human-robot interaction within domestic spaces.

Despite the importance, handling arbitrarily moving objects in a 3D space remains a significant challenge in robotics, primarily due to the complexities of planning coordinated arm and hand movements while simultaneously tracking an object's trajectory. Conventional methodologies designed for grasping static objects are often inadequate in dynamic environments, as they may fail when the object moves. However, even simply adding an object tracking system is insufficient for successful grasping. To understand that, consider, for example, the basic approach of selecting a single grasp from a planner and then continuously transforming it as the object moves. This strategy can lead to jerky or discontinuous robot movements as the grasp

target dynamically changes. Even if a controller is implemented to enhance smoothness and reactiveness, the robot's workspace is often constrained, and there is no guarantee that the object will not move into challanging configurations near singularities or joint limits. Additionally, the originally selected grasp may no longer be feasible as the object moves. For these reasons, it is essential that robots be equipped with advanced controllers that possess superior decision-making capabilities to manage the uncertainties inherent in perceived exteroceptive information, such as fluctuating object positions and orientations.

In this context, in this chapter, a novel visual-servoing formulation for grasping moving objects is explored by building upon the LoCoMo planner [3](see 3.3.2 for more details) and the grasp-ranking metric it provides. The LoCoMo approach is used to generate a ranked list of stable grasp poses based on the perceived point cloud data of the scene object. This is then integrated with a dual quaternion-based visual servoing method to demonstrate the task of tracking and grasping arbitrarily moving objects. The proposed approach then takes advantage of the dual quaternion spatial distance error between the current end-effector pose and the LoCoMo-provided pre-grasp pose to define a dynamic grasp re-ranking metric that select the most feasible grasp for the moving object. In this framework, the quality of multiple grasp configurations and their proximity to the gripper are simultaneously evaluated in real-time. Consequently, the robot is able to dynamically switch between grasps as the object moves, thereby maintaining a secure and adaptive relative distance which facilitates grasping.

Moreover, there is a high possibility that the robot may encounter singular configurations while executing grasp tracking. To prevent this, a dual quaternion-based control strategy is employed, which ensures a singularity-free representation while also exploring the robot's null-space to maintain smooth motion and avoid joint limits. This strategy allows for uninterrupted tracking and grasping of moving objects, even in complex scenarios.

To this end, the key contributions are:

- A dual quaternion-based position-based visual servoing (PBVS) scheme in the manipulator's joint space has been developed. This controller is designed to be robust against singularities. Particularly, it shows robustness near regions prone to representation singularities as well as joint limits. Furthermore, the dual-quaternion representation ensures a coupling between position and orientation, allowing the manipulator to perform continuous, uninterrupted motion while tracking and grasping moving objects in dynamic environments.

- Extreme Robotics's lab previous work on stationary unknown object grasping (LoCoMo), has been extended to grasp arbitrarily moving objects. For this purpose, a novel dynamic ranking metric has been developed, combining the dual quaternion spatial distance error with a vantage-point tree (vp-tree) algorithm. This metric enables the real-time selection of the most feasible grasp during tracking trajectories, and adjusting dynamically to changes in the object's position and orientation. The integration of LoCoMo with the dual

quaternion-based PBVS allows for a seamless transition between grasps, ensuring that the robot can maintain effective control.

7.2 Problem statement

As mentioned earlier, the problem of grasping moving objects is formulated as a PBVS problem where the main objective is to regulate the pose of a robotic system based on visual information. Traditionally, such servoing systems are described using the axis-angle notation for rotations, with position and orientation often treated separately. However, this decoupling can lead to inefficiencies and potential inaccuracies, particularly in dynamic environments where both position and orientation need to be tightly coupled and managed concurrently.

Let $\underline{\boldsymbol{x}}(t)$ be the dual quaternion representing the current robot state derived from vision measurements, and let $\underline{\boldsymbol{x_d}}(t)$ represent the desired pose. Generally, the visual servoing problem can then be formulated as finding a control law $\mathbf{u}(t)$ that regulates the dual quaternion error between these poses. Specifically, the problem is expressed as

$$\mathbf{u}(t) = \underset{\mathbf{u}(t)}{\arg\min} \left[E(\underline{\boldsymbol{x}}(t), \underline{\boldsymbol{x_d}}(t)) \right], \tag{7.1}$$

where $E(\underline{\boldsymbol{x}}(t), \underline{\boldsymbol{x_d}}(t)) = \underline{\boldsymbol{e}}(t)$ is the dual quaternion error function, defined as

$$\underline{\boldsymbol{e}}(t) = \underline{\boldsymbol{x}}^*(t)\underline{\boldsymbol{x_d}}(t). \tag{7.2}$$

Here, $\underline{\boldsymbol{x}}^*(t)$ denotes the conjugate of the current pose $\underline{\boldsymbol{x}}(t)$ (see Appendix B for an overview of dual quaternion algebra). The control objective is to find the control input $\mathbf{u}(t)$ that drives this error to zero, thereby aligning the manipulator's end-effector with the desired pose.

In the present formulation, given the perceived point cloud of an object, the LoCoMo grasp planner is used to derive an initial ranked set of grasp poses in the world frame as $\underline{\boldsymbol{\mathcal{X}}}_g = \{\underline{\boldsymbol{x}}_{g1}, \underline{\boldsymbol{x}}_{g2}, \ldots, \underline{\boldsymbol{x}}_{gK}\}$. The corresponding pre-grasp poses are represented as $\underline{\boldsymbol{\mathcal{X}}}_{pg} = \{\underline{\boldsymbol{x}}_{pg1}, \underline{\boldsymbol{x}}_{pg2}, \ldots, \underline{\boldsymbol{x}}_{pgK}\}$, where each of them is associated with a corresponding quality score $\mathcal{R} = \{r_1, r_2, \ldots, r_K\}$. These poses are then transformed to the object frame using the initial pose inferred from the visual tracking module, resulting in $\underline{\boldsymbol{\mathcal{X}}}^o_g(t)$ and $\underline{\boldsymbol{\mathcal{X}}}^o_{pg}(t)$, which are subsequently used to set the reference pose for visual servoing, $\underline{\boldsymbol{x_d}}(t)$. Specifically, $\underline{\boldsymbol{x_d}}(t)$ is determined by

$$\underline{\boldsymbol{x_d}}(t) = \underset{\underline{\boldsymbol{x}} \in \{\underline{\boldsymbol{\mathcal{X}}}^o_{pg}(t), \underline{\boldsymbol{\mathcal{X}}}^o_g(t)\}}{\arg\min} \left[\mathcal{G}(\mathcal{R}, \underline{\boldsymbol{x}}, \underline{\boldsymbol{x}}_{EF}(t), \underline{\boldsymbol{x}}_o(t)) \right], \tag{7.3}$$

where $\underline{\boldsymbol{x}}_{EF}$ is the current gripper pose, $\underline{\boldsymbol{x}}_o$ the object pose and $\mathcal{G}(\cdot)$ is a function that balances grasp quality and the robot's end-effector motion (i.e., the error between the current and desired pose).

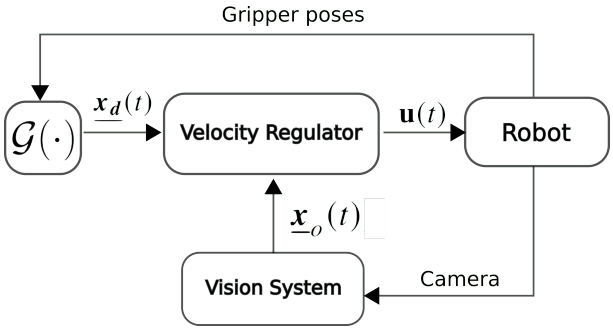


Figure 7.1: Control diagram illustrating the dual quaternion-based visual servoing process. It shows how the system updates the manipulator's pose online based on the object's visually tracked pose. $\mathcal{G}(\cdot)$ selects the input to the velocity regulator.

The dual quaternion-based visual servoing controller continuously updates the desired manipulator's end-effector pose based on the object's visual tracking, thus enabling the system to dynamically adapt to changes in the object's position and orientation. The manipulator's trajectory is adjusted in real-time, allowing the end-effector to smoothly transition to the desired grasp pose, even as the object moves unpredictably. This process is illustrated in the control diagram shown in Figure 7.1. For simplicity, the time variable t will be omitted from the notation in the remainder of this chapter.

7.3 Background

This section provides an overview of the background concepts employed in the proposed method. As the primary control equations use dual quaternions, a review of dual quaternion algebra is provided in Appendix B. The main concepts relevant to visual servoing, particularly PBVS, are discussed in Section 7.3.1. Additionally, the Vantage-Point tree (VP-tree) used in the grasp selection algorithm is introduced in Section 7.3.2. Finally, although the LoCoMo algorithm was discussed in detail in Section 3.3.2.

7.3.1 Visual servoing

Visual servoing is a class of control strategies that uses visual data from one or more cameras to control the motion of a robot. This approach enables tasks such as locating a robot within its environment with respect to visual features or visually following objects. Typically, two main camera configurations are employed in visual servoing setups: **eye-in-hand**, where the camera is mounted on the robot's end effector and calibrated to account for the robot's motions, and **eye-to-hand**, where the cameras are mounted within the scene, observing the robot's hand from a stationary position, with the robot's position and orientation determined relative to the camera's frame of reference.

Regardless of the camera configuration, the objective of the visual servoing task is to develop a control law that, given a set of visual inputs $\mathbf{x}(t)$ and a desired pose $\mathbf{x}_d$, the error $E(\mathbf{x}(t), \mathbf{x}_d)$[1] can be regulated. Furthermore, there are two main types of visual servoing schemes: Image-Based Visual Servoing (IBVS), in which $\mathbf{x}(t)$ consists of image features varying with time, and PBVS, where $\mathbf{x}(t)$ consists of 3D positional and orientational data derived from the camera's view [34]. This chapter considers a PBVS with scene-mounted cameras. For a more in-depth exploration of visual servoing techniques, the reader is referred to [34, 35].

7.3.1.1 Position-based visual servoing (PBVS)

In PBVS, the visual input $\mathbf{x}(t)$ is used to compute the current position and orientation (pose) of the object or the robot's end-effector in a global coordinate frame. This pose is then compared to a desired pose $\mathbf{x}_d$ to compute the error vector $\mathbf{e} = E(\mathbf{x}, \mathbf{x}_d)$, which is subsequently used to generate control commands that drive the robot towards the target pose. Usually, the most straightforward approach for solving visual servoing problems is to design a velocity controller such that the velocity $\dot{\mathbf{x}}$ can be derived as

$$\dot{\mathbf{x}} = \mathbf{L}\omega_c \tag{7.4}$$

with $\mathbf{L}$ being the interaction matrix (or feature Jacobian) , which maps the visual input to the robot's motion $\mathbf{x}$ and ω_c being the twist of the robot's end-effector[2].

If an error in $\mathbb{R}^3$ is considered (i.e. $\mathbf{e} = \mathbf{x} - \mathbf{x}_d$, then the error derivative can similarly be expressed as $\dot{\mathbf{e}} = \mathbf{L}_e\omega_c$ with $\mathbf{L} = \mathbf{L}_e$ for stationary objects. It is noted that for a moving object, the error dynamics must account for the object's motion as well, leading to a more general form of the error rate of change [35]

$$\dot{\mathbf{e}} = \mathbf{L}_e\omega_c - \dot{\mathbf{x}}_d. \tag{7.5}$$

In cases where the object's motion is slow relative to the camera's motion, and the control system is fast and efficient, the contribution of $\dot{\mathbf{x}}_d$ may be negligible, allowing for the approximation $\dot{\mathbf{x}}_d \approx 0$ to simplify the control law. With the object motion is neglected the velocity can be expressed as

$$\omega_\mathbf{c} = -\mathbf{K}\mathbf{L}_e^\dagger\mathbf{e} \tag{7.6}$$

for an exponential decrease of the error, such that for $\mathbf{K} > 0$ the error dynamic is $\dot{\mathbf{e}} = -\mathbf{K}\mathbf{e}$ and with $\mathbf{L}_e^\dagger$ being the pseudo-inverse of the interaction matrix $\mathbf{L}_e$ (i.e., $\mathbf{L}_e^\dagger = \left(\mathbf{L}_e^T\mathbf{L}_e\right)^{-1}\mathbf{L}_e^T$). This

[1] In this section, a general representation for visual servoing is used. If the variables are quaternions (or dual quaternions), the error is given by $E(\underline{\boldsymbol{x}}(t), \underline{\boldsymbol{x}}_d) = \underline{\boldsymbol{e}} = \underline{\boldsymbol{x}}^*\underline{\boldsymbol{x}}_d$. If the variables are in $\mathbb{R}^3$, then $E(\mathbf{x}(t), \mathbf{x}_d) = \mathbf{e} = \mathbf{x}(t) - \mathbf{x}_d$.Alternative error functions can also be defined.

[2] Though visual servoing is typically described using the matrix formulations with the interaction matrix $\mathbf{L} \in \mathbb{R}^{k\times 6}$ for a twist vector in $\mathbb{R}^6$ and k visual features, alternative approaches (such as the one taken in this work using dual quaternion algebra) may not define $\mathbf{L}$ explicitly. Instead, the mapping from visual input to control actions is handled implicitly through dual quaternion operations and frame transformations.

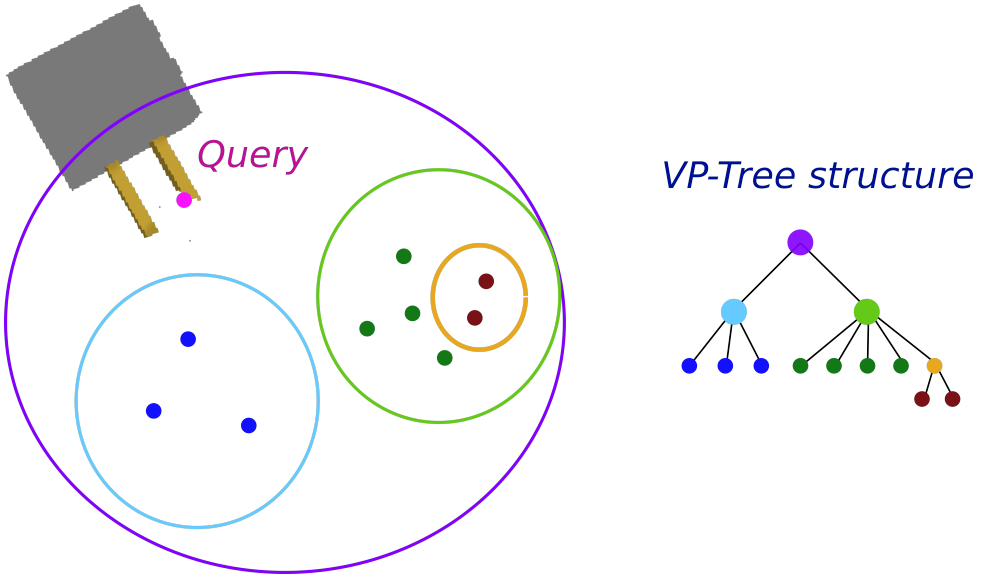


Figure 7.2: VP-tree when a new query is made and tree structure.

control law ensures that the visual error **e** decreases exponentially over time, assuming the object motion remains negligible.

7.3.2 Vantage-point tree

VP-trees, first introduced in [205] and illustrated in Figure 7.2 are tree-like data structures that are used for executing efficient nearest neighbor search in a given metric space, which is crucial in real-time systems where quick decision-making is required. Following the process for the tree construction and search are detailed.

VP-tree construction The construction of a VP-tree begins by selecting a data point from the dataset to serve as the vantage point. The remaining data points are then partitioned based on their distance from this vantage point. This process is repeated recursively for each partition, resulting in a binary tree structure where each node represents a subset of data points and their corresponding distances from the selected vantage point.

Formally, given a set of data points $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, a vantage point x_v is selected from $\mathcal{X}$. The set is then divided into two subsets: an inner set $\mathcal{X}_{\text{in}}$, containing points within a certain distance r_v from x_v, and an outer set $\mathcal{X}_{\text{out}}$, containing points farther than r_v. The distance metric used for partitioning is typically the Euclidean distance, but it can be adapted to other metrics depending on the specific application.

VP-tree search Once the VP-tree is constructed, it facilitates efficient nearest-neighbour searches within the dataset. During a search query, the current data point is compared against the points stored in the VP-tree. The tree structure allows for a rapid search to find the data point

that is nearest to the query point, minimising the distance metric used

$$x_{\text{nearest}} = \arg\min_{x \in X} \|x - x_q\|. \tag{7.7}$$

Here, $\|x - x_q\|$ represents the distance between the data point x and the query point x_q. The VP-tree's efficiency in organising and querying the data significantly reduces computational load, making it suitable for real-time applications where quick retrieval of nearest neighbours is essential [205].

7.4 Methodology

In this section, the methodology for grasping moving objects using a visual servoing controller and a grasp re-ranking scheme is detailed. Section 7.4.1 presents an overview of the main pipeline of the method. Section 7.4.2 provides an in-depth explanation of the controller design, including a stability proof. Finally, Section 7.4.3 describes the approach for grasp selection and the re-ranking scheme.

7.4.1 Overview

The detailed control scheme for the proposed dual-quaternion-based visual servoing method is illustrated in Figure 7.3. The pipeline is comprised of five stages: ① visual tracking, ② grasp planning, ③ visual servo control in the robot's joint space, ④ a secondary task projected into the null-space, and ⑤ testing and execution.

Before executing stages ①-⑤, an initial offline preprocessing step is performed where the object's point cloud is acquired. The LoCoMo grasp planner then generates an initial set of grasps, denoted as $\underline{\boldsymbol{X}}_g$ (grasp poses) and $\underline{\boldsymbol{X}}_{pg}$ (pre-grasp poses), along with their corresponding scores $\mathcal{R}$, such that $\mathcal{R} = \{r_1, r_2, \ldots, r_K\}$ is defined from (3.8) and each score r is associated with a specific grasp pose in $graspsetdq_g$. Here, each grasp $\underline{\boldsymbol{x}} \in \{\underline{\boldsymbol{X}}_g, \underline{\boldsymbol{X}}_{pg}\}$ is represented as a dual quaternion ($\underline{\boldsymbol{x}} \in \mathbb{H} \otimes \mathbb{D}$), while each ranking $r \in \mathcal{R}$ is a scalar value ($r \in \mathbb{R}$). Once this initial grasp computation is complete, the method can proceed.

In stage ①, the object is tracked, and its current pose $\underline{\boldsymbol{x}}_o(t)$ is updated at each time step. Tracking is achieved using fiducial markers placed on the object's surface [69, 68]. It is worth noting that this tracking approach can be easily replaced with any other fast and reliable tracking algorithms.

With the object tracking data and the precomputed grasp list, stage ② involves selecting the current desired pose $\underline{\boldsymbol{x}}_d$ as described in (7.3). The planner uses VP-trees with a metric based on dual quaternion error to search for the nearest and highest-ranked grasp in the set ($\underline{\boldsymbol{X}}_g$ or $\underline{\boldsymbol{X}}_{pg}$) that is close to the current robot configuration. The decision to use either $\underline{\boldsymbol{X}}_g$ or $\underline{\boldsymbol{X}}_{pg}$ to set $\underline{\boldsymbol{x}}_d$ is made automatically. Initially $\underline{\boldsymbol{X}}_{pg}$ is used for following the object, and $\underline{\boldsymbol{X}}_g$ is used when the

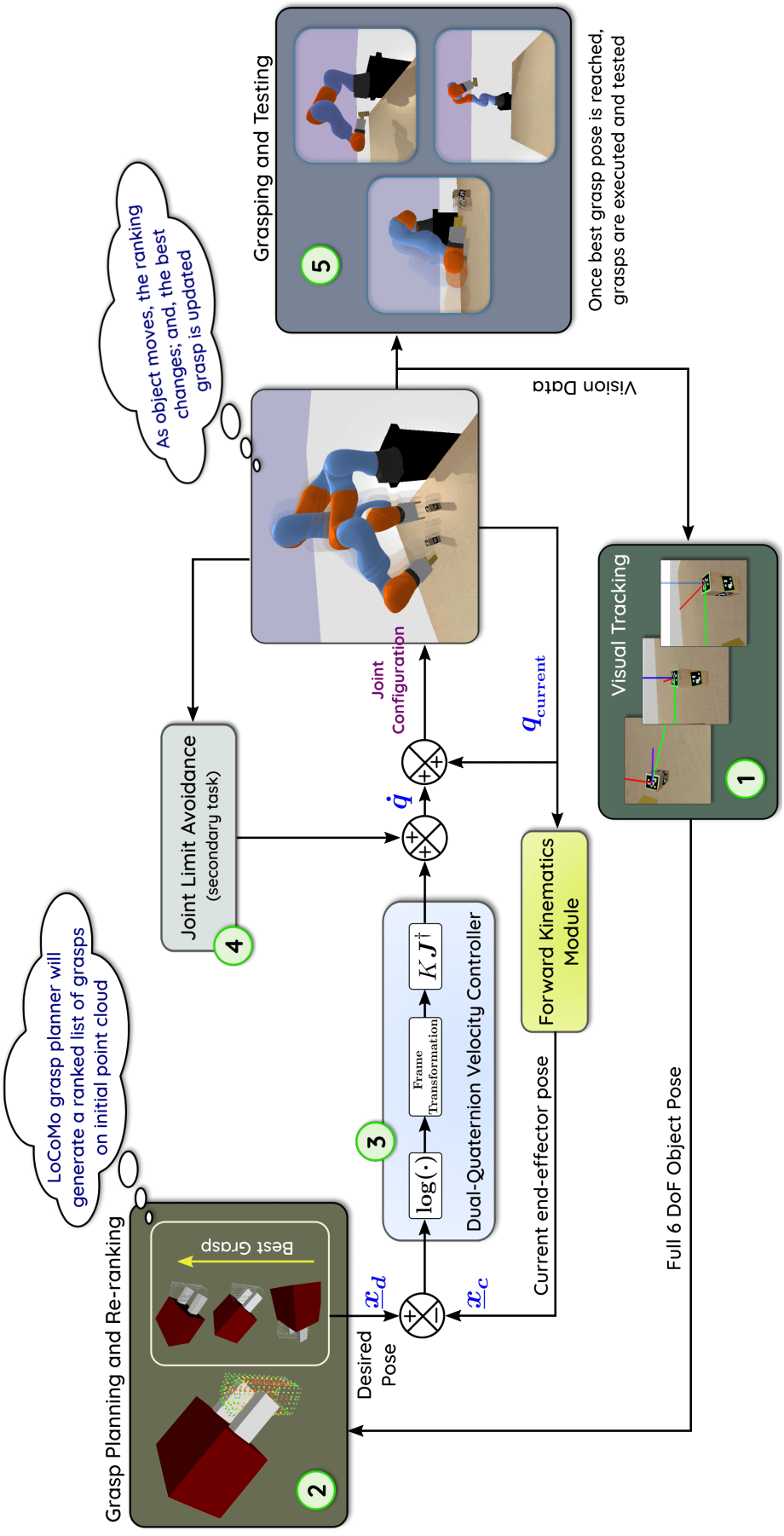


Figure 7.3: Control diagram of the visual servoing system for grasping dynamic objects. The process is organised into five stages, as marked by circled numbers. Concurrently, the object's position is estimated by tracking attached markers. Initially, a scene-mounted depth camera captures a point cloud of the target object, which the LoCoMo grasp planner uses to generate preliminary grasp options. As the object moves, the desired pose for visual servoing—specifically, the pre-grasp pose provided by the grasp planner—is continuously updated through a re-ranking process. Once visual servoing aligns and the object is successfully grasped, a stability check is performed.

robot is ready to grasp. When the tracking error is sufficiently low, the system will switch to $\underline{\boldsymbol{X}}_g$ for grasp execution. This switching mechanism is illustrated in Figure 7.4, where the object transitions from configuration ***1*** to configuration ***2***[3].

Stage ③ implements the main control block of the method, which minimises the dual quaternion error with a control law such as the one defined in (7.1). Additionally, in stage ④, the robot's redundancy is taken advantage of to perform a secondary task, in this case as joint-limit avoidance, by projecting this task into the robot's null-space. This helps the robot avoid challenging positions near singular configurations, especially during complex motions that approach the limits of the robot's workspace. Finally, in stage ⑤, after the grasp is executed, the robot performs stability tests to evaluate the quality of the grasp.

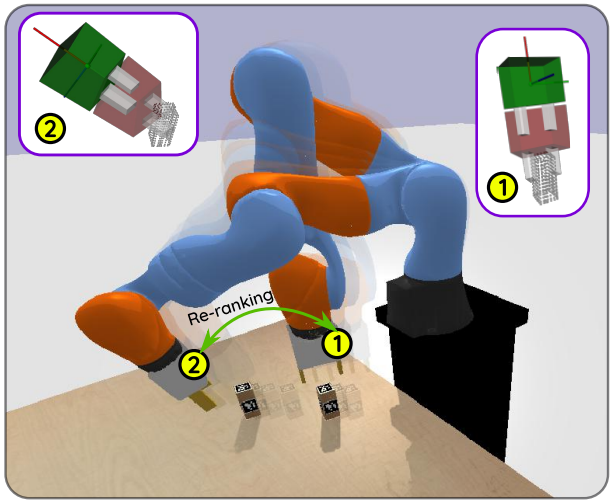


Figure 7.4: Example of grasp switching during grasp tracking.

7.4.2 Controller design

In this section, the primary dual quaternion control law is derived. Here definitions from dual quaternion algebra are used. For a comprehensive overview of these definitions, the reader is referred to Appendix B.

7.4.2.1 Visual servo velocity control

Let the current relative pose between the robot's end effector and the object be denoted as $\underline{\boldsymbol{x}}_c$, such that,

$$\underline{\boldsymbol{x}}_c = \underline{\boldsymbol{x}}_o^* \underline{\boldsymbol{x}}_{EF}, \tag{7.8}$$

[3]Although the grasping could be done using only $\underline{\boldsymbol{X}}_g$, having a pre-grasp pose ($\underline{\boldsymbol{X}}_{pg}$) offset along the robot's approach axis is advantageous as it reduces the risk of collision between the end-effector and the object during approach.

where $\underline{\boldsymbol{x}}_o$ represents the object's pose tracked by the camera and transformed into the world frame, and $\underline{\boldsymbol{x}}_{EF}$ is the robot's end-effector pose obtained from the robot's forward kinematics, also in the world frame. Given the desired relative configuration $\underline{\boldsymbol{x}}_d$, the error is defined as in (7.1), or more specifically as,

$$\underline{\boldsymbol{e}} = \left(\underline{\boldsymbol{x}}_c\right)^* \underline{\boldsymbol{x}}_d. \tag{7.9}$$

To minimise this error and ensure system convergence, previous works have proposed a logarithmic error dynamic, which leads to a stable closed-loop control system [78, 79],

$$\dot{\underline{\boldsymbol{e}}} = K\log\left(\underline{\boldsymbol{e}}\right)\underline{\boldsymbol{e}}. \tag{7.10}$$

for $K > 0$. Then, using the body twist definition from (B.12), the time derivative of $\underline{\boldsymbol{x}}_c$ is given by

$$\dot{\underline{\boldsymbol{x}}}_c = \frac{1}{2}\underline{\boldsymbol{x}}_c\,\underline{\boldsymbol{\omega}}_c^B, \tag{7.11}$$

where $\underline{\boldsymbol{\omega}}_c^B$ represents the current end-effector twist. By also calculating the first-order time derivative of the error in (7.9), the expression becomes,

$$\dot{\underline{\boldsymbol{e}}} = \left(\dot{\underline{\boldsymbol{x}}}_c\right)^* \underline{\boldsymbol{x}}_d. \tag{7.12}$$

Then, using (7.11), and the property that for a pure dual quaternion (as is the case for $\underline{\boldsymbol{\omega}}_c^B$) $(\underline{\boldsymbol{\omega}}_c^B)^* = -\underline{\boldsymbol{\omega}}_c^B$, the time-derivative of the error $\dot{\underline{\boldsymbol{e}}}$ can be rewritten as

$$\dot{\underline{\boldsymbol{e}}} = \left(\frac{1}{2}\underline{\boldsymbol{x}}_c\,\underline{\boldsymbol{\omega}}_c^B\right)^* \underline{\boldsymbol{x}}_d = -\frac{1}{2}\,\underline{\boldsymbol{\omega}}_c^B\underline{\boldsymbol{x}}_c^*\underline{\boldsymbol{x}}_d = -\frac{1}{2}\,\underline{\boldsymbol{\omega}}_c^B\underline{\boldsymbol{e}}. \tag{7.13}$$

Finally, given the desired closed-loop dynamics from (7.10), the current twist in the end-effector's body frame can be defined as,

$$\begin{aligned} K\log\left(\underline{\boldsymbol{e}}\right)\underline{\boldsymbol{e}} &= -\frac{1}{2}\,\underline{\boldsymbol{\omega}}_c^B\underline{\boldsymbol{e}}, \\ \underline{\boldsymbol{\omega}}_c^B &= -2K\log(\underline{\boldsymbol{e}}). \end{aligned} \tag{7.14}$$

7.4.2.2 Robot controller

To perform visual servoing with a robotic manipulator, velocities cannot be directly applied, instead, a control law must be defined in the joint space. For the proposed controller, the decision is made to use the inertial twist rather than the body-frame twist. The transformation from the inertial frame to world coordinates is achieved using the adjoint operation. Specifically, as shown in (B.14), the relationship is given by $\underline{\boldsymbol{\omega}}_c^I = \mathrm{Ad}(\underline{\boldsymbol{x}}_c)\underline{\boldsymbol{\omega}}_c^B$. This transformation is depicted in the "frame transformation block" in stage ③ of Figure 7.3. For simplicity, the inertial twist will be denoted as $\underline{\boldsymbol{\omega}}$, and unless otherwise stated, all variables are considered within the robot's

base (world) frame.

Let $\boldsymbol{J} \in \mathbb{R}^{6\times N_{\text{DoF}}}$ represent the dual quaternion-based geometric Jacobian, as defined in [63]. This Jacobian $\boldsymbol{J}$ maps the dual quaternion twist to the robot's joint space, thereby enabling the definition of the controller as follows

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - K_r \boldsymbol{J}^{\dagger} \operatorname{vec}_6 \left(\underline{\boldsymbol{\omega}}\right), \tag{7.15}$$

where $\boldsymbol{q}_k$ is the current joint configuration, $\boldsymbol{q}_{k+1}$ is the joint configuration to be commanded to the robot, $K_r > 0$ is the proportional gain for the control and $\boldsymbol{J}^{\dagger}$ is the damped pseudo-inverse of the Jacobian, calculated as $\boldsymbol{J}^{\dagger} = \boldsymbol{J}^T \left(\boldsymbol{J}\boldsymbol{J}^T + \lambda \boldsymbol{I}_6\right)^{-1}$, with λ being a small scalar gain that ensures extra robustness during matrix inversion.

Finally, incorporating the frame transformation, the control law can be expressed as,

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - K_r \boldsymbol{J}^{\dagger} \operatorname{vec}_6 \left(\operatorname{Ad}\left(\underline{\boldsymbol{x}}_c\right) \log(\underline{\boldsymbol{e}})\right). \tag{7.16}$$

It is important to note that although (7.14) and (7.16) differ from the traditional vector-based visual servoing law in (7.6), their underlying principles are consistent. Both (7.14) and the expression $\underline{\boldsymbol{\omega}} = \operatorname{Ad}\left(\underline{\boldsymbol{x}}_c\right) \log(\underline{\boldsymbol{e}})$ link the error derived from visual features to the robot's twist in a similar manner.

7.4.2.3 Joint limit avoidance

A significant challenge in visual servoing tasks, particularly when dealing with moving objects, is the risk of the robot reaching its joint limits, which can lead to problematic configurations. When robots operate within confined workspaces and perform dynamic tasks involving moving targets, there is a high likelihood that some joints will come close to their physical limits. This situation can result in reduced dexterity, the occurrence of singularities, or even potential damage to the robot's mechanical structure. To mitigate this issue, an effective strategy is to exploit the robot's kinematic redundancy. By projecting a secondary task, such as joint limit avoidance, onto the robot's null-space, the primary visual servoing task can be accomplished with enhanced robustness.

The null-space of a robot's kinematic structure refers to the set of joint motions that do not affect the end-effector's position or orientation in space. In other words, movements within the null-space allow the robot to adjust its joint configurations without disturbing the execution of the primary control task. Mathematically, the null-space is defined as the set of solutions to the homogeneous equation $\mathbf{J}\mathbf{q} = \mathbf{0}$, where $\mathbf{J}$ is the robot's Jacobian matrix and $\mathbf{q}$ represents the joint configuration. To incorporate the null-space into a control law, a null-space projector is defined as

$$\mathbf{P} = \mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}, \tag{7.17}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{J}^{\dagger}$ is the pseudo-inverse of the Jacobian.

For the secondary task of joint limit avoidance, the mapping $\boldsymbol{J}_s = \boldsymbol{q} - \bar{\boldsymbol{q}}$ is defined, where $\bar{\boldsymbol{q}}$ is the mean position for each joint. The task cost to be minimised is given by

$$C_{\text{null}} = \frac{1}{2} \sum (\boldsymbol{q} - \bar{\boldsymbol{q}})^2 , \tag{7.18}$$

then (7.16) can be extended by using the projector in (7.17) as follows

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - K_r \boldsymbol{J}^{\dagger} \operatorname{vec}_6 \left(\underline{\boldsymbol{\omega}} \right) + K_s \boldsymbol{P} \boldsymbol{J}_s^{-1} C_{\text{null}}, \tag{7.19}$$

where $K_s < 0$ is the gain for the null-space control term, resulting in a new control law that incorporates joint limit avoidance.

7.4.2.4 Stability analysis

In this section, the Lyapunov stability proof for the proposed visual servo velocity control is presented. To prove the stability of the controller the pure dual quaternion norm can be defined as in [78, 79],

$$N(\underline{\boldsymbol{e}}) = 2\|(\log(\underline{\boldsymbol{e}}))\| = 2\sqrt{\langle(\log(\underline{\boldsymbol{e}})), (\log(\underline{\boldsymbol{e}}))\rangle}. \tag{7.20}$$

Stemming from that, a Lyapunov candidate function can be defined as

$$V(\underline{\boldsymbol{e}}) = \frac{1}{2} N^2(\underline{\boldsymbol{e}}), \tag{7.21}$$

which is positive definite by definition, i.e., the norm (7.20) must always be positive. Differentiating (7.21) results in

$$\begin{aligned} \dot{V}(\underline{\boldsymbol{e}}) &= \frac{1}{2} \dot{N}^2(\underline{\boldsymbol{e}}), \\ \dot{V}(\underline{\boldsymbol{e}}) &= 2 \frac{d}{dt} \langle(\log(\underline{\boldsymbol{e}})), (\log(\underline{\boldsymbol{e}}))\rangle. \end{aligned} \tag{7.22}$$

Then, by setting $\underline{\boldsymbol{x}}_l = \log(\underline{\boldsymbol{e}})$ and from the definition of the inner product in (B.16),

$$\frac{d}{dt} \langle \underline{\boldsymbol{x}}_l, \underline{\boldsymbol{x}}_l \rangle = - \frac{2 \left(\dot{\underline{\boldsymbol{x}}}_l \underline{\boldsymbol{x}}_l + \underline{\boldsymbol{x}}_l \dot{\underline{\boldsymbol{x}}}_l \right)}{2} = \langle \underline{\boldsymbol{x}}_l, \dot{\underline{\boldsymbol{x}}}_l \rangle. \tag{7.23}$$

Which direclty leads to,

$$\dot{V}(\underline{\boldsymbol{e}}) = \langle 2(\log(\underline{\boldsymbol{e}})), 2(\frac{d}{dt} \log(\underline{\boldsymbol{e}})) \rangle. \tag{7.24}$$

Finally, using the definition proposed in [78, 79],

$$\dot{V}(\underline{\boldsymbol{e}}) = 2\langle(\log(\underline{\boldsymbol{e}})), (\underline{\boldsymbol{\omega}})\rangle, \tag{7.25}$$

which from (7.14) becomes,

$$\dot{V}(\underline{\boldsymbol{e}}) = 2\langle(\log(\underline{\boldsymbol{e}})), (-2K\log(\underline{\boldsymbol{e}}))\rangle. \tag{7.26}$$

Then, by applying the definition of $V(\underline{\boldsymbol{e}})$ from (7.21),

$$\dot{V}(\underline{\boldsymbol{e}}) = -KV(\underline{\boldsymbol{e}}), \tag{7.27}$$

which means the function is negative for any $K > 0$, with its energy decreasing exponentially and thus ensuring the error $\underline{\boldsymbol{e}}$ is locally exponentially stable. That is, it converges exponentially to zero as time progresses.

7.4.3 Grasp planning and re-ranking

The final step in the proposed algorithm for grasping moving objects involves creating a function $\mathcal{G}(\cdot)$, as defined in (7.3), which selects the desired grasp pose $\underline{\boldsymbol{x}}_d$ from either $\underline{\boldsymbol{\mathcal{X}}}_{pg}$ (pre-grasp poses) or $\underline{\boldsymbol{\mathcal{X}}}_{g}$ (grasp poses). This step is illustrated as stage ② in Figure 7.3. The selection process is twofold:

(i) The set $\underline{\boldsymbol{\mathcal{X}}}_{pg}$ or $\underline{\boldsymbol{\mathcal{X}}}_{g}$ is selected based on a distance threshold. Specifically, if the robot has sufficiently converged to a pre-grasp pose $\underline{\boldsymbol{\mathcal{X}}}_{pg}$, it will attempt to grasp the object, setting $\underline{\boldsymbol{\mathcal{X}}}_{g}$ as the desired goal and once that also converges a secondary flag to initiate grasping is activated.

(ii) Once a generic set $\underline{\boldsymbol{\mathcal{X}}}$ (which may be either $\underline{\boldsymbol{\mathcal{X}}}_{pg}$ or $\underline{\boldsymbol{\mathcal{X}}}_{g}$, together with their associated $\mathcal{R}$) is selected, an optimal grasp configuration $\underline{\boldsymbol{x}} \in \underline{\boldsymbol{\mathcal{X}}}$ is chosen. The grasp selection within the set is performed using a VP-tree-based search, which considers both the quality metric $\mathcal{R}$ and the available grasp configurations $\underline{\boldsymbol{\mathcal{X}}}$.

7.4.3.1 VP-tree Search for Grasp Selection

When grasping moving objects, a naive approach would be to select the top-ranked grasp from the set $\underline{\boldsymbol{\mathcal{X}}}$ and follow it throughout the entire trajectory. However, during the object's movement, this grasp candidate might become unfeasible due to changes in the object's orientation or position, or it may lead the robot into challenging configurations where singularities become unavoidable. One possible solution to this problem is to simply use the list of grasps provided in the set $\underline{\boldsymbol{\mathcal{X}}}$—if the first grasp becomes unfeasible, the next best alternative can be selected. However, making switching decisions based solely on grasp quality is not optimal. The grasp quality metric does not account for the spatial configuration of the grasp, and simply selecting the next best grasp may result in large, abrupt changes that can negatively impact tracking and convergence.

To alleviate this problem, the function $\mathcal{G}(\cdot)$ for selecting the next best grasp should consider the robot's current configuration and search for the closest high-ranked grasp. As detailed in

Section 7.3.2, VP-trees are well-suited for quickly performing searches in metric spaces, making them an ideal choice for implementing the function $\mathcal{G}(\cdot)$.

Let $\underline{\boldsymbol{x}}_o$ represent the current pose of the object and $\underline{\boldsymbol{x}}_{EF}$ represent the current configuration of the end-effector. If the grasp set $\underline{\boldsymbol{\mathcal{X}}}$ is used directly in its current form, the VP-tree would need to be rebuilt at every time step to account for the movement of the object, which is computationally intensive. This is particularly inefficient for tasks such as visual tracking and grasping moving objects, where reactiveness is needed. To address this, instead of using the grasp poses directly in the world frame, the grasp set $\underline{\boldsymbol{\mathcal{X}}}$ is transformed into the object frame, resulting in the set $\underline{\boldsymbol{\mathcal{X}}}^o$. That is,

$$\underline{\boldsymbol{\mathcal{X}}}^o = (\underline{\boldsymbol{x}}_o)^o \underline{\boldsymbol{\mathcal{X}}}. \tag{7.28}$$

Similarly, the end-effector pose $\underline{\boldsymbol{x}}_{EF}$ is converted into the relative pose $\underline{\boldsymbol{x}}_c$, as described in (7.8). This transformation is important because it allows the system to consider the grasps relative to the moving object, making the poses invariant to the object's movement and therefore alleviating the requirement to rebuild the VP-tree at every time step.

Then to build the VP-tree, a distance function in a metric space must be defined. For the proposed method, the dual quaternion distance is used. For two poses $\underline{\boldsymbol{x}}_1$ and $\underline{\boldsymbol{x}}_2$, the distance is given by

$$x_{\text{dist}}(\underline{\boldsymbol{x}}_1, \underline{\boldsymbol{x}}_2) = \| \text{ vec}_6 \left(\log(1 - \underline{\boldsymbol{x}}_1^* \underline{\boldsymbol{x}}_2) \right) \|, \tag{7.29}$$

where $\log(\cdot)$ it the dual quaternion logarithm, which maps the dual quaternion error to its tangent space and $\| \cdot \|$ denotes the vector norm.

Once the VP-tree is constructed, given a query point $\underline{\boldsymbol{x}}_q$ (which in this work is $\underline{\boldsymbol{x}}_q = \underline{\boldsymbol{x}}_c$), the search can be performed as follows:

$$\{\underline{\boldsymbol{x}}_d^{(k)}\}_{k=1}^{\mathcal{K}} = \underset{\substack{\underline{\boldsymbol{x}} \subset \underline{\boldsymbol{\mathcal{X}}}^o \\ |\underline{\boldsymbol{x}}| = \mathcal{K}}}{\arg\min} \left[\sum_{k=1}^{\mathcal{K}} x_{\text{dist}}(\underline{\boldsymbol{x}}_d^{(k)}, \underline{\boldsymbol{x}}_q) \right], \tag{7.30}$$

where the $\mathcal{K}$ closest candidates are selected. A further kinematic check is performed, and any kinematically unfeasible grasps are removed from the list.

Finally, with the remaining $\mathcal{K}$ grasps $\{\underline{\boldsymbol{x}}_d^{(k)}\}_{k=1}^{\mathcal{K}}$, a re-ranking scheme is applied. By defining

$$\Upsilon(\underline{\boldsymbol{x}}) = \frac{x_{\text{dist}}(\underline{\boldsymbol{x}})}{x_{\text{dist}}^{\max} - x_{\text{dist}}^{\min}}, \quad \text{with } x_{\text{dist}}^{\max} \neq x_{\text{dist}}^{\min}, \tag{7.31}$$

as the normalised distance for each candidate, and using the LoCoMo quality metric,

$$\{r(\underline{\boldsymbol{x}}^{(k)})\}_{k=1}^{\mathcal{K}} = \{r(\underline{\boldsymbol{x}}) : \underline{\boldsymbol{x}} \in \{\underline{\boldsymbol{x}}_d^{(k)}\}_{k=1}^{\mathcal{K}}\}, \tag{7.32}$$

associated with the selected grasps, the new grasp score can be defined as

$$\mathcal{G}(\underline{\boldsymbol{x}}, r) = r\Upsilon(\underline{\boldsymbol{x}}). \tag{7.33}$$

The final desired grasp pose $\underline{\boldsymbol{x}}_d$ is then selected as

$$\underline{\boldsymbol{x}}_d = \underset{\underline{\boldsymbol{x}} \in \{\underline{\boldsymbol{x}}_d^{(k)}\}_{k=1}^{\mathcal{K}}}{\arg\max} \left[\mathcal{G}(\underline{\boldsymbol{x}}, r(\underline{\boldsymbol{x}}))\right]. \tag{7.34}$$

Note that in (7.9), $\underline{\boldsymbol{x}}_d$ is already expressed as a relative pose, so there is no need for an additional transformation back to the world frame at this point.

Finally, depending on the trajectory and the noise, the proposed planner may choose to switch frequently, and this can result in a chattering problem in the controller. To account for that a hysteresis is set over the switching strategy. This means that the control reference pose will only change if the new grasp ranking has improved by an amount greater or equal than a pre-defined switching threshold δ [132]. This, therefore allows for a reactive and smooth reach-to-grasp trajectory.

7.5 Experimental validations

In this section, the designed controller and the grasp re-ranking scheme are validated through a series of simulation experiments. First, the convergence of the controller is analysed in detail. Subsequently, experiments are conducted to evaluate the robot's ability to successfully grasp objects, with particular attention given to the impact of null-space utilisation and the re-ranking scheme on overall performance.

7.5.1 Experimental setup

Similar to the methodology used in previous chapters, experiments were conducted using the open-source PyBullet engine [40] with a 7-DoF robotic arm equipped with a parallel jaw gripper. Three objects from the YCB object dataset [31] were selected for evaluation: the Mustard Bottle, the Bleach Cleanser Bottle, and the Woodblock. These objects were chosen for their compatibility with the gripper and their suitability for marker placement on their surfaces. It is worth noting that the CAD models of the objects were downloaded and modified to include textures with fiducial markers.

To create a more realistic visual servoing environment, a virtual 3D camera was simulated for both pose tracking and the initial point cloud acquisition. The camera was initially moved to three different positions around the object, and the point clouds captured from these positions were stitched together to generate a complete point cloud for the LoCoMo grasp planner. For visual object pose tracking, ArUco fiducial marker tracking [69] from OpenCV was used [92].

The dual quaternion-based controllers and kinematics were implemented with the DQRobotics library [7].

The results were reported following the same grasp evaluation protocol as outlined in the previous chapter (presented in [17]). Given that the current work focuses on grasping moving objects, various trajectories were simulated for each object. The evaluation step, including object lifting, rotating, and shaking, was applied once a successful grasp was achieved.

For the experiments involving moving objects, five different trajectories were defined:

1. A horizontal line moving from the left side of the table to the right.
2. A vertical line, where the object moved away from the robot.
3. A diagonal line from the bottom left to the top right corner.
4. An ellipsoid with radius of $(15, 8)$ cm.
5. A sine wave with an amplitude of 15 cm and a frequency of 0.02 Hz.

For each trajectory, two tests were conducted: (i) the object was moved in space without any rotations; and (ii) a more challenging setup, where the object was moved with additional arbitrary rotations. In this case, a rotation step of 2.7rad/s is applied to the object's yaw.

7.5.2 Controller convergence analysis

This section presents a convergence analysis of the proposed method. First, Section 7.5.2.1 examines the controller's performance with stationary objects. Next, Section 7.5.2.2 explores the controller's behaviour when tracking moving objects. Finally, Section 7.5.2.3 analyses the impact of the re-ranking strategy on the control scheme. For consistency and comparison, most experiments in this section are conducted using the "woodblock" object, with a few results for the other two objects presented on the full controller for completeness.

7.5.2.1 Stationary objects

This test serves as an initial validation of the controller's performance, focusing on its ability to accurately reach and grasp an object that remains fixed in position. The object was placed centrally on the table, and the top-ranked feasible grasping position was selected.

Figure 7.5 presents the tracking results as the robot approaches the stationary object. Specifically, Figure 7.5(a) illustrates the 3D trajectory of the end-effector towards the object, while Figure 7.5(b) and (c) show the position and rotation tracking, respectively. Note that the discontinuity seen in the plots is due to controller switching between the feasible grasps. As the robot converges to the selected pre-grasp pose, the planner updates the goal pose to the final grasping configuration. This switch momentarily increases the tracking error, but the controller quickly readjusts, demonstrating its robustness.

Figure 7.6 depicts the error plots corresponding to the translational and rotational components of the dual quaternion. As shown in Figure 7.6(a) and (b), the translational and rotational errors exhibit smooth convergence, with a noticeable spike at the moment of goal switching. This spike is a natural consequence of the sudden change in the target pose, leading to a temporary increase in error. However, the controller effectively compensates for this, and the error decreases smoothly afterwards.

It is important to highlight that the dual quaternion-based controller inherently couples translation and rotation. This coupling leads to smoother convergence, as both translational and rotational components of the pose are adjusted synergistically. However, due to these synergies, the end-effector's trajectory does not necessarily follow a straight-line path. Instead, the path is often more curved as the controller balances adjustments in both translation and rotation simultaneously as can be seen in Figure 7.5(a). This behaviour is particularly advantageous in tasks requiring precise control and tracking, as it reduces abrupt movements and potential oscillations, ensuring a smoother and stable approach to the target.

While the coupling of translation and rotation in dual quaternion-based control has been a subject of debate [60], it proves particularly effective for tracking tasks that require smooth control. By simultaneously adjusting both translational and rotational aspects of motion, this approach leads to a more stable and predictable grasping process.

7.5.2.2 Moving object tracking

In this section, the robot's ability to track a moving object is analysed, with a focus on how the null-space of the controller influences performance. The object was subjected to an ellipsoid trajectory within the robot's task space. Figures 7.7 and 7.8 illustrate the tracking and error convergence without null-space control (as in (7.16)), while Figures 7.9 and 7.10 present the corresponding results with null-space control activated. Particularly, the experiment without the null-space control took significantly extended to complete. This lead to longer error plots shown in Figure 7.8, where the the convergence is harder to see. To address this, the figures include zoomed-in views of key moments, particularly during the initial convergence and the grasping phase, where the error spikes before settling again.

Nevertheless, both controllers managed to successfully follow the target trajectories and regulated the error. Once it was sufficiently low, the robot was able to grasp the object. In both cases, the trajectory was smooth, demonstrating the robustness of the controllers. Similar to the experiments with stationary objects, the distinct phases of pre-grasp and grasp can be observed in all figures. As expected, a comparison between the two methods reveals that the null-space controller achieved convergence much faster.

Specifically, regarding the speed of convergence, the experiment without null-space required nearly 4000 iterations to finish, whereas the experiment with the null-space controller completed in just over 1000 iterations. Further, Figure 7.8 shows that the error in the controlled error without

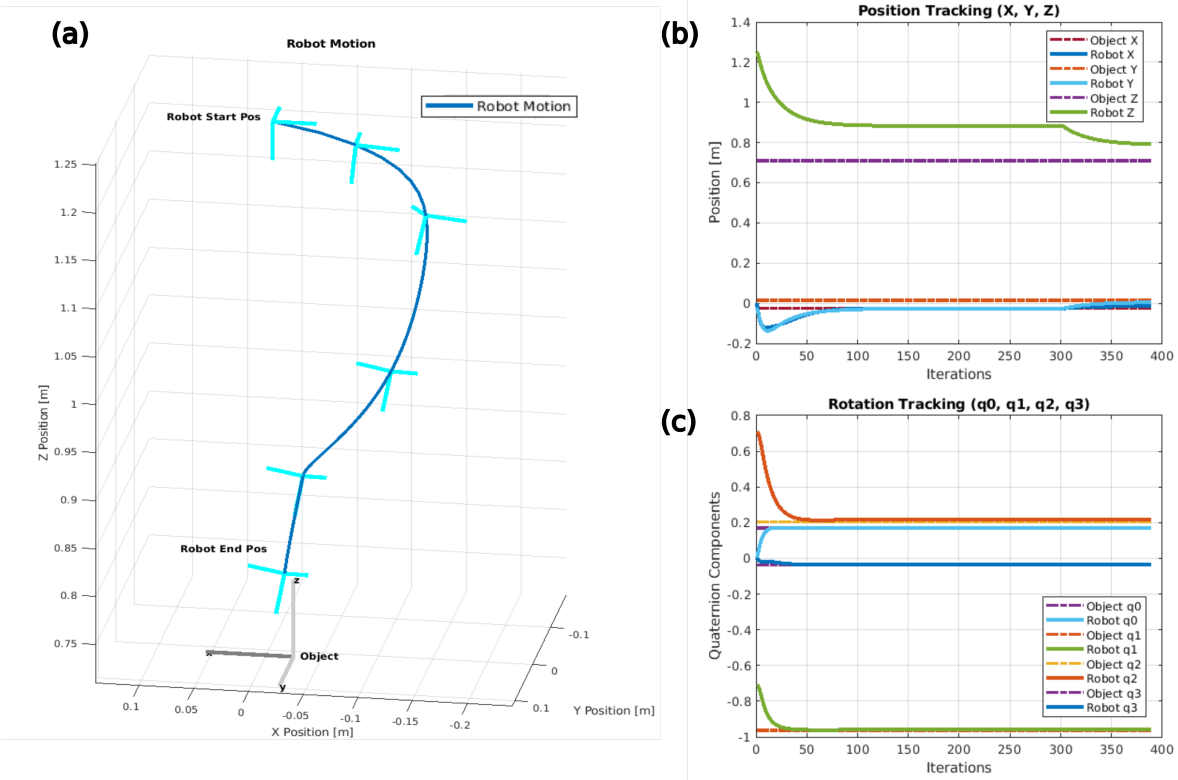


Figure 7.5: Tracking plots for grasping a stationary object with the proposed controller. Here (a) is the 3D trajectory of the end-effector towards the object (grey frame) and some of the the partial poses of the robot motion are represented by the blue frame plotted over its trajectory. (b) shows the position tracking over time, and (c) the rotation tracking.

null-space is noticeably noisier. This is likely due to the lack of joint-limit avoidance, which, when enforced by the null-space controller, keeps the robot in more stable configurations, thereby promoting smoother and more efficient convergence. Additionally, the increased noise makes it harder for the grasp planner to switch from the pre-grasp to the final grasp pose, which leads to a longer time for the system to stabilize.

7.5.2.3 Re-ranking strategy

This section presents the result which incorporates both the null-space secondary task for avoiding joint limits and the grasp re-ranking scheme. Similar to the experiments in the previous section, Figure 7.11 illustrates the tracking of the "Woodblock" object, in this case following a diagonal trajectory (trajectory 3 in the list from Section 7.5.1). Figure 7.12 displays the corresponding error convergence plots. In all figures, the moments when the re-ranking algorithm is activated and when there is a switch in the desired goal pose are highlighted.

Figure 7.11(a) shows when the re-ranking algorithm is triggered. Despite a change in direction, the trajectory remained smooth, allowing for effective tracking and grasping. Figures 7.11(b) and (c) display how each dual quaternion component continues to track the object's

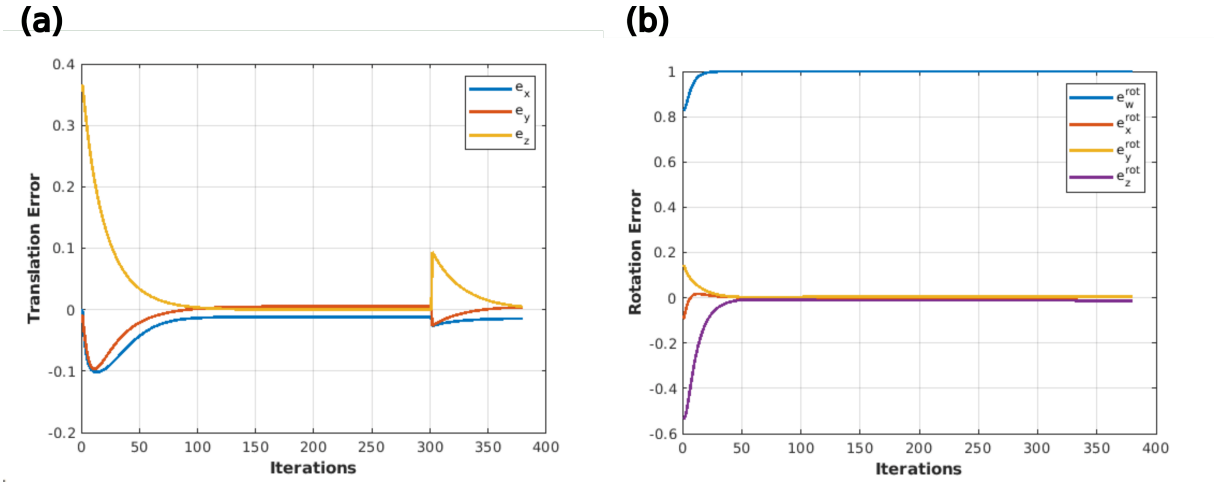


Figure 7.6: Error convergence plots for grasping a stationary object. In (a) is the translational error and (b) the rotational error during the grasping process.

reference, even as the distance to the reference pose changes. This is expected, as the new pose $\underline{\boldsymbol{x}}_d$ introduced by the switch is defined relative to the object, according to (7.28). Figure 7.12 shows the translational and rotational error plots, highlighting when re-ranking occurs and when the final grasping pose is set. The plots reveal a temporary increase in error during the switch, but the error quickly reconverges with minimal noise, demonstrating the controller's robustness.

Figure 7.13 shows error convergence plots for the 'Bleach Cleanser' and 'Mustard Bottle,' along with screenshots highlighting re-ranking and switch activation moments. Unlike the symmetrical 'Woodblock,' the complex shapes of these bottles lead to more challenging grasp configurations, causing slower convergence times. This often increases the distance between the end-effector and gripper, raising the value of Υ in (7.31) and triggering re-ranking more frequently. Despite this, the error is quickly reduced again, enabling successful grasping.

7.5.3 Grasping Arbitrarily Moving Objects

To comprehensively evaluate the proposed approach, a second set of experiments was conducted to quantitatively analyse grasp success for moving objects. All five trajectories listed in Section 7.5.1 were tested, both with and without rotational motion, for each of the three objects analysed through the Benchmark [17], where sequential lifting, shaking, and rotating tests were performed. If the object slips during any of these stages, the test is marked as a failure, and the remaining stages are not performed.

Figure 7.14 presents screenshots of the robot grasping moving objects using the proposed method. The results for all three objects: "Woodblock", "Mustard Bottle", and "Bleach Cleanser" are summarised in Tables 7.1, 7.2, and 7.3, respectively. The number of times the controller switched grasping configurations (re-ranking), the time elapsed from the start of the control loop until grasping, and the overall grasp success rate have been reported. Additionally, results

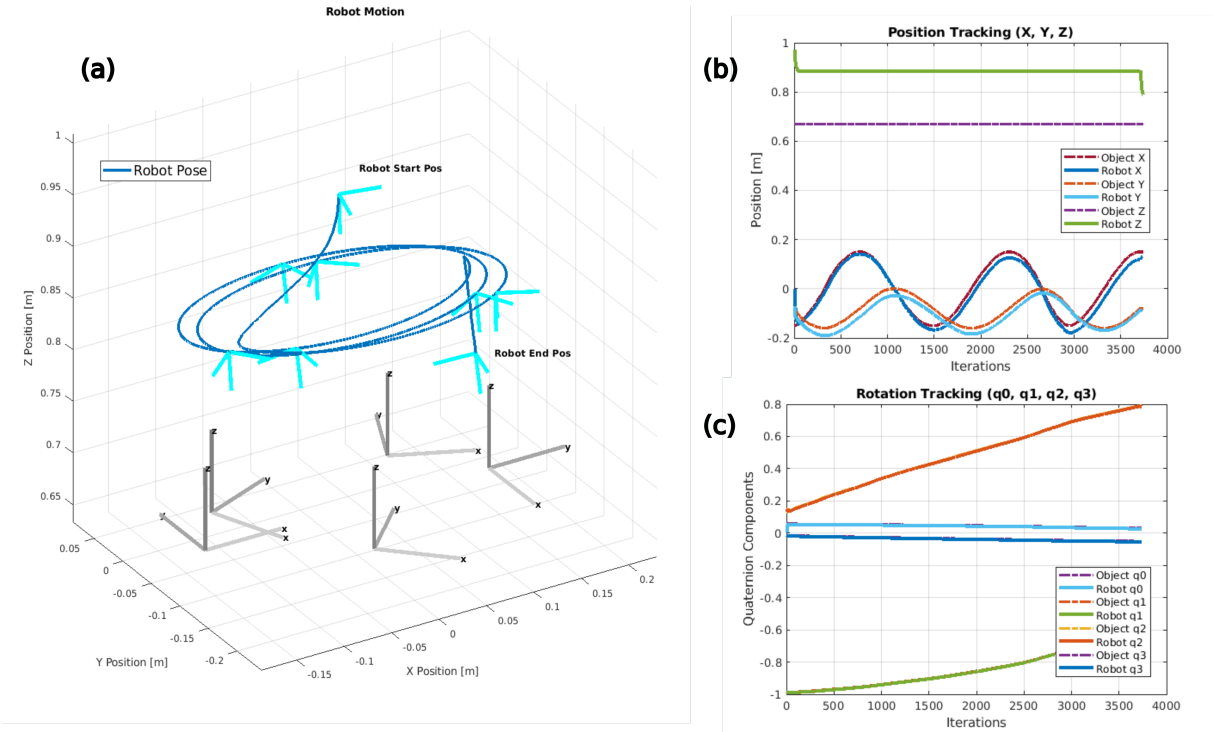


Figure 7.7: Tracking plots for grasping an object following an ellipsoid trajectory. The controller used corresponds to the one presented in (7.16), i.e., without null-space integration. (a) shows the 3D trajectory of the end-effector towards the object, with grey frames representing object motion, and blue ones showing the end-effector pose during various moments of the trajectory. In (b) the position tracking over time, is shown and (c) the rotation tracking.

are compared with and without null-space control, as well as with and without grasp re-ranking enabled. For each trajectory, tests were repeated three times, with the average of these trials reported in the tables.

The results clearly demonstrate that the full method, with both null-space control and re-ranking enabled, significantly outperformed the alternative approaches. For the three objects tested, success rates of $[\mathbf{80}\%, \mathbf{80}\%, \mathbf{70}\%]$ were achieved using the full method, compared to $[60\%, 60\%, 40\%]$ for the method without null-space control and $[50\%, 30\%, 30\%]$ for the method without re-ranking. Notably, a $\mathbf{100}\%$ success rate was achieved in following and grasping the first two objects when the full method was applied.

In contrast, the controller without null-space control exhibited difficulties in converging in certain cases due to joint constraints which led to an increased number of pose switches. This suggests that the absence of null-space control required more frequent re-ranking to adjust for these constraints.

The method without re-ranking performed poorly, likely because it lacked trajectory adjustments. Re-ranking is crucial for stable grasp tracking, as it lets the system dynamically choose the best grasp configuration as the object moves. Without this, the system is more likely to select suboptimal or unstable grasps, increasing the risk of failure.

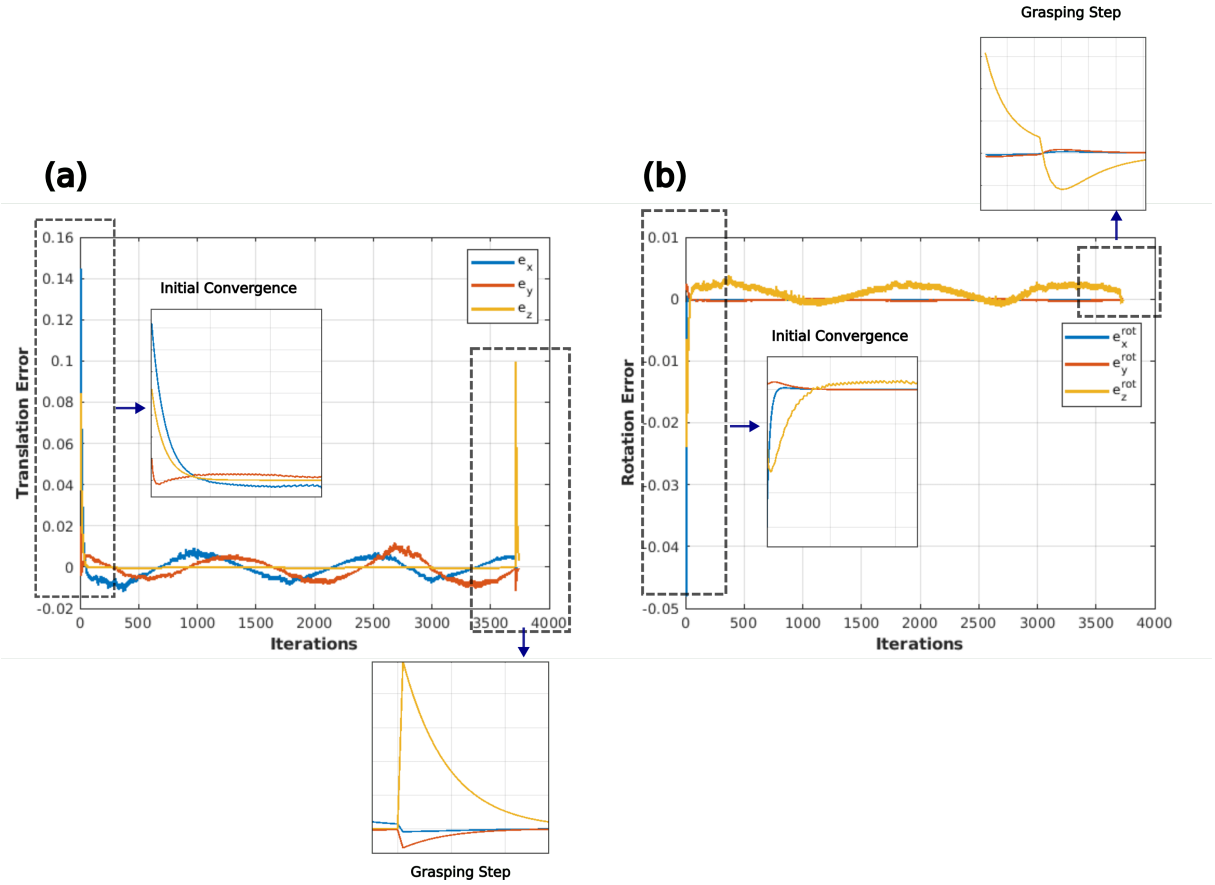


Figure 7.8: Error convergence plots for grasping an object following an ellipsoid trajectory using the control law from (7.16), i.e., without null-space integration. (a) The translational error is shown, and (b) the rotational error during the grasping process is shown. Interest regions of the plots are zoomed in.

Tables 7.1, 7.2, and 7.3 further illustrate the benefits of incorporating null-space control, showing significant improvements in success rates across all dynamic grasping tests. For instance, with the woodblock, the full method achieved a 100% success rate in the lifting test, 80% in the rotational test, and 80% in the shaking test. In contrast, when the null-space control was removed, these success rates dropped to 70% across all tests. Similar trends were observed for the other objects, with substantial decreases in success rates when the null-space term was omitted. Furthermore, the impact of re-ranking is also clear across all experiments. The controller achieved an overall 96.7% success rate across tested trajectories when re-ranking was enabled, compared to only 60% without it. This significant improvement highlights the role of re-ranking in adapting the grasp configuration in real-time, ensuring that the robot maintains an optimal grasp as the object moves unpredictably.

Table 7.1: Performance of dynamic grasping with the "Woodblock" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking.

Trajectory	Null-Space + Switching					Without Null-Space					Without Re-rank				
	Switch[1]	Time[2]	Lift[3]	Rot.[3]	Shake[3]	Switch[1]	Time[2]	Lift[3]	Rot.[3]	Shake[3]	Switch[1]	Time[2]	Lift[3]	Rot.[3]	Shake[3]
H. Line	1	28.81	100	100	100	0	11.07	100	100	100	NA	26.54	100	100	100
H. Line + Rot.	1	32.63	100	100	100	3	-	-	-	-	NA	30.42	100	100	100
V. Line	2	19.85	100	0	0	1	72.42	0	0	0	NA	10.93	0	0	0
V. Line + Rot.	1	19.31	100	100	100	1	29.71	0	0	0	NA	-	-	-	-
Diag.	1	19.63	100	100	100	3	17.15	100	100	100	NA	-	-	-	-
Diag. + Rot.	1	22.28	100	100	100	0	20.82	100	100	100	NA	29.4	100	100	100
Circle	0	88.29	100	100	100	0	11.32	100	100	100	NA	81.11	100	100	100
Circle + Rot.	3	91.32	100	100	100	4	97.10	0	0	0	NA	79.27	100	100	100
Wave	2	52.93	100	0	0	9	161.77	100	100	100	NA	-	-	-	-
Wave + Rot.	4	77.35	100	100	100	7	134.11	100	100	100	NA	-	-	-	-

[1] Number of times the re-ranking switched the reference pose.
[2] Time in seconds from the start pose until the object is grasped.
[3] Percentage of success (%) of lifting, Rotational and Shaking tests.

Table 7.2: Performance of dynamic grasping with the "Bleach Cleanser" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking.

Trajectory	Null-Space + Switching					Without Null-Space					Without Re-rank				
	Switch	**Time**	**Lift**	**Rot.**	**Shake**	**Switch**	**Time**	**Lift**	**Rot.**	**Shake**	**Switch**	**Time**	**Lift**	**Rot.**	**Shake**
H. Line	2	30.85	100	100	100	3	48.15	100	100	100	NA	48.8	100	100	100
H. Line + Rot.	2	39.92	100	100	100	3	38.39	100	100	100	NA	-	-	-	-
V. Line	2	43.66	100	0	0	2	49.87	0	0	0	NA	-	-	-	-
V. Line + Rot.	2	37.76	100	100	0	2	31.08	100	100	0	NA	-	-	-	-
Diag.	4	110.26	100	100	100	3	39.55	100	100	100	NA	-	-	-	-
Diag. + Rot.	2	31.45	100	100	100	2	50.41	100	100	100	NA	-	100	-	-
Circle	3	76.81	100	100	100	0	62.44	0	0	0	NA	81.11	100	100	100
Circle + Rot.	5	88.07	100	100	100	4	214.04	100	0	0	NA	82.44	100	100	100
Wave	0	82.44	100	100	100	2	46.29	100	100	100	NA	41.29	100	100	100
Wave + Rot.	2	35.73	100	100	100	2	56.10	100	100	100	NA	156.32	0	0	0

Table 7.3: Performance of dynamic grasping with the "Mustard Bottle" object across various testing trajectories and conditions: null-space with switching, without null-space, and without re-ranking.

Trajectory	Null-Space + Switching					Without Null-Space					Without Re-rank				
	Switch	Time	Lift	Rot.	Shake	Switch	Time	Lift	Rot.	Shake	Switch	Time	Lift	Rot.	Shake
H. Line	0	26.68	100	0	0	2	59.09	100	100	100	NA	49.90	100	100	0
H. Line + Rot.	0	40.94	100	100	100	3	38.39	100	100	100	NA	37.88	100	100	0
V. Line	-	-	-	-	-	-	-	-	-	-	NA	52.32	0	0	0
V. Line + Rot.	2	91.75	100	100	100	2	97.56	100	100	100	NA	65.49	100	100	100
Diag.	2	45.31	100	100	100	5	83.55	100	0	0	NA	-	-	-	-
Diag. + Rot.	1	45.31	100	100	100	1	32.90	100	0	0	NA	-	-	-	-
Circle	5	88.07	100	100	100	7	197.10	0	0	0	NA	145.87	100	100	100
Circle + Rot.	3	179.42	100	100	100	3	66.40	100	100	100	NA	-	-	-	-
Wave	1	29.83	100	100	0	1	47.80	0	0	0	NA	48.09	100	100	0
Wave + Rot.	3	45.69	100	100	100	3	0	0	0	0	NA	48.57	100	100	100

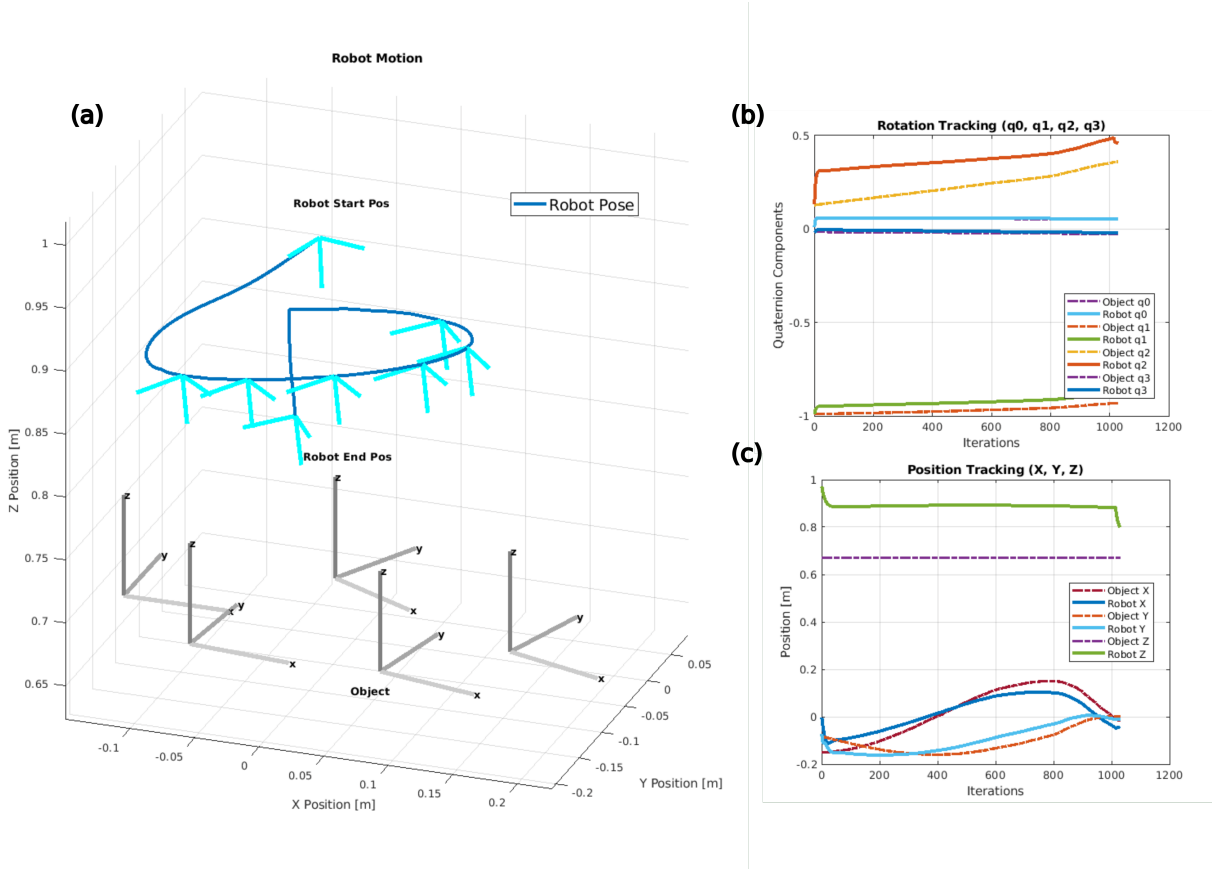


Figure 7.9: Tracking plots for grasping an object following an ellipsoid trajectory using the null-space controller from (7.19). (a) depicts the 3D trajectory of the end-effector towards the object, with grey frames representing object motion at some instants and blue frames showing the end-effector pose during various moments of the trajectory. In (b) the position tracking over time is shown, and in (c) the rotation tracking.

7.6 Conclusion

In this chapter, we presented a dual quaternion-based visual servoing controller for grasping moving objects, incorporating a secondary null-space task for joint limit avoidance and a novel re-ranking strategy. This approach, which utilizes the LoCoMo grasp quality metric, dual quaternion error, and a vantage-point tree-based search, demonstrated smooth and efficient object tracking even with perturbations. Experimental results highlighted the controller's convergence and underscored the importance of the secondary task and re-ranking strategy for robust and stable grasping.

Future research could explore integrating learning-based tracking strategies to enhance adaptability, real-time grasp generation techniques for complex objects, and incorporating object velocity into the control law to improve robustness. Additional enhancements might include dual quaternion tracking with neural networks, model predictive control for more adaptive strategies, and integrating obstacle avoidance for better safety. Combining these advancements with path planning mechanisms could further optimize trajectories for efficient, collision-free movements.

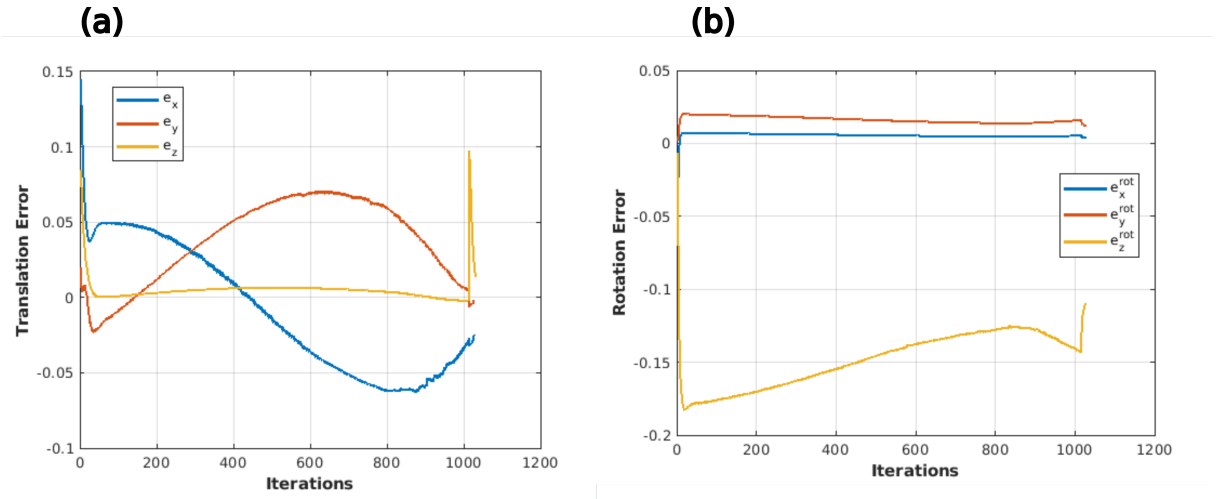


Figure 7.10: Error convergence plots for grasping an object following an ellipsoid trajectory using the null-space control law from (7.19). (a)is the translational error, and (b) the rotational error during the grasping process.

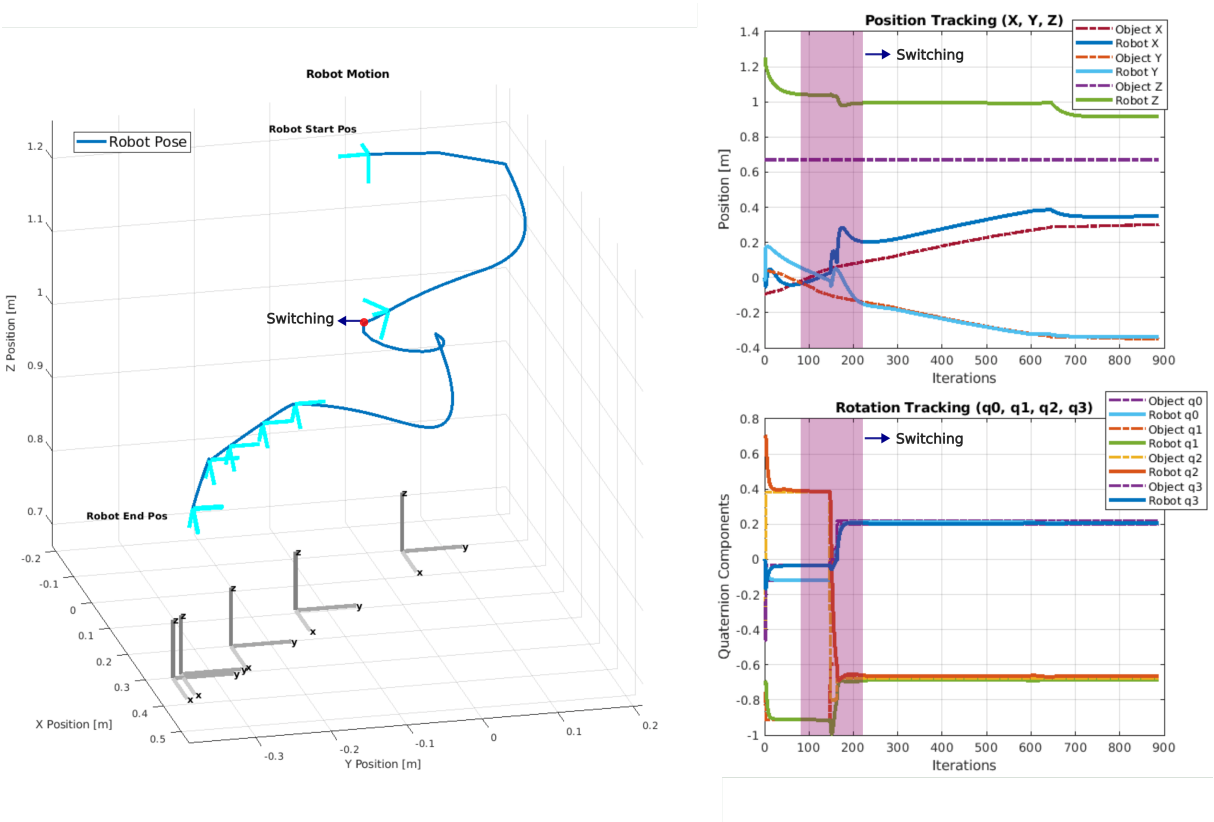


Figure 7.11: Tracking plots for grasping an object along a diagonal trajectory using the full controller from (7.19), with grasp re-ranking enabled. The moment of the grasp configuration switch is highlighted. (a) shows the 3D trajectory of the end-effector towards the object, with grey frames representing object motion and blue frames showing the end-effector pose at various points along the trajectory. (b) shows the position tracking over time. (c) The rotation tracking.

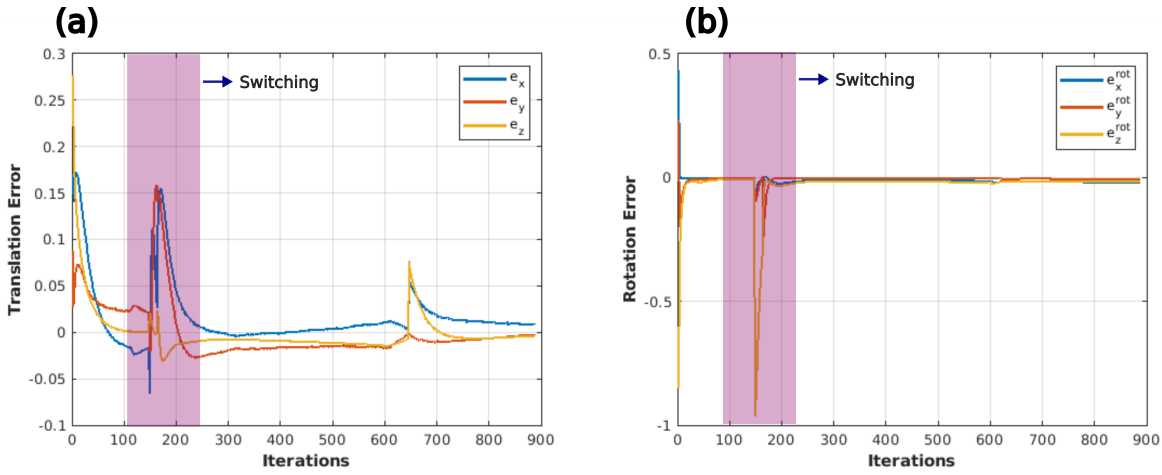


Figure 7.12: Error convergence plots for grasping an object along a diagonal trajectory using the full control law from (7.19), with grasp re-ranking enabled. The figure highlights the increase in error when the grasp re-ranking algorithm is activated. (a) depicts the translational error during the grasping process and (b) the rotational error during the grasping process.

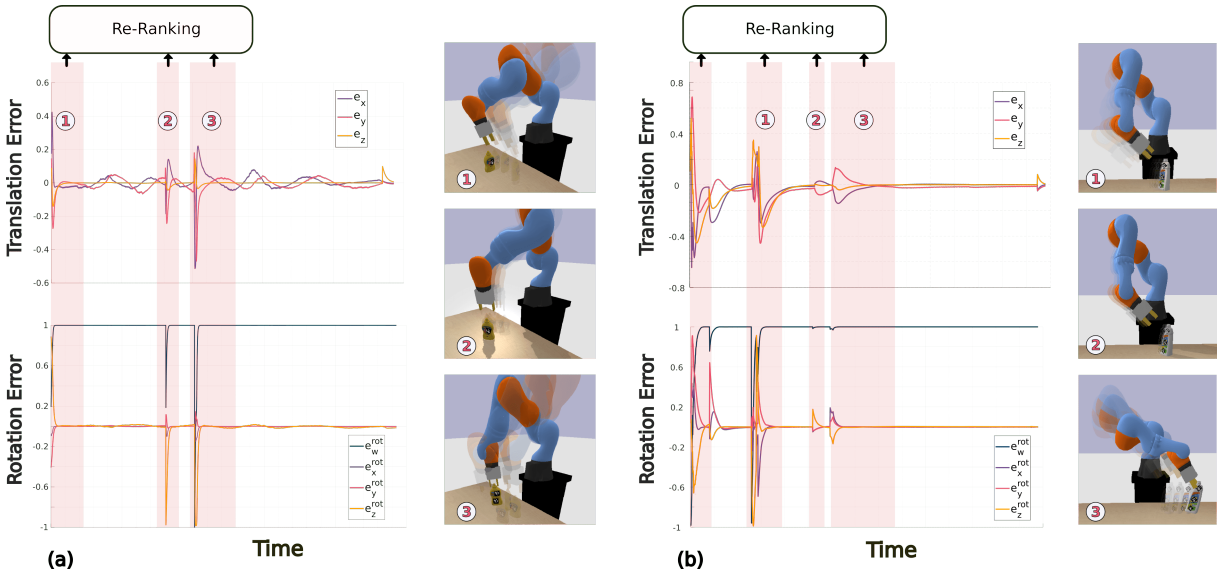


Figure 7.13: Error convergence plots using the full controller for (a) the "Mustard Bottle" object following an ellipsoidal trajectory with object rotation (trajectory 4 from the list in Section 7.5.1), and (b) the "Bleach Cleanser" object following a horizontal trajectory (1 from the list in 7.5.1) with rotation. In both cases, the moments of grasp re-ranking, when switching occurs, are highlighted.

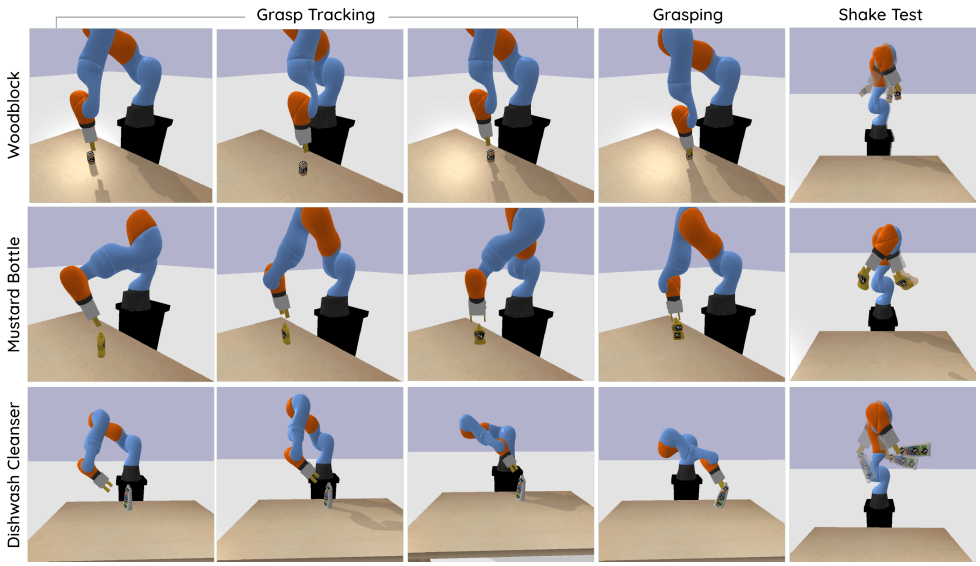


Figure 7.14: Screenshots depicting the robotic grasping of moving objects. The first three columns show various positions during the trajectory, the fourth column displays the object being grasped, and the final column shows the grasp stability test.

Part V

Closing remarks & supplementary materials

Chapter 8

Conclusion and future works

A primary objective in robotics involves the deployment of robots in unknown environments where minimal prior information is available, and perception uncertainty is frequently encountered. Indeed, contemporary research is increasingly shifting away from predictable, structured applications within factory floors and laboratories, towards more complex environments such as collaborative factory settings, households, and field applications. Throughout this thesis, various strategies have been proposed to address these challenges, particularly with respect to how objects can be meaningfully grasped and manipulated and how perception uncertainty and object dynamics should be treated.

To address the issue of grasping, the initial focus of this work was placed on representing tasks and task-aware grasps over the surface of an object. This representation was further combined with template-based approaches to facilitate the transfer of information to other deformed instances of the same object, as well as to different objects within the same category. In Chapter 3, the concept of using the Functional Maps Framework (FM) to transfer grasp information between rigid parts of deformed instances of the same object was introduced. By assuming that local parts of an object remain rigid during deformation and by mapping between two instances of an object, grasp transfer was made possible. As grasp transfer was treated as a segmentation transfer problem, a fine adjustment step to compensate for errors was needed. This approach was extended in Chapter 4, where instead of directly mapping rigid parts and transforming their poses, a framework for skill transfer, based on transferring functions defined over the surface of an object, was developed. The FM framework, combined with an interpolation-based imitation learning path planning approach, was applied to generalise skill transfer to different tasks, environments, and objects. Furthermore, in Chapter 5, a method was proposed that integrated shape completion based on Gaussian Process Regression (GPR) with FM-based transfer for grasp cost functions, enabling meaningful results even when only partial views of an object were available. To improve grasping stability, the inherent uncertainty of predicted surfaces was incorporated into the grasp cost function, expanding the graspable region onto unobserved areas while avoiding regions of high uncertainty.

Subsequently, the focus shifted towards the active exploration of objects. In Chapter 6, a method for active shape exploration based on Bayesian optimisation was proposed. Given a partially observed instance of an object, a method was introduced to simultaneously complete the shape of the object and enhance grasp stability of multi-finger grasps—particularly the probability of force closure was explored as a metric to be optimised. This method facilitated the identification of high-quality grasps (or refined versions thereof) alongside improved shape estimates, which is particularly beneficial when non-symmetric parts of an object are occluded (e.g. when a handle was not initially visible).

Finally, the challenge of grasping objects that move unpredictably was addressed in Chapter 7. This was approached through visual servoing, where a dual quaternion-based visual servoing method was proposed. Dual quaternions, offering a non-minimal, singularity-free representation that couples rotational and translational motion, enabled the achievement of a smooth controller. However, since representation singularities are not the only singularities a robot may encounter, the controller was further refined with a null-space projected secondary task and a re-ranking scheme that allowed for grasps to be switched before they became unfeasible. This controller demonstrated a high level of success over limited workspaces and challenging trajectories.

While these strategies contribute significantly to the current state of the art in robotic grasping, particularly in task-driven grasps, skill execution, and strategies for handling the inherent uncertainties and dynamic nature of unstructured environments, numerous challenges still remain. Following, potential future directions for this work are outlined.

8.1 Limitations

While the methods developed in this thesis have advanced the state of the art in robotic grasping under uncertainty and in unstructured environments, several limitations remain that may affect their applicability in real-world scenarios.

First, the use of the FM framework for transferring affordance functions or rigid pose information of a segmented object part is beneficial. However, these methods operate under the assumption that object shapes should be nearly isometric—i.e., geodesic distances within the shape should be preserved. Methodologies for handling non-isometric deformations or achieving scale invariance were not explored. Furthermore, this assumption limits the effective transfer between objects that are similar but structurally different, such as transferring information between a cup and a mug with a handle, or between a teapot and a pitcher. Likewise, for tasks involving some types of articulated objects, such as opening a drawer, challenges arise. When the drawer is closed, its surface appears as part of a single, continuous cabinet structure. However, once opened, the spatial relationships between different parts change, breaking the isometry condition assumed by the FM framework. As a result, meaningful transfer of grasping or task information is not feasible unless the system explicitly models the articulation of the drawer in advance, al-

lowing it to account for these non-isometric transformations. Additionally, the transfer primarily focuses on geometric information, while the task-related information—represented as functions over the surface of an object—relies on kinesthetic demonstrations or user input. These functions lack explicit semantic meaning, which limits their interpretability and adaptability to novel tasks.

With regard to task representation, while the proposed method can generalise to a variety of tasks, it remains constrained by the functional definition. Complex tasks that involve multiple stages or intricate interactions between multiple objects were not considered. The current framework does not explicitly address task sequencing, reasoning about dependencies between sub-tasks, or scenarios where multiple objects need to be manipulated in coordination.

For objects subject to occlusions, GP-based shape completion was employed. To improve efficiency, a sparse GP model was selected; however, this method remains unsuitable for real-time modelling or tracking due to computational constraints. Another limitation of the chosen GP model is the inherent smoothness of the function. While differentiability is advantageous for certain applications, it presents challenges when modelling sharp edges and discontinuities, which are common in many real-world objects. As a result, the reconstructed shapes may lack fine-grained details, potentially affecting downstream grasping performance.

Regarding both the BO method for active shape exploration and the VS approach for grasping moving objects, no real-world experiments were conducted. Although these methods are promising for practical deployment, and there are no major concerns regarding the sim-to-real transferability, several challenges must still be addressed. These include effective motion planning for real-time execution, workspace limitations that may affect grasp feasibility, and robust object tracking—particularly in active exploration scenarios where objects may not be continuously observed, as discussed in Chapter 6. Addressing these challenges is essential for the successful deployment of these approaches in dynamic, unstructured environments.

8.2 Future Directions

Part-Based Matching for Intra-Class Matching In this thesis, and particularly in Chapter 5, partial shape matching was explored by estimating full shapes before transferring information. This strategy expands the grasp search space as uncertainty can be used to safely estimate grasps in occluded regions. However, one drawback of estimating a full shape is that it does not allow for part-based matching, which could be useful for inter-class grasp or affordance transfer (e.g., by matching the handle of a mug to the handle of a teapot). Alternative techniques for non-rigid partial matching that enable such tasks could be explored in future research.

Priors for Bayesian Optimisation The presented method for Bayesian optimisation can be initialised with prior object information. In this work, grasp exploration was initialised by send-

ing the robot to explore a few configurations based on randomly selected samples. Alternatively, a grasp planner could be used to suggest initial grasps that could then be refined as the shape structure evolves. A more interesting alternative would be to incorporate prior knowledge from different objects from the same class, transferred using the proposed approaches in Part II of this thesis. The grasp transfer algorithm from Chapter 3 could be used to directly transfer a selected set of grasps from one object to another, further guiding the search to regions of interest. Moreover, insights from Chapter 5 could be employed to address the challenges associated with partiality.

Motion Prediction and Markerless Object Tracking In this thesis, object tracking was accomplished with the aid of markers. A natural extension of this work would involve the incorporation of markerless tracking algorithms, using features such as [8], into the pipeline. Additionally, predicting the object's motion could further enhance the visual servoing approach of Chapter 7. Kalman filters (particularly the Extended and Unscented variants) could be employed for this purpose. Alternatively, learning-based approaches for motion estimation, which have been shown to improve results as demonstrated in [9], could be explored. For an end-to-end method that leverages the synergies between the linear and angular components of motion, learning based dual quaternion-based strategies for motion estimation, as suggested in [170], could be investigated. Furthermore, these could be integrated with more advanced control techniques, such as Model Predictive Control, to enhance reactivity.

References

[1] Ahmed Abdelreheem et al. “Zero-shot 3d shape correspondence”. In: *SIGGRAPH Asia 2023 Conference Papers*. 2023, pp. 1–11.

[2] Jad Abou-Chakra et al. *Physically Embodied Gaussian Splatting: A Realtime Correctable World Model for Robotics*. 2024. DOI: 10.48550/arxiv.2406.10788. arXiv: 2406.10788 [cs.RO].

[3] M. Adjigble et al. “Model-free and learning-free grasping by Local Contact Moment matching”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2018, pp. 2933–2940.

[4] Maxime Adjigble et al. “3d spectral domain registration-based visual servoing”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 769–775.

[5] Maxime Adjigble et al. “An assisted telemanipulation approach: combining autonomous grasp planning with haptic cues”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 3164–3171.

[6] Maxime Adjigble et al. “Spectgrasp: Robotic grasping by spectral correlation”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 3987–3994.

[7] Bruno Vilhena Adorno and Murilo Marques Marinho. “DQ Robotics: A Library for Robot Modeling and Control”. In: *IEEE Robotics and Automation Letters* 28.3 (Sept. 2021), pp. 102–116. ISSN: 1070-9932. DOI: 10.1109/MRA.2020.2997920. URL: https://ieeexplore.ieee.org/document/9136790/.

[8] Ayush Aggarwal, Rustam Stolkin, and Naresh Marturi. “Unsupervised learning-based approach for detecting 3D edges in depth maps”. In: *Scientific Reports* 14.1 (2024), p. 796.

[9] Iretiayo Akinola et al. “Dynamic grasping with reachability and motion awareness”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 9422–9429.

[10] Sara Aldhaheri et al. “Underwater robot manipulation: Advances, challenges and prospective ventures”. In: *OCEANS 2022-Chennai*. IEEE. 2022, pp. 1–7.

[11] P.K. Allen et al. “Automated tracking and grasping of a moving object with a robotic hand-eye system”. In: *IEEE Transactions on Robotics and Automation* 9.2 (1993), pp. 152–165. DOI: 10.1109/70.238279.

[12] F Allmendinger, S Charaf Eddine, and B Corves. “Coordinate-invariant rigid-body interpolation on a parametric C1 dual quaternion curve”. In: *Mechanism and Machine Theory* (Mar. 2018), pp. 731–744.

[13] Kirsty Aquilina, David AW Barton, and Nathan F Lepora. “Tactile control for object tracking and dynamic contour following”. In: *Robotics and Autonomous Systems* 178 (2024), p. 104710.

[14] Paola Ardón et al. “Affordances in robotic tasks–a survey”. In: *arXiv preprint arXiv:2004.07400* (2020).

[15] C Atkeson and S Schaal. “Learning tasks from a single demonstration”. In: *Proceedings of International Conference on Robotics and Automation*. Vol. 2. 1997, pp. 1706–1712.

[16] M. Aubry, U. Schlickewei, and D. Cremers. “The wave kernel signature: A quantum mechanical approach to shape analysis”. In: *IEEE international conference on computer vision workshops (ICCV workshops)*. 2011, pp. 1626–1633.

[17] Yasemin Bekiroglu et al. “Benchmarking Protocol for Grasp Planning Algorithms”. In: *IEEE Robotics and Automation Letters* 5.2 (2019), pp. 315–322.

[18] P. J. Besl and N. D. McKay. “A method for registration of 3-D shapes”. In: *IEEE Transactions on Pattern analysis and Machine Intelligence,* 14.2 (1992), pp. 239–256.

[19] Silvia Biasotti et al. “Recent trends, applications, and perspectives in 3d shape similarity assessment”. In: *Computer Graphics Forum*. Vol. 35. 6. Wiley Online Library. 2016, pp. 87–119.

[20] A Bicchi. “On the force-closure property of robotic grasping”. In: *IFAC* 27.14 (1994), pp. 213–218.

[21] A. Bicchi and V. Kumar. “Robotic grasping and contact: a review”. In: *IEEE international Conference on Robotics and Automation (ICRA)*. Vol. 1. 2000, 348–353 vol.1. DOI: 10.1109/ROBOT.2000.844081.

[22] Alexander Bierbaum et al. “Grasp affordances from multi-fingered tactile exploration using dynamic potential fields”. In: *IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2009, pp. 168–174.

[23] Aude Billard and Danica Kragic. “Trends and challenges in robot manipulation”. In: *Science* 364.6446 (2019), eaat8414. DOI: 10.1126/science.aat8414.

[24] Marten Björkman et al. “Enhancing visual perception of shape through tactile glances”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 3180–3186.

[25] Jeannette Bohg et al. “Data-driven grasp synthesis—a survey”. In: *IEEE Transactions on Robotics* 30.2 (2013), pp. 289–309.

[26] Jeannette Bohg et al. “Mind the gap-robotic grasping under incomplete observation”. In: *2011 IEEE international Conference on Robotics and Automation*. IEEE. 2011, pp. 686–693.

[27] Michael Bombile and Aude Billard. “Bimanual dynamic grabbing and tossing of objects onto a moving target”. In: *Robotics and Autonomous Systems* 167 (2023), p. 104481.

[28] Christoph Borst, Max Fischer, and Gerd Hirzinger. “Grasping the dice by dicing the grasp”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 4. IEEE. 2003, pp. 3692–3697.

[29] Benjamin Busam, Tolga Birdal, and Nassir Navab. “Camera pose filtering with local regression geodesics on the riemannian manifold of dual quaternions”. In: *IEEE International Conference on Computer Vision Qorkshops*. 2017, pp. 2436–2445.

[30] S Calinon. “Learning from demonstration (programming by demonstration)”. In: *Encyclopedia of Robotics* (2018), pp. 1–8.

[31] Berk Calli et al. “The ycb object and model set: Towards common benchmarks for manipulation research”. In: *Iinternational Conference on Advanced Robotics (ICAR)*. IEEE. 2015, pp. 510–517.

[32] Antonio Candelieri. “A Gentle Introduction to Bayesian Optimization”. In: *2021 Winter Simulation Conference (WSC)*. 2021, pp. 1–16. DOI: 10.1109/WSC52266.2021.9715413.

[33] J. Castanheira et al. “Finding safe 3D robot grasps through efficient haptic exploration with unscented Bayesian optimization and collision penalty”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1643–1648.

[34] François Chaumette and Seth Hutchinson. “Visual servo control. I. Basic approaches”. In: *IEEE Robotics and Automation Letters* 13 (2006), pp. 82–90.

[35] François Chaumette and Seth Hutchinson. “Visual servo control. II. Advanced approaches [Tutorial]”. In: *IEEE Robotics and Automation Letters* 14.1 (2007), pp. 109–118.

[36] Dong Chen, Vincent Dietrich, and Georg von Wichert. “Precision grasping based on probabilistic models of unknown objects”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 2044–2051.

[37] Sirui Chen et al. “Diffsrl: Learning dynamical state representation for deformable object manipulation with differentiable simulation”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 9533–9540.

[38] Yiting Chen et al. “Sliding touch-based exploration for modeling unknown object shape with multi-fingered hands”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 8943–8950.

[39] Ping Yong Chua, T Ilschner, and Darwin G Caldwell. “Robotic manipulation of food products–a review”. In: *Industrial Robot: An International Journal* 30.4 (2003), pp. 345–354.

[40] Erwin Coumans and Yunfei Bai. *Pybullet, a python module for physics simulation for games, robotics and machine learning*. 2016.

[41] Jinda Cui and Jeff Trinkle. “Toward next-generation learned robot manipulation”. In: *Science robotics* 6.54 (2021), eabd9461.

[42] Shaowei Cui et al. “Grasp state assessment of deformable objects using visual-tactile fusion perception”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 538–544.

[43] Konstantinos Daniilidis. “Hand-eye calibration using dual quaternions”. In: *The International Journal of Robotics Research* 18.3 (1999), pp. 286–298.

[44] Cristiana De Farias et al. “Dual quaternion-based visual servoing for grasping moving objects”. In: *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2021, pp. 151–158.

[45] Cristiana De Farias et al. “Grasp Transfer for Deformable Objects by Functional Map Correspondence”. In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2022, pp. 735–741. DOI: 10.1109/ICRA46639.2022.9812141. URL: https://deformable-workshop.github.io/icra2022/.

[46] Norman Di Palo and Edward Johns. “DINOBot: Robot manipulation via retrieval and alignment with vision foundation models”. In: *arXiv preprint arXiv:2402.13181* (2024).

[47] Stanimir Dragiev, Marc Toussaint, and Michael Gienger. “Gaussian process implicit surfaces for shape estimation and grasping”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2011, pp. 2845–2850.

[48] Stanimir Dragiev, Marc Toussaint, and Michael Gienger. “Uncertainty aware grasping and tactile exploration”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 113–119.

[49] Alin Drimus et al. “Classification of rigid and deformable objects using a novel tactile sensor”. In: *15th International Conference on Advanced Robotics (ICAR)*. IEEE. 2011, pp. 427–434.

[50] Haonan Duan et al. “Robotics dexterous grasping: The methods based on point cloud and deep learning”. In: *Frontiers in Neurorobotics* 15 (2021), p. 658280.

[51] Tom Duckett et al. “Agricultural robotics: the future of robotic agriculture”. In: *arXiv preprint arXiv:1806.06762* (June 2018).

[52] Simon Duenser et al. “Interactive robotic manipulation of elastic objects”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3476–3481.

[53] Roberto M Dyke et al. “SHREC 2020 Track: Non-rigid shape correspondence of physically-based deformations”. In: *Eurographics Workshop on 3D Object Retrieval* (2020).

[54] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD’96. Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[55] Sara Fabbri et al. “Disentangling representations of object and grasp properties in the human brain”. In: *Journal of Neuroscience* 36.29 (2016), pp. 7648–7662.

[56] Cristiana de Farias et al. “Grasp Transfer for Deformable Objects by Functional Map Correspondence”. In: *Proceedings of the 2nd Workshop on Representing and Manipulating Deformable Objects*. 2022. URL: https://deformable-workshop.github.io/icra2022/.

[57] Cristiana de Farias et al. “Simultaneous Tactile Exploration and Grasp Refinement”. In: *In Bridging the Gap between Data-driven and Analytical Physics-based Grasping and Manipulation II, ICRA 2021*. 2022. URL: https://deformable-workshop.github.io/icra2022/.

[58] Cristiana de Farias et al. “Simultaneous Tactile Exploration and Grasp Refinement for Unknown Objects”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3349–3356. DOI: 10.1109/LRA.2021.3063074.

[59] Cristiana de Farias et al. “Task-Informed Grasping of Partially Observed Objects”. In: *IEEE Robotics and Automation Letters* 9.10 (2024), pp. 8394–8401. DOI: 10.1109/LRA.2024.3445633.

[60] João Gutemberg Farias, Edson De Pieri, and Daniel Martins. “A Review on the Applications of Dual Quaternions”. In: *Machines* 12.6 (2024), p. 402.

[61] Thomas Feix, Ian M Bullock, and Aaron M Dollar. “Analysis of human grasping behavior: Object characteristics and grasp type”. In: *IEEE. Transactions on Haptics* 7.3 (2014), pp. 311–323.

[62] Carlo Ferrari and John F Canny. “Planning optimal grasps”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 3. 1992, pp. 2290–2295.

[63] Luis Felipe Cruz Figueredo, Bruno Vilhena Adorno, and João Yoshiyuki Ishihara. "Robust H-infinity kinematic control of manipulator robots using dual quaternion algebra". In: *Automatica* (June 2021), pp. 1–8.

[64] Peter I Frazier. "A tutorial on Bayesian optimization". In: *arXiv preprint arXiv:1807.02811* (2018).

[65] Gabriela Zarzar Gandler et al. "Object shape estimation and modeling, based on sparse Gaussian process implicit surfaces, combining visual data and tactile exploration". In: *Robotics and Autonomous Systems* 126 (2020), p. 103433. ISSN: 0921-8890. DOI: 10.1016/j.robot.2020.103433. URL: https://www.sciencedirect.com/science/article/pii/S0921889019303495.

[66] Wei Gao and Russ Tedrake. "kpam 2.0: Feedback control for category-level robotic manipulation". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 2962–2969.

[67] Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

[68] S. Garrido-Jurado et al. "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming". In: *Pattern Recognition* 51 (2016), pp. 481–491.

[69] Sergio Garrido-Jurado et al. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6 (2014), pp. 2280–2292.

[70] A Gasparetto and L Scalera. "A Brief History of Industrial Robotics in the 20th Century". In: *Advances in Historical Studies* 8.1 (2019), pp. 24–35.

[71] Sebastian Geidenstam et al. "Learning of 2D grasping strategies from box-based 3D object approximations." In: *Robotics: Science and Systems*. Vol. 2008. 2009.

[72] James J Gibson. "The theory of affordances". In: *Hilldale, USA* 1.2 (1977), pp. 67–82.

[73] Jared Glover, Daniela Rus, and Nicholas Roy. "Probabilistic models of object geometry with application to grasping". In: *The International Journal of Robotics Research* 28.8 (2009), pp. 999–1019.

[74] GPy. *GPy: A Gaussian process framework in python*. http://github.com/SheffieldML/GPy. 2012.

[75] Mona Gridseth, Katharina Hertkorn, and Martin Jagersand. "On visual servoing to improve performance of robotic grasping". In: *12th Conference on Computer and Robot Vision*. IEEE. 2015, pp. 245–252.

[76] Feida Gu et al. "A Survey on Robotic Manipulation of Deformable Objects: Recent Advances, Open Challenges and New Frontiers". In: *arXiv preprint arXiv:2312.10419* (2023).

[77] William Rowan Hamilton. *Elements of quaternions*. Longmans, Green, & Company, 1866.

[78] D. Han, Q. Wei, and Z. Li. "A Dual-quaternion Method for Control of Spatial Rigid Body". In: *IEEE International Conference on Networking, Sensing and Control*. 2008, pp. 1–6. DOI: 10.1109/ICNSC.2008.4525172.

[79] Dapeng Han et al. "Control of oriented mechanical systems: A method based on dual quaternion". In: *IFAC Proceedings Volumes* 41.2 (2008), pp. 3836–3841.

[80] Yunhai Han et al. "Learning generalizable vision-tactile robotic grasping strategy for deformable objects via transformer". In: *IEEE/ASME Transactions on Mechatronics* (2024).

[81] Mohammed Hassanin, Salman Khan, and Murat Tahtali. "Visual affordance and function understanding: A survey". In: *ACM Computing Surveys (CSUR)* 54.3 (2021), pp. 1–35.

[82] Hannah B. Helbig and Marc O. Ernst. "Optimal integration of shape information from vision and touch". In: *Experimental Brain Research*. 2007, pp. 1432–1106.

[83] U. Hillenbrand and M. A. Roa. "Transferring functional grasps through contact warping and local replanning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2012, pp. 2963–2970.

[84] Martin Hjelm et al. "Representations for cross-task, cross-object grasp transfer". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 5699–5704.

[85] N. Houshangi. "Control of a robotic manipulator to grasp a moving target using vision". In: *IEEE International Conference on Robotics and Automation*. 1990, 604–609 vol.1. DOI: 10.1109/ROBOT.1990.126048.

[86] Jiang Hua et al. "Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning". In: *Sensors* 21.4 (2021), p. 1278.

[87] Isabella Huang et al. "DefGraspSim: Physics-Based Simulation of Grasp Outcomes for 3D Deformable Objects". In: *IEEE Robotics and Automation Letters* 7.3 (2022), pp. 6274–6281. DOI: 10.1109/LRA.2022.3158725.

[88] Kanrun Huang and Tucker Hermans. "Building 3d object models during manipulation by reconstruction-aware trajectory optimization". In: *arXiv preprint arXiv:1905.03907* (2019).

[89] Ahmed Hussein et al. "Imitation learning: A survey of learning methods". In: *ACM Computing Surveys (CSUR)* 50.2 (2017), pp. 1–35.

[90] A Ijspeert, J Nakanishi, and S Schaal. "Movement imitation with nonlinear dynamical systems in humanoid robots". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2002.

[91] AJ Ijspeert et al. "Dynamical movement primitives: learning attractor models for motor behaviors". In: *Neural computation* 25.2 (2013), pp. 328–373.

[92] Itseez. *Open Source Computer Vision Library*. https://github.com/itseez/opencv. 2015.

[93] P Jiménez. "Survey on model-based manipulation planning of deformable objects". In: *Robotics and computer-integrated manufacturing* 28.2 (2012), pp. 154–163.

[94] Roland S Johansson. "Sensory control of dexterous manipulation in humans". In: *Hand and brain*. Elsevier, 1996, pp. 381–414.

[95] Roland S Johansson. "Sensory input and control of grip". In: *Novartis Foundation Symposium*. Wiley Online Library. 1998, pp. 45–58.

[96] Mohsen Kaboli, Kunpeng Yao, and Gordon Cheng. "Tactile-based manipulation of deformable objects with dynamic center of mass". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2016, pp. 752–757.

[97] Rainer Kartmann, Danqing Liu, and Tamim Asfour. "Semantic scene manipulation based on 3d spatial object relations and language instructions". In: *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2021, pp. 306–313.

[98] Michael Kazhdan and Hugues Hoppe. "Screened poisson surface reconstruction". In: *ACM Transactions on Graphics (ToG)* 32.3 (2013), pp. 1–13.

[99] Farshad Khadivar et al. "Online active and dynamic object shape exploration with a multi-fingered robotic hand". In: *Robotics and Autonomous Systems* 166 (2023), p. 104461.

[100] Fouad F Khalil and Pierre Payeur. "Robotic interaction with deformable objects under vision and tactile guidance-a review". In: *2007 International Workshop on Robotic and Sensors Environments*. IEEE. 2007, pp. 1–6.

[101] Seungsu Kim, Ashwini Shukla, and Aude Billard. "Catching objects in flight". In: *IEEE Transactions on Robotics* 30.5 (2014), pp. 1049–1065.

[102] A. Koessler et al. "An efficient approach to closed-loop shape control of deformable objects using finite element models". In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2021, pp. 1637–1643. DOI: 10.1109/ICRA48506.2021.9560919.

[103] Marek Kopicki et al. "One-shot learning and generation of dexterous grasps for novel objects". In: *The International Journal of Robotics Research* 35.8 (2016), pp. 959–976.

[104] Marek S Kopicki, Dominik Belter, and Jeremy L Wyatt. "Learning better generative models for dexterous, single-view grasping of novel objects". In: *The International Journal of Robotics Research* 38.10-11 (2019), pp. 1246–1267.

[105] Oliver Kroemer, Scott Niekum, and George Konidaris. "A review of robot learning for manipulation: Challenges, representations, and algorithms". In: *The Journal of Machine Learning Research* 22.1 (2021), pp. 1395–1476.

[106] Robert Krug, Yasemin Bekiroglu, and Máximo A Roa. “Grasp quality evaluation done right: How assumed contact force bounds affect wrench-based quality metrics”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 1595–1600.

[107] Riddhiman Laha et al. “Coordinate invariant user-guided constrained path planning with reactive rapidly expanding plane-oriented escaping trees”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 977–984.

[108] Riddhiman Laha et al. “Point-to-Point Path Planning Based on User Guidance and Screw Linear Interpolation”. In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 85451. American Society of Mechanical Engineers. 2021, V08BT08A010.

[109] M. Laskey et al. “Multi-armed bandit models for 2D grasp planning with uncertainty”. In: *IEEE International Conference on Automation Science and Engineering (CASE)*. 2015, pp. 572–579. DOI: 10.1109/CoASE.2015.7294140.

[110] Michael Laskey et al. “Learning robust bed making using deep imitation learning with dart”. In: *arXiv preprint arXiv:1711.02525* (2017).

[111] Tran Nguyen Le, Fares J Abu-Dakka, and Ville Kyrki. “Sponge: Sequence planning with deformable-on-rigid contact prediction from geometric features”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 10596–10603.

[112] John M Lee. *Riemannian manifolds: an introduction to curvature*. Vol. 176. Springer Science & Business Media, 2006.

[113] Robert Lee et al. “Sample-efficient learning of deformable linear object manipulation in the real world through self-supervision”. In: *IEEE Robotics and Automation Letters* 7.1 (2021), pp. 573–580.

[114] Qujiang Lei, Jonathan Meijer, and Martijn Wisse. “A survey of unknown object grasping and our fast grasping algorithm-C shape grasping”. In: *International Conference on Control, Automation and Robotics (ICCAR)*. IEEE. 2017, pp. 150–157.

[115] Nathan F Lepora, Kirsty Aquilina, and Luke Cramphorn. “Exploratory tactile servoing with active touch”. In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 1156–1163.

[116] K. Li et al. “Simultaneous Localization and Mapping Using a Novel Dual Quaternion Particle Filter”. In: *21st Int. Conf. on Information Fusion*. 2018, pp. 1668–1675. DOI: 10.23919/ICIF.2018.8455347.

[117] Miao Li et al. “Dexterous grasping under shape uncertainty”. In: *Robotics and Autonomous Systems* 75 (2016), pp. 352–364.

[118] Ying Li, Jiaxin L Fu, and Nancy S Pollard. "Data-driven grasp synthesis using shape matching and task-based pruning". In: *IEEE Transactions on Visualization and Computer Graphics* 13.4 (2007), pp. 732–747.

[119] Hsien-Chung Lin et al. "A Framework for Robot Grasp Transferring with Non-rigid Transformation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 2941–2948.

[120] Hanzhong Liu et al. "Multi-fingered tactile servoing for grasping adjustment under partial observation". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 7781–7788.

[121] Jirong Liu et al. "Target-referenced reactive grasping for dynamic objects". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 8824–8833.

[122] Q. Lu, M. Van der Merwe, and T. Hermans. "Multi-Fingered Active Grasp Learning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 8415–8422. DOI: 10.1109/IROS45743.2020.9340783.

[123] Wen Hao Lui and Ashutosh Saxena. "Tangled: Learning to untangle ropes with RGB-D perception". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 837–844. DOI: 10.1109/IROS.2013.6696448.

[124] K.M. Lynch and F.C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017. ISBN: 9781316609842. URL: https://books.google.co.uk/books?id=86G0nQAACAAJ.

[125] M. Madry, D. Song, and D. Kragic. "From object categories to grasp transfer using probabilistic reasoning". In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2012, pp. 1716–1723.

[126] Jeffrey Mahler et al. "Dex-Net 1.0: A cloud-based network of 3D objects for robust grasp planning using a Multi-Armed Bandit model with correlated rewards". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1957–1964. DOI: 10.1109/ICRA.2016.7487342.

[127] Jeffrey Mahler et al. "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics". In: *arXiv preprint arXiv:1703.09312* (2017).

[128] Jeffrey Mahler et al. "Dex-net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning". In: *IEEE international Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 5620–5627.

[129] Jeffrey Mahler et al. "Gp-gpis-opt: Grasp planning with shape uncertainty using gaussian process implicit surfaces and sequential convex programming". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 4919–4926.

[130] Lucas Manuelli et al. “kpam: Keypoint affordances for category-level robotic manipulation”. In: *The International Symposium of Robotics Research*. Springer. 2019, pp. 132–157.

[131] Wolfram Martens et al. “Geometric Priors for Gaussian Process Implicit Surfaces”. In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 373–380. DOI: 10.1109/LRA.2016.2631260.

[132] Naresh Marturi et al. “Dynamic grasp and trajectory planning for moving objects”. In: *Autonomous Robots* 43 (2019), pp. 1241–1256.

[133] Naresh Marturi et al. “Towards advanced robotic manipulation for nuclear decommissioning: A pilot study on tele-operation and autonomy”. In: *2016 International Conference on Robotics and Automation for Humanitarian Applications (RAHA)*. IEEE. 2016, pp. 1–8.

[134] Matthew T. Mason. “Toward Robotic Manipulation”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 1.Volume 1, 2018 (2018), pp. 1–28. ISSN: 2573-5144. DOI: 10.1146/annurev-control-060117-104848.

[135] Jan Matas, Stephen James, and Andrew J Davison. “Sim-to-real reinforcement learning for deformable object manipulation”. In: *Conference on Robot Learning*. PMLR. 2018, pp. 734–743.

[136] Simone Melzi et al. “ZoomOut: Spectral Upsampling for Efficient Shape Correspondence”. In: *ACM Transactions on Graphics* (2019).

[137] Mark Meyer et al. “Discrete differential-geometry operators for triangulated 2-manifolds”. In: *Visualization and mathematics III*. Springer. 2003, pp. 35–57.

[138] R. Monica and J. Aleotti. “Point Cloud Projective Analysis for Part-Based Grasp Planning”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4695–4702.

[139] Andriy Myronenko and Xubo Song. “Point set registration: Coherent point drift”. In: *IEEE Transactions on Pattern analysis and Machine Intelligence,* 32.12 (2010), pp. 2262–2275.

[140] Ashvin Nair et al. “Combining self-supervised learning and imitation for vision-based rope manipulation”. In: *IEEE international Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 2146–2153.

[141] NDA Blog. *The Role of Robotics in Nuclear Decommissioning*. Accessed: 2024-08-24. 2023. URL: https://nda.blog.gov.uk/the-role-of-robotics-in-nuclear-decommissioning/.

[142] Rhys Newbury et al. “Deep Learning Approaches to Grasp Synthesis: A Review”. In: *IEEE Transactions on Robotics* 39.5 (2023), pp. 3994–4015. DOI: 10.1109/TRO.2023.3280597.

[143] Hai Nguyen and Hung La. “Review of deep reinforcement learning for robot manipulation”. In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE. 2019, pp. 590–595.

[144] José Nogueira et al. “Unscented Bayesian optimization for safe robot grasping”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 1967–1972.

[145] Valerio Ortenzi et al. “Object handovers: a review for robotics”. In: *IEEE Transactions on Robotics* 37.6 (2021), pp. 1855–1873.

[146] Simon Ottenhaus et al. “Visuo-haptic grasping of unknown objects based on gaussian process implicit surfaces and deep learning”. In: *IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2019, pp. 402–409.

[147] Maks Ovsjanikov et al. “Computing and processing correspondences with functional maps”. In: *SIGGRAPH ASIA 2016 Courses*. 2016, pp. 1–60.

[148] Maks Ovsjanikov et al. “Functional maps: a flexible representation of maps between shapes”. In: *ACM Transactions on Graphics (ToG)* 31.4 (2012), pp. 1–11.

[149] Evangelos Papadopoulos et al. “Robotic manipulation and capture in space: A survey”. In: *Frontiers in Robotics and AI* 8 (2021), p. 686723.

[150] A Paraschos et al. “Probabilistic movement primitives”. In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc. 2013.

[151] Andreas ten Pas et al. “Grasp Pose Detection in Point Clouds”. In: *The International Journal of Robotics Research* 36.13–14 (2017), pp. 1455–1473.

[152] David Paulius et al. “Functional object-oriented network for manipulation learning”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 2655–2662.

[153] Florian T Pokorny, Yasemin Bekiroglu, and Danica Kragic. “Grasp moduli spaces and spherical harmonics”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 389–396.

[154] Florian T Pokorny et al. “A database for reproducible manipulation research: CapriDB–Capture, Print, Innovate”. In: *Data in brief* 11 (2017), pp. 491–498.

[155] Mila Popović et al. “Grasping unknown objects using an early cognitive vision system for general scene understanding”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2011, pp. 987–994.

[156] Jana Procházková and Dalibor Martišek. “Notes on iterative closest point algorithm”. In: *17th Conference on Applied Mathematics*. 2018, pp. 876–884.

[157] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

[158] H Ravichandar et al. "Recent advances in robot learning from demonstration". In: *Annual Review of Control, Robotics, and Autonomous Systems* 3 (2020), pp. 297–330.

[159] Jing Ren et al. "Continuous and Orientation-Preserving Correspondences via Functional Maps". In: *ACM Trans. Graph.* 37.6 (Dec. 2018). ISSN: 0730-0301. DOI: 10.1145/3272127.3275040.

[160] Jing Ren et al. "Structured regularization of functional map computations". In: *Computer Graphics Forum*. Vol. 38. 5. Wiley Online Library. 2019, pp. 39–53.

[161] Yu Ren, Ronghan Chen, and Yang Cong. "Autonomous Manipulation Learning for Similar Deformable Objects via Only One Demonstration". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 17069–17078. DOI: 10.1109/CVPR52729.2023.01637.

[162] D. Rodriguez and S. Behnke. "Transferring category-based functional grasping skills by latent space non-rigid registration". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2662–2669.

[163] Diego Rodriguez et al. "Transferring grasping skills to novel instances by latent space non-rigid registration". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4229–4236.

[164] Yusuf Sahillioğlu. "Recent advances in shape correspondence". In: *The Visual Computer* 36.8 (2020), pp. 1705–1721.

[165] R. Saltus et al. "Dual Quaternion Visual Servo Control". In: *IEEE Conference on Decision and Control (CDC)*. 2020, pp. 5956–5961.

[166] Jose Sanchez et al. "Blind manipulation of deformable objects based on force sensing and finite element modeling". In: *Frontiers in Robotics and AI* 7 (2020), p. 73.

[167] Jose Sanchez et al. "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey". In: *The International Journal of Robotics Research* 37.7 (2018), pp. 688–716.

[168] Anik Sarker, Anirban Sinha, and Nilanjan Chakraborty. "On screw linear interpolation for point-to-point path planning". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 9480–9487.

[169] Philipp Schmidt et al. "Grasping of unknown objects using deep convolutional neural networks based on depth images". In: *IEEE international Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 6831–6838.

[170] Andreas Schwung, Johannes Pöppelbaum, and Pradeep C Nutakki. "Rigid body movement prediction using dual quaternion recurrent neural networks". In: *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*. Vol. 1. IEEE. 2021, pp. 756–761.

[171] Jon M Selig. *Geometric fundamentals of robotics*. Springer Science & Business Media, 2004.

[172] B. Shahriari et al. "Taking the Human Out of the Loop: A Review of Bayesian Optimization". In: *Proceedings of the IEEE* 104.1 (2016), pp. 148–175. DOI: 10.1109/JPROC.2015.2494218.

[173] Bokui Shen et al. "Acid: Action-conditional implicit visual dynamics for deformable object manipulation". In: *arXiv preprint arXiv:2203.06856* (2022).

[174] Bruno Siciliano, Oussama Khatib, and Torsten Kröger. *Springer handbook of robotics*. Vol. 200. Springer, 2008.

[175] Anthony Simeonov et al. "Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation". In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2022.

[176] Anthony Simeonov et al. "SE(3)-Equivariant Relational Rearrangement with Neural Descriptor Fields". In: *Proceedings of The 6th Conference on Robot Learning*. Ed. by Karen Liu, Dana Kulic, and Jeff Ichnowski. Vol. 205. Proceedings of Machine Learning Research. PMLR, 2023, pp. 835–846. URL: https://proceedings.mlr.press/v205/simeonov23a.html.

[177] Simons. *SCHUNK Dextrous Hand 2.0 (SDH 2.0) Datasheet*. http://www.schunk.com. Version: April 10th, 2008. Bahnhofstrasse 106-134, 74348 Lauffen / Neckar, Germany, Apr. 2008. URL: http://www.schunk.com.

[178] Nicolas Sommer and Aude Billard. "Multi-contact haptic exploration and grasping with tactile sensors". In: *Robotics and Autonomous Systems* 85 (2016), pp. 48–61.

[179] Dan Song et al. "Task-based robot grasp planning using probabilistic inference". In: *IEEE Transactions on Robotics* 31.3 (2015), pp. 546–561.

[180] Jingyi Song et al. "Robust task-based grasping as a service". In: *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2020, pp. 22–28.

[181] T. Stouraitis, U. Hillenbrand, and M. A. Roa. "Functional power grasps transferred through warping and replanning". In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2015, pp. 4933–4940.

[182] J. Stückler and S. Behnke. “Efficient deformable registration of multi-resolution surfel maps for object manipulation skill transfer”. In: *IEEE international Conference on Robotics and Automation (ICRA)*. 2014, pp. 994–1001.

[183] Ahmet Tekden, Marc Peter Deisenroth, and Yasemin Bekiroglu. “Grasp Transfer Based on Self-Aligning Implicit Representations of Local Surfaces”. In: *IEEE Robotics and Automation Letters* 8.10 (2023), pp. 6315–6322. DOI: 10.1109/LRA.2023.3306272.

[184] Hao Tian et al. “Transferring Grasp Configurations using Active Learning and Local Replanning”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 1622–1628.

[185] S. Tokumoto, Y. Fujita, and S. Hirai. “Deformation modeling of viscoelastic objects for their shape control”. In: *IEEE international Conference on Robotics and Automation*. Vol. 1. 1999, 767–772 vol.1. DOI: 10.1109/ROBOT.1999.770067.

[186] Nikolaus Vahrenkamp et al. “Part-based grasp planning for familiar objects”. In: *2016 IEEE-RAS 16th international conference on humanoid robots (humanoids)*. IEEE. 2016, pp. 919–925.

[187] Bruno Vilhena Adorno. “Robot Kinematic Modeling and Control Based on Dual Quaternion Algebra — Part I: Fundamentals.” working paper or preprint. Feb. 2017. URL: https://hal.archives-ouvertes.fr/hal-01478225.

[188] Jonathan Vorndamme et al. “Integrated Bi-Manual Motion Generation and Control shaped for Probabilistic Movement Primitives”. In: *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*. IEEE. 2022, pp. 202–209.

[189] Vitalis Vosylius and Edward Johns. “Few-Shot In-Context Imitation Learning via Implicit Graph Alignment”. In: *7th Annual Conference on Robot Learning*. 2023. URL: https://openreview.net/forum?id=CnKf9TyYtf2.

[190] T. Wada, S. Hirai, and S. Kawamura. “Indirect simultaneous positioning operations of extensionally deformable objects”. In: *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications*. Vol. 2. 1998, 1333–1338 vol.2. DOI: 10.1109/IROS.1998.727484.

[191] Angelina Wang et al. “Learning robotic manipulation through visual planning and acting”. In: *arXiv preprint arXiv:1905.04411* (2019).

[192] Ruicheng Wang et al. “Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 11359–11366.

[193] Shaoxiong Wang et al. “3d shape perception from monocular vision, touch, and shape priors”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1606–1613.

[194] J. Weisz and P. K. Allen. “Pose error robust grasping from contact wrench space metrics”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. May 2012, pp. 557–562.

[195] Bowen Wen et al. “You only demonstrate once: Category-level manipulation from single visual demonstration”. In: *Robotics: Science and Systems (RSS)*. 2022.

[196] Oliver Williams and Andrew Fitzgibbon. “Gaussian process implicit surfaces”. In: *Gaussian Processes in Practice*. 2006.

[197] Thomas Wimmer, Peter Wonka, and Maks Ovsjanikov. *Back to 3D: Few-Shot 3D Keypoint Detection with Back-Projected 2D Features*. 2024. DOI: 10.48550/arxiv.2311.18113. arXiv: 2311.18113 [cs.CV].

[198] Rina Wu et al. “Cross-Category Functional Grasp Tansfer”. In: *arXiv preprint arXiv:2405.08310* (2024).

[199] Y Wu and Y Demiris. “Towards one-shot learning by imitation for humanoid robots”. In: *IEEE international Conference on Robotics and Automation (ICRA)*. IEEE. 2010.

[200] Jingyi Xiang et al. “TrackDLO: Tracking Deformable Linear Objects Under Occlusion With Motion Coherence”. In: *IEEE Robotics and Automation Letters* 8.10 (2023), pp. 6179–6186. DOI: 10.1109/LRA.2023.3303710.

[201] Jingyi Xu et al. “Minimal work: A grasp quality metric for deformable hollow objects”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 1546–1552.

[202] Natsuki Yamanobe et al. “A brief review of affordance in robotic manipulation research”. In: *Advanced Robotics* 31.19-20 (2017), pp. 1086–1101.

[203] Mengyuan Yan et al. “Self-supervised learning of state estimation for manipulating deformable linear objects”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2372–2379.

[204] Zhengkun Yi et al. “Active tactile object exploration with gaussian processes”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 4925–4930.

[205] Peter N Yianilos. “Data structures and algorithms for nearest neighbor search in general metric spaces”. In: *Soda*. Vol. 93. 194. 1993, pp. 311–21.

[206] Hang Yin, Anastasia Varava, and Danica Kragic. “Modeling, learning, perception, and control methods for deformable object manipulation”. In: *Science Robotics* 6.54 (2021), eabd8803.

[207] Tianhe Yu et al. “One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning”. In: *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*. Ed. by Hadas Kress-Gazit et al. 2018. DOI: 10.15607/RSS.2018.XIV.002. URL: http://www.roboticsproceedings.org/rss14/p02.html.

[208] Yuanxin Wu et al. “Strapdown inertial navigation system algorithms based on dual quaternions”. In: *IEEE Transactions on Aerospace and Electronic Systems* 41.1 (2005), pp. 110–132.

[209] Irati Zamalloa et al. “Dissecting robotics-historical overview and future perspectives”. In: *arXiv preprint arXiv:1704.08617* (2017).

[210] Gioele Zardini, Joseph Lorenzetti, and Marco Pavone. *Principles of Robot Autonomy*. 2024.

[211] Maryam Zare et al. “A survey of imitation learning: Algorithms, recent developments, and challenges”. In: *IEEE Transactions on Cybernetics* (2024).

[212] M Zeestraten et al. “An approach for imitation learning on Riemannian manifolds”. In: *IEEE Robotics and Automation Letters* (2017).

[213] Hanbo Zhang et al. “Robotic grasping from classical to modern: A survey”. In: *arXiv preprint arXiv:2202.03631* (2022).

[214] Hanbo Zhang et al. “Visual Manipulation Relationship Network for Autonomous Robotics”. In: *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. 2018, pp. 118–125. DOI: 10.1109/HUMANOIDS.2018.8625071.

[215] Zengzhi Zhao et al. “Grasp prediction and evaluation of multi-fingered dexterous hands using deep learning”. In: *Robotics and Autonomous Systems* 129 (2020), p. 103550.

[216] Jihong Zhu et al. “Challenges and Outlook in Robotic Manipulation of Deformable Objects”. In: *IEEE Robotics and Automation Letters* 29.3 (2022), pp. 67–77.

[217] Claudio Zito et al. “Sequential trajectory re-planning with tactile information gain for dexterous grasping under object-pose uncertainty”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2013, pp. 4013–4020.

Appendix A

Expanded Tables for PFM

In this Appendix the expanded tables for Chapter 5 are presented.

A.1 Method Comparison Tables

Here, the full results summarised in Table 5.2 are presented. Tables A.1 and A.2 refer respectively to the results for the open and pour function with the target bottle in pose 1 (aligned with the source) and Tables A.3 and A.4 refer to the bottles in pose 2–rotated 90 degrees. Best results are in bold.

A.2 Expanded Full Results Table

Here, the full results summarised in Table 5.3 are presented. The following tables each detail the results for one object and function. The entries in the tables represent the outcomes of each test conducted over five trials, following the protocol outlined in Section 5.5.3. A successful experiment is marked as 1 and a failure as 0, with averages reflecting values within this range. Additionally, it is important to note that the objects were placed in different configurations, denoted as **PX**. When a particular function is not feasible in a given configuration, these instances are omitted from the tables.

Table A.1: Pose 1 Open Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates.

	FM					CPD					ICP				
Cuts	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]
X-Cuts															
C1	0.2860	0.3376	**0.6410**	0.3347	20.00	**0.1273**	**0.2179**	0.7787	**0.1939**	**60.00**	0.1789	0.2643	0.8196	0.1947	20.00
C2	**0.0743**	**0.0970**	**0.2532**	**0.0808**	80.00	0.2534	0.4009	0.9840	0.3111	80.00	0.2551	0.3738	0.9840	0.2733	**100.00**
C3	**0.0353**	**0.0452**	**0.1372**	**0.0288**	**100.00**	0.3225	0.4724	0.9851	0.3466	**100.00**	0.2842	0.4172	0.9851	0.3057	**100.00**
Y-Cuts															
C1	**0.1416**	**0.1832**	**0.4566**	**0.1608**	60.00	0.1846	0.3087	0.9719	0.2628	60.00	0.1744	0.2634	0.9338	0.1974	**80.00**
C2	**0.0327**	**0.0513**	**0.1680**	**0.0512**	**100.00**	0.2469	0.4038	0.9834	0.3225	**100.00**	0.2226	0.3455	0.9834	0.2644	**100.00**
C3	**0.0206**	**0.0294**	**0.1059**	**0.0263**	**100.00**	0.2559	0.4133	0.9851	0.3299	**100.00**	0.2398	0.3590	0.9851	0.2673	**100.00**
Z-Cuts															
C1	**0.6903**	**0.7046**	**0.8438**	**0.1415**	**20.00**	0.8120	0.8333	0.9851	0.1876	0.00	0.7845	0.8001	0.9851	0.1568	0.00
C2	**0.2502**	**0.3081**	**0.7286**	**0.1814**	**100.00**	0.5174	0.5975	0.9851	0.2990	0.00	0.3661	0.4513	0.9847	0.2641	**100.00**
C3	0.2733	0.3315	**0.6640**	**0.2013**	80.00	0.3188	0.4643	0.9851	0.3427	**100.00**	**0.2164**	**0.3281**	0.9851	0.2469	**100.00**

[1] MAE: Mean Average Error
[2] RMSE: Root Mean Square Error
[3] MAX: Maximum Error
[4] STD: Standard Deviation of Error
[5] CP: Average percentage of success in grasping the correct object part.

Table A.2: Pose 1 Pour Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates.

	FM					CPD					ICP				
Cuts	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]
X-Cuts															
C1	**0.1987**	**0.2228**	**0.4198**	**0.1612**	60.00	0.3957	0.4478	0.9339	0.2100	**100.00**	0.3996	0.4704	0.9332	0.2902	80.00
C2	**0.1043**	**0.1316**	**0.3623**	**0.1313**	**100.00**	0.2639	0.3598	0.9333	0.2539	**100.00**	0.2339	0.3174	0.9333	0.2785	**100.00**
C3	**0.0407**	**0.0632**	**0.2854**	**0.0627**	**100.00**	0.2294	0.3241	0.9563	0.2404	**100.00**	0.2030	0.2814	0.9365	0.2683	**100.00**
Y-Cuts															
C1	**0.1151**	**0.1287**	**0.2843**	**0.0593**	80.00	0.3022	0.3702	0.9119	0.2177	**100.00**	0.3033	0.3694	0.9171	0.3297	**100.00**
C2	**0.0914**	**0.1440**	**0.4202**	**0.1213**	**100.00**	0.2265	0.3104	0.9314	0.2226	**100.00**	0.2281	0.2979	0.9331	0.2815	**100.00**
C3	**0.0674**	**0.0816**	**0.2391**	**0.0759**	**100.00**	0.2093	0.3012	0.9587	0.2279	**100.00**	0.2300	0.3068	0.9454	0.2885	**100.00**
Z-Cuts															
C1	0.2739	0.3075	0.4783	0.1582	0.00	**0.1117**	**0.1326**	**0.2519**	**0.0901**	0.00	0.1424	0.1793	0.6613	0.1791	0.00
C2	0.1486	**0.1828**	**0.5674**	**0.1690**	60.00	0.1880	0.2629	0.7897	0.2499	**100.00**	**0.1361**	0.2228	0.7921	0.1916	**100.00**
C3	0.3302	0.3796	**0.8476**	0.3794	60.00	0.2470	0.3771	0.9587	0.3069	**100.00**	**0.2018**	**0.3231**	0.9587	**0.2935**	**100.00**

[1] MAE: Mean Average Error
[2] RMSE: Root Mean Square Error
[3] MAX: Maximum Error
[4] STD: Standard Deviation of Error
[5] CP: Average percentage of success in grasping the correct object part.

Table A.3: Pose 2 Open Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates.

	FM					CPD					ICP				
Cuts	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]
X-Cuts															
C1	0.2732	0.3125	0.7063	0.3037	0.00	**0.2031**	**0.2857**	**0.9532**	**0.2729**	0.00	0.2300	0.3343	0.8196	0.2427	0.00
C2	**0.0955**	**0.1131**	**0.2122**	**0.0930**	**80.00**	0.3826	0.5181	0.9840	0.4192	0.00	0.4020	0.5460	0.9840	0.3696	0.00
C3	**0.1915**	**0.2307**	**0.4122**	**0.1755**	**60.00**	0.4527	0.5895	0.9851	0.4232	0.00	0.4774	0.6128	0.9851	0.3842	0.00
Y-Cuts															
C1	**0.1541**	**0.1858**	**0.4464**	**0.1680**	**60.00**	0.2812	0.4009	0.9669	0.3118	0.00	0.2922	0.4174	0.9740	0.2981	0.00
C2	**0.0401**	**0.0552**	**0.1945**	**0.0505**	**80.00**	0.3832	0.5172	0.9834	0.3707	0.00	0.3995	0.5366	0.9834	0.3582	0.00
C3	**0.0844**	**0.1147**	**0.3437**	**0.1025**	**100.00**	0.4096	0.5452	0.9851	0.3881	0.00	0.4297	0.5656	0.9851	0.3679	0.00
Z-Cuts															
C1	**0.8038**	**0.8202**	**0.9418**	0.1631	0.00	0.8469	0.8563	0.9851	0.1261	0.00	0.8694	0.8762	0.9851	**0.1091**	0.00
C2	**0.3559**	**0.3910**	**0.6352**	**0.1869**	**40.00**	0.6067	0.6688	0.9851	0.3172	0.00	0.6453	0.6978	0.9851	0.2654	0.00
C3	**0.3456**	**0.4159**	**0.7645**	**0.3203**	**40.00**	0.4753	0.5839	0.9851	0.3704	0.00	0.5066	0.6098	0.9851	0.3394	0.00

[1] MAE: Mean Average Error
[2] RMSE: Root Mean Square Error
[3] MAX: Maximum Error
[4] STD: Standard Deviation of Error
[5] CP: Average percentage of success in grasping the correct object part.

Table A.4: Pose 2 Pour Function: Comparison of function transfer methods (FM, CPD, ICP) using various error metrics and success rates.

	FM					CPD					ICP				
Cuts	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]	MAE[1]	RMSE[2]	MAX[3]	STD[4]	CP (%)[5]
X-Cuts															
C1	**0.1872**	**0.2238**	**0.4908**	**0.1855**	**40.00**	0.5432	0.6059	0.9371	0.2893	0.00	0.5744	0.6237	0.9371	0.2430	0.00
C2	**0.0818**	**0.1123**	**0.3154**	**0.1111**	**100.00**	0.4287	0.5197	0.9339	0.3381	0.00	0.4445	0.5389	0.9371	0.3047	0.00
C3	**0.0390**	**0.0528**	**0.2952**	**0.0516**	**100.00**	0.3760	0.4748	0.9736	0.3713	0.00	0.3860	0.4924	0.9587	0.3057	0.00
Y-Cuts															
C1	**0.2922**	**0.3466**	**0.6335**	0.2988	**20.00**	0.4363	0.5097	0.9219	0.3192	0.00	0.5028	0.5654	0.9223	**0.2585**	0.00
C2	**0.1762**	**0.1988**	**0.3665**	**0.1771**	**40.00**	0.4080	0.4951	0.9628	0.3533	0.00	0.4368	0.5247	0.9371	0.2907	0.00
C3	**0.1475**	**0.1771**	**0.4482**	**0.1758**	**80.00**	0.4047	0.4959	0.9587	0.3688	0.00	0.4170	0.5113	0.9587	0.2958	0.00
Z-Cuts															
C1	0.1487	0.1933	0.4374	0.1404	0.00	0.1636	0.2344	0.9683	0.2307	0.00	**0.1086**	**0.1302**	**0.2519**	**0.0718**	0.00
C2	0.3101	**0.3408**	**0.6052**	0.2771	**40.00**	**0.2756**	0.3502	0.9723	0.3001	0.00	0.2737	0.3462	0.8082	**0.2119**	0.00
C3	**0.2857**	**0.3318**	**0.7731**	0.3296	**60.00**	0.4003	0.5019	0.9609	0.3890	0.00	0.4135	0.5174	0.9587	**0.3110**	0.00

[1] MAE: Mean Average Error
[2] RMSE: Root Mean Square Error
[3] MAX: Maximum Error
[4] STD: Standard Deviation of Error
[5] CP: Average percentage of success in grasping the correct object part.

Table A.5: Bottle Pour Function

Table A.6: Data for the bottle experiments with the pour function

Bottle	P1					P2					P3				
	C.D.	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]	**C.D.**	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]	**C.D.**	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]
B1	0.0035	0.8	0.8	0.8	0.8	0.0038	1.0	1.0	1.0	1.0	0.0062	0.8	1.0	1.0	1.0
B2	0.0047	1.0	1.0	1.0	1.0	**Symmetrical**					0.0041	1.0	1.0	1.0	1.0
B3	0.0047	1.0	0.8	0.8	0.8	0.0062	1.0	0.8	0.8	0.8	0.0017	1.0	1.0	1.0	0.8
B4	0.0033	1.0	1.0	1.0	1.0	**Symmetrical**					0.0002	1.0	0.8	0.8	0.8
B5	0.0063	1.0	1.0	1.0	1.0	0.0034	1.0	1.0	1.0	1.0	0.0083	1.0	1.0	1.0	1.0
Mean	**0.0045**	**0.96**	**0.92**	**0.92**	**0.92**	**0.0044**	**1.0**	**0.9333**	**0.9333**	**0.9333**	**0.0041**	**0.96**	**0.96**	**0.96**	**0.92**

[*] Average of success in grasping the correct object part.

[‡] Average of success – reported out of 5 trials.

Table A.7: Bottle Open Function

Table A.8: Data for the bottle experiments with the open function

Bottle	P1					P2					P3				
	C.D.	Part*	Lift‡	Shake‡	Rot.‡	C.D.	Part*	Lift‡	Shake‡	Rot.‡	C.D.	Part*	Lift‡	Shake‡	Rot.‡
B1	0.0035	0.8	0.8	0.8	0.6	0.0038	0.8	0.8	0.8	0.8	0.0003	1.0	1.0	1.0	1.0
B2	0.0048	1.0	1.0	1.0	1.0	Symmetrical					0.0040	0.8	0.8	0.8	0.8
B3	0.0045	0.6	0.8	0.8	0.8	0.0061	1.0	1.0	1.0	1.0	0.0002	0.8	1.0	1.0	1.0
B4	0.0034	1.0	1.0	1.0	0.8	Symmetrical					0.0003	1.0	1.0	1.0	1.0
B5	0.0067	1.0	1.0	0.8	0.8	0.0035	1.0	1.0	1.0	1.0	0.0083	1.0	1.0	1.0	1.0
Mean	**0.0046**	**0.88**	**0.92**	**0.88**	**0.80**	**0.0044**	**0.9333**	**0.9333**	**0.9333**	**0.9333**	**0.0026**	**0.92**	**0.96**	**0.96**	**0.96**

* Average of success in grasping the correct object part.
‡ Average of success – reported out of 5 trials.

Table A.9: Bottle Handover Function

Table A.10: Data for the bottle experiments with the handover function

Bottle	P3				
	C.D.	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]
B1	0.0050	1.0	1.0	1.0	0.8
B3	0.0041	1.0	1.0	1.0	0.8
B5	0.0002	1.0	1.0	1.0	0.8
B6	0.0002	1.0	1.0	1.0	1.0
B8	0.0085	1.0	0.8	0.8	0.8
Mean	**0.0036**	**1.0**	**0.96**	**0.96**	**0.84**

[*] Average of success in grasping the correct object part.

[‡] Average of success – reported out of 5 trials.

Table A.11: Mug Store Function

Table A.12: Data for the mug experiments with the store function

Mug	P1					P2				
	C.D.	**Part***	**Lift‡**	**Shake‡**	**Rot.‡**	**C.D.**	**Part***	**Lift‡**	**Shake‡**	**Rot.‡**
M1	0.0030	1.0	0.6	0.6	0.6	0.0002	1.0	1.0	1.0	1.0
M2	0.0020	1.0	1.0	1.0	1.0	0.0015	1.0	1.0	1.0	1.0
M3	0.0042	1.0	1.0	1.0	0.8	0.0053	0.8	0.8	0.8	0.8
M4	0.0034	1.0	1.0	1.0	0.8	0.0025	1.0	1.0	1.0	1.0
M5	0.0033	1.0	1.0	1.0	0.8	0.0030	1.0	1.0	1.0	1.0
Mean	**0.0032**	**1.0**	**0.92**	**0.92**	**0.8**	**0.0025**	**0.96**	**0.96**	**0.96**	**0.96**

* Average of success in grasping the correct object part.
‡ Average of success – reported out of 5 trials.

Table A.13: Mug Drink Function

Table A.14: Data for the mug experiments with the drink function

Mug	**P1**					**P2**				
	C.D.	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]	**C.D.**	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]
M1	0.0024	1.0	1.0	0.7750	0.7750	0.0003	1.0	1.0	1.0	1.0
M2	0.0020	1.0	1.0	1.0	0.8	0.0018	1.0	0.8	0.8	0.8
M3	0.0042	1.0	1.0	1.0	1.0	0.0053	1.0	1.0	1.0	0.8
M4	0.0034	1.0	1.0	1.0	0.8	0.0025	1.0	1.0	1.0	1.0
M5	0.0033	1.0	0.8	0.8	0.6	0.0030	1.0	1.0	1.0	1.0
Mean	**0.0030**	**1.0**	**0.96**	**0.915**	**0.795**	**0.0026**	**1.0**	**0.96**	**0.96**	**0.92**

[*] Average of success in grasping the correct object part.

[‡] Average of success – reported out of 5 trials.

Table A.15: Mug Stack Function

Table A.16: Data for the mug experiments with the stack function

Mug	P1					P2				
	C.D.	**Part***	**Lift‡**	**Shake‡**	**Rot.‡**	**C.D.**	**Part***	**Lift‡**	**Shake‡**	**Rot.‡**
M1	0.0013	1.0	0.6	0.6	0.4	0.0002	1.0	1.0	0.8	0.8
M2	0.0020	0.4	0.8	0.8	0.8	0.0019	0.4	1.0	1.0	0.8
M3	0.0041	1.0	0.8	0.8	1.0	0.0053	1.0	1.0	1.0	0.8
M4	0.0033	1.0	1.0	1.0	1.0	0.0026	0.6	1.0	1.0	1.0
M5	0.0033	1.0	1.0	1.0	0.8	0.0030	1.0	1.0	0.8	0.8
Mean	**0.0028**	**0.88**	**0.84**	**0.84**	**0.8**	**0.0026**	**0.8**	**1.0**	**0.92**	**0.84**

* Average of success in grasping the correct object part.
‡ Average of success – reported out of 5 trials.

Table A.17: Mug Handover Function

Mug	P1					P2				
	C.D.	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]	**C.D.**	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]
M1	0.0030	0.8	0.8	0.8	0.8	0.0011	1.0	1.0	1.0	0.8
M2	0.0020	0.75	1.0	1.0	1.0	0.0019	0.6	1.0	1.0	0.8
M3	0.0041	1.0	1.0	0.8	0.8	0.0053	1.0	1.0	1.0	1.0
M4	0.0033	1.0	1.0	1.0	0.8	0.0025	1.0	1.0	1.0	1.0
M5	0.0034	0.8	0.8	0.8	0.8	0.0030	1.0	1.0	1.0	1.0
Mean	**0.0032**	**0.87**	**0.92**	**0.88**	**0.84**	**0.0028**	**0.92**	**1.0**	**1.0**	**0.92**

Table A.18: Data for the mug experiments with the handover function

[*] Average of success in grasping the correct object part.

[‡] Average of success – reported out of 5 trials.

Table A.19: Fruit Handover Function

Table A.20: Data for the fruit experiments with the handover function

Fruit	P1					P2				
	C.D.	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]	**C.D.**	**Part**[*]	**Lift**[‡]	**Shake**[‡]	**Rot.**[‡]
F1	0.0039	1.0	1.0	0.8	0.75	0.0049	0.8	0.8	0.8	0.8
F2	0.0042	0.8	1.0	1.0	1.0	0.0041	0.6	1.0	1.0	1.0
F3	0.0056	0.8	1.0	1.0	1.0	0.0038	1.0	1.0	1.0	0.8
F4	0.0037	1.0	1.0	1.0	1.0	0.0075	0.6	0.6	0.6	0.6
F5	0.0077	1.0	1.0	1.0	1.0	0.0096	1.0	0.8	0.8	0.8
Mean	**0.0050**	**0.92**	**1.0**	**0.96**	**0.95**	**0.0060**	**0.8**	**0.84**	**0.84**	**0.8**

[*] Average of success in grasping the correct object part.
[‡] Average of success – reported out of 5 trials.

Table A.21: Fruit Stack Function

Table A.22: Data for the fruit experiments with the stack function

Fruit	P1					P2				
	C.D.	Part*	Lift‡	Shake‡	Rot.‡	C.D.	Part*	Lift‡	Shake‡	Rot.‡
F1	0.0034	1.0	1.0	1.0	0.8	0.0051	1.0	1.0	1.0	1.0
F2	0.0043	1.0	0.6	0.6	0.6	0.0042	1.0	1.0	1.0	0.8
F3	0.0056	1.0	0.8	0.8	0.8	0.0070	1.0	1.0	1.0	1.0
F4	0.0033	0.8	0.8	0.8	0.8	0.0070	0.8	1.0	1.0	1.0
F5	0.0076	1.0	1.0	1.0	1.0	0.0078	0.8	1.0	1.0	0.8
Mean	**0.0049**	**0.96**	**0.84**	**0.84**	**0.8**	**0.0062**	**0.92**	**1.0**	**1.0**	**0.92**

* Average of success in grasping the correct object part.

‡ Average of success – reported out of 5 trials.

Appendix B

Dual Quaternions

Dual quaternion algebra is an extension of classical quaternion algebra that integrates the principles of dual numbers. This combination provides a powerful mathematical framework for representing rotations and translations in a coupled manner. The following sections introduce the fundamentals of quaternion algebra and subsequently, dual quaternion algebra. For a more comprehensive review of dual quaternions, readers are directed to [60] for an in-depth analysis and to [187] for a foundational overview and its applications in robotic kinematic modelling.

B.1 Quaternions

Quaternions are an extension of imaginary numbers introduced by Hamilton in the 19th century [77]. They are often used to represent three-dimensional rotations due to their ability to avoid the singularities and ambiguities associated with other rotation representations, such as Euler angles. Let $\hat{\imath}$, $\hat{\jmath}$, and $\hat{k}$ be the three quaternion unit vectors, defined such that $\hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = \hat{\imath}\hat{\jmath}\hat{k} = -1$. The algebra of quaternions is generated by the basis elements 1, $\hat{\imath}$, $\hat{\jmath}$, and $\hat{k}$, yielding the non-commutative set

$$\mathbb{H} \triangleq \left\{ \eta + \boldsymbol{\mu} \mid \boldsymbol{\mu} = \mu_1 \hat{\imath} + \mu_2 \hat{\jmath} + \mu_3 \hat{k},\ \eta, \mu_1, \mu_2, \mu_3 \in \mathbb{R} \right\}, \tag{B.1}$$

where, for any quaternion $\boldsymbol{x} \in \mathbb{H}$, η and $\boldsymbol{\mu}$ are the real and imaginary (or vector) parts, respectively. The quaternion conjugate is defined as $\boldsymbol{x}^* \triangleq \eta - \boldsymbol{\mu}$, which in turn allows the definition of the quaternion norm as $\|\boldsymbol{x}\|^2 = \boldsymbol{x}\boldsymbol{x}^* = \boldsymbol{x}^*\boldsymbol{x}$.

A significant subset of quaternions is the unit quaternions, defined as those quaternions with a norm of 1,

$$\mathcal{S}^3 \triangleq \{ \boldsymbol{x} \in \mathbb{H} \mid \|\boldsymbol{x}\| = 1 \}. \tag{B.2}$$

Unit quaternions are particularly useful for representing spatial rotations. Another important

subset is the pure quaternions,

$$\mathbb{H}_0 \triangleq \{\boldsymbol{x}_0 \mid \eta + \boldsymbol{\mu} \in \mathbb{H},\, \eta = 0\}\,, \tag{B.3}$$

which are used to represent spatial translations. These subsets play crucial roles in the representation of rigid body motions. The vector isomorphism under $\mathbb{H}_0$ facilitates operations such as inner and vector products, as well as the group multiplication of unit quaternions. This property ensures that for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{S}^3$, their quaternion multiplication $\boldsymbol{x}_3 = \boldsymbol{x}_1\boldsymbol{x}_2$ will also result in a unit quaternion, i.e., $\boldsymbol{x}_3 \in \mathcal{S}^3$.

Moreover, a unit quaternion representing a rotation by an angle ϕ around an axis $\boldsymbol{n} \in \mathbb{H}_0$ can be expressed as,

$$\boldsymbol{x} = \cos\left(\frac{\phi}{2}\right) + \boldsymbol{n}\sin\left(\frac{\phi}{2}\right). \tag{B.4}$$

This compact representation offers significant computational advantages and stability, making it a preferred choice in computer graphics, robotics, and aerospace applications.

B.1.1 Quaternion Products

Dual quaternion algebra extends the classical quaternion framework by integrating dual numbers, which consist of a real part and a dual part. This extension enables the simultaneous representation of rotation and translation in a unified algebraic structure, making dual quaternions a powerful tool for rigid body transformations in robotics. Given two pure quaternions $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{H}_0$, their inner and vector products are defined as:

$$\langle \boldsymbol{x}_1, \boldsymbol{x}_2 \rangle \triangleq -\frac{\boldsymbol{x}_1\boldsymbol{x}_2 + \boldsymbol{x}_2\boldsymbol{x}_1}{2}, \tag{B.5}$$

$$\boldsymbol{x}_1 \times \boldsymbol{x}_2 \triangleq \frac{\boldsymbol{x}_1\boldsymbol{x}_2 - \boldsymbol{x}_2\boldsymbol{x}_1}{2}. \tag{B.6}$$

These operations closely mirror the (inner) dot product and cross product in vector algebra, thereby providing a familiar framework within the quaternion algebra for manipulating and interpreting spatial relationships.

B.2 Dual Numbers

Dual numbers are an extension of the real number system, analogous to how complex numbers extend real numbers by introducing an imaginary unit. A dual number is composed of a primary part and a dual part, expressed as

$$d = r + \varepsilon d', \tag{B.7}$$

where r and d' are real numbers, and ε is the dual unit, characterised by the property $\varepsilon^2 = 0$ and $\varepsilon \neq 0$. The nilpotent nature of the dual unit ε—meaning its square equals zero—differentiates

dual numbers from complex numbers, where the imaginary unit has a square equal to -1.

Dual numbers are particularly useful rigid body mechanics and robotics, as they enable the simultaneous representation of quantities associated with both position and velocity, or rotation and translation, within a single mathematical framework. The primary part r typically corresponds to the primary physical quantity, such as position or rotation, while the dual part d' represents the associated infinitesimal change, such as velocity or translation.

B.3 Dual Quaternions

Dual quaternions extend the concept of imaginary numbers by incorporating quaternions with dual numbers, resulting in a non-minimal, non-commutative, and singularity-free mathematical framework for describing rigid body motions [171]. Formally, dual quaternions are defined as the set

$$\mathbb{H} \otimes \mathbb{D} \triangleq \left\{ \underline{\boldsymbol{x}} = \boldsymbol{x_P} + \varepsilon \boldsymbol{x_D} \mid \boldsymbol{x_P}, \boldsymbol{x_D} \in \mathbb{H} \right\}, \tag{B.8}$$

where ε is the nilpotent dual unit, as introduced in (B.7). In this context, $\boldsymbol{x_P}$ represents the primary quaternion part, and $\boldsymbol{x_D}$ represents the dual part of the dual quaternion $\underline{\boldsymbol{x}}$.

Analogous to quaternions, the conjugate of a dual quaternion is defined as $\underline{\boldsymbol{x}}^* \triangleq \boldsymbol{x}_P^* + \varepsilon \boldsymbol{x}_D^*$, and the norm of a dual quaternion is given by $\left\|\underline{\boldsymbol{x}}\right\|^2 = \underline{\boldsymbol{x}}\underline{\boldsymbol{x}}^*$. The subset of unit dual quaternions, defined as

$$\underline{\boldsymbol{S}} \triangleq \left\{ \underline{\boldsymbol{x}} \in \mathbb{H} \otimes \mathbb{D} \mid \left\|\underline{\boldsymbol{x}}\right\| = 1 \right\}, \tag{B.9}$$

forms the Lie group $\mathrm{Spin}(3) \ltimes \mathbb{R}^3$, which double covers the special Euclidean group $\mathrm{SE}(3)$. The inverse of a unit dual quaternion $\underline{\boldsymbol{x}} \in \underline{\boldsymbol{S}}$ is simply its conjugate $\underline{\boldsymbol{x}}^*$.

Additionally, the set of pure dual quaternions is defined as

$$\mathbb{H}_0 \otimes \mathbb{D} \triangleq \left\{ \underline{\boldsymbol{x_0}} = \boldsymbol{x_P} + \varepsilon \boldsymbol{x_D} \mid \boldsymbol{x_P}, \boldsymbol{x_D} \in \mathbb{H}_0 \right\}, \tag{B.10}$$

where $\boldsymbol{x_P}$ and $\boldsymbol{x_D}$ are pure quaternions (i.e., quaternions with zero real parts). For any pure dual quaternion $\underline{\boldsymbol{x_0}} \in \mathbb{H}_0 \otimes \mathbb{D}$, the conjugate satisfies $\underline{\boldsymbol{x_0}}^* = -\underline{\boldsymbol{x_0}}$.

B.3.1 Rigid Body Displacement and Twist

Full rigid body displacement, encapsulating both rotation and translation can be represented using dual quaternions. Let $\boldsymbol{x}_t \in \mathbb{H}_0$ represent a translation, and $\boldsymbol{x}_r \in \mathcal{S}^3$ a rotation of a rigid body. The complete rigid body displacement can then be described by the dual quaternion

$$\underline{\boldsymbol{x}} = \boldsymbol{x}_r + \varepsilon \frac{1}{2} \boldsymbol{x}_t \boldsymbol{x}_r, \tag{B.11}$$

where $\underline{\boldsymbol{x}}$ combines the rotation $\boldsymbol{x}_r$ with the translation $\boldsymbol{x}_t$, simultaneously capturing the full spatial displacement of the rigid body.

The kinematic equation for the body twist, which describes the instantaneous velocity of the rigid body, can also be expressed using dual quaternions. Given a dual quaternion $\underline{\boldsymbol{x}}$ representing the current displacement of the rigid body, the time derivative of this displacement, $\dot{\underline{\boldsymbol{x}}}$, is related to the body twist $\underline{\boldsymbol{\omega}}^B \in \mathbb{H}_0 \otimes \mathbb{D}$ through the equation:

$$\dot{\underline{\boldsymbol{x}}} = \frac{1}{2}\underline{\boldsymbol{x}}\,\underline{\boldsymbol{\omega}}^B, \tag{B.12}$$

where $\underline{\boldsymbol{\omega}}^B$ is a pure dual quaternion representing the twist, which includes both angular velocity and linear velocity components.

B.3.2 Frame Transformations and Inertial Twist

In the context of rigid body motions, efficiently describing transformations between different reference frames is crucial. Let ${}^I\underline{\boldsymbol{x}}_B \in \underline{\boldsymbol{S}}$ represent the transformation between the inertial frame $\mathbb{F}^I$ and the moving body frame $\mathbb{F}^B$ and $\underline{\boldsymbol{\omega}}^B$ be the body twist, defined as in (B.12). The inertial twist, $\underline{\boldsymbol{\omega}}_I$ which describes the instantaneous motion of the body in the inertial frame, can be expressed as

$$\underline{\boldsymbol{\omega}}_I = {}^I\underline{\boldsymbol{x}}_B\,\underline{\boldsymbol{\omega}}^B\,{}^I\underline{\boldsymbol{x}}_B^*, \tag{B.13}$$

where $\underline{\boldsymbol{\omega}}^B \in \mathbb{H}_0 \otimes \mathbb{D}$, and ${}^I\underline{\boldsymbol{x}}_B^*$ is the conjugate of the transformation.

The adjoint transformation, which facilitates the mapping of twists and other quantities between different frames, is then defined as

$$\mathrm{Ad}(\underline{\boldsymbol{x}})\underline{\boldsymbol{y}} \triangleq \underline{\boldsymbol{x}}\,\underline{\boldsymbol{y}}\,\underline{\boldsymbol{x}}^*, \tag{B.14}$$

where $\underline{\boldsymbol{x}} \in \underline{\boldsymbol{S}}$ is a unit dual quaternion and $\underline{\boldsymbol{y}} \in \mathbb{H}_0 \otimes \mathbb{D}$ is a pure dual quaternion representing a twist or other quantity in the body frame.

Using the adjoint transformation from (B.14), the inertial twist can be rewritten as:

$$\underline{\boldsymbol{\omega}}_I = \mathrm{Ad}({}^I\underline{\boldsymbol{x}}_B)\underline{\boldsymbol{\omega}}^B, \tag{B.15}$$

which expresses the inertial twist as a transformed version of the body twist in the inertial frame.

B.3.3 Pure Dual Quaternion Products

In dual quaternion algebra, the pure dual quaternions set, $\mathbb{H}_0 \otimes \mathbb{D}$ is usually used to represent both rotational and translational components of rigid body twists and screws. Similar to vector algebra, pure dual quaternions allow the definition of inner (dot) products and cross products.

Thus, let $\underline{\boldsymbol{x}}_1$ and $\underline{\boldsymbol{x}}_2$ be two pure dual quaternions. The inner (dot) product of these two pure dual quaternions is defined as

$$\langle \underline{\boldsymbol{x}}_1, \underline{\boldsymbol{x}}_2 \rangle \triangleq -\frac{\underline{\boldsymbol{x}}_1\underline{\boldsymbol{x}}_2 + \underline{\boldsymbol{x}}_2\underline{\boldsymbol{x}}_1}{2}. \tag{B.16}$$

Similarly, given the pure dual quaternions, $\underline{\boldsymbol{x}}_1$ and $\underline{\boldsymbol{x}}_2$ the cross product is given by

$$\underline{\boldsymbol{x}}_1 \times \underline{\boldsymbol{x}}_2 \triangleq \frac{\underline{\boldsymbol{x}}_1\underline{\boldsymbol{x}}_2 - \underline{\boldsymbol{x}}_2\underline{\boldsymbol{x}}_1}{2}. \tag{B.17}$$

This operation results in another pure dual quaternion that is orthogonal to both $\underline{\boldsymbol{x}}_1$ and $\underline{\boldsymbol{x}}_2$, similar to the cross product in vector algebra.

B.3.4 Vector Maps

In dual quaternion algebra, it is often necessary to convert between the algebraic representation of a dual quaternion and its equivalent vector form for computational and analytical purposes. These vector mappings allow for efficient computation and integration with algorithms that operate on standard vector spaces. This section introduces the operators used for one-to-one mapping between dual quaternions and vectors.

The vec_8 operator maps a dual quaternion $\underline{\boldsymbol{x}} = \boldsymbol{x_P} + \varepsilon \boldsymbol{x_D}$ to an 8-dimensional vector. This mapping includes both the real and dual components of the dual quaternion, providing a complete representation of the dual quaternion in vector form. Specifically, for a dual quaternion $\underline{\boldsymbol{x}}$, the vec_8 map is defined as

$$\text{vec}_8(\underline{\boldsymbol{x}}) \triangleq \begin{bmatrix} \eta_P \\ \mu_{1P} \\ \mu_{2P} \\ \mu_{3P} \\ \eta_D \\ \mu_{1D} \\ \mu_{2D} \\ \mu_{3D} \end{bmatrix}, \tag{B.18}$$

where $\boldsymbol{x_P} = \eta_P + \mu_{1P}\hat{\imath} + \mu_{2P}\hat{\jmath} + \mu_{3P}\hat{k}$ is the real part of the dual quaternion, and $\boldsymbol{x_D} = \eta_D + \mu_{1D}\hat{\imath} + \mu_{2D}\hat{\jmath} + \mu_{3D}\hat{k}$ is the dual part.

The vec_4 operator maps a single quaternion (either the real or dual part of a dual quaternion)

to a 4-dimensional vector. For a quaternion $\boldsymbol{x} = \eta + \mu_1\hat{\imath} + \mu_2\hat{\jmath} + \mu_3\hat{k}$, the vec_4 map is defined as

$$\text{vec}_4(\boldsymbol{x}) \triangleq \begin{bmatrix} \eta \\ \mu_1 \\ \mu_2 \\ \mu_3 \end{bmatrix}. \tag{B.19}$$

Finally, the vec_6 operator is used to map the translational and rotational components of a pure dual quaternion into a 6-dimensional vector. Given a pure dual quaternion $\underline{\boldsymbol{x}} = \boldsymbol{x_P} + \varepsilon\boldsymbol{x_D}$, the vec_6 map is defined as:

$$\text{vec}_6(\underline{\boldsymbol{x}}) \triangleq \begin{bmatrix} \mu_{1P} \\ \mu_{2P} \\ \mu_{3P} \\ \mu_{1D} \\ \mu_{2D} \\ \mu_{3D} \end{bmatrix}, \tag{B.20}$$

where $\boldsymbol{x_P} = \mu_{1P}\hat{\imath} + \mu_{2P}\hat{\jmath} + \mu_{3P}\hat{k}$ is the pure quaternion representing primary part, and $\boldsymbol{x_D} = \mu_{1D}\hat{\imath} + \mu_{2D}\hat{\jmath} + \mu_{3D}\hat{k}$ represents the dual part.

B.3.5 Logarithm and Exponential of Dual Quaternions

In dual quaternion algebra, it is also possible to define the logarithm and exponential functions which are important for transforming between different representations of rigid body motions. These operations are particularly useful in the context of interpolation, integration, and the development of control laws in robotics.

The logarithm of a unit dual quaternion $\underline{\boldsymbol{x}} \in \underline{\boldsymbol{S}}$ is defined as the operation that extracts the rotation and translation parameters embedded within the dual quaternion, resulting in a pure dual quaternion. This operation maps elements from the Lie group of dual quaternions, $\text{Spin}(3) \ltimes \mathbb{R}^3$, to the Lie algebra, which is the tangent space at the identity element of the group. The logarithm of a unit dual quaternion is expressed as

$$\log(\underline{\boldsymbol{x}}) = \frac{\phi\boldsymbol{n}}{2} + \varepsilon\frac{\boldsymbol{p}}{2}, \tag{B.21}$$

where $\boldsymbol{n}$ is the unit quaternion representing the axis of rotation, ϕ is the rotation angle, and $\boldsymbol{p}$ represents the translation component of the rigid body motion. The logarithm effectively decouples the rotational and translational components encoded in the dual quaternion, making these parameters explicit and accessible for further manipulation.

The exponential function of a dual quaternion is the inverse operation to the logarithm. It reconstructs the unit dual quaternion from its rotational and translational components, mapping

elements from the Lie algebra back to the Lie group. Given a pure dual quaternion $\underline{\boldsymbol{x}}_0 = \boldsymbol{x_P} + \varepsilon \boldsymbol{x_D}$, the exponential function is used to reconstruct the dual quaternion as

$$\exp(\underline{\boldsymbol{x}}_0) = \exp(\boldsymbol{x_P}) + \varepsilon \left(\exp(\boldsymbol{x_P}) \cdot \boldsymbol{x_D}\right), \tag{B.22}$$

where the exponential of the primary part is given by

$$\exp(\boldsymbol{x_P}) = \cos\left(\|\boldsymbol{x_P}\|\right) + \frac{\boldsymbol{x_P}}{\|\boldsymbol{x_P}\|} \sin\left(\|\boldsymbol{x_P}\|\right). \tag{B.23}$$

This exponential map allows for reconstructing rigid body transformations ensuring that the transformation remains within the unit dual quaternion set $\underline{\boldsymbol{S}}$.