INFERRING GEOMETRIC CONSTRAINTS IN HUMAN DEMONSTRATIONS FOR ROBOT PROGRAMMING

by

Guru Subramani

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Mechanical Engineering)

at the

UNIVERSITY OF WISCONSIN–MADISON

2019

Date of final oral examination: 06/05/2019

The dissertation is approved by the following members of the Final Oral Committee:
Michael Zinn, Associate Professor, Mechanical Engineering
Michael Gleicher, Professor, Computer Sciences
Dan Negrut, Professor, Mechanical Engineering
Suresh Krishnan, Professor, Mechanical Engineering
Bilge Mutlu, Associate Professor, Computer Sciences

© Copyright by Guru Subramani 2019
All Rights Reserved

Acknowledgments

My mentors, my family, my friends, and my peers have influenced who I am, who I want to be, and how I value myself. They have expressed their confidence in me, have trusted my judgment and most importantly have forgiven my mistakes with compassion.

My parents have sacrificed so much for me. They have always placed us, my brother and I, first before themselves or anything else. This is something I can merely acknowledge but cannot return in equal value. My brother, Gowthaman, has been my first and dearest friend. A friend, no matter what, who would always be there by my side.

Professor Michael Zinn, my advisor, has always been supportive and encouraging. If it were not for him, I probably would not have done my Ph.D. I could not have expected more from an advisor. Professor Michael Gleicher has always put time and effort into the professional development of his students. It has been a privilege working with him. I also thank Professor Bilge Mutlu for his advice.

My friends have been so understanding and patient with me. It is difficult for me to see myself here without their encouragement. I will miss my lab mates at the Reach lab and the UW Graphics lab. Their presence made the whole experience all the more fun and enjoyable.

I would like to acknowledge a few of my fellow collaborators for their contributions related to this dissertation. Mike Hagenow a fellow member of the Reach lab, was instrumental in the development and evaluation of some of the constraint inference algorithms. We made a good team and I will always value his friendship. Pragathi Praveena, a fellow member of the Graphics lab, helped evaluate the design and usability of the tongs.

I thank my collaborators and committee for their support and guidance. I also thank the Wisconsin Alumni Research Foundation (WARF), the National Science Foundation (NSF) and others for financially supporting my research which I am so passionate for.

The journey was well worth it. Thank you for helping me make this happen.

Contents

Contents ii

Abstract v

1 Introduction 1
- *1.1 Thesis statement* 7
- *1.2 Constraint inference approach* 9
- *1.3 Evaluation of constraint inference approach* 10

2 Methods to record human demonstrations 12
- *2.1 Instrumented Tongs* 13
- *2.2 Constraint Saber* 15

3 Mathematical modeling of constraints 17
- *3.1 General scleronomic geometric constraint model* 18
- *3.2 Relationship between constraint models and reaction forces and moments* 19
- *3.3 Resolving reaction forces and moments from wrench measurements* 20
- *3.4 Mathematical models of geometric constraints* 21

4 Fitting pose and wrench measurements to constraint models 31
- *4.1 Kinematic error objective* 32
- *4.2 Wrench error objective* 32
- *4.3 Maximum likelihood estimate* 33
- *4.4 Robust regression* 34

5 Model selection 36
- *5.1 Error metrics* 36
- *5.2 Model Generality and Penalization* 38
- *5.3 Null Hypothesis* 39

6 Segmentation 41

7 Experimental Evaluation of Constraint Inference 43
- *7.1 Experimental setup* 43
- *7.2 Influence of wrench data when fitting constraint models* 44
- *7.3 Evaluation of Model Selection* 50
- *7.4 Situated Demonstration* 53

8 Robust robot replay using identified constraints 56
- *8.1 Related Work* 59
- *8.2 Salient features of approach* 62
- *8.3 Mathematical framework* 63
- *8.4 Robust constrained motion replay system* 66
- *8.5 Experimental Setup* 69
- *8.6 Experiments* 71
- *8.7 Conclusion* 77

9 Potential avenues for future work 78

10 Conclusion 81

A Recognizing tactile actions 82
- *A.1 Background* 83
- *A.2 Key Idea* 84
- *A.3 Methods* 84
- *A.4 Experimental Setup for Evaluating Methodology* 91
- *A.5 Results* 97
- *A.6 Conclusions and Discussion* 106

B Recognizing constraints from position and force signals 108
- *B.1 Point model of a constrained object* 109
- *B.2 Detailed Overview of Approach* 114
- *B.3 Experimental Evaluation* 124

Bibliography129

Abstract

Many physical actions, such as pulling out a drawer or wiping a table may be modeled as geometric constraints. Geometric constraints restrict the motion of objects and generate reaction forces and moments when objects interact with them. Prior knowledge of the constraint is vital for robots to manipulate these constrained objects robustly. This dissertation focuses on methods to infer geometric constraints in human demonstrations by using both pose and wrench measurements. Specifically, it describes how wrench measurements can improve overall constraint inference.

The developed constraint inference approach addresses three problems: model selection (what the constraint type is), model regression (what the model parameters are) and demonstration segmentation (where they occur). It fits arbitrary scleronomic constraint models (constraints that impose restrictions on position and orientation independent of time) to kinematic and wrench measurements. It performs model selection by comparing models using position, orientation, force, and moment error metrics.

The constraint inference approach was evaluated using data from human participants performing tasks consisting of constraints. Experimental results conclude that wrench measurements improve both regression and model selection when compared to competing kinematic approaches. Wrench measurements help in situations where kinematic information is insufficient to determine the constraint, allowing constraint inference in short human demonstrations, typical in real tasks.

The inferred constraint geometry translate to constraint frames that are useful for robot control. Constraint frames map to hybrid position-force control parameters that enable robust robot interaction with the constraint. A proof of concept teaching by demonstration system called Robust Replay validates the utility of the constraint inference approach. Robust Replay converts a single human demonstration consisting of constraints into a robot program which is robust to deviations in the known geometry of the physical constraint. The approach decomposes a task into subtasks, identifies geometric constraints within those subtasks, and maps each subtask to an appropriate hybrid control description.

1 Introduction

Physical constraints are commonplace. Opening a door, erasing a whiteboard, drawing on paper, manipulating levers on a lathe, turning a steering wheel in a car, are just a few instances of the many different physical interactions that can be represented as physical constraints. Physical constraints impose restrictions on the motion of objects in the environment. They permit application of forces against the physical constraint while preventing motion of the constrained object. Representing physical interactions as physical constraints provide information about the permissible motion of the object in the environment and the permissible directions forces and moments may be applied to them.

Many of these physical constraints can be abstracted with geometric representations. For example, the motion of an iron against an ironing table can be interpreted as planar motion represented by a plane. The position of the iron is restricted to the plane of the table and the orientation of the iron is restricted to rotations in the plane of the table. Multiple physical constraints can map to the same geometric representation. For example, hinged objects such as a steering wheel or a cabinet door can all be abstracted to rotation about a fixed axis.

Figure 1.1: Common physical constraints we interact with every day. A physical constraint restricts motion in certain directions. A whiteboard eraser is restricted to move in the plane of the board. A door handle is restricted to rotating about an axis. A drawer is restricted to translate in one direction.

Articulated robot arms are frequently required to interact with these physical constraints. When a robot arm interacts with a constrained object, the end effector may also be restricted by the constraint. Figure 1.2 shows a robot interacting with a glass panel. The end effector of the robot is restricted to the planar surface of the glass panel. The robot must push against the panel with sufficient force to successfully wipe the panel without shattering it. It must also move against the panel safely without violating the constraint where violation would result in either the robot penetrating the glass or losing contact.

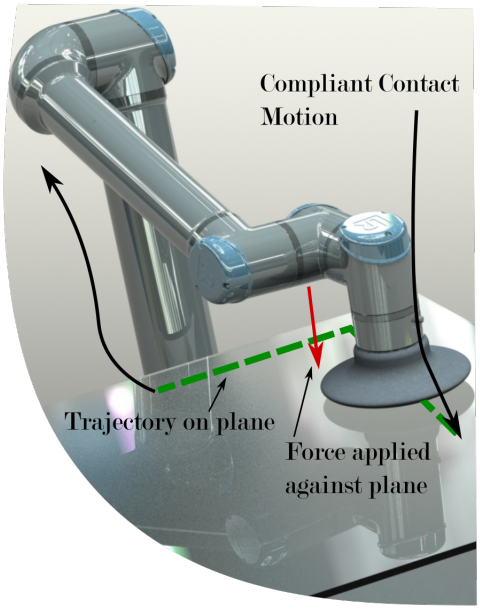

Figure 1.2: Glass panel wiping task.

Information pertaining to the physical constraint enables robust robot interaction with the constraint. When a robot interacts with a constraint, it creates a closed kinematic chain. If the constraint is known, the robot can restrict its joints to permissible configurations, so it does not violate this closed kinematic chain. It can also apply safe levels of force against the constraint to achieve the appropriate task-specific interaction. With regards to robot control, it allows the selection of an appropriate control approach necessary to interact with the constraint (33; 57). This allows position control along unconstrained directions and applying force control along constrained directions. Knowledge of the constraint also enables planning robot motion along the constraint (7; 26; 66).

Information about the constraint such as constraint geometry is typically specified manually. In existing systems, a teach pendant interface is used to specify the appropriate controller necessary to interact with the constraint. Such specification requires training, and novice users may not have the skills required to program a

robot using hybrid control. It may also require specification of geometric primitives that may not be easily determined. For example, consider the task of wiping a table. If the table is oblique, it may be difficult to precisely determine the orientation of the plane necessary to program the robot to push against the plane. In some tasks, the required trajectory against the constraint is important and in such circumstances, this method is cumbersome to use because it does not facilitate the natural transfer of motion (8; 59; 73).

This suggests that there is a need to create better input method that facilitates the programming of constraint interactions. One approach is to use a *human demonstration* to record a constraint interaction (3). Demonstrations have been an effective way for humans to communicate tasks. A person can observe the performed task, deduce the important aspects of the task, and learn the operations that must be performed to complete the task. Along the same lines, a human demonstration can also be used to describe a task containing a constraint to a robot.

Human demonstrations can be recorded using different types of input. The input method and type of physical measurements have a large influence on the ability to either recognize or infer an action (35) and in turn, constraints in a demonstration. Actions may be recognized from images or video to create robot programs using task-level planners (19; 74; 55). Geometric constraints can also be inferred from raw video. In (67), the motion of the flat surface of a door is inferred and converted to a trajectory which is used to determine revolute motion.

Interaction with physical constraints naturally generate forces and moments and are a composition of both kinematic trajectories and applied forces and moments. As humans, we regularly use tactile feedback and proprioception to recognize actions, so it lends to the idea of using forces and moments for action recognition too. Recording kinematic trajectories and applied forces and moments when interacting with constraints may provide sufficient information to infer constraints and program them for robot motion. Figure 1.3 shows a human demonstration and associated kinematic, force, and moment trajectories recorded in the demonstration.

Typically, a constraint interaction is part of a task. Consider the task of microwaving a dish. The task may consist of grasping the dish and moving it into

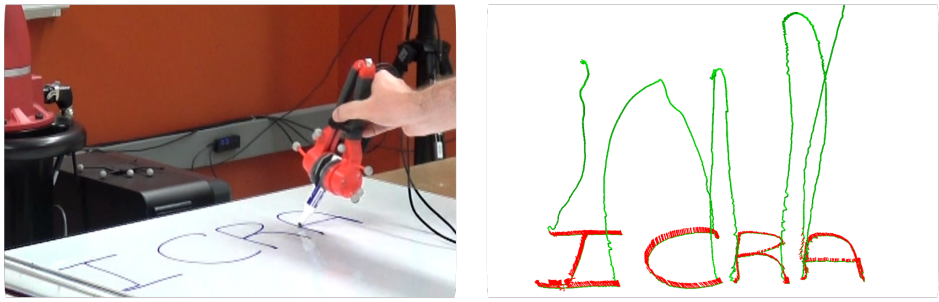

Figure 1.3: Human demonstration and corresponding kinematic, force and moment trajectories.

the microwave as well as the constraint interaction of closing the microwave door. The constraint interaction of *closing the microwave door* should be inferred from the demonstration. This involves segmenting the demonstration to determine *when* the constraint interaction occurs in the demonstration as well as determining the constraint type and its geometric parameters.

The representation of the inferred constraint is important for robot control. The constraint model must communicate sufficient information about the constraint so a robot can interact with the constraint appropriately. In the microwave example, the constraint is an *axial rotation constraint* (i.e., motion can only be about an axis of rotation). The inferred constraint model must inform the robot that it can only rotate the microwave door about the axis but cannot move it in other directions restricting its motion to 1 degree of freedom.

Inferring the constraint from a single demonstration of the task avoids the demonstrator from repeating the demonstration, thereby saving the demonstrator time and effort. A long task may consist of a short but important constraint interaction, and the demonstrator may not want to provide multiple demonstrations. Constraint interactions are frequently of short duration when they are performed in the context of the task. Therefore it is necessary that constraint inference methods are capable of inferring the constraint with as little information as possible. Fig. 1.4

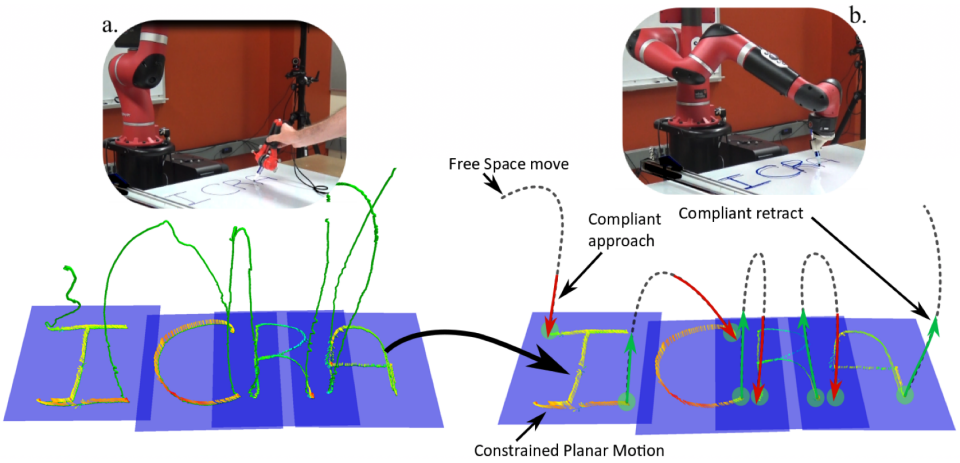

Figure 1.4: Human demonstration and corresponding kinematic, force and moment trajectories.

shows the sawyer robot replaying tasks using inferred geometric constraints.

This discussion leads to the requirements for an effective constraint inference system:

- **Automatic segmentation:** Determine when the constraint occurs in the demonstration.
- **Demonstration efficiency:** Ability to infer the constraint in short demonstrations.
- **Constraint modeling:** Incorporating a mathematical modelling approach that enables the description of multiple different types of constraints.
- **Model selection:** Determining the correct constraint model from many different possible models in the demonstration.

Prior methods do not address many of these requirements.

Automatic segmentation: Novice users may not know what a constraint is and thus may be incapable of determining where constraints occur in demonstrations.

Prior approaches either query the user to provide segmentation (54) or require the entire demonstration to contain the constraint interaction (6; 4; 5).

Demonstration efficiency: Constraints may need to be inferred from short demonstrations. In (4), 20 seconds of demonstration time is required to infer the constraint. While the inferred constraint model is general and robot motions can be transferred to other constraint models, many tasks may not incorporate 20 seconds constrained motion in the demonstration.

Constraint modeling: The constraint inference method needs to incorporate models that sufficiently describe the constraint. Simple constraint models which involve the inference of simple geometric primitives such as planes, lines, and arcs may be insufficient to program robot interaction. They do not describe the orientation constraints imposed on the robot when it interacts with these constraints. In (50; 54), only simple constraint models are used. While more complex models are incorporated in (15; 45), (constraint interaction is modeled as contacts between known polyhedral, cylindrical, and spherical tools), prior information about the kind of contacts, the tool, and environment geometry is necessary. In many applications tool geometry may not be known. In (24) constraints are represented as lower dimensional task manifolds learned through manifold learning. While this can encode complicated constraint regions, it does not allow a compact, semantic, parameterized representation of a constraint (for example, a hinge constraint is defined by its axis of rotation).

Model Selection: Model selection may manifest itself in either identifying the number of parameters required to fit a general model or selecting an appropriate model given a set of candidate models. In (67), both aspects of model selection are present. A gaussian mixture model with varying model parameters represents the general model. Only two specific models are employed; *prismatic* (1 DOF linear translation) and *revolute* (1 DOF rotational referred to in this dissertation as an *axial rotation constraint*). Many constraint interactions cannot be represented by just *prismatic* and *revolute* constraints. For example, a commonly occurring constraint is a *planar constraint* which is exemplified by an eraser moving against a plane. Another constraint interaction is a *point-on-plane constraint* which is exemplified by

the motion of a pen against paper where the pen tip is constrained to the plane of the paper. The model selection problem increases in difficulty as more models are introduced.

Most prior work only incorporate kinematic information for constraint inference (4; 67; 54; 24; 50). Incorporating wrench measurements (measurements of forces and moments) recorded when performing the demonstration may improve the ability to perform constraint inference. Wrench measurements generally include both reaction forces and moments generated by the constraint as well as other wrenches such as friction or inertia. Friction will always occur when the objects in contact slide against one another.

Wrench measurements have been incorporated in earlier work (45) to infer geometric constraints but effects such as friction are not modeled, reducing the performance of these systems. Understanding the applicability of wrench measurements for constraint inference is still an unexplored topic. Wrench measurements provide reciprocal information to velocity measurements (16). This lends to the idea that wrench measurements provide additional information to kinematic measurements and improve geometric constraint inference.

1.1 Thesis statement

This discussion leads to the following thesis statement:

Force and moment measurements help in inferring geometric constraints in human demonstrations.

This dissertation provides an approach to infer geometric constraint from human demonstrations. This enables novice users to provide information pertaining to constraints in the environment to a robot. Competing approaches do not solve this problem completely. They ignore important constraint interactions or do not address the applicability to novice users. Apart from just the goal of building better constraint inference methods, the key contribution of this work is establishing that force and moment measurements improve geometric constraint inference.

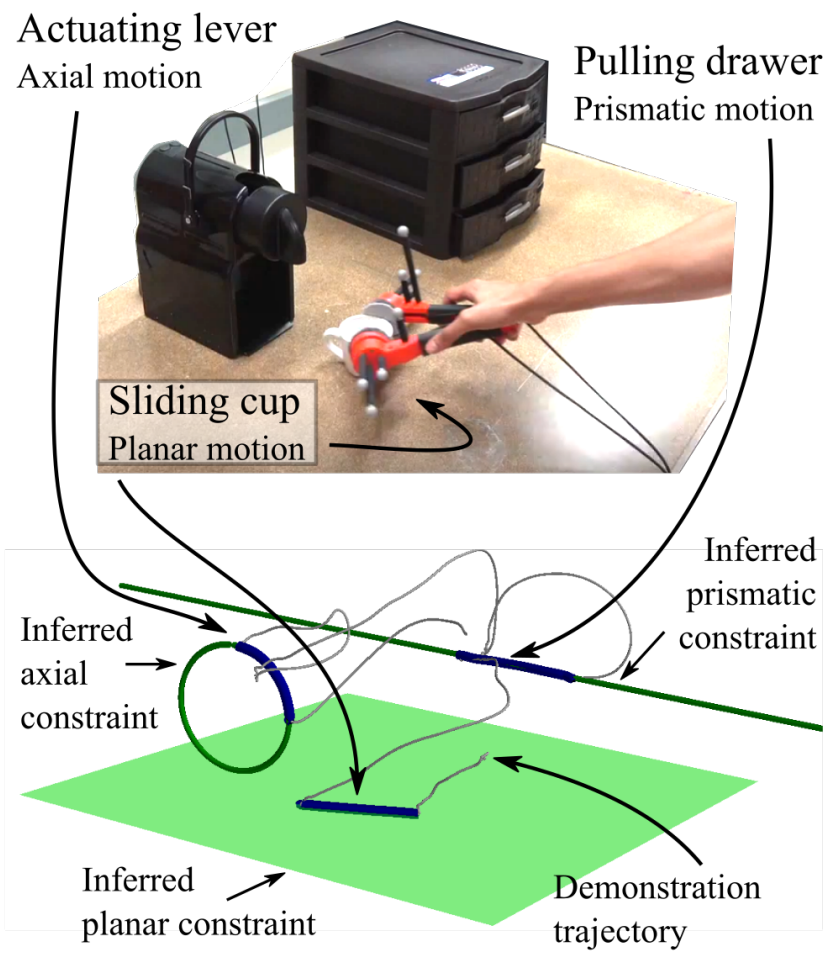

Figure 1.5: Top: Demonstration of an espresso-making task using instrumented tongs consisting of sliding an espresso cup, pulling out a drawer and actuating an espresso lever. Center: Inferred constraints in the task. Our approach infers the samples associated with each constraint (segmentation), the type of constraint (model selection) and model parameters of each constraint (model regression) in the task.

1.2 Constraint inference approach

We use instrumented tools, described in chapter 2, to record kinematic and wrench trajectories to infer constraints. Our approach is to partition the human demonstration into constrained motion and unconstrained motion, fit all candidate constraint models to each constrained motion segment, and select the appropriate constraint models from the fit models. Fig. 1.5 shows an espresso-making task, and the corresponding inferred constraints using our constraint inference approach. We organize the constraint inference as the following steps:

1. **Constraint modeling:** creating a library of candidate constraint models
2. **Demonstration segmentation:** determining when constraints occur in the demonstration
3. **Model regression:** identifying the parameters associated with a candidate constraint model
4. **Model selection:** identifying the most suitable constraint type from the candidate constraint models.

Constraint Modeling: Our constraint inference approach requires defined mathematical representations of the constraints. We use models based on human interpretable geometric primitives to represent constrained motion such as the *planar constraint* (motion of wiping a table), *point-on-plane constraint* (motion of a pen on paper), *axial rotation constraint* (the motion of a hinged door) and many more. These constraints are represented as mathematical equations that the tool or object must satisfy when under the influence of the constraint. They come under the class of constraints called scleronomic constraints. Chapter 3 describes the scleronomic constraint model and the construction of specific constraint models. This section also derives the relationship between the reaction wrenches (reaction forces and moments) imparted on constrained objects and the generalized scleronomic constraint model.

Demonstration segmentation: Applications may require determining when constraints occur in a demonstration, especially when demonstrations represent a larger task. We discuss methods to segment the constrained motion from demonstrations in Chapter 6 (after model regression and model selection to cover the more important contributions first). Our approach takes advantage of the reaction forces and moments that exist during constraint interaction.
Model regression: In Chapter 4, we show how nonlinear least squares regression can be used to fit geometric constraint models to kinematic and wrench measurements. Our method derives reaction forces and moments from measured wrenches by considering the contributing dynamics from friction. This is described in section 3.3. We show that by simultaneously fitting with position and force data, we are able to fit certain constraints using significantly less data compared to fitting with kinematic data.
Model selection: Once all models are fit to the constrained segments, each model fit must be evaluated, and the appropriate model must be selected. The regression objectives are inadequate to perform this comparison because they evaluate to different units and scales. Instead, kinematic and wrench error metrics are derived for the general scleronomic constraint model to provide a consistent approach to evaluate models. These error metrics cannot always be used directly as higher degree of freedom constraint models will over-fit. In Chapter 5, we describe how wrench measurements can be used to significantly improve model selection even in the presence of multiple models that agree kinematically. We also present a penalized error metric similar to that of Akaike, which penalizes models based on their degrees of freedom and number of parameters.

1.3 Evaluation of constraint inference approach

We empirically evaluate the system's ability to perform automatic segmentation, efficiently use available demonstration data, and discern the correct model from multiple candidate models. We focus on the use of wrench measurements in fitting and model selection against competing methods that use only kinematic

measurements. The system's overall performance is evaluated on demonstrations of an espresso-making task. Evaluations are performed using human participants data. Chapter 7 describes these evaluations.

Recorded human demonstrations are used to program robots interact with physical constraints robustly. These programs are determined by mapping inferred geometric constraints to their appropriate robot control algorithms e.g. hybrid control, and corresponding kinematic trajectories. Chapter 8 describes the *Robust Replay system* developed to replay human demonstrations consisting of physical constraints using articulated robot arms.

2 Methods to record human demonstrations

The approach used to record the human demonstration influences the applicability of the demonstration for action inference and subsequently robot programming. Ideally, the demonstrator should not be distracted by the input method. The input method must be as *transparent* as possible to the demonstrator so the demonstrator can focus on performing the task. In our work we use *natural demonstrations* using hand-held instrumented tools to describe tasks to a robot. Some of the different approaches to provide human demonstrations are discussed below:

Kinesthetic teaching involves moving a robot with one's hands and using the robot's sensors (e.g. joint encoders) to record the demonstration. This is the most widely used input method. Sometimes only poses of the robot are recorded by the demonstrator and a teach pendent is used to create a robot program around these poses. Extracting smooth trajectories is a challenge with kinesthetic teaching because it relies on the transparency of the robot and it's ability to be back-driven. Applied forces and torques may be important to the task and measuring them may require an additional force-torque sensor at the tool (in spite of inherent torque sensors in the robot joints as these would measure the demonstrator's applied forces on the robot). Applied forces and moments in a task can also be recorded by having the robot replay the kinesthetically taught trajectory in position control and recording the joint torques experienced by the robot when it executes the task as done in (39). This method works when the robot interacts with compliant and soft objects.

Teleoperation involves remote controlled guidance of a robotic manipulator. The success of a telerobotic system greatly depends on the kind of remote used as input (56). Telerobotic systems may also incorporate force feedback which may directly be measured from the robot but may require a specialized haptic device to render. Telerobotic manipulation also adds cognitive load on the demonstrator because the demonstrator is required to know the environment the robot manipulates in.

Kinesthetic teaching and *teleoperation* both record the demonstration using the sensors located on the robot such as encoders and joint torque sensors. In the most

naive implementation of task replay, the encoder joint positions can be replayed on the robot directly to recreate the demonstration (albeit, this method may not be reliable). Both these methods suffer from their ability to be *transparent* to the demonstrator.

Natural demonstrations are performed directly by the user, either with their hands or a hand held tool. Demonstrations performed directly using the demonstrator's hand are difficult to capture completely because demonstrators often use dexterous in-hand manipulations (8). Tracking only the hand ignoring the fingers may be insufficient to completely capture the actions by the demonstrator. Handheld tools enable accurate recording of the exact motion performed by the demonstrator. These tools may be designed to prevent demonstrators from performing actions that cannot be performed by the robot. Force torque sensors and fiducial markers may be rigidly attached to hand held devices to accurately (and completely) record the interaction. Many tasks may require a task specific end effector tool (such as a buffing tool) while other tasks may require a robot gripper to manipulate objects in the environment. An effective natural demonstration tool will be convenient to use, can be tracked reliably and tracked trajectories will provide a complete description of the action performed.

We introduce two instrumented tools used for natural demonstrations, the *Instrumented Tongs* and the *Constraint Saber*.

2.1 Instrumented Tongs

The instrumented tongs is similar to a robot gripper. The design is inspired from conventional kitchen tongs. (56) Although the tongs have limited manipulation capabilities, it prevents users from performing highly dexterous tasks that cannot be performed by a robot with a two finger gripper. It also provides sufficient tactile feedback when performing tasks.

Design

The instrumented tongs consist of two rigid, 3D printed arms made from PLA plastic. A hinge consisting of a steel pin with plastic bushings holds the arms together. The tongs are held open with a flat *spring steel* spring. Two inset mounting surfaces enable attachment of force-torque sensors. Multiple prongs provide locations to attach motion capture markers. A Plastic plate is attached to the tool side of the force torque sensors. To prevent two point grasps caused by the rigid plates, layers of foam padding with different density are attached. The padding consists of a low-density thick foam layer (10mm) followed by a high density thin foam layer (1mm). A final layer of high friction 3M Gripping material (3M TB400) was used. The instrumented tongs provide a reliable grasp of objects but the grasp may not be rigid because the foam pads introduce compliance when objects are grasped.

Sensors

The current version of the tongs uses an Optitrack motion capture system which employs reflective markers that are rigidly attached to the tongs and cameras to record the motion of these markers. We track both arms as independent rigid bodies. A pair of ATI Mini40 force-torque sensors, one in each arm, provide accurate measurement of the applied forces and torques when grasping objects. Objects are directly in contact with the tool side of the force torque sensors (through the padding) enabling a complete measurement of the applied forces and moments on the objects. Figure 2.1 provides the section view of the instrumented tongs. The design of the tongs is provided at: `https://github.com/uwgraphics/HRI2019Tongs`

Potential improvements

Realistic environments may have objects that can occlude motion capture cameras. Inertial tracking with IMUs would greatly improve the tracking performance in such circumstances. The tongs are not a perfect mapping to a robot gripper because robot grippers come in many shapes and forms. Since the tongs are natural to

humans, it may be worthwhile to design robot grippers to mimic the design of the tongs.

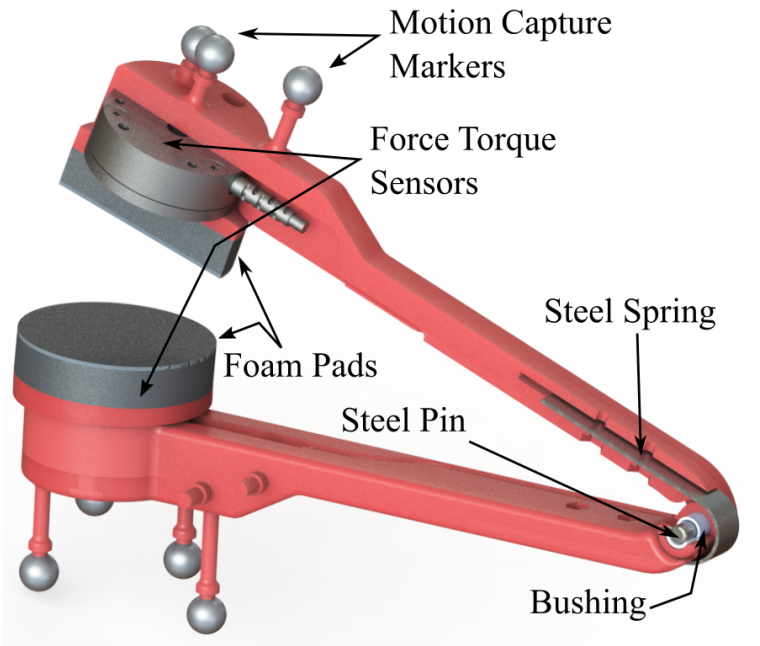

Figure 2.1: **Instrumented tongs**

2.2 Constraint Saber

The constraint saber is similar to a robot with an application specific end effector tool. The tool has no moving parts and can be considered a single rigid body. A single force torque sensor measures applied forces when interacting with the environment. Motion capture markers are attached to track the device. A handle is attached on one side and the application specific tool on the other. The measurement frame for both the force torque sensor and the motion capture markers are aligned. The kinematic trajectories of the tool along with the measured forces and moments from the force torque sensor completely describe the rigid body interaction with the environment. Figure 2.2 describes the construction of the Constraint Saber.

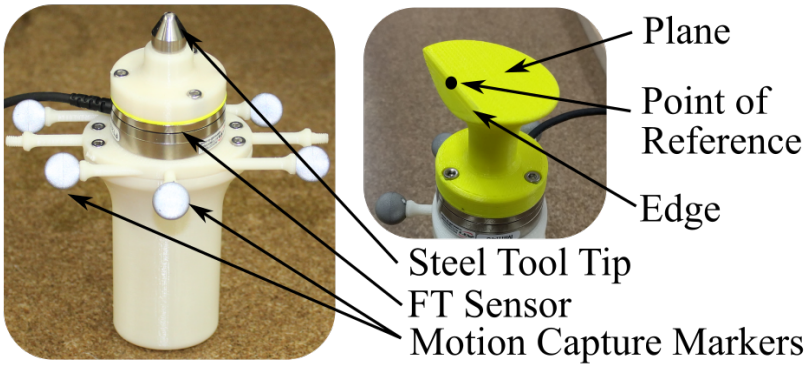

Figure 2.2: **Constraint saber, different end effectors may be attached to this hand-held demonstration tool.**

3 Mathematical modeling of constraints

When a robot end effector or other object moves under the influence of a constraint, the degrees of freedom of its rigid body motion are restricted to a subset in SE(3). These restrictions may be represented as mathematical relationships between defined geometry on the body (e.g., a point on the body) and defined global geometry (e.g., a plane in the environment). For example, consider a pen moving against paper. The motion of the pen tip is constrained to the planar surface of the paper. In this example, the rigid body is the pen, the location of the pen tip is the defined geometry on the rigid body and the planar surface represents the global geometry. Geometric constraint models consist of equations that define the possible configurations of the rigid body.

Geometric constraints impose restrictions on the configurations on the rigid body. Applying forces and moments on a constrained rigid body creates reaction forces and moments between the rigid body and the constraint. The principle of virtual work expresses that reaction forces and moments caused by the constraint do not produce work. This property can be used to determine the permissible reaction forces and reaction moments of the constraint. It is also necessary to consider additional forces that may act on the rigid body such as friction, inertia, and gravity.

3.1 General scleronomic geometric constraint model

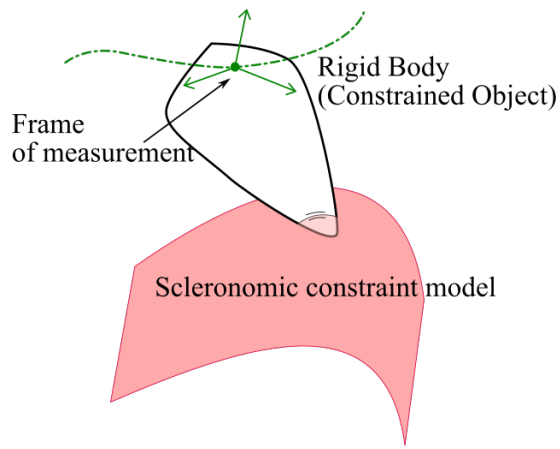

Figure 3.1: **Scleronomic constraint**

Scleronomic constraints are geometric constraint models that only impose restrictions on the position and orientation (i.e. generalized coordinates of the rigid body) and do not vary with time(21).

Consider a 6-degree of freedom rigid body of negligible inertial properties located in space through translational coordinates $r \in \mathbb{R}^3$ and orientation coordinates represented by a unit quaternion $q \in SU(2)$. [1] The measurement reference frame is the fixed frame on this rigid body. The rigid body is constrained by k constraint equations $\Phi : SE(3) \Rightarrow \mathbb{R}^k$:

$$\Phi(r, q) = 0 \tag{3.1}$$

The scleronomic constraint model represented by equation (3.1) may be instantiated with a specific constraint model (e.g. with a *planar constraint*) with associated model parameters α which configure the constraint (e.g. the position and orientation of the plane). Thus, Φ becomes $\Phi(r, q, \alpha)$.

[1] Scalars are italicized (s). Vectors are normal text (v) and are column vectors in the context of a matrix. Matrices are bold ($\mathbf{M}$).

In this work, a few of the geometric constraint models we consider are shown in Fig. 3.2. These constraints were chosen as examples of commonly occurring geometric primitives. Our approach can be extended to other models satisfying the scleronomic constraint formulation.

3.2 Relationship between constraint models and reaction forces and moments

The formulation in this section is standard in multi-body dynamics literature (23) and is provided here as a review. We determine the permissible linear and angular velocities using virtual displacements and then use the principle of virtual work to determine the permissible reaction forces and moments.

The first partial derivatives of constraint equations (3.1) are given by:

$$\delta\Phi = \Phi_r \delta r + \Phi_q \delta q = 0 \tag{3.2}$$

Where $\Phi_r \delta r$ and $\Phi_q \delta q$ are the partial derivatives of (3.1) and $(\delta r, \delta q)$ are the virtual displacements. Equation (3.2) may also be written using virtual rotation variable $\delta\pi$. The virtual rotation variable is related to the angular velocity of a rigid body. This relationship is similar to how the virtual displacement δp is related to linear velocity.

$$\delta\Phi = \Phi_r \delta r + \Phi_\pi \delta\pi = 0 \tag{3.3}$$

Equation (3.3) is similar to the Pfaffian constraint form widely used in robotics literature (14). The permissible reaction forces $f_r \in \mathbb{R}^3$ and reaction moments $n_r \in \mathbb{R}^3$ that the constraint applies to the constrained body must satisfy the virtual work equation (37) because constraint reaction forces and moments do not produce work. The principle of virtual work requires:

$$\delta r^T f_r + \delta\pi^T n_r = 0 \tag{3.4}$$

Combining equations (3.3) and (3.4) using the Lagrange multiplier theorem:

$$\Phi_r{}^T\lambda + f_r = 0 \tag{3.5}$$

$$\Phi_\pi^T\lambda + n_r = 0 \tag{3.6}$$

where $\lambda \in \mathbb{R}^k$ are the Lagrange multipliers. Equations (3.5) and (3.6) provide a relationship between the constraint equations Φ and the permissible reaction forces and moments. $\Phi_\pi \delta\pi$ is computed from $\Phi_q \delta q$ as follows:

$$\Phi_\pi = \Phi_q \frac{1}{2}\mathbf{G}^T\mathbf{A}^T \tag{3.7}$$

with the following defined variables:
Quaternion $q = \begin{bmatrix} e_0 & e^T \end{bmatrix}^T$ where $e_0 = q_0$ and $e = [q_1 \; q_2 \; q_3]^T$.

$$\mathbf{A} =: \left(e_0^2 - e^Te\right) I + 2ee^T + 2e_0\tilde{e} \tag{3.8}$$

$$\mathbf{G} =: \begin{bmatrix} -e & -\tilde{e} + e_0\mathbf{I} \end{bmatrix} \tag{3.9}$$

where

$$\tilde{e} = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix} \tag{3.10}$$

and $\mathbf{I}$ is the identity matrix.

3.3 Resolving reaction forces and moments from wrench measurements

Equations (3.5) and (3.6) relate the reaction forces and moments with the constraint equations described by equation (3.1). Interactions with the physical environment involve forces such as kinetic friction, inertia, gravity, and also task specific forces such as springs and dampers. We assume quasi-static motion and thus only model

kinetic friction in our analysis of constrained motion. As many physical constraint interactions are of short duration, there may be insufficient information in the demonstration to confidently estimate dynamic effects such as inertia. In cases where demonstrations are highly dynamic involving objects with large inertia or involving interaction with task specific devices such as springs, this approach may not be directly applicable and such dynamic effects must be modeled and incorporated appropriately.

Quasi-static motion against constraints generates frictional forces due to the sliding contact between surfaces. Forces attributed to kinetic friction oppose the motion of the constrained object. For example, when writing on paper (*point-on-plane constraint*), friction force opposes the motion of the pen tip. Friction forces may manifest themselves differently for each constraint and there is no general way to describe them.

However, friction forces and moments can be modeled for each constraint with certain assumptions. We derive a relationship between the reaction forces and moments with the measured forces and moments as functions of the parameters of the constraint:

$$f_r = F_r(r, q, v, \omega, f, n, \alpha) \tag{3.11}$$

$$n_r = N_r(r, q, v, \omega, f, n, \alpha) \tag{3.12}$$

where the measured quantities are forces (f), moments (n), linear velocity (v) and angular velocity (ω) and the parameters of the constraint are α. The specific formulation of equations (3.11) and (3.12) for each constraint model is provided in Appendix 3.4. The resolving of reaction forces and moments for the *point-on-plane* constraint is provided here.

3.4 Mathematical models of geometric constraints

Consider a 6-degree of freedom rigid body of negligible inertial properties located in space through 3 translational coordinates $r \in \mathbb{R}^3$ and rotation coordinates repre-

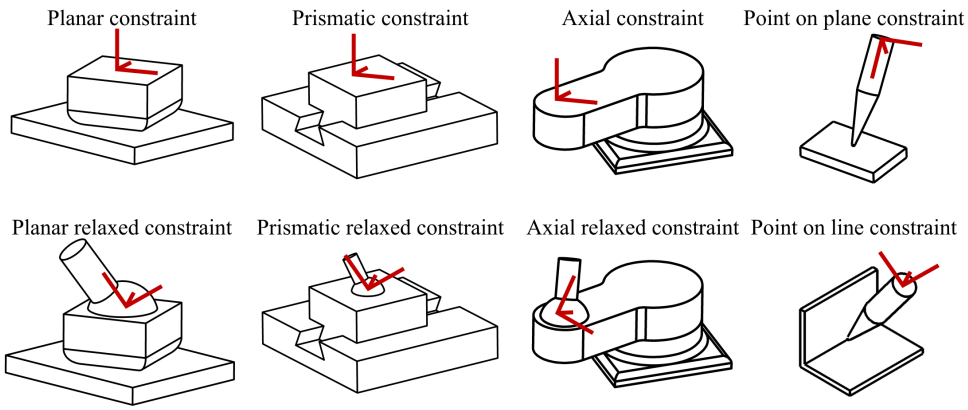

Figure 3.2: Geometric constraints used in this paper. The three constraint models *planar constraint*, *prismatic constraint* and *axial constraint* enforce constraints on the orientation of the constrained object. The *relaxed* counterparts are similar constraint models except with removed orientation constraints. The *point-on-plane* and *point-on-line* constraint models contain representations of the location of contact with respect to the measured position on the object. The measurement frame is shown in red.

sented either with a unit quaternion $q \in SU(2)$ or an orthogonal rotational matrix $\mathbf{A}(q) \in SO(3)$. Together (r,q) form the body coordinates p.

Each constraint type has a set of parameters that parameterize the geometry of the constraint. Let the parameters of the constraint be α, then equation Φ becomes $\Phi(p, \alpha)$.

Point-on-plane constraint

The *point-on-plane constraint* describes a point on the rigid body constrained to a plane in the environment such as a pencil tip (point) moving across paper (plane). An illustrated representation of this constraint is shown in Fig. 3.3.

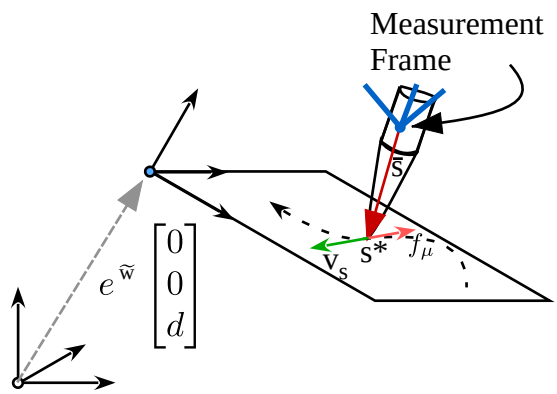

Figure 3.3: *Point-on-plane constraint* model. The point s* is constrained to the planar surface. $\bar{s}$ is a fixed vector in the measurement frame between the measurement frame and s*. The global x-y plane is transformed using the exponential map, $e^{\tilde{w}}$, to create the planar surface.

A plane may be specified by applying a general displacement (i.e. translation and rotation) transformation of the x-y plane which involves:

1. translating the x-y coordinate plane along the z-axis
2. rotating the translated plane about the origin.

We represent the rotation transformation using two exponential coordinates $w = [w_x, w_y, 0] \in \mathbb{R}^2$ corresponding to $e^{\tilde{w}} \in SO(3)$, the exponential map, which is equivalent to a rotation matrix with an axis of rotation in the x-y plane. Rodrigues' rotation formula (48) is used to compute $e^{\tilde{w}}$. $\tilde{w}$ is the skew symmetric matrix defined in equation (3.10) in the appendix. The third term of w is zero because rotations about the z - axis (perpendicular to the plane) do not alter the plane's geometry. The translation is represented by $d \in \mathbb{R}$. The normal vector on this plane is represented by $e^{\tilde{w}}[0\ \ 0\ \ 1]^T$ and the shifted origin of the x-y plane is represented by $e^{\tilde{w}}[0\ \ 0\ \ d]^T$.

A point P on the plane satisfies the following equation:

$$\left(e^{\tilde{w}}[0\ \ 0\ \ d]^T - P\right)^T e^{\tilde{w}}[0\ \ 0\ \ 1]^T = 0 \tag{3.13}$$

which specifies that the dot product between a vector within the plane and the plane normal is zero.

Consider a point $s^* \in \mathbb{R}^3$ (in the global reference frame) defined as a point fixed to the rigid body and constrained to the plane. s is a vector directed from the origin of the measurement reference frame to the point s* (vector defined in the global reference frame). The local reference frame counterpart of s is $\bar{s}$ (whose values do not change with the motion of the measurement frame):

$$s^* \equiv r + s \equiv r + \mathbf{A}(q)\bar{s} \tag{3.14}$$

where $\mathbf{A}(q)$ is the orthogonal rotation matrix equivalent to the rotation represented by q.

The constraint is satisfied when s* is constrained to the plane, i.e. equation $\Phi : SE(3) \Rightarrow \mathbb{R}^1$ is:

$$\Phi_1 \equiv \left(e^{\tilde{w}}[0\ \ 0\ \ d]^T - r - \mathbf{A}(q)\bar{s}\right)^T e^{\tilde{w}}[0\ \ 0\ \ 1]^T = 0 \tag{3.15}$$

The parameters of this constraint are $\alpha = (\bar{s}, d, w_x, w_y)$, six variables. The constraint equation removes one degree of freedom leaving 5 degrees of freedom.

Friction Model: The goal is to resolve the reaction wrenches from the measured wrenches (i.e. determine expressions for $F_r(r, q, v, \omega, f, n, \alpha)$ and $N_r(r, q, v, \omega, f, n, \alpha)$). We assume that inertia of the constrained object is negligible and the only wrenches acting on the object are friction and reaction wrenches.

The force and moment balance equation for the rigid body at the measurement frame are as follows:

$$f = f_\mu + F_r \tag{3.16}$$

$$n = n_\mu + N_r \tag{3.17}$$

where f is the measured force, f_μ is the friction force, F_r is the reaction force, n is the measured moment, n_μ is the friction moment, and N_r is the reaction moment

acting at the measurement frame.

Point s* on the rigid body is in contact with the plane. s* moves with velocity v_s. The friction force acts at point s* on the rigid body and is directed along v_s ($\widehat{v_s}$ is a unit vector in this direction). Any force in the direction of v_s is assumed to be friction. f_μ is the friction force located at point s* directed along v_s:

$$f_\mu = f \cdot v_s \widehat{v_s} \tag{3.18}$$

where v_s is given by:

$$v_s = v + \omega \times s \tag{3.19}$$

and s is the vector from the measurement frame to point s* in the global frame.

We assume that at s*, no moment is transferred (including friction moment) between the rigid body and the plane. The moment induced by the friction forces at s* is given by:

$$n_\mu = s \times f_\mu \tag{3.20}$$

Combining equations (3.16) through (3.20):

$$F_r = f - f \cdot v_s \widehat{v_s} \tag{3.21}$$

$$N_r = n - s \times f \cdot v_s \widehat{v_s} \tag{3.22}$$

Planar constraint

The planar constraint is exemplified by an eraser moving against a whiteboard. The rigid body can only rotate about a vector perpendicular to the plane, and all points within the rigid body translate parallel to this plane. The formulation of the plane is similar to that in section 3.4. The origin of the local coordinate frame on the body is coincident with the plane.

$$\Phi_1 \equiv \left(e^{\tilde{w}}[0 \;\; 0 \;\; d]^T - r\right)^T e^{\tilde{w}}[0 \;\; 0 \;\; 1]^T = 0 \tag{3.23}$$

A unit vector $\bar{t}$ is defined to force the rigid body perpendicular to the plane:

$$\Phi_2 \equiv (\mathbf{A}\bar{t}\,)^T e^{\tilde{w}}[1\ \ 0\ \ 0]^T = 0 \tag{3.24}$$

$$\Phi_3 \equiv (\mathbf{A}\bar{t}\,)^T e^{\tilde{w}}[0\ \ 1\ \ 0]^T = 0 \tag{3.25}$$

$$\Phi_{\mathbf{p}1} \equiv \bar{t}^T\bar{t} - 1 = 0 \tag{3.26}$$

Unity of $\bar{t}$ is enforced by equation (3.26). The parameters of this constraint are $\alpha = (\bar{t}, d, w_x, w_y)$, a total of 6 variables. The rigid body has 3 degrees of freedom.
Friction Model: Friction force is in the direction of linear velocity and friction moment is in the direction of angular velocity.

$$f_\mu = f \cdot v\widehat{v} \tag{3.27}$$

$$n_\mu = n \cdot \omega\widehat{\omega} \tag{3.28}$$

Planar relaxed constraint

The *planar relaxed constraint* is represented by equation (3.23). The friction model is the same as the *planar constraint*.

Prismatic constraint

The prismatic constraint represents translational motion in one direction. It is similar to pulling out a drawer. All points on the rigid body translate identically. This constraint requires a representation of the axis. Similar to the plane, the axis of translation can be generated by applying a general displacement (i.e. translation and rotation) transformation of the z coordinate axis which is equivalent to:

1. translating the z axis in the x-y plane
2. rotating the translated axis about the origin.

This is represented by two exponential coordinates $\mathbf{w} = [w_x, w_y, 0] \in \mathbb{R}^2$ and two translational coordinates $d_x, d_y \in \mathbb{R}$. The third term in $\mathbf{w}$ is zero because

rotations about the z - axis produce a line that could be produced by an alternative translation motion. This defines the axis in the global reference frame. The tangent to this axis is $e^{\tilde{w}}[0\ \ 0\ \ 1]^T$ and the translated origin is $e^{\tilde{w}}[d_x\ d_y\ 0]^T$. $e^{\tilde{w}}[1\ \ 0\ \ 0]^T$ and $e^{\tilde{w}}[0\ \ 1\ \ 0]^T$ represent vectors perpendicular to this axis. The origin of the local coordinate frame of the body is coincident with this line/axis:

$$\Phi_1 \equiv \left(e^{\tilde{w}}[d_x\ d_y\ 0]^T - r\right)^T e^{\tilde{w}}[1\ \ 0\ \ 0]^T = 0 \tag{3.29}$$

$$\Phi_2 \equiv \left(e^{\tilde{w}}[d_x\ d_y\ 0]^T - r\right)^T e^{\tilde{w}}[0\ \ 1\ \ 0]^T = 0 \tag{3.30}$$

The orientation of the rigid body is fixed using the following equations:

$$\Phi_3 \equiv (\mathbf{A}\bar{s})^T e^{\tilde{w}}[1\ \ 0\ \ 0]^T = 0 \tag{3.31}$$

$$\Phi_4 \equiv (\mathbf{A}\bar{s})^T e^{\tilde{w}}[0\ \ 1\ \ 0]^T = 0 \tag{3.32}$$

$$\Phi_5 \equiv (\mathbf{A}\bar{t})^T e^{\tilde{w}}[1\ \ 0\ \ 0]^T = 0 \tag{3.33}$$

$$\Phi_{\mathbf{p}1} \equiv \bar{s}^T\bar{s} - 1 = 0 \tag{3.34}$$

$$\Phi_{\mathbf{p}2} \equiv \bar{s}^T\bar{t} = 0 \tag{3.35}$$

where $\bar{s}$ and $\bar{t}$ are unit vectors fixed on the rigid body. Equations (3.31) through (3.26) and (3.34) prevent the body from rotating. Equations (3.34) and (3.26) forces unity of s and t respectively. This rigid body has one degree of freedom of motion. The constraint parameters are: $\alpha = (\bar{t}, \bar{s}, d_x, d_y, w_x, w_y)$.

Friction Model: Friction force is in the direction of linear velocity. This model does not permit angular rotation so we assume zero friction moment.

$$f_\mu = f \cdot v\hat{v} \tag{3.36}$$

$$n_\mu = 0 \tag{3.37}$$

Prismatic relaxed constraint

The *prismatic relaxed constraint* is obtained by only using equations (3.29) and (3.30). The friction model is given by equations (3.27) and (3.28).

Point-on-line constraint

The *point-on-line constraint* represents a point on a rigid body constrained to a line in space. It is similar to a pen moving against a straight edge.

Consider a point $s^* \in \mathbb{R}^3$ in the global reference frame defined as a fixed point on the rigid body. s is a vector directed from the origin of the local reference frame of the body to the rigidly fixed point s^* in the global reference frame. The local reference frame counterpart of s is $\bar{s}$:

$$s^* \equiv r + s \equiv r + \mathbf{A}(q)\bar{s} \tag{3.38}$$

where $\mathbf{A}(q)$ is the orthogonal rotation matrix equivalent to the rotation represented by q. s^* on the rigid body is coincident with a line represented by parameters (d_x, d_y, w_x, w_y):

$$\Phi_1 \equiv \left(e^{\tilde{w}}[d_x \ d_y \ 0]^T - r - \mathbf{A}(q)\bar{s}\right)^T e^{\tilde{w}}[1 \ \ 0 \ \ 0]^T = 0 \tag{3.39}$$

$$\Phi_2 \equiv \left(e^{\tilde{w}}[d_x \ d_y \ 0]^T - r - \mathbf{A}(q)\bar{s}\right)^T e^{\tilde{w}}[0 \ \ 1 \ \ 0]^T = 0 \tag{3.40}$$

Friction Model: The friction model is identical to the friction models for the *point-on-plane constraint*.

Axial rotation constraint

The axial rotation constraint is similar to a door knob or a hinged door, all points on the rigid body rotate about an axis and translations are not permitted.

Consider a point on the rigid body s* that is on the axis of rotation and in the plane perpendicular to the axis containing the coordinate frame origin. A rigid point on the axis of rotation defined as $(d_x, d_y, d_z) \in \mathbb{R}^3$ constrains the point s*. The vector $\bar{s}$ is perpendicular to this axis. The rigid body must not rotate about vector $\bar{s}$ so unit vector $\bar{t}$ is introduced to enforce this condition. The constraint equations are (3.26), (3.35) and the following:

$$\Phi_1 \equiv \ r + \mathbf{A}\bar{s} - [d_x, d_y, d_z]^T = 0 \tag{3.41}$$

$$\Phi_2 \equiv (\mathbf{A}\bar{t}\ )^T e^{\tilde{w}}[0\ \ 0\ \ 1]^T = 0 \tag{3.42}$$

$$\Phi_3 \equiv (\mathbf{A}\bar{s}\ )^T e^{\tilde{w}}[0\ \ 0\ \ 1]^T = 0 \tag{3.43}$$

Constraint parameters are $\alpha = (\bar{t}, \bar{s}, d_x, d_y, d_z, w_x, w_y)$. This constraint has 1 degree of freedom.

Friction Model: The friction wrenches and reaction wrenches are coupled in this model. The principle of virtual work enables the computation of reaction forces and reaction wrenches from the corresponding measured quantities. Let's assume all measured forces are reaction forces, then friction forces are zero:

$$f_\mu = 0 \tag{3.44}$$

The principle of virtual work requires that wrenches are orthogonal to twists as per Equation 3.4 (substituting virtual displacements with velocities). The residual of equation 3.4 provides the corresponding friction work W_μ:

$$W_\mu = v \cdot f + \omega \cdot n \tag{3.45}$$

The friction work comes from the friction moment about the axis of rotation. The friction moment is directed about the axis of rotation.

$$n_\mu = \frac{W_\mu}{|\omega|}\hat{\omega} \tag{3.46}$$

Axial relaxed constraint

The *axial relaxed constraint* also has parameters $(d_x, d_y, d_z, w_x, w_y)$ to configure the axis of rotation and the center of rotation. The radius $l \in \mathbb{R}$ is included to fully define the constraint.

$$\Phi_2 \equiv \left\| \mathbf{r} - [d_x, d_y, d_z]^T \right\| - \mathbf{l}^2 = 0 \tag{3.47}$$

$$\Phi_1 \equiv \left(r - [d_x, d_y, d_z]^T \right)^T e^{\tilde{w}} [0 \;\; 0 \;\; 1]^T = 0 \tag{3.48}$$

Equation (3.47) restricts the body origin to a sphere of radius l centered about (d_x, d_y, d_z). Equation (3.48) restricts the body origin to a plane containing (d_x, d_y, d_z) with normal $e^{\tilde{w}}[0 \;\; 0 \;\; 1]^T$.

The friction model is represented by equations (3.27) and (3.28)

Degrees of freedom and model parameters per constraint

Constraint	**DOF**	**number of model parameters**
Point-on-plane	5	6
Point-on-line	4	7
Planar	3	6
Planar relaxed	5	3
Prismatic	1	10
Prismatic relaxed	4	4
Axial rotation	1	11
Axial relaxed	4	8

Table 3.1: Degrees of freedom (DOF) and number of model parameters for each constraint

4 Fitting pose and wrench measurements to constraint models

The quality of the fit model depends on the available information in the demonstration. Indeterminate information will lead to degenerate model, such as fitting a line with one point. Incorporating wrench measurements along with pose measurements mitigates this problem because wrench measurements compliment pose measurements. Pose measurements provide information about the permissible configurations of the rigid body while wrench measurements provide information about the impermissible directions of motion. Therefore, incorporating wrench measurements during fitting allows fitting models with fewer samples of data (see figure 4.1).

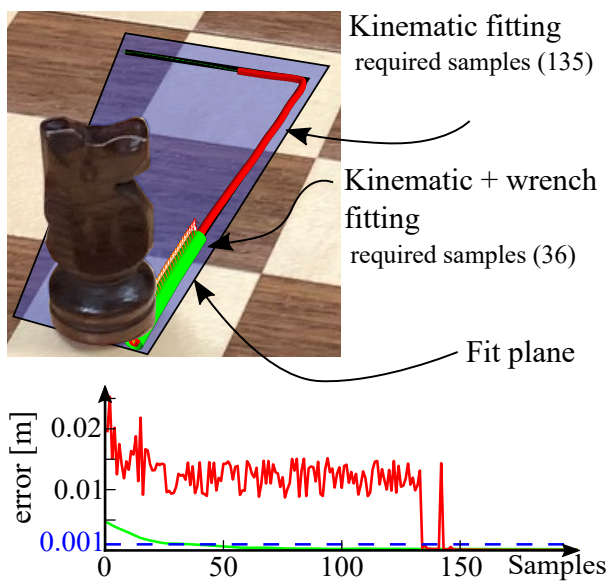

Figure 4.1: A User demonstrates a chess move by sliding the knight against the chess board. Fitting the *planar-relaxed constraint* using kinematic information [green] is compared against fitting with a combination of pose and wrench measurements [red]. Kinematic-Wrench fitting is able to identify the plane using considerably fewer samples than kinematic fitting.

Each geometric constraint model is configured by model parameters α. The task is to fit model parameters α using N samples consisting of recorded position, orientation, applied force and applied moment information. The generalized constraint equations are used to formulate a kinematic error objective. Equations (3.5) and (3.6) are used to formulate a wrench error objective. The two error objectives are cooperative and included in a summed objective to fit recorded demonstration data to a constraint model.

4.1 Kinematic error objective

Let the parameters of the constraint be α. Let the ith sample of data be represented with a tuple $p_i = (r_i, q_i, f_i, n_i, v_i, \omega)$. The kinematic error objective across the N samples of data $\mathbf{p} := \{p_1, ..., p_N\}$ is given by:

$$K_{error}(\alpha, \mathbf{p}, \mathbf{w}) = \sum_{i=1}^{N} w_i \|\Phi(p_i, \alpha)\| \tag{4.1}$$

where $\|\square\|$ is the L(2) norm and $\mathbf{w} := \{w_1, ..., w_N\}$ are the weights on each sample. Sample weights allow increasing the influence of certain samples over others. Some samples included in the regression may not be part of the constraint or may be caused by measurement errors and should have less influence when fitting. For the kinematic error objective, only (r, q) is required from tuple p.

4.2 Wrench error objective

A wrench error objective is developed to incorporate wrench measurements during fitting. Equations (3.5) and (3.6) provide a relationship between the constraint equations and the applied forces and moments in a demonstration. To apply equations (3.5) and (3.6) for regression, the Lagrange multipliers must be determined jointly with the parameters of the constraint. This may be formulated as a weighted least squares objective:

$$W_{error}(\alpha, \mathbf{p}, \mathbf{w}) = \sum_{i=1}^{N} w_i \min_{\lambda} \left(\|\Phi_r^T(p_i, \alpha)\lambda + f_{ri}\| + \|\Phi_\pi^T(p_i, \alpha)\lambda + n_{ri}\| \right) \tag{4.2}$$

Equations (3.5) and (3.6) are linear in λ and may be solved using linear least squares regression. The least squares error of $\mathbf{A}\lambda = b$ is denoted as $LSQ(\mathbf{A}, b)$ where $\mathbf{A}$ and b are given by:

$$\mathbf{A} = \begin{bmatrix} \Phi_r^T \\ \Phi_\pi^T \end{bmatrix} \tag{4.3}$$

and

$$b = \begin{bmatrix} -f_r \\ -n_r \end{bmatrix} \tag{4.4}$$

Equation (4.2) may be reformulated as:

$$W_{error}(\alpha, \mathbf{p}, \mathbf{w}) = \sum_{i=1}^{n} w_i LSQ \left(\begin{bmatrix} \Phi_r^T(p_i, \alpha) \\ \Phi_\pi^T(p_i, \alpha) \end{bmatrix}, \begin{bmatrix} -f_{ri} \\ -n_{ri} \end{bmatrix} \right) \tag{4.5}$$

We do not directly measure the reaction forces and moments. Equations (3.11) and (3.12) are used to represent f_r and n_r. The reaction forces and reaction moments are jointly estimated by including their expressions in the optimization objective.

4.3 Maximum likelihood estimate

We need to determine the most likely model parameters α given model Φ and demonstrator data $\mathbf{p}$. The maximum likelihood estimate of model parameters α based on observations $\mathbf{p}$ is $\hat{\alpha}_{MLE}$ and can be be estimated as:

$$\hat{\alpha}_{MLE} = \arg\min_{\alpha} \left(W_{error}(\alpha, \mathbf{p}, \mathbf{w}) + K_{error}(\alpha, \mathbf{p}, \mathbf{w}) \right) \tag{4.6}$$

The two objectives W_{error} and K_{error} are cooperative. Experimental evaluation indicates that changing the weights between the two objectives has little effect on

the overall fit.

4.4 Robust regression

Segmentation error and non-gaussian noise from sensor measurements may cause a direct nonlinear least squares optimization to fit poorly. All of the samples may not be part of the constraint. We use iterative reweighted least squares (IRLS) (11) to estimate the sample weights associated with the constraint jointly with the model parameters which helps to reject outliers.

The approach initializes equal weight values to each sample. It performs an optimization using a standard non-linear optimization algorithm (in this work we use BFGS (13)) using these weights. It calculates the errors associated with each sample and down weights samples that have large error. A re-weighting step is used to normalize the weights using a regularization function to prevent aggressive down-weighting of samples so all samples continue to have an influence on the objective. Using these new weighted values, the algorithm performs another iteration of optimization.

The process of finding the most likely value of α is give in Algorithm 1. The exit criteria for the algorithm is a maximum number of iterations N. Fig. 4.2 shows how selection of N affects fit quality.

Algorithm 1 Iterative reweighted least squares regression

```
procedure RobustRegression(X, w, N)
    initialize(w)                                  ▷ set all weights to 1
    for j = 1 ... N do                             ▷ number of iterations
        w ← w / sum (w)                            ▷ Normalization
        α ← FitFunction(X,w)                       ▷ BFGS minimization
        E ← ErrorFunction(X,α,w)                   ▷ vector of fit errors
        w ← 1/(1 + E^0.1)                          ▷ Regularization
    return α                                       ▷
```

Using the knight moving across the chessboard example shown in Fig. 4.1, we intentionally segment the demonstration incorrectly so a portion of the demon-

stration is in free space. When fitting the model without iterative reweighted least squares (IRLS) regression, the fit plane is offset because of outliers caused by incorrect segmentation. After 100 iterations of IRLS the fit converges to the true constraint model.

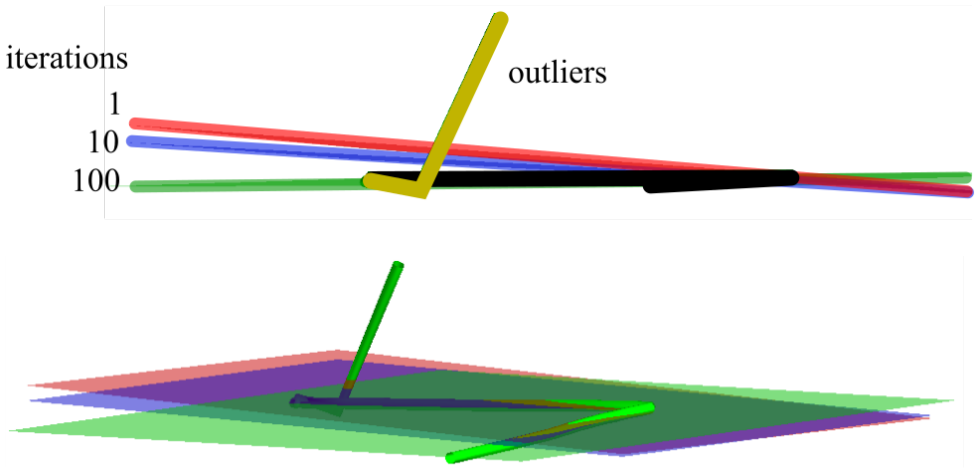

Figure 4.2: Bottom: The perspective view of the demonstration of moving the knight is shown. The demonstration is intentionally segmented incorrectly. Iterative reweighted least squares is used to robustly fit the demonstration. 100 iterations of IRLS fits the model to the constraint.

5 Model selection

Once a set of candidate models are fit, it is necessary to identify the model most consistent with the measured data. The difficulty of the model selection problem is a function of the number of candidate models and the uniqueness of these models.

To simplify the model comparison process, we create a single selection metric that can be used across models. In our formulation, the optimization fit error is not appropriate as the models consist of regressor functions that are not guaranteed to have consistent units. To allow for comparison, kinematic and wrench error metrics are defined in terms of measured quantities (m, rad, N, Nm). These metrics are combined into a single error metric by counting measurements above a threshold for each metric. We refer to this process as error tallying.

5.1 Error metrics

Position and force information allow for the construction of separate kinematic and wrench error metrics. An error metric is defined as a measure of distance between a measurement (r, q, f, n) and the constraint model.

Kinematic error metrics

Given that a constraint can enforce both a position and an orientation, there is a need for two kinematic metrics. A position distance $D_r(\Phi(r^*, q^*), r, q)$ and an orientation distance $D_q(\Phi(r^*, q^*), r, q)$ are defined for a pose measurement $(r, q) \in SE(3)$ where (r^*, q^*) is coincident with the constraint (satisfies Φ). Evaluation of (r^*, q^*) consists of finding the point on the constraint that is closest in overall distance to the measured point. Determining these values is posed as a least squares regression where (r^*, q^*) are solved simultaneously:

$$\hat{r}^*, \hat{q}^* = \underset{r^*, q^*}{\arg\min} \begin{pmatrix} & \|r^* - r\| + \angle(\mathrm{conj}(q) \star q^*) \\ + & W\Phi(r^*, q^*)^T \Phi(r^*, q^*) \\ + & W\|q^{*T} q^* - 1\|_2 \end{pmatrix} \tag{5.1}$$

where W is a large weight (1000) on the constraint equations to ensure that r^* and q^* are fixed to the constraint. $\angle$ is the rotation angle of a quaternion, $\star$ represents quaternion multiplication and conj represents the conjugate of a quaternion. $\angle(\mathrm{conj}(q) \star q^*)$ computes the angle between q and q*.

The kinematic error metrics, D_r and D_q can subsequently be computed as:

$$D_r = \|\hat{r}^\star - r\| \tag{5.2}$$

$$D_q = \angle(\mathrm{conj}(q) \star \hat{q}^\star) \tag{5.3}$$

Wrench error metrics

Measured forces and moments provide metrics that quantify inconsistency with the constraint model, Φ. By computing the Lagrange multipliers individually for each measurement (see Section 4.2), a force and moment error can be written as the residual of the virtual work equations previously defined in equations (3.5) and (3.6). The values are normalized with respect to the measured forces and moments to remove a dependence on magnitude:

$$E_f = \|\Phi_r^T \lambda + f\| / \|f\| \tag{5.4}$$

$$E_n = \|\Phi_\pi^T \lambda + n\| / \|n\| \tag{5.5}$$

Combined error metric

For model selection, there is a desire to create a single metric capable of disambiguation between candidate models. Any sort of weighted sum of the individual error metrics contains inherent bias dependent on the range of motion and level of

applied forces and moments. To circumvent this, we introduce a combined error metric (CE) that looks at how many samples of the constrained motion are above the measurement error for each of the individual metrics (defined as p_{SE}, q_{SE}, f_{SE}, n_{SE}).:

$$CE = \sum_{n} \begin{pmatrix} (D_r(n) > p_{SE}) \\ + (D_q(n) > q_{SE}) \\ + (E_f(n) > f_{SE}) \\ + (E_n(n) > n_{SE}) \end{pmatrix} + 1 \tag{5.6}$$

Section 7.3 demonstrates setting the threshold for each of the error metrics based on an estimate of the experimental setup accuracy. One tally is added to each model's combined error after the entire tallying process to allow conversion to a likelihood metric without the issue of division by zero.

$$\hat{L_m} = \frac{CE(m)^{-1}}{\sum_m CE(m)^{-1}} \tag{5.7}$$

5.2 Model Generality and Penalization

When considering only kinematics, it is very likely that multiple models will fit a constrained motion well. This is because a demonstration may be consistent with models of varying complexity. Consider a demonstration of an *axial constraint* as shown in Fig 5.1.

While the motion follows an exact axial path, the position and orientation data is also consistent with the more general model of a plane. This suggests the use of a penalized likelihood such as AIC/BIC which corrects for issues with more general models (2; 67). When including wrench measurements, the motion contains forces that are inconsistent with the *planar* model allowing for proper detection without the use of penalization. While there are extreme cases where a set of pose and wrench measurements may be consistent with two different models, it is highly unlikely to occur in actual measurements. So while it may make sense to penalize a kinematic model selection, our method which incorporates a force and moment

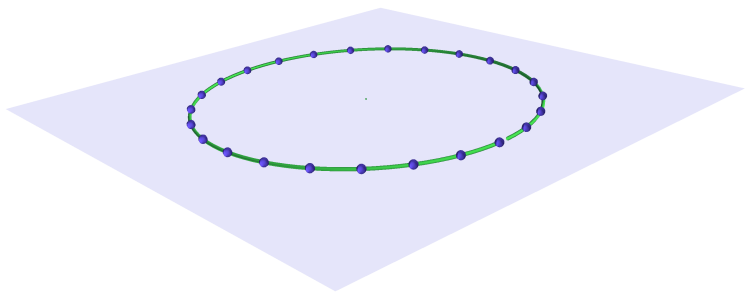

Figure 5.1: Demonstration of *axial* motion (green) that is also fit by a *planar constraint* (blue) . The kinematic data contains no data inconsistent with the planar model. Wrench measurements include forces in a direction that cannot be described by the planar model.

error does not require penalization. Section 7.3 explains construction of a penalized kinematic metric and compares the kinematic, penalized kinematic, and combined error methods.

5.3 Null Hypothesis

The fitting and model selection procedures determine the best model that matches the data for a particular segment. Sometimes there just may not be a constraint at all. In such cases a *Null Hypothesis* is used and described as follows.

An error threshold for each error metric (position, orientation, force and moment) is applied to each sample in the segment and contribute to cumulative tallies V_r, V_q, V_f and V_n. A tally is awarded when the error is more than the threshold. The thresholds are set above the system's measurement noise. If the percentage of cumulative tallies for any metric is greater than an acceptable maximum per-

centage, the segment is discarded. Additionally, any segment consisting of only a few samples will easily fit a given model, but does not lend to confidence that the data is actually constrained. For example, if the segment is only two points, the *prismatic relaxed* model will always fit with zero error. To avoid this case, we discard segments of very short duration (in our work, less than or equal to five samples). The implemented numerical values of the thresholds are described in table 7.3 in chapter 7.

6 Segmentation

There is no strict pattern for when constraints occur in a demonstration. A demonstration may contain a single constraint, multiple different constraints, or may visit the constraint multiple times. Free space motion may occur in-between two different constraint interactions or sometimes the motion might directly transition between different constraints. Regression and model selection requires predetermined locations of the constraints. While user-specified segmentation is possible, it might be desirable to make the process automatic to avoid training users how to segment.

Prior work has explored the segmentation problem. Appendix A (69) describes a way to use the frequency content in force and moment signals to infer actions in demonstrations. This is not applicable to constraint interaction as constraint interaction may not produce sufficient vibratory excitation. One approach (70) is to fit each constraint type over a small sliding window. Similar constraint fits are clustered over the entire demonstration to determine the constraints over the entire demonstration. However this is computationally expensive and limited to simple constraint models. It also requires significant tuning of heuristics by the user. This approach is described in appendix B. CHAMP (50) uses change point detection to determine constraint locations and can detect transitions between constraints without free-space motion, but this method is limited to simple models. Another approach is to use a generic action recognition approach.

The method used in this work exploits measured reaction forces that occur when interacting with constraints. If a measured force exceeds a specified threshold (e.g. twice the measurement noise of the system), then a constraint could be present. The result is a Boolean list of contiguous samples indicating possible constrained motion segments and free space motion segments. One caveat of this simple segmentation method is that free space motion must occur between each constraint interaction. While our experimental demonstrations are designed to meet this assumption, other methods that identify transitions between constraints or that incorporate

multiple constraints within a single model could be considered as future work.

Motions other than constraint interaction can register forces such as in the case of gravity or accelerating inertias. As a result, this segmentation approach is highly susceptible to false positive segments. To compensate for this, we consider that an identified segment may or may not represent constrained motion. Incorrect segments are removed by applying a Null Hypothesis described in section 5.3.

7 Experimental Evaluation of Constraint Inference

The constraint inference approach is evaluated with three experiments:

- **An evaluation of model fitting using pose and wrench measurements**, that shows that high degree of freedom models can be fit with shorter demonstrations when incorporating wrench measurements. (Section 7.2)
- **An evaluation of model selection** that shows improved accuracy using the combined error metric and penalization. (Section 7.3)
- **An evaluation of the complete constraint inference method** on a situated demo consisting of an espresso making task. (Section 7.4)

7.1 Experimental setup

Experimental data was collected from 9 participants recruited from the university student population. Subjects use instrumented tongs (56) (described in chapter 2) which are equipped with ATI mini40 force-torque sensors and Opti-track motion capture markers to perform demonstrations. The instrumented tongs, provide measured position, orientation, force, and moment trajectories sampled at 120 Hz (after appropriate resampling and filtering).

When the tongs grasp objects in the environment, there is relative slip between the tongs and the grasped object. Therefore constraint models that do not incorporate orientation constraints are considered when using pose measurements from the tongs. These correspond to *prismatic relaxed*, *planar relaxed* and *axial relaxed* constraints. When orientation of the constrained body is important, a motion capture coordinate frame is attached to the object to track its precise position and orientation. Applied forces and moments measured from the tongs are transformed to this coordinate frame. The constraints where orientation is important are *point-on-plane*, *point-on-line*, *prismatic*, *axial* and *planar* constraints.

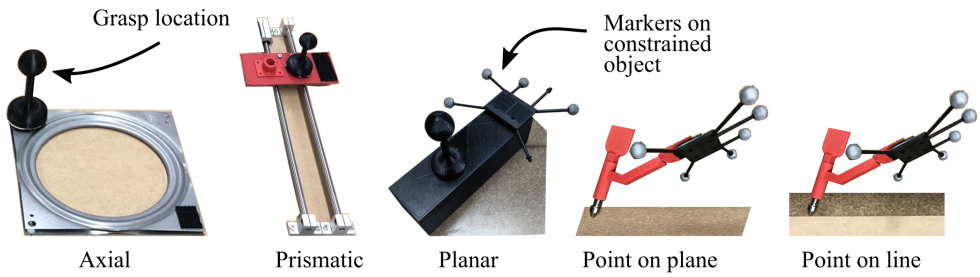

Figure 7.1: Physical constraints used in experiment.

An experimental test bed with different physical constraints, shown in Fig. B.6, was constructed. Each constraint has a grasp location where participants use to grasp and manipulate the constraint.

7.2 Influence of wrench data when fitting constraint models

An experiment to evaluate fit accuracy as a function of demonstration duration was conducted. Participants interacted with each physical constraint on the test bed. Two 10 second demonstrations were recorded per constraint. For each constraint, randomly sampled contiguous segments of data of different lengths were fit to the corresponding constraint model. A fit error that uses the position error metric described in equation (5.2) is computed. A more detailed description of the fit error and acceptable error for each model is described in 7.2.

Fig. 7.2 shows a regression analysis for constraints with large number of degrees of freedom. Fit accuracy as a function of contiguous samples is evaluated. For models with high degrees of freedom (*point-on-plane* - 5 DOF, *planar relaxed* - 4 DOF, *planar* - 3 DOF, and *point-on-line* - 4 DOF) considerably fewer samples of kinematic and wrench data are required to fit models.

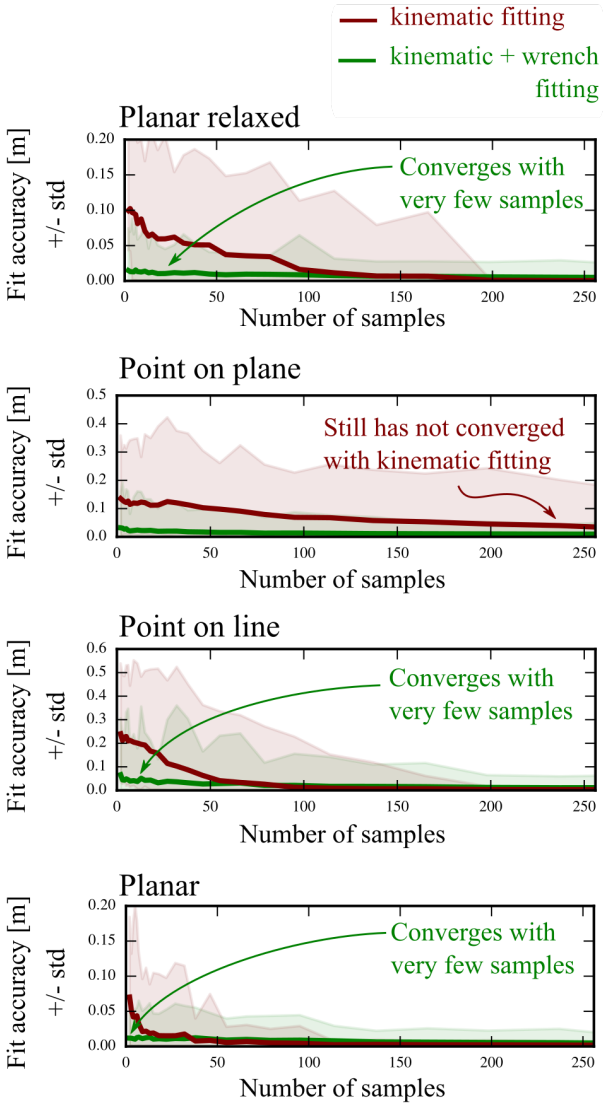

Figure 7.2: Comparison between fit accuracy in meters (see section 7.2) verses number of contiguous samples in a demonstration for high degree of freedom models. Fitting with kinematic and wrench measurements [green line]. Fitting with kinematic measurements [red line]. Shaded regions correspond to the first standard deviation about the mean. Constraints with higher degrees of freedom converge with fewer number of samples when wrench measurements are incorporated during the fitting process.

Wrench measurements provide additional information and compensate for degenerate kinematic information that occur in short demonstrations allowing a constraint to be fit with fewer samples. Fig. 7.3 shows a comparison between fitting the *planar relaxed constraint* model with pose and wrench measurements for a particular participant's demonstration. Short demonstrations on the *planar relaxed* constraint may consist of primarily linear motion which leads to degenerate planes when fitting with only kinematic data. Motion in a straight line will not completely determine the orientation of the plane as many different planes may fit the same line.

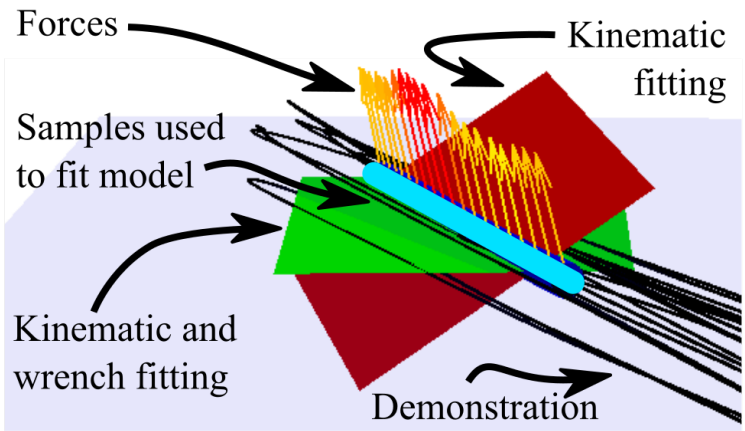

Figure 7.3: An example demonstration of the *planar relaxed constraint* from Demonstrator 6. Using a small portion in the demonstration (cyan) of 0.2 seconds fitting with only kinematic data (red plane) does not produce the correct model. Including reaction wrenches in the analysis fits a better plane (green) to the demonstration.

For models with fewer degrees of freedom such as the *prismatic*, *axial rotation*, *axial relaxed*, and *planar* constraints, the wrench method yields similar results to the kinematic-only fitting method. This is expected since high numbers of kinematic constraints lead to many directions of admissible forces and moments which limits the utility in fitting. For example, in the *prismatic* model, there is only a single degree of freedom that is not constrained. However, this is the direction of motion which is also the direction in which friction occurs. Since we assume that forces in

this direction could be friction, they are removed. The remaining two directions are acceptable reaction forces. As a result, there is no additional information added from including wrench measurements in fitting for this model. Fig. 7.4 shows a regression analysis of number of samples verses fit error for the constraints with a small number of degrees of freedom. The *axial relaxed* and *axial rotation* models have high nonlinearity and sometimes the fits are incorrect due to convergence to a local minimum. This explains the large standard deviation for both models even when using a large number of samples.

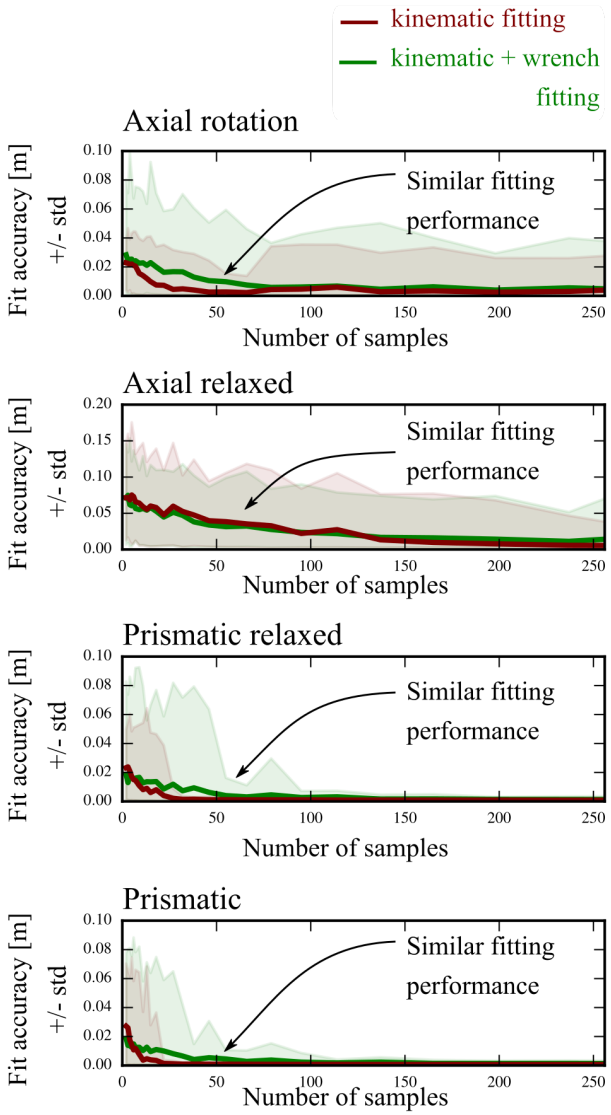

Figure 7.4: Fit accuracy in meter (see section 7.2) verses number of contiguous samples in a demonstration for each constraint. Fitting with kinematic and wrench measurements [green line]. Fitting with kinematic measurements [red line]. Shaded regions correspond to the first standard deviation about the mean. Constraints with lower degrees of freedom do not benefit when wrench measurements are incorporated during the fitting process.

In this experimental setup, the orientations of forces are derived from motion capture orientation measurements. As a result, the force and moment errors tend to be higher based on the propagated orientation error. This can be seen in the increased noise floor for the wrench fitting method in the regression plots (i.e. the wrench method converges to a higher error than the kinematic method). While this is a product of the experimental setup, it demonstrates that it may still be beneficial to consider using kinematic fitting for constraints where the force and moments add little information (e.g. *prismatic constraint*) to avoid cases when the forces and moments have a significant noise floor and to simplify the dimensionality of the non-linear optimization.

Wrench measurements improve fitting of models with high degrees of freedom over short demonstration duration. They help in situations when kinematic information is insufficient to completely define the constraint model. For constraint models with high degrees of freedom we recommend that wrench measurements are incorporated in the fitting process. For models with low degrees of freedom, wrench measurements can be omitted to speedup computation time and remove complexity in the optimization process.

Evaluation of fit error

Each fit was evaluated against either a) the **entire** recorded demonstration or b) an independent demonstration recorded with the experimenter. The demonstration used to evaluate the fit is referred to as the *ground truth demonstration*. The independent ground truth demonstration was used for the *point-on-plane* and *point-on-line* constraint models because in some user demonstrations, the entire demonstration's data was insufficient to fit the constraint model to the true constraint and would resulted in degenerate fits. A limitation of the the ground truth demonstration is that it may incorporate an error of up to 1 cm due to the inability to accurately reproduce the configuration of the constraint during ground truth recording.

An error of 1 cm is expected for the *relaxed constraint* models since the pose of the constraint is measured using the tongs and the tongs grasp gradually slips

during the course of the entire demonstration. All other models are tracked using an independent motion capture frame which has an advertised accuracy of a millimeter.

Fit error plots for Fig. 7.2 and Fig. 7.4 are generated by aggregating results from all 144 demonstrations (9 demonstrators X 2 demonstrations per constraint X 8 constraint models). Each demonstrator performs two 10 second demonstrations of each constraint type. Participants were instructed to interact with the constraint without violating the constraint (e.g. not lifting the planar constraint object off of the plane). The *point on plane* and constraints requires orientation information to fit the constraint models appropriately. Participants were instructed to move their wrist more during these constraint interactions.

Multiple demonstration lengths are selected from the demonstration and evaluated against all the samples of the ground truth demonstration. 5 randomly sampled contiguous segments from each demonstration are selected per segment length. The length of segments are logarithmicly spaced from 2 to 256 samples. The mean fit error and first standard deviation are reported where the line indicates the mean and the shaded area indicates the bounds on the first standard deviation. The fit error for each fit is calculated using the position error metric computed using equation (5.2). The average of the per sample error (across the whole ground truth demonstration) is used to compute an individual fit error.

7.3 Evaluation of Model Selection

The error tallying method described in equation (5.6) requires thresholds for each of the error metrics. The metric values are set based on an estimate of the data collection accuracy. The force and moment error represent the component orthogonal to the permissible force and moment directions. Since the errors are normalized with respect to the measured force and moment, the formulation is the equivalent to evaluating the sine of the angle between the measured and error vectors. This relationship is illustrated for the planar constraint in Fig. 7.5. The error for the force and moment is set to the same angle as the orientation error (2 degrees). The force

and moment is then $\sin(2^\circ) \approx 0.06$. The full list of parameter values is provided in table 7.3.

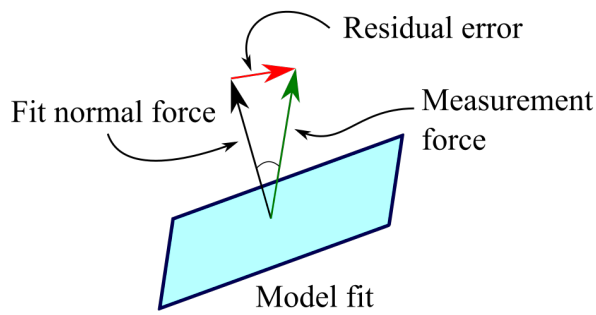

Figure 7.5: The force error measures the orthogonal direction to the permissible force direction. As a result, the normalized residual error resembles the sine of the angle between the measurement and fit normal force. This is demonstrated for a planar model.

We construct a penalized kinematic selection method for comparison that is similar to that of Akaike. In the case of our constraint formulation, a combination of degrees of freedom and model parameters provides a better analog to model generality than the model parameters alone. The penalized likelihood metric is defined as follows:

$$m = \arg\min_{m}(\alpha DOF(m) + \beta k - \ln L(m)) \tag{7.1}$$

where DOF refers to the kinematic degrees of freedom of a model and k is the number of model parameters (Corresponding values for the constraint models used in this paper are provided in table 3.4. Eighty training demonstrations are performed (10 per constraint model) to fit the weighted parameters α and β.The training set accuracy of the kinematic method is maximized with $\alpha = 1.69$ and $\beta = 0.40$ (accuracy: 0.65).

Using data from the 9 participants, model selection is compared between the kinematic, penalized kinematic, and combined error methods. Demonstrations are

split into 3.33 second segments (sufficient samples for fitting). Since the demonstrations are sufficiently long, all demonstrations are fit with kinematic data. Since model selection is requisite on the constraint being properly fit, fits that do not match the performed constraint are manually discarded. We consider the eight constraint models from Fig. 8.2. Choice of these models, some of which are highly similar (e.g. *planar* and *planar relaxed*), intentionally complicates the selection process and allows for a finer comparison between methods. The results appear in Fig. 7.6.

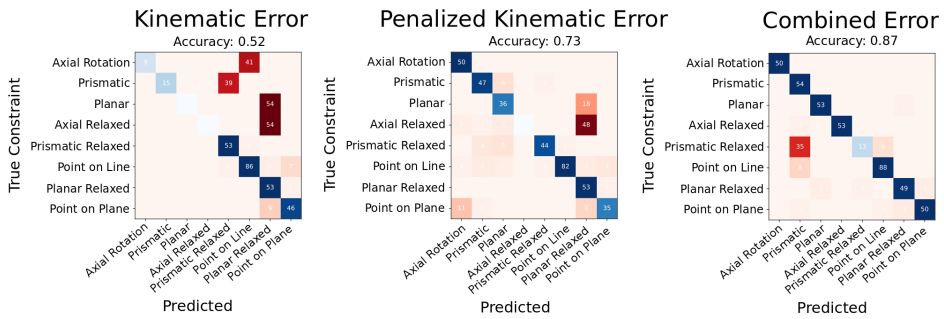

Figure 7.6: Comparison between kinematic and combined methods for model selection. The use of a penalized likelihood greatly improves kinematic model selection, but the method incorporating wrench measurements has a much higher overall classification.

In the kinematic-only tallying method, misses can be attributed to model generality. For example, the *axial relaxed* constraint is always identified as *planar relaxed*. When only kinematic measurements are considered, this is a likely case. There is always a plane that can describe an axial motion and moreover any small errors in the position might still be consistent with the plane model, but not necessarily with the axial model. Two other models, (*planar constraint* and *prismatic constraint*), are consistently misidentified as their relaxed versions. As a reminder, the relaxed version of the constraint does not enforce a rigid orientation. Similar to the previous case, any small errors when the orientation is rigid favor the relaxed model in the selection process. The kinematic method also misidentifies the rigid axial constraint as a point on line constraint. In this case, both models do not receive any tallies

and as a result, the selection is arbitrary based on the order in which the models are selected. Penalization of the kinematic method increases classification accuracy from 0.52 to 0.73 correcting for many of the issues with the general model. However, even with correction, the *axial relaxed* model is still identified as a *planar relaxed* constraint.

Use of the combined error greatly improves the classification accuracy (0.87). The only model that is consistently misidentified is the *prismatic relaxed constraint*. In most cases, the identified model is the rigid version of the constraint. The relaxed model does not enforce an orientation which consequently means there can be no generated moments. When a participant interacts with the tongs, the grip can slip leading to changes in orientation, but for the most part, the generated moments and rigidity may be more consistent with the rigid model. As such, these cases are not necessarily even misses and more represent cases of demonstrations that fall between the two models.

Threshold	**Value**
Position	2 mm
Orientation	2 deg
Force	$\sin(2^\circ) \approx 0.06$
Moment	$\sin(2^\circ) \approx 0.06$
Null Hypothesis Max %	0.75

Table 7.1: Threshold values for error tallying

7.4 Situated Demonstration

We evaluate our system on a proof-of-concept situated task. The task involves preparing an espresso using the instrumented tongs as an input device. Fig. 1.5 shows the experimental setup.

1. The demonstrator first slides an espresso cup under the espresso machine spout (*planar-relaxed constraint*).

2. The demonstrator then uses the instrumented tongs to pull out a drawer (*prismatic-relaxed constraint*) containing a single coffee pod.

3. Using the tongs the demonstrator carries the pod to the espresso machine and places the pod inside.

4. Finally the demonstrator uses the tongs to close the lever (*axial-relaxed constraint*) of the espresso machine.

Using the system described in this paper, the demonstration is parsed for the three possible constraint models: *planar relaxed*, *prismatic relaxed*, and *axial relaxed* constraints. The three relaxed models are used as the instrumented tongs have slip directions and do not enforce a rigid orientation when grasping the constrained objects. The two 1-DOF models, *prismatic relaxed* and *axial relaxed* models are fit using the kinematic method, to avoid the increased noise floor of the experimental setup.

The results of the constraint inference are in Fig. 7.7. The system correctly identifies 37 out of the 48 constraints in 18 demonstrations (2 per participant). Note that there are only 48 constraints because in 6 demonstrations, the participants lifted the cup rather than sliding it, so there is no planar motion constraint.

The espresso setup evaluates the method on a realistic task where the constraints may not be as precise as the constraints in the earlier experiments. For example, the drawer used for linear motion is plastic and has 1.5cm on play in its constrained motion. Additionally, the espresso machine lever (*axial relaxed constraint*) can be difficult to grasp leading to some demonstrations that do not resemble axial motion. As expected, a few constraint interactions were misclassified by model selection or rejected by the null hypothesis. In spite of the imprecise constraints, the system correctly infers a majority of the constraints.

The null hypothesis was successful in rejecting non-constrained motion that exceeded the force threshold, such as when participants hold the coffee pod and accelerate towards the espresso maker. In the 18 demonstrations, the null hypothesis also rejected 20 of these motions.

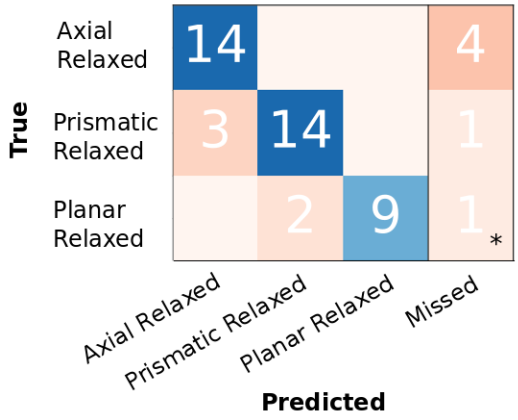

Figure 7.7: Confusion matrix of the end-to-end system on 18 espresso making demonstrations (9 participants, 2 trials, 3 potential constraints per task). In 6 demonstrations, the participant lifted the cup rather than sliding it, so there was no planar constraint to be detected (*).

Fig. 1.5 shows the results for one of the participants where the method is successful. Our approach correctly selected the model 77% of the time. It identified the correct locations of constraints rejecting 20 extraneous segments through the null hypothesis. Correctly inferred constraints were all correctly fit (as shown in Fig 1.5). Constraints were identified using a single demonstration in which the demonstrator performed them naturally with minimal instruction.

8 Robust robot replay using identified constraints

Robots are frequently required to interact with physical constraints. For example, a robot may need to wipe a table, actuate levers and knobs when tending a machine, or open a dishwasher door to pull out its rack. Some tasks may require following known paths precisely such as when drawing against a whiteboard. A key challenge of physical constraint interaction is that physical constraints impart restrictions to the objects that interact with them. A task may require following a path along the constraint precisely while robustly interacting with the physical constraint. Hybrid position - force control (or hybrid control) is used when robots interact with these physical constraints because it provides robustness to variations in the known location of the physical constraint as well as the possibility of applying controlled forces against the constraint. Hybrid control achieves this by applying position control along unconstrained degrees of freedom and force control along constrained degrees of freedom of the physical constraint. Consequently, hybrid control requires the physical locations of the task geometry which may not be easy to measure (e.g. the location of an oblique planar surface) and may also require operators trained in the use of hybrid control.

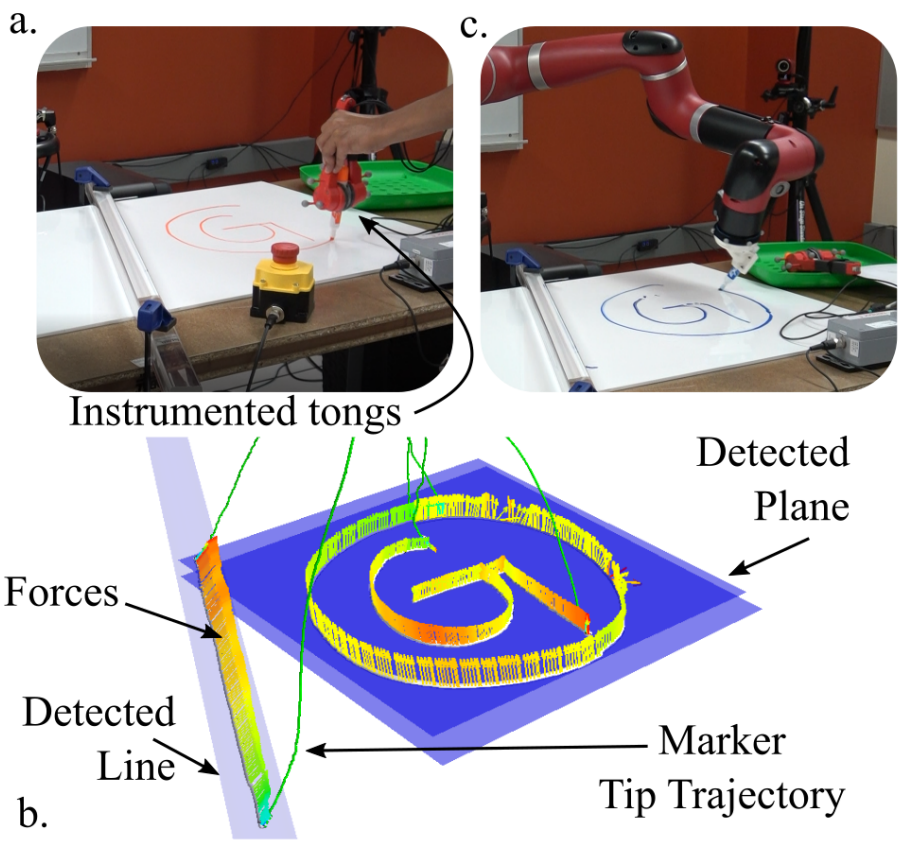

Figure 8.1: **a. A human demonstrator draws on a whiteboard using instrumented tongs designed to record demonstrations. The demonstrator first draws a *G* then using the straight edge draws a line and finally draws a circle around the *G*. b. The demonstration is segmented and annotated by inferring the geometric constraints in the demonstration. c. Using the inferred constraints and annotated demonstration, the robot performs the task by matching the kinematic motion closely while maintaining appropriate contact forces against the board.**

We introduce an approach that simplifies the creation of constraint-aware, hybrid control for tasks that involve following a prescribed path along a geometric constraint. Our approach enables *robust replay* of a provided human demonstration by transforming it into a robot program that uses constraint-aware, hybrid control where appropriate. Our work builds on recent methods for inferring geometric

constraints in recorded demonstrations and couples them to standard hybrid position - force control methods available on industrial robots. The key challenge is to automatically adapt the information available from the demonstration into a form usable by the controller available on the robot. Our key idea is that we can use inferred constraint geometry to set the frame of the hybrid position - force controller and to generate feasible kinematic trajectories that are consistent with the constraint. This allows our method to apply correct forces in the constrained directions and match kinematic goals in unconstrained directions.

We apply our approach to implement *robust replay* of a single recorded human demonstration that involve physical constraints. Given a recording of human movements, our approach allows a robot to reproduce the movements closely while being robust to perturbations in geometry and reproduce forces that are important to the task. For example, given a recording of a person drawing with a marker, our methods allow the robot to reproduce the drawing, applying appropriate forces to adequately maintain contact between the marker and drawing surface while being robust to changes in position of the drawing surface, or imperfections in its smoothness.

Our method decomposes a recording of a human performing a task into subtasks, identifies which subtasks have constraints, determines the type and geometric parameters of these constraints, computes a feasible robot trajectory that follows the demonstration subject to the constraints, and creates a program that executes the trajectory using hybrid position - force control to provide robust, constraint-aware execution of tasks segments where appropriate. We infer the kinematic constraint using using the approach elaborated in prior chapters and use the inferred constraint to determine the parameters of hybrid position - force control. We extend *RelaxedIK*(58) for feasible trajectory generation, introducing *Constraint Augmented RelaxedIK* (CaRIK), a novel solver that generates feasible trajectories against geometric constraints while avoiding singularities, self and known object collisions, and joint position limits while following paths closely. We build on available hybrid position - force control methods (9) to achieve robust interaction with the constraint. The result is an approach that can take a single human demonstration of the task as

input and allows a robot to replay the task in a manner that matches the kinematic tool path closely, applies appropriate contact forces, and is robust to changes in the constraining objects or imperfections in execution.

Hybrid position - force control allows the creation of robust control policies. Our work shows that it is possible to apply such robust control by inferring constraints from a single demonstration. Potential advantages of such an approach include reducing the need for multiple demonstrations, avoiding multiple robot trials required for learning and an explicit representation of the constraints that improve user interpretability. We demonstrate our approach with a variety of constraint types, and provide empirical evidence of its ability to achieve robust execution.

8.1 Related Work

Learning by demonstration (3) aims to teach robots through observations of humans demonstrating tasks. A robot can be taught new tasks either using multiple demonstrations or a single demonstration. Multiple demonstrations are used for generalization - the robot learns to perform a task by fitting a general model to several demonstrations as in (72) and (39). In the case of a single demonstration (one-shot learning), the robot is equipped to recognize and perform actions that form the building blocks of complex tasks as similar to (54). In this work, a single demonstration of a complex task is replayed robustly through inferring geometric constraints so hybrid control may be applied.

Physical tasks may be learned using models that generalize multiple human demonstrations. Rozo et al. (61), use hidden markov models (HMM) and gaussian mixture regression (GMR) to learn robot policies. A similar approach by Kalakrishnan et al. (31), uses Dynamic Movement Primitives (DMPs) coupled with Path Integrals (PI^2) to learn a policy using reinforcement learning. These approaches may be applied to various different types of tasks but require multiple demonstrations and training. Our approach uses a single demonstration to robustly follow a path closely. It is capable of following large scale complex paths using a single

human demonstration which may not be possible with the above approaches. It also infers geometric constraints enabling the use hybrid control.

General constraint models may be estimated in a human demonstration to create robot programs. Berenson et al. (7), represented constraints as Task Space Regions (TSRs) which are subsets of SE(3) describing the permissible end effector configuration. Li et al. (41), encoded task manifolds as a Constrained Object Manipulation (COM) task. Englert et al. (17), learn motions under contact using inverse KKT, setting up the constraint learning problem by learning cost functions that solve an optimal control problem. Armesto et al. (4; 5), estimate both linear constraints as well as the null space policy for the robot with a fast closed form solution. This technique may be limited to linear, planar and surface constraints as well as require more data because their method enables generalization and retargeting to other similar constraint surfaces. In (65) probabilistic models are used to learn torque controllers for kinematic and force constraints from multiple human demonstrations. While all these methods learn the geometric constraint through a generalizable model, they require a significant amount of information to determine the constraint. Though our method lacks the ability to generalize to arbitrarily shaped constraint models, it only requires a single demonstration to replay tasks consisting of standard geometric constraints robustly.

Standard geometric constraints such as *prismatic*, *planar* or *point contact* constraints and others may be inferred from human demonstrations. *C-Learn* by Perez-D'Arpino and Shah (54), uses different grasp approaches and key-frames to determine standard geometry such as prismatic constraints, posture constraints, and Task Space Regions(TSRs). This approach provides a semantic constraint representation but can only identify simple constraints such as lines and postures of the robot. It also requires user specified segmentation of motion. In Subramani et al. (70), simple geometric constraints such as lines, arcs and planes are inferred using position and force information. Replaying human demonstrations involving contacts between known polyhedral, cylindrical, and spherical tools and the environment has been explored in detail (45; 15). This approach estimates the contact state in a *contact graph* with prior information about the kind of contacts, the tool, and environment

geometry that occur in the task. In contrast, our method does not estimate contact states. It estimates geometric constraints in the demonstration with no prior knowledge of the environment or tool geometry.

Planning robot motion over a constraint is a hard problem (7; 66; 26) which involves generating robot trajectories over constraints autonomously. In contrast, this work tries to match the demonstrated tool path closely such as in (58) and (43) . Motion planning languages involving geometric constraints, such as eTaSL/eTC (1) or (63), provide a means to program robots to perform complex tasks involving geometric constraints. However they require specification of the task and it's corresponding parameters which may not be possible for novice users.

Execution of robot motion under the influence of a known constraint is a widely explored topic in robotics. Hybrid position - force control (also termed as impedance control, compliance control or admittance control depending on the robot hardware and implementation) (57; 33; 75), which is commercially available on many collaborative robotic platforms from KUKA, Rethink Robotics, Universal Robotics, Kinova and others is used when robots are programmed to interact physically with the environment. The robot programmer uses the task space formalism concept (9) to specify the hybrid controller parameters when programming these commercial robots. This involves complex specification of hybrid control parameters that depend on the geometry of the task which may not be trivial to specify. Our work performs this specification automatically with minimal user input (apart from the demonstration itself) by setting the appropriate hybrid control parameters for the task by inferring the constraint.

There are two ways to program robots to perform these physical actions, a) through a careful modeling approach or b) through a generalization approach. Some physical actions can easily be modeled analytically and robot motion control algorithms may be developed around these models. For example, consider the action of wiping a white board. The model may consist of the equation of the plane, the trajectory of the robot tool against the plane, and the kinematics of the robot. The properties of the model map directly to existing robot motion control algorithms such as path planning or hybrid control. A generalization approach

would correspond to using a general purpose model that is fit to the task using prior experience.[1]

Generalization approaches make sense when it is intractable to model the physical task. (For example, the fine motor manipulation of objects in an anthropomorphic robot hand(52).) These result in robot policies that are difficult to interpret by humans but enable robots to learn and execute complex physical actions.

8.2 Salient features of approach

The problem of robustly replaying human demonstrations involving geometric constraints with minimal user input is tackled here. Robustness is achieved by using inferred geometry from the demonstration to calculate the appropriate *hybrid position - force control*. We infer geometric constraints using the method described in this work which is an extension of (71). We incorporate *RelaxedIK* (58) a purely kinematic method that generates feasible joint trajectories under the influence of multiple kinematic goals. Combining constraint inference from (71), trajectory generation from *RelaxedIK* and *hybrid position - force control* provides a robust approach to constrained motion replay called *Constraint Augmented RelaxedIK*. Our solution involves the following:

1. A human demonstrator uses a hand-held instrumented tool to record a task consisting of geometric constraints. The tool records positions, orientations, forces and moments during the demonstration.

2. The type and locations of the geometric constraints in the demonstration are inferred using the approach elaborated earlier in this document similar to our prior work in (71). The *constraint frame*s (similar to the *Task Frame Formalism* described in (9)), which succinctly describes the degrees of freedom of a constraint, is determined from the inferred constraint models.

[1] A significant portion of teaching by demonstration literature focuses on task generalization from human demonstrations. Dynamic movement primitives are a popular tool to trajectory generalization (72; 12; 25; 49; 27; 31; 51; 34; 47; 36; 10). Other models include using Gaussian Mixture Models (GMM) (32) and deep neural networks (22)

3. Joint trajectories compatible with the constraint are generated using *Constraint Augmented RelaxedIK* (CaRIK) so paths on the constraint may be followed closely. CaRIK builds on *RelaxedIK* (58) which generates kinematically feasible joint trajectories that follow a demonstrated path.

4. The generated kinematic trajectory is augmented with hybrid position - force control (hybrid control) informed by the *constraint frame*. Hybrid control provides robustness during constraint interaction.

The framework is embedded in a simple programming construct called a behavior list. Actions are inferred from a human demonstration and converted into a series of behaviors. These behaviors map to joint trajectories and hybrid position - force control settings which are standard features on many commercially available robotic platforms.

8.3 Mathematical framework

Constraint frame from a geometric constraint model

The degrees of freedom of many standard geometric constraints may be represented by a coordinate frame which we call the *constraint frame*. The *constraint frame* specifies the directions of motion degrees of freedom and constrained degrees of freedom of the constraint.

Consider the prismatic constraint (e.g. pulling out a drawer). The *prismatic* constraint has one unconstrained degree of freedom along its *axis of translation*. The *constraint frame* for this constraint may be configured such that the frame's z-axis is aligned with the constraint's *axis of translation*.

The *axis of translation* can be generated by applying a general displacement (i.e. translation and rotation) transformation of the z coordinate axis:

1. translating the z axis in the x-y plane

2. rotating the translated axis about the origin.

We represent the rotation transformation using two exponential coordinates $\mathbf{w} = [w_x, w_y, 0] \in \mathbb{R}^2$ corresponding to $e^{\tilde{\mathbf{w}}} \in SO(3)$, the exponential map, which is equivalent to a rotation matrix with an axis of rotation in the x-y plane. Rodrigues' rotation formula (48) is used to compute $e^{\tilde{\mathbf{w}}}$. Two coordinates $\mathbf{d} = [d_x, d_y, 0] \in \mathbb{R}^2$ are used to represent the translation of the axis. The coordinates $\mathbf{w}$ and $\mathbf{d}$ are used to locate the *axis of translation* and in turn the *constraint frame*. Other geometric constraints may be represented by the *constraint frame* using a similar development.

The *constraint frame* informs the parameters of hybrid position - force control. Hybrid control allows specifying position or force control along each coordinate axis at the end effector of the robot. Position control is required along the motion degrees of freedom while force control is required along the constrained degrees of freedom for desirable robot interaction.

Inferring the constraint model

Geometric constraints are inferred from a recorded human demonstration. A recording tool is used to record the demonstration. We abstract the demonstration as a 6-degree of freedom rigid trajectory of the tool parameterized by $p \in SE(3)$ space. When the recording tool is attached to the constraint, it slides along the motion degrees of freedom of the constraint.

The type and location of the geometric constraints are inferred using the approach described in earlier chapters. This provides the geometric constraints in the form of equation (3.1). Figure 8.2 shows the orientation of the constraint frame that our approach infers from the identified constraint for each constraint type.

Robot replay against identified constraint

RelaxedIK (58) is a method to generate kinematically feasible joint trajectories that follow a demonstrated path closely in free space. *RelaxedIK* is a purely kinematic method and is only applicable to free space motion replay. It is an optimization based inverse kinematics solver which uses a weighted sum of objectives to solve the feasible inverse kinematics problem:

$$\Theta = \arg\min_{\theta} \ \ \mathbf{f}(\theta) \text{ s.t. } \mathbf{c}_i(\theta) \ \geqslant \ \mathbf{0}, \ \ \mathbf{c}_e(\theta) \ = \ \mathbf{0} \\ l_i \leqslant \Theta_i \leqslant u_i, \forall i \tag{8.1}$$

where $\mathbf{c}_i(\theta)$ is a set of inequality constraints, $\mathbf{c}_e(\theta)$ is a set of equality constraints, l_i and u_i values define the upper and lower bounds for the robot's joints, and $\mathbf{f}$ is an objective function.

$$\mathbf{f}(\theta) = \sum_{i=1}^{k} w_i f_i(\theta) \tag{8.2}$$

where w_i is a static weight value for each term, and $f_i(\theta)$ is an objective-term function that encodes a single sub-goal. These sub-goals (described in detail in (58)) correspond to avoiding self and known object collisions, kinematic singularities and generating smooth motions while following the demonstrated paths closely.

We extend *RelaxedIK* in this work to create *Constraint Augmented RelaxedIK* (CaRIK) which is capable of following paths against a geometric constraint. When the robot replays the motion recorded by the tool, the end effector takes the place of the measurement tool. Thus the end effector must satisfy the geometric constraint. The forward kinematics $\mathsf{FK}(\theta)$ provide the end effector configuration of the robot:

$$p = \mathsf{FK}(\theta) \tag{8.3}$$

where $\theta \in \mathbb{R}^n$ represents the n joint angles of the robot. The end effector must satisfy the constraint. Combining (3.1) and (8.3):

$$\Phi(\mathsf{FK}(\theta)) = 0 \tag{8.4}$$

The robot must follow the recorded path closely while maintaining engagement with the constraint.

To enforce the end effector configuration to the constraint, equations (3.1) and (8.3) are combined to create a new objective and added to (8.2) with an appropriate

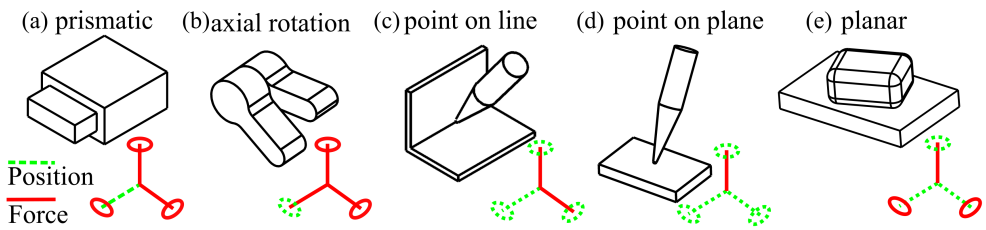

Figure 8.2: **Geometric constraints used in this work. The frame depicts the axes for force control (red solid) and position control (green dotted). Straight lines indicate linear DOF whereas ovals indicate rotational DOF.**

weight w_{gc} to create $\mathbf{f}_{CaRIK}$:

$$\begin{aligned} f_{gc}(\theta) &= \Phi(FK(\theta))^T \Phi(FK(\theta)) \\ \mathbf{f}_{CaRIK}(\theta) &= w_{gc} f_{gc}(\theta) + \mathbf{f}_{RIK}(\theta) \end{aligned} \tag{8.5}$$

where $\mathbf{f}_{RIK}$ represents the objectives defined in equation (8.2).

8.4 Robust constrained motion replay system

The robust replay system records a human demonstration and replays the motions on a robot with minimal user input. The individual actions in the demonstration are identified and converted into robot behaviors. The corresponding joint trajectories and hybrid control parameters are inferred and computed from the demonstration automatically for each behavior. These behaviors are assembled in series to create a robot program called a *Behavior List*.

The demonstration is first segmented by inspecting the forces and moments. The action recognition approach outlined in (69) and described in appendix A is used to identify gripper actions. Thresholding of the reaction forces/moments is used to segment the motions into free space motion and constrained motion. The system requires free space motion between constraint interactions. For example, in Figure 8.1 the three constraint interactions, two *point on plane* and one *point on line* have free space motion in between them. We assume that the free space motions are

mainly used to set up the constrained actions. The constraints are first processed and then the free space motion is adapted accordingly.

The generated behavior list is executed on the robot as follows. A path is planned to a position a few seconds before the constraint interaction occurs. From this point forward, the path is followed by the robot in position control mode (free space motion behavior) until just before it comes in contact with the constraint. The robot end effector is set to have compliance and comes in contact with the constraint (safe contact transition behavior). Once the robot comes in contact with the constraint, the robot follows the path of the recording tool along the constraint in hybrid control mode (constrained motion behavior). The safe contact transition behavior is applied again when retracting from the constraint. The process is repeated for the remainder of the constraints. These behaviors are described in detail as follows:

Free space motion behavior

The *free space motion behavior* corresponds to moving the robot end effector in joint space position control so it follows a prescribed trajectory. The robot joint angle trajectories for this behavior are created using *RelaxedIK*.

Safe contact transition behavior

This behavior corresponds to safely transitioning between free space and constrained motion. This is achieved by following the approach/retract trajectory using *RelaxedIK* and applying compliance control with moderate stiffness.

Transitioning between motion in free-space to motion over a constraint must be performed safely. The robot should approach the constraint from the right direction and come in contact applying safe levels of force. Small errors in the known location of the constraint may cause the robot to come in contact with the constraint before the free space motion trajectory is completed. During this time, if the robot is commanded in position control mode, contact with the constraint will cause the robot to exert large forces against the constraint. To prevent this, the robot is commanded in compliance control with moderate stiffness (end effector simulates

a spring) to simulate position control with a compliance at the end effector. This enables safe contact with the constraint even in the presence of small errors in the system.

Constrained motion behavior

The inferred geometric constraint is used to create robust robot motion against the constraint. The constrained motion behavior is generated by applying *Constraint Augmented RelaxedIK* described in section 8.3 which creates kinematically consistent motion with appropriate hybrid position - force control. The parameters of the hybrid position - force controller is a frame with modes describing the choice of position/compliance or force control along the task space axes (position and orientation axes). From the inferred constraint, our method extracts the *constraint frame* which informs the motion and constrained degrees of freedom of the constraint. This enables the hybrid controller to apply force control along the constrained degrees of freedom and position control along motion degrees of freedom. The *constraint frame* provides sufficient information to configure the hybrid position - force controller for robust interaction with the constraint. This enables the precise replay of large scale trajectories against a constraint and apply recorded forces/moments to maintain contact and robust interaction with the constraint.

In contrast, a pure kinematic approach that does not infer the constraint and set appropriate hybrid control will result in undesirable behavior. Factors such as sensor errors, rigid registration errors between coordinate frames of sensors, ideal interpretations of non-ideal objects in the real world (a white board that has slight imperfections represented as a plane), etc., prevent a pure kinematic approach. Closing a kinematic chain between the robot end effector and a physical constraint in pure position control is not advisable because even small errors may cause undesirable interaction between the robot and the constraint. Our approach solves this problem by applying forces/moments along the rigid directions of the constraint. The constraint in turn enforces the appropriate position/orientation (rather than the position controller) of the robot.

8.5 Experimental Setup

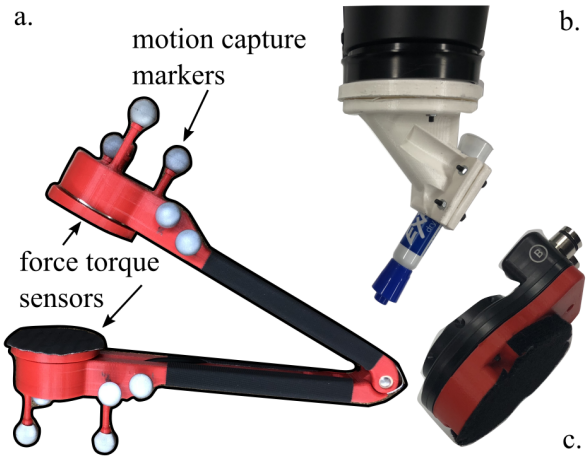

Figure 8.3: **a. Instrumented tongs described in chapter 2 - a hand held device used to record human demonstrations. Motion capture markers measure the location of the tool and force torque sensors measure interaction forces and moments. b. Marker end effector tool c. A whiteboard eraser end effector tool.**

We constructed a Teaching by Demonstration system to evaluate our robust constrained motion replay approach. The system records a human demonstration using instrumented tongs, recognizes actions in the demonstration and creates a robot program through the generation of a behavior list automatically with minimal user intervention.

Equipment

Custom-made instrumented tongs described in chapter 2 are is used to record human demonstrations. Two ATI Mini40 force torque sensors are affixed to the tongs grasping surface to measure the forces during grasp and OptiTrack - Motion Capture Markers are attached to capture the motion of the tongs.

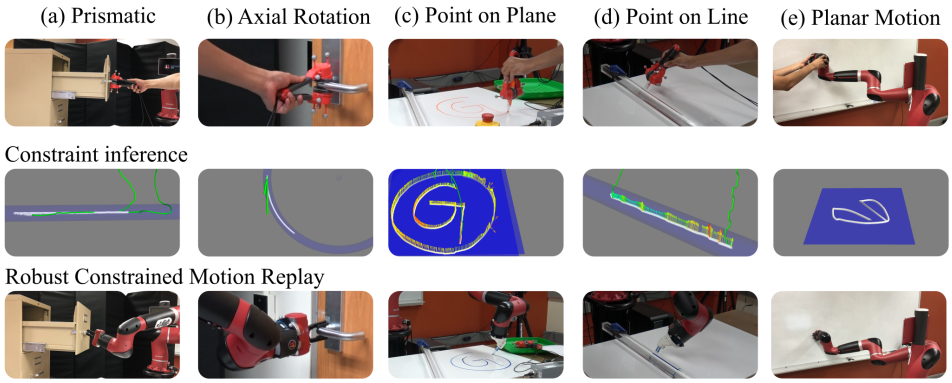

Figure 8.4: **The various constraints evaluated. Row 1: The human demonstration of the constrained task. Row 2: Inference of the constraint in the demonstration. The transparent blue shapes indicate the inferred geometry. The white line represents the samples of the demonstration associated with the constraint. The green line represents the free-space motion. Row 3: The robot executing the task.**

A Rethink Robotics' Sawyer robot was used to execute the recorded tasks. Custom end effector tools were designed to allow the Sawyer robot to grasp and/or interact with different types of constraints. Figure 8.3 shows these tools.

Hybrid position - force control on the Sawyer robot

Motions were programmed using the Sawyer's hybrid position - force controller which works in conjunction with its joint trajectory controller. The Sawyer robot implements its position - force controller through impedance control. The controller enables isolation and decoupling of control between each of the task space degrees of freedom. Either impedance control or force control may be applied along each task space axis (including rotational degrees of freedom). Position control is achieved using impedance control with a high stiffness setting.

Free-space motion is achieved using the Sawyer's joint angle position controller while constrained motion is achieved using the Sawyer's hybrid position - force controller. To ensure safe collisions with the environment, transitions between

free-space and constrained motion are controlled using the impedance controller with the stiffness values set to moderate levels (linear stiffness: 500 N/m, rotational stiffness: 5 Nm/rad).

8.6 Experiments

The focus of the experimental evaluation is to showcase the benefits of inferring geometric constraints in human demonstrations to automatically generate programs that robustly replay trajectories against constraints.

Robust replay of constraints

We demonstrate the constrained replay of motion on the *point on plane*, *point on line*, *axial rotation*, *prismatic motion* and *planar motion* constraints. The supplementary video shows the recorded motion of these constraints and the corresponding replay of the task. Figure 8.4 shows the set of tested constraint types described below.

Pulling out a drawer - Prismatic constraint

The opening of a filing cabinet's drawer is shown in Figure 8.4(a) and is an example of a *prismatic constraint* with one degree of freedom of motion. As described in section 8.3, our method sets the constraint frame's z-axis along the line of motion of the drawer. It configures the hybrid controller to apply force control along all axes except along the Z position axis where position control is applied instead.

Opening a door - Axial rotation constraint

The opening of a door, shown in Figure 8.4(b) is an example of the axial rotation constraint which has one degree of freedom of motion. Our method orients the constraint frame's z-axis along the hinge and positions it at the hinge. Force control is used along all axes except about the Z orientation axis which is set to position

control. The robot end effector rigidly grasps the door and the rotation of the end effector forces the door to open.[2]

Drawing on a whiteboard - Point on plane constraint

Free-form drawing on a whiteboard, shown in Figure 8.4(c), is an example of a *point on plane* constraint which has 5 degrees of freedom of motion. Only the direction perpendicular to the plane is constrained. Our method sets the orientation of the constraint frame with its z-axis parallel to the plane normal. The z-axis direction is set to force control and all other axes are set to position control.

CaRIK closely matches both the position of the tool tip and the orientation of the tool. It is capable of tracing the same shape closely while avoiding collisions with the whiteboard because the inferred constraint is used for collision avoidance. The robot draws the desired shape correctly by applying the recorded forces in the demonstration. Figure 8.5 shows that the orientation of the tongs is similar to the orientation of the pen tool on the robot.

[2]The version of the Sawyer's software (Intera 5.2) used for these experiments only allows setting the orientation of the tool frame and not its position. The default (and only) position of the frame is at the end effector. This limitation causes slip between the door handle and the end effector of the robot causing incomplete opening of the door. To correctly execute the task with this limitation, the X and Y linear stiffnesses are set to moderate stiffness values(linear stiffness: 500N/m, rotational stiffness: 5Nm/rad). With these settings, the robot correctly opens the door.

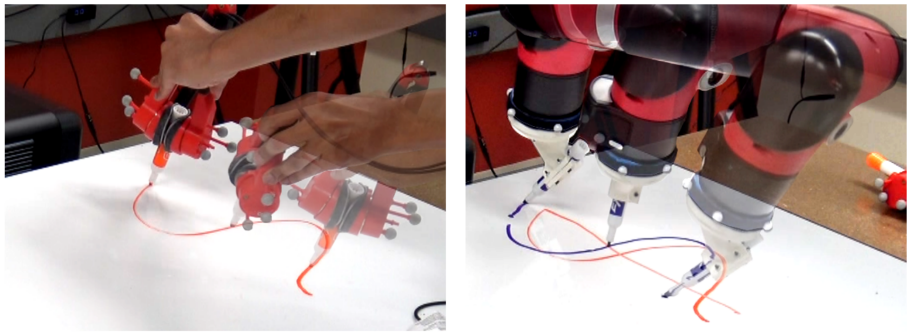

Figure 8.5: **Time lapse of demonstration. Left: The tongs are used to create the S shaped drawing. The orientation of the tongs changes over the trajectory. Right: The robot replays the motion following both position and orientation of the recorded tongs motion.**

Drawing against a straight edge - Point on line constraint

The drawing of a line on a whiteboard using a straight edge, shown in Figure 8.4(d), is an example of a *point on line* constraint. This constraint has 4 degrees of freedom of motion. The linear directions perpendicular to the line are constrained by the straight-edge. Our approach sets the constraint frame orientation with its z - axis parallel to the line. The x and y - axis linear motions are controlled in force control while the remaining axes are controlled in position control. This enables the robot to apply appropriate force to maintain contact with the edge of the line while moving along the line.

Kinesthetic replay of the planar motion constraint task

We use kinesthetic teaching to demonstrate erasing motion against a board to show that our approach is compatible with kinesthetic demonstrations too. As the robot cannot record forces during kinesthetic demonstrations, default force values are provided in software.

The eraser motion against the board, shown in Figure 8.4(e), is an example of the planar motion constraint and has 3 degrees of freedom of movement, two

translational along the plane and one rotational in the plane of the board. Our approach sets the constraint frame orientation with its z - axis perpendicular to the board. It applies force control along the Z position axis and about the X and Y orientation axes and position control for all other degrees of freedom.

Replay of tasks consisting of multiple constraints

Inferring the type and location of the constraint in the demonstration enables replay of complex tasks that contain multiple constraints. A task consisting of multiple constraints (*point on line* and *point on plane*) is shown in figure 8.1. Our approach uses the inferred geometric constraints in the demonstration to create a behavior list with behaviors containing the appropriate joint trajectories and hybrid control parameters with minimal user intervention.

Evaluating robustness

Any recorded demonstration will have some errors. As a result, sufficiently precise kinematic measurements of the demonstration may be intractable. Our method provides robust execution of motion by using the inferred constraint to set the appropriate constraint frame for hybrid position - force control.

Robust execution of the prismatic constraint - a drawer

We perform the same experiment described in section 8.6. To test the robustness of the replayed task, the filing cabinet is physically translated perpendicular to the direction of the prismatic motion of the cabinet drawer while the robot executes the task.

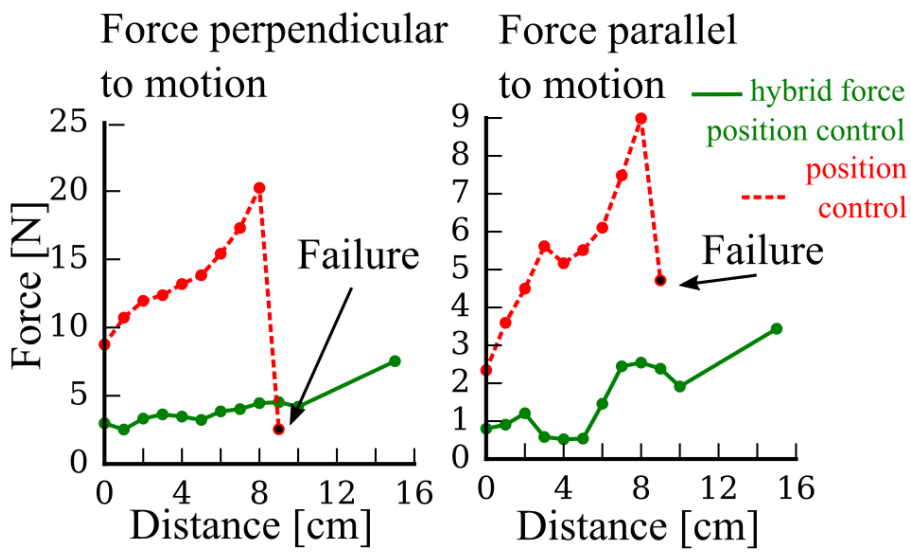

Figure 8.6: **Average forces experienced by the end effector when executing the drawer motion as a function of the perturbation distance of the filing cabinet. Red dashes indicate position control. Green line indicates using the hybrid position - force controller with force and position control applied to the appropriate axes. The position controller experiences significantly more forces than the hybrid position - force controller.**

We quantitatively evaluate how robust the robot is to perturbations when the constraint provides the parameters of the hybrid position - force controller. The hybrid position - force control method is compared to position control (For safe execution on the robot, moderate stiffness(500N/m, 5Nm/rad) for position control is used.)

The filing cabinet is perturbed up to a distance of 16 cm perpendicular to the direction of motion. Figure 8.6 shows the plot of the magnitude of force experienced by the end effector as a function of the perturbation of the constraint. The position controller exerts significantly higher forces than the hybrid position - force controller. After a large amount of perturbation, the position controller fails with the gripper slipping from the handle.

Knowledge of the constraint enables intelligent application of the hybrid position - force controller and facilitates robust constrained motion execution.

Importance of forces for robust execution

Forces are important when executing a task. When drawing against a straight edge as shown in Figure 8.4(d), the demonstrator uses forces to maintain contact between the marker tip and the straight edge allowing the edge to guide the motion of the marker. When the robot executes this task, forces are applied against the two constrained degrees of freedom.

In Figure 8.7, the blue lines indicate the executed robot motion when the applied forces by the robot are the same as those in the demonstration. Even after a 10 degree perturbation, the robot maintains contact with the edge. When the applied forces are removed, we see that the trajectory loses its ability to follow the edge. This deviation is more pronounced with the 10 degree perturbation. Simply replaying the demonstration's forces without setting the appropriate hybrid control frame inferred from the constraint fails to perform the task completely because the forces required to move the robot along the line are different from the forces recorded during the demonstration.

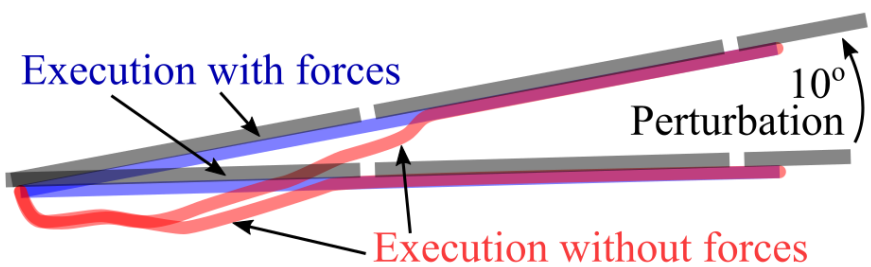

Figure 8.7: **Forces should be applied upwards towards the constraint (gray line) to maintain consistent contact with the constraint. The blue line indicates robot motion when forces are replayed. The red line indicates robot motion when forces are absent. In the absence of forces the robot deviates from the prescribed path. Even after the straight edge is perturbed by 10 degrees the robot still manages to follow the edge when forces are applied.**

8.7 Conclusion

This chapter introduces an approach to robustly replay tasks consisting of physical constraints. It uses the inferred geometric constraint in the task to determine constrained and unconstrained degrees of freedom of the constraint. This information is used to configure the robot's hybrid position - force controller and generate feasible kinematic trajectories that follow paths closely along the constraint. The robust replay system takes a human demonstration as input and replays the task robustly with minimal user intervention.

The developed system is a proof of concept and preliminary step towards an overall robust solution to replaying a demonstration and more importantly the demonstrator's intent. Our solution is limited to standard geometric constraints that can be described with the *constraint frame* concept. It also does not actively adapt to large changes in geometry or use any visual information to identify the location of objects in the demonstration to accommodate for changes in the location of these objects between the time of the demonstration and the time of execution on the robot. Another limitation is that only interactions that can be represented through standard geometric constraints are considered.

The system requires an evaluation from a user's perspective, measuring how the human demonstrator affects overall performance. Developing a system with specific performance metrics focused on the human's preferences is important for successful adaptation into mainstream robotic systems.

9 Potential avenues for future work

Substantial progress must be made before *versatile* robot arms are introduced in domestic environments. The three major hurdles that need to be addressed are, a) reducing the robot's cost without compromising performance, b) developing methods for users to communicate tasks to robots, and c) performing dexterous physical manipulations in cluttered, evolving and unstructured environments.

The work in this dissertation focuses on (b), inferring geometric constraints in human demonstrations to facilitate better communication between humans and robots. Translating these inferred geometric constraints into robot programs has been described in chapter 8 but it is a preliminary, proof-of-concept system. The limitations of the proof-of-concept system highlight many potential avenues for future work:

- Integrate visualization approaches to communicate the inferred robot task.
- Improved interfaces for novice users to manipulate the inferred task and check if the inferred task is correct.
- Automatic adaptation of the robot task to a changing environment.
- Inferring collaborative robot tasks where a human may interact with the robot when the robot performs the task.

The methods described in this work only use kinematic and wrench trajectories. Integrating computer vision by including raw image input would improve recognition accuracy and bring out other important aspects of the demonstration (42). For example, if the demonstration involves pulling out a drawer, the video of the motion of the drawer may provide information such as grasp location with respect to the geometry of the drawer. Computer vision may be helpful to track objects and landmarks in the workspace. This would eliminate the requirement of a new demonstration each time the workspace configuration changes. The trajectories

in the demonstration cannot be used directly but need to be retargetted to a new configuration or a suitable planning approach used instead.

The inferred actions may not be correct and motivates error checking. The order of actions and the individual properties of actions may not be compatible with each other. Checking for these problems would lend to a more robust system. Methods to visualize these errors in a user interpretable format and get appropriate user feedback is also essential. Advances in augmented reality make it possible to superimpose the robot's policy onto the environment; however it is still unclear how these policies should be presented to the user.

Actions inferred from multiple demonstrations can be consolidated to improve task inference instead of just using a single demonstration. If the actions in a task are not ordered and consist of logic then this logic may be deduced from multiple demonstrations. Deducing the task plan from multiple demonstrations would enable robots to reason about the demonstration. For example, consider a user demonstrating a task consisting of opening the microwave and setting the time; multiple demonstrations of different objects could teach the robot which objects require what time to set. Inferring actions from raw demonstrations provide a layer of semantic abstraction which task planners can use to deduce robot task policies.

The developed recognition and inference methods may have other applications. For instance, if the constraint inference approach was online then the inferred constraint could be used in shared control during teleoperation. The robot could infer the constraint interaction and adapt its control approach to match the constraint. Sometimes two different actions may have a very similar robot policy and if one action is already known, the same skills could be transferred to the new policy. This transfer of skill could be made possible by inferring the action and finding similarities between the inferred action and an action recalled from prior experience. For example, if a robot is trained to insert an electrical outlet plug then it could apply the same policy to inserting a USB stick. The tactile action recognition approach described in chapter A could be refactored to detect similar actions and transfer learnt policies. Online tactile action recognition could be used in shared control in a similar way.

Currently, there is a huge push in learning tasks in an end-to-end fashion, from input to final robot execution(40). Integrating physics based models along with more general machine learning models may improve overall performance. For example, if a deep reinforcement learning algorithm was used to learn how to wipe a table, it may be interesting to see if integrating the model of a plane and a human demonstration of the task in the neural network would improve robot learning.

10 Conclusion

Forces and moments are generated when interacting with the environment. Measuring these forces and moments allows recognizing and inferring physical actions in human demonstrations.

This dissertation introduces methods to infer geometric constraints from human demonstrations consisting of position, orientation, force and moment signals. It shows that including force and moment measurements improve constraint inference when compared to competing kinematic approaches. The recognition and inference of these constraints enables robots to robustly interact with the environment.

Robot arms seem to be infinitely useful but it is hard to see how everyone can use them. If we make it easier for everyone to program robots, people may just find unique and interesting ways to use them!

A Recognizing tactile actions

Recognizing the type and timing of tactile force actions is useful in a range of robotics applications. Some actions are against geometric constraints. Other actions are more fine grained and do not involve general constrained motion. For example, actions such as sudden contact, friction and slip. Identifying force actions exerted on objects such as gripping, snapping or releasing is particularly useful because it allows reasoning about the task at a semantic level of abstraction. Performing such action recognition from force measurements is attractive because forces are often necessary to differentiate between actions in circumstances where other sources of information are insufficient.

Recognition of force actions is commonly done in an *ad hoc* manner. Specific recognition algorithms can be devised for specific actions, however, such solutions do not generalize. Therefore, considerable effort may be required as the set of different actions to be recognized grows and more complex actions are considered. Our goal is to provide a force action recognition method that easily generalizes and scales to many diverse action types.

We introduce an approach for performing recognition of tactile actions based on measurement of forces. Our method is based on machine learning and can be trained to classify a wide range of action types, providing not only the type of the recognized action, but also its timing. The key idea is to transform the measured force signals into *feature vectors* by using the Continuous Wavelet Transform (CWT)(44). The CWT provides a concise description of the temporal neighborhood for every sample in a signal. Given a set of training motions recorded as force signals, along with action labels for each time step, our approach constructs a classifier that can label each time step of new motions.

We demonstrate and evaluate our approach in a system that records human tactile actions using an instrumented set of tongs. Considering only the force measurements from the tongs, our system is able to correctly label tactile actions such as inserting a cartridge, having an object slip from a grasp, and twisting a marker cap. It is sufficiently sensitive to distinguish grasping a soft object from a

hard one. The system requires no per-action programming: it can be trained on new actions by simply providing a few labeled examples of the action.

The contributions of this chapter include:

- We provide a technique using the Continuous Wavelet Transform with the Complex Morlet wavelet that encodes force measurements as features amenable to a variety of machine learning methods.
- We show the utility of this CWT technique in an approach that applies supervised machine learning to action recognition.
- We show that tactile actions can be recognized from force measurements alone.

A.1 Background

While others have developed action recognition systems using computer vision approaches (55; 29), we focus on action recognition from forces. Many task specific solutions to the problem of recognizing tactile actions from forces have been considered in the literature. Examples include contact-state detection for the peg in the hole problem (28), slip detection(46; 68) and snap detection (60). However, each of these solutions involves specialized algorithms and therefore are difficult to generalize. In contrast, our approach can be trained to identify many different kinds of actions.

The tactile action recognition problem demands a representation of the force signals that provides both a compact expression of vibration and transient phenomena but also the ability to localize temporal events. The CWT meets these demands. It has many applications in image and signal processing such as clustering (20), detecting seismic events (76) and analyzing EEG signals (62). We apply the CWT technique to tactile action recognition.

An alternative is to completely forgo useful signal characterization and use the raw signal directly in the recognition process. Such approaches, for example Con-

volution Neural Networks(C-NN)(38), typically require many training examples, and are therefore inappropriate for our applications.

A.2 Key Idea

The central problem in tactile action recognition is to represent the measured force signal in a manner that is amenable to classification techniques. Our key idea is to use the *continuous wavelet transform* (CWT) to transform a scalar valued force signal into a vector-valued one where each time step is represented by a *feature vector*. Vector valued input signals are similarly transformed into concatenated wider feature vectors.

The CWT, with appropriate choice of wavelet (we use the Complex Morlet Wavelet), effectively summarizes the vibration and transient signal phenomena around each time step in a concise descriptor of the time step. Unlike other time-frequency characterization methods like the short time fourier transform(STFT), the CWT avoids windowing effects. We sample the wavelet at k scales, so that each time step of a scalar force signal is represented by a descriptor of length k. A vector valued force signal with vector length m has each time step represented by a feature vector of length $m * k$. We perform scaling and transformations to the CWT so the feature representation is suitable for machine learning algorithms.

After wavelet transformation, we treat each time step independently. In a training signal of length N, each time step has an associated action label. Hence we build a set of N pairs of $k * m$ length descriptor vectors and labels for training. Subsequent test signal labeling is done by applying the trained classifier to each time step of the test signal independently.

A.3 Methods

The purpose of this work is to detect the type and timing of force actions. We use the CWT of the measured force signal using the complex morlet wavelet, as a set of features for representing force actions. The complex morlet wavelet extracts

vibrational content from a signal. The intuition behind choosing the complex morlet wavelet is that many physical interactions between contacting objects are characterized by the frequency content of the signal. For example, contact with a hard object would result in a rapidly changing signal containing significant high frequency content while interactions with softer and/or heavier objects might result in greater low frequency content. Further, most physical interactions can be characterized by a linear system. Since higher order linear systems vibrate, and tend to amplify certain frequencies but not others, like the resonant frequency in second order systems, it can be seen that most interactions involve vibrational effects.

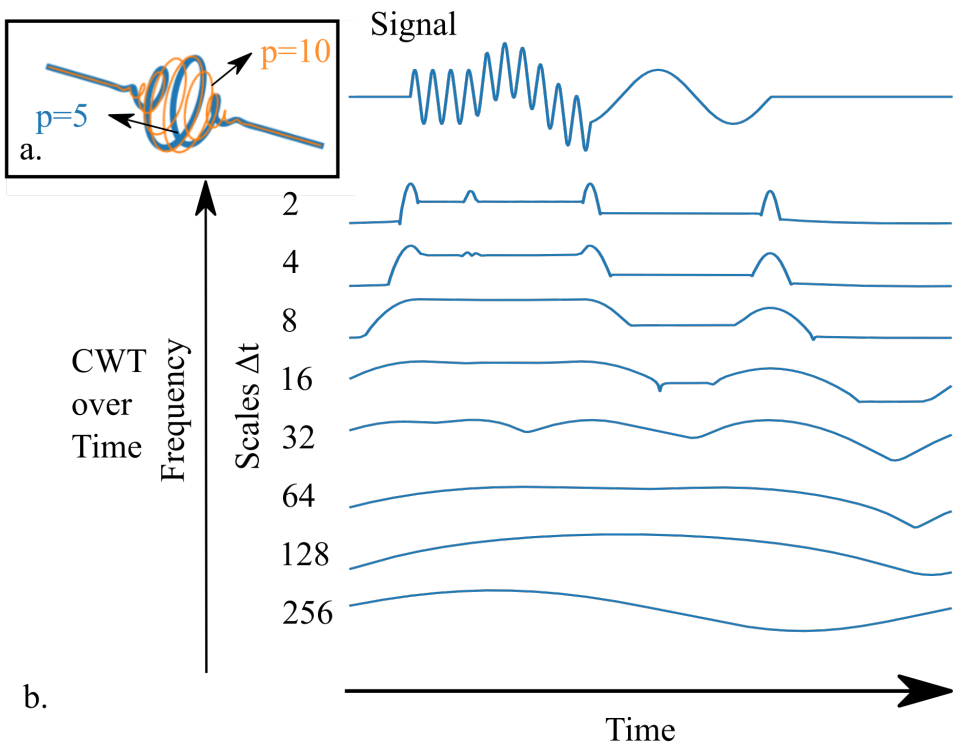

Figure A.1: **a. Complex Morlet Wavelet - Complex sinusoid windowed by a Gaussian at different p values. b. Signals represent the magnitude of the CWT coefficients over time on the test signal.**

Continuous Wavelet Transform with the Complex Morlet Wavelet

The continuous wavelet transform of a signal can be understood as a transform of a signal which compares the shifted and scaled form of a prototype kernel function sometimes referred to as the analyzing wavelet Ψ. This comparison is achieved by performing a convolution of the measured time signal with different scaled versions of the analyzing wavelet. We can define the kernel wavelet at different scaled versions of the analyzing wavelet as:

$$\Psi_{(a)} = \Psi\left(\frac{t}{a}\right) \tag{A.1}$$

where a is a scaling factor. Hence the CWT of a square integrable signal $f(t)$ using an analyzing wavelet Ψ is defined as:

$$X_{(a,b)} = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} \Psi^* \left(\frac{t-b}{a}\right) f(t)\, dt : a \neq 0 \tag{A.2}$$

where b is time shift parameter. Equation A.2 is a convolution of scaled and translated version of the analyzing wavelet. We must still define the analyzing wavelet function. We use the Complex Morlet Wavelet (30) given by the following equation:

$$\Psi t = \pi^{-1/4} e^{ipt} e^{\frac{-t^2}{2}} \tag{A.3}$$

where the parameter p indicates the trade off between frequency and time resolution. On closer inspection we see that the Complex Morlet wavelet is a complex sine wave windowed with a Gaussian kernel centered at $t = 0$. Fig. A.1a. shows the Morlet wavelet at different values of p.

Computing the CWT

We perform the CWT of a discrete signal f of length N in the Fourier space using the discrete Fourier transform. From the convolution theorem the CWT is given by:

$$X_{a,b} = \sum_{k=1}^{N-1} \bar{f}_k \bar{\Psi}^* (a\omega_k) e^{i\omega_k b \Delta t} \tag{A.4}$$

where $\bar{f}$ is the discrete Fourier transform given by:

$$\bar{f}_k = \frac{1}{N} \sum_{n=0}^{N-1} f_n e^{\frac{-2\pi i k n}{N}} \tag{A.5}$$

where the angular frequency ω is defined as:

$$\omega_k = \begin{cases} \frac{2\pi k}{N\Delta t} & : k \leqslant \frac{N}{2} \\ \frac{-2\pi k}{N\Delta t} & : k > \frac{N}{2} \end{cases} \tag{A.6}$$

Finally the CWT provides a coefficient for every time shift b and frequency(scale) a of the wavelet function. If the analyzing wavelet is a complex valued function then the CWT of that wavelet will also be complex valued. We can determine a power and phase spectrum from these coefficients given by:

$$M_{a,b} = \left\| X_{(a,b)} \right\| \tag{A.7}$$

$$\phi_{a,b} = \arg\left(X_{(a,b)}\right) \tag{A.8}$$

Fig. A.1b. provides the magnitudes of the CWT with the Complex Morlet wavelet on a test signal composed of two different frequencies.

Using the Complex Morlet Wavelet coefficients as Features for Machine Learning

The use of the Complex Morlet Wavelet (as the analyzing wavelet function shown in equation A.3) require the selection of parameter, p, which controls the trade-off between frequency resolution and time resolution. For our analysis we chose a value of $p = 5$ balancing the need to (approximately) meet the admissibility criteria

(73) of the wavelet while maintaining sufficient temporal resolution. To create a usable representation of the coefficients, we perform additional transformations:

- Use a logarithmic scale distribution: A linear scale distribution would create too many features and is highly redundant.
- Decompose the CWT complex coefficients into real-valued magnitude and phase components: The magnitude provides the intensity of force, while phase provides a measure of duration.
- Logarithmically scale the magnitudes: All the features have similar scales.
- Discard low frequency scales: Very low frequency scales do not characterize local behavior.
- Stack the CWT features from multiple channels in a signal: Force signals are usually vector quantities and contain multiple channels.

The CWT is a highly redundant transform if one takes all possible scales into account. The scale distribution is given by:

$$a_j = a_0 2^{j\Delta j}; where j = 0, 1, ...J \tag{A.9}$$

$$J = \Delta j^{-1} \log_2 \left(N \frac{\Delta t}{a_0} \right) \tag{A.10}$$

For our experiments we chose $\Delta j = 1$ and $a_0 = 2$. We found that this value of j was sufficient for sampling scales, albeit smaller values will give richer CWT coefficients. The DFT algorithm we use introduces windowing artifacts at the beginning and the end of the signal. It is useful to pad the signal at each end. The amount of padding depends on the width of the lowest frequency scale used.

Splitting the complex coefficients into magnitudes and phases allows resolving between actions of the same magnitude but with different phase. Finally standardizing the magnitude by computing their natural logarithm is useful as signal amplitudes can vary by orders of magnitude. In particular, high frequency signals

generally have much lower amplitudes when compared to low frequency ones but are critical in identifying actions such as contacts with objects.

In cases where we have more than one channel in a signal, we stack up the coefficients to create a combined representation. If we have q channels in a signal and use k scales then the features X_b representing time instance b of the signal would be:

$$X_b = \begin{bmatrix} M_{b1}^T & M_{b2}^T & \dots & M_{bq}^T & \phi_{b1}^T & \phi_{b2}^T & \dots & \phi_{bq}^T \end{bmatrix}^T \tag{A.11}$$

Once we compute this, each sample b of the signal has a *feature vector* X_b. For a signal of length N there will be N such feature vectors which can be expressed as examples in an unlabeled data set. Note that these examples have order as they correspond to samples in the signal. If we were to provide labels to each of these examples then we could treat these as members of a labeled data set. Since the examples are associated with a sample time, the labels for these examples are also associated with that same sample time. Hence we can define a new time series $L(t)$ which evaluates to the label names corresponding to the signal at time t.

A Sample Scenario for Supervised Learning

Consider a sample data set consisting of k signals containing multiple channels with different portions of their signals labeled. We can consider a training set T_r with $k-1$ signals and test set T_s with the last signal k. The supervised learning procedure would be as follows:

1. For each signal in T_r, compute the CWT features to create their associated *examples*. An *example* is a pair, consisting of a feature vector and an associated label name, corresponding to a specific sample time.

2. Use these *examples* to train a model.

3. For signal T_s compute the CWT feature vectors at each sample time.

4. Predict labels at each sample instance of the test signal T_s to get a label time series L.

5. Filter the time series L.

Choice of Supervised Learning Algorithm

Since we have abstracted the signal representation into a set of features at every time instance, it is possible to use almost any conventional classifier for the supervised learning task. In practice we see that the Neural Network gives us the best execution time and relatively good accuracy when compared to other methods we tested including, Naive Bayes, Decision Trees, SVMs and K-Nearest Neighbor. Note that feature representation affords us the flexibility to choose many algorithms. For the experimental evaluations performed here, we used a Neural Network from Scikit-learn (53) with default options which contains one hidden layer with 100 neurons.

Filtering Predicted Labels

The raw results from classification when arranged in time contain chattering. Since the predicted label names are in the form of a time series, we can filter them. Some filtering methods are:

- Implement a probability thresholding filter which removes labels classified with a probability less than a particular threshold and sets them to unclassified.
- Define a condition that, if an action is predicted to occur for less time than a factor of the minimum duration of that particular action in the signals used for training, it is labeled as unclassified. We refer to this as the Drop Heuristic.
- Assign discrete numbers to each type of label and filter these using a median filter.

Why not other Wavelets?

Using the Morlet wavelet allows encoding both phase and magnitude information. If we were to use the real Morlet wavelet we would not be able to distinguish between phase and magnitude information. This is a particularly bad choice since this wavelet would phase in and phase out, changing the magnitude of the wavelet in a signal, even for one whose magnitude is constant. Alternatively, if we were to use the Haar wavelet, we would not be able to extract phase information.

To demonstrate this we compute the CWT features for a trapezoidal test signal using:

- Complex Morlet Wavelet with magnitude and phase
- Complex Morlet Wavelet with only magnitude
- Derivative of Gaussian Wavelet (DOG)
- Haar Wavelet (73)

Fig. A.2. shows this comparison. We see that the Complex Morlet Wavelet with both magnitude and phase, is the only wavelet that can distinguish between increasing and decreasing portions of the test signal.

A.4 Experimental Setup for Evaluating Methodology

Data Collection System

We developed an object manipulation device to record user performed tasks. While for this work, we do not use the kinematic (motions) information, the device is fully capable of recording its spatial location and the grasp configuration through an HTC Vive VR controller and an encoder. Force interactions are recorded using Optoforce force sensors. Data collection and synchronization was performed using ROS. The body was designed and 3D printed in-house. This is shown in Fig. A.3. We record 3 axis force values from each sensor. We convert these 6 signals into 4

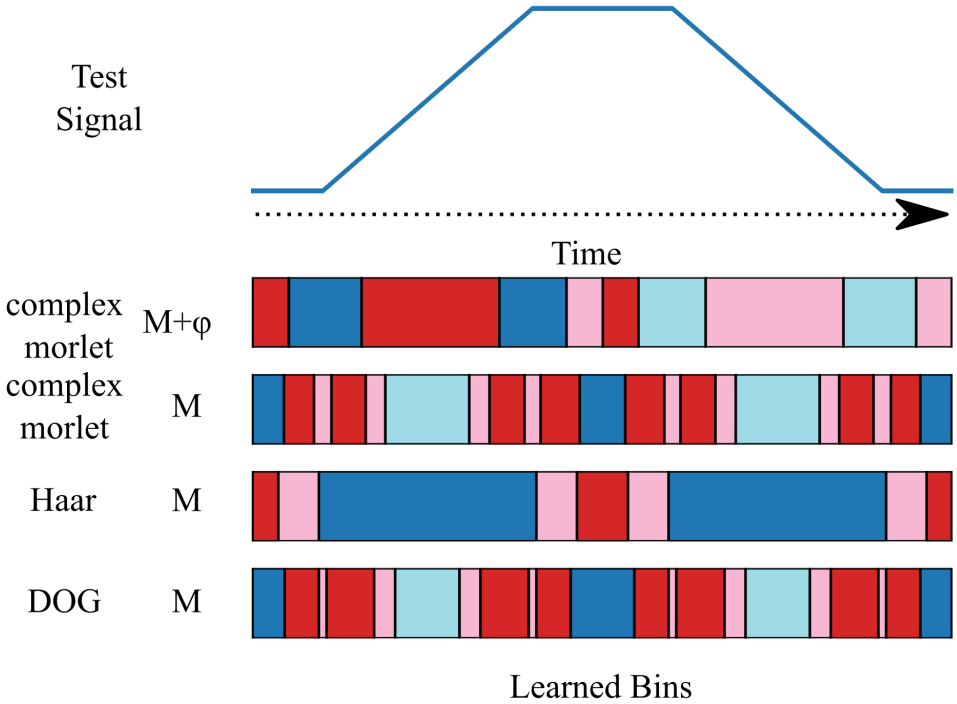

Figure A.2: **Clustering CWT coefficients using 4 clusters with the k-means algorithm over the Trapezoidal test signal. Row 1 shows the complex morlet wavelet capable of distinguishing between rising and falling edges while other wavelets cannot. Colors correspond to different clusters. M - Magnitude, φ - Phase**

signals by computing the normal and tangential components of the native force signals. This transforms the signals into a form that is approximately invariant to forces in the plane tangential to the force sensors.

Experimental Evaluation Methodology

A user performs specific tasks which are recorded by the instrumented device described in the previous section. For clarity, we will define three words and their relationships:

- Force Action: Something that can be represented by a single action which describes what is happening, like a verb.

- Demonstration: A single recording of a set of signals while performing a series of force actions.
- Task: A specific choreography of a demonstration.

A task is the overall objective of the demonstration. For example, a task could be the process of picking up a bottle and placing it in a different location. A demonstration is an instance of a task. Each task can be broken up into different force actions like grab, still or release. We can think of force actions as primitives to create tasks.

The data of one demonstration is essentially a time series of multiple signals and each time instance is labelled with a force action label. In our experiments, a human labels sections of the time series. Each time step is labeled with a force action in each demonstration. We define a set of force action labels for each task. In each experiment two coders (in consensus) define which locations in the force signal correspond to force actions but independently code the demonstrations allotted to them. Each task is recorded with a frame by frame, time synchronized video

Figure A.3: **Instrumented tongs(earlier version to the tongs described in chapter 2) used to manipulate objects.**

recordings of the task. The human coders code the task demonstration by looking at the force signal, using the video recording as an aid.

Sample Demonstration

Fig. A.4 shows the force signals for a sample demonstration. The choreography of the force actions that occur during a demonstration of the task is also shown. In this experiment a lego block is picked up from a table and fastened on to another lego, then it is removed and placed back on the table. From a set of demonstrations and human coded labels we learn a model which is capable of classifying each point in time with a label. During transitions between force actions, the classified labels chatter between two different label names. We mitigate this by applying the filtering techniques described in A.3. We used a median filter of window size 10, a probability threshold of 80% and a drop factor of 1/8.

Experimental Evaluation Metrics

In evaluating the model we made use of the following metrics:

Qualitative Metric TIMELINES

In each timeline plot pair, the upper plot displays the human coded labels while the lower plot displays the predicted label. If the timelines in the pairs (ie. upper and lower plots) match then the trained classifier has predicted every label correctly. A qualitative measure of the models performance can be obtained by examining the alignment of the upper (human coded labels) and lower (predicted labels) timelines. To generate the timelines we train and test on demonstrations in a leave one out type cross validation setup. In other words, for a particular task, we test on one demonstration and train the model using the others. In this fashion, we test on all demonstrations and plot the results in the form of timelines of predicted labels. The colors correspond to the individual action labels. The white color corresponds to unlabeled examples. If they occur in the human coded time line then these time

samples are dropped during training. If the algorithm has insufficient confidence about the label then it labels them white.

Quantitative Metric

We compute a confusion matrix which provides information on which label is misclassified with what other label. To build this matrix, we use the same cross validation technique described above unless stated otherwise.

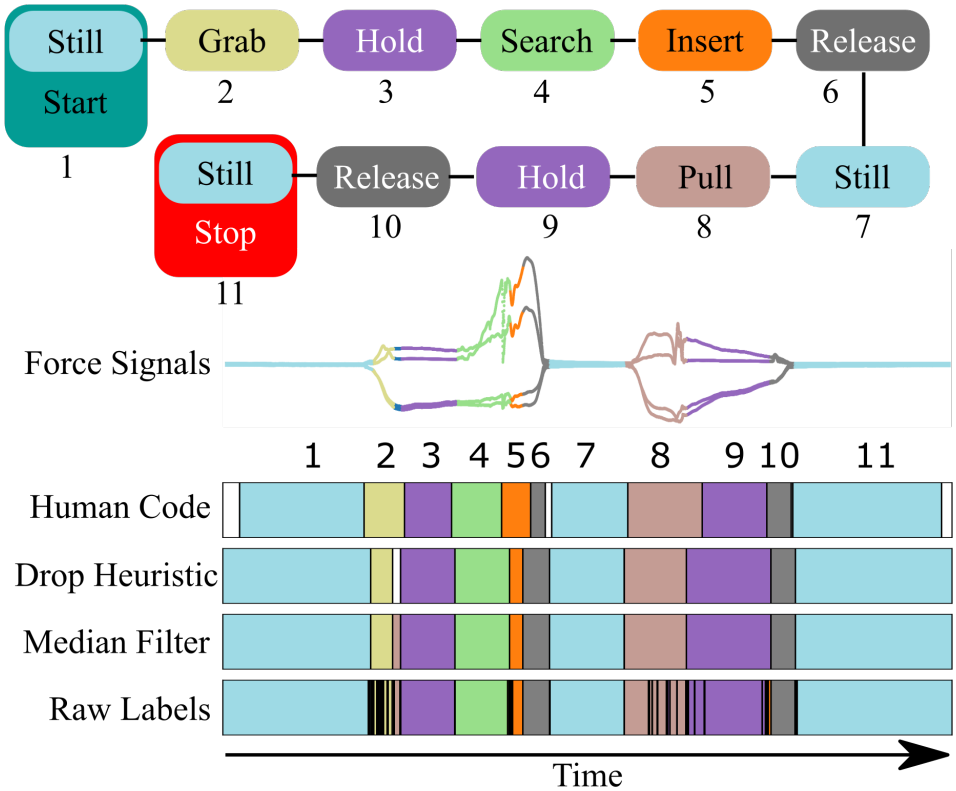

Figure A.4: **Top - Semantic timeline of Lego assembly and disassembly task with the corresponding labels numbered according to their occurrence in the human coded timeline. Bottom - Human Code corresponds to the labels given by a human coder, Drop Heuristic corresponds to applying the Drop Heuristic and the Median Filter. Median filter corresponds to only applying the median filter to the classified labels. The white spaces correspond to unlabeled samples.**

Since these are force signals and the location of a force action is not well defined it is difficult to come up with a fair quantitative metric to describe the algorithms accuracy. At the interfaces between two different force actions, it is very difficult for the human coder to consistently code the force action. Since these are signals, the adjacent force actions bleed into each other. Some force actions like still, dominate others. Hence, mere classification accuracy is not a good way of reporting the algorithm's behavior.

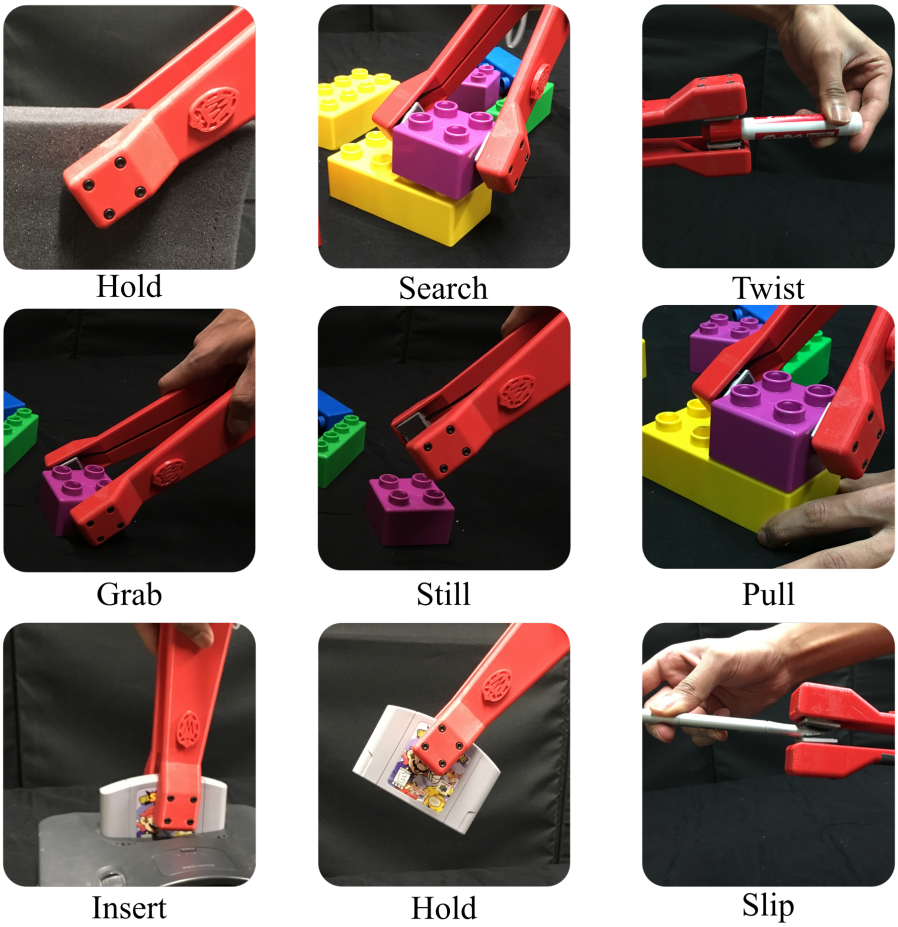

Figure A.5: **Force actions and their corresponding object manipulations**

A.5 Results

In this section we validate our feature representation method by performing two sets of experiments:

- Detecting force actions in different demonstrations of the same task. Training set and test set include demonstrations of the same type of task.
- Detecting force actions in demonstrations of a different task. Training set includes demonstrations from one type of task. Test set includes demonstrations from a different task.

Detecting force actions in similar tasks

We ran multiple experiments to see if we could identify the same force action in different demonstrations of the same task. Fig. A.5 shows a glossary of actions with their label names. We chose a few scenarios which were found in robotics literature, namely:

- Peg in the Hole problem
- Slip and Friction detection
- Distinguishing between hard and soft objects

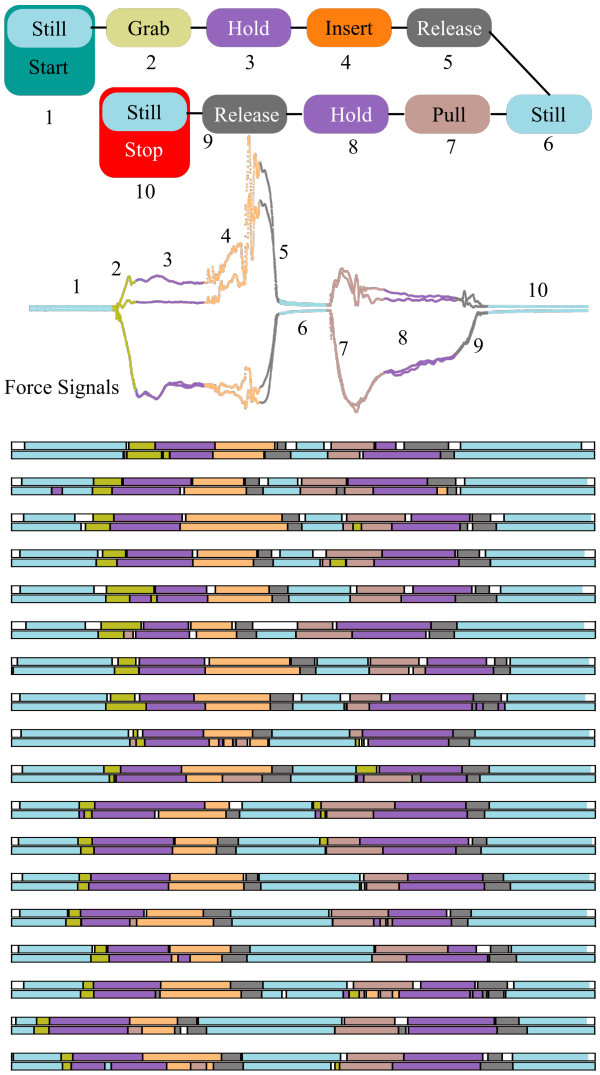

Figure A.6: **Summary of the Nintendo 64® trials. Top - Timeline of force actions in numerical order as they occur. Bottom - Summary of all the demonstrations' predicted labels and human coded labels for each demonstration of the Nintendo 64 task. In each pair, the upper color coded timeline bar corresponds to the human coded labels. The lower bar corresponds to the predicted labels. Notice that the predicted force actions align with the human coded force actions. TIMELINES MUST BE VIEWED AS PAIRS AND IN COLOR, see Sec. A.4**

Peg in the Hole Problem - Inserting a Nintendo 64® Cartridge

In this task we pick up a Nintendo 64® cartridge and insert it into the console. The cartridge clicks into place during insertion. Fig. A.6 shows the choreography of the task along with the force signals and associated force action labels. Fig. A.6 displays an experimental evaluation of the demonstrations. We see that the pull action is mis-classified frequently. On close inspection we see that this is not surprising as it might be due to coding errors. The human coder has difficulty coding the pull and grab action consistently between demonstrations. Although the coder can observe the force signals along with the video recording of the task, the video aid is not that helpful as the cartridge is enclosed in the game console and occluded during insertion. We can also see that the pull action is confused with the grab action. While the human coder does not consider a grab before a pull action, the algorithm occasionally finds a grab action. Intuitively, the cartridge must be grabbed before it can be pulled out of the console.

Peg in the Hole Problem - LEGO® Block Assembly

We evaluated the algorithm by examining LEGO® block assembly. This task is characteristic of tasks found in various assembly tasks that involve both interference and transition fits between parts. A defining characteristic of LEGO® block assembly is the need to fumble for the proper mating configuration between parts, referred to here as the *search* force action. From Fig. A.7 we see that the algorithm consistently identifies the search actions and distinguishes this from the hold and insert actions.

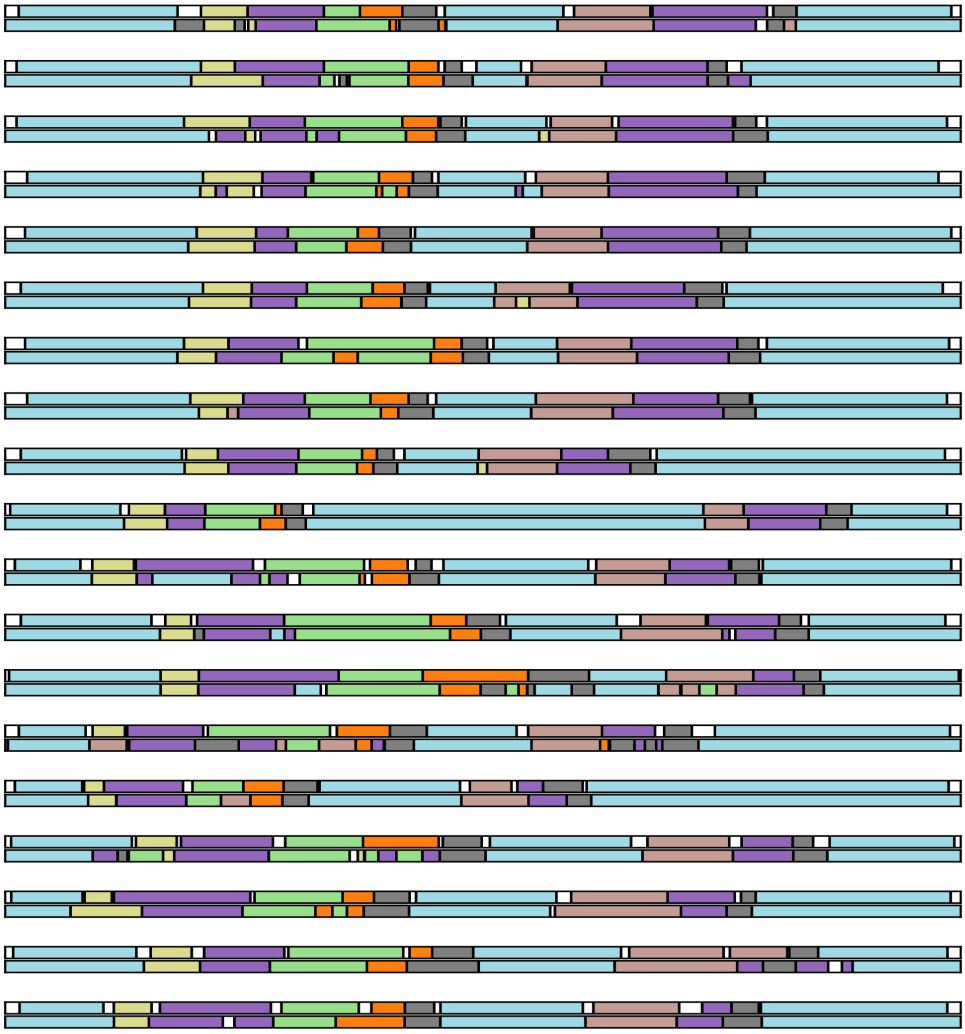

Figure A.7: **Summary of the LEGO® trials. Please refer to Fig. A.4 for the label names. In each pair of timelines, the upper timeline corresponds to the human coded labels and the lower timeline corresponds to the predicted labels. Notice that in almost every trial the order and locations of the force actions are predicted correctly. TIMELINES MUST BE VIEWED AS PAIRS AND IN COLOR, see Sec. A.4**

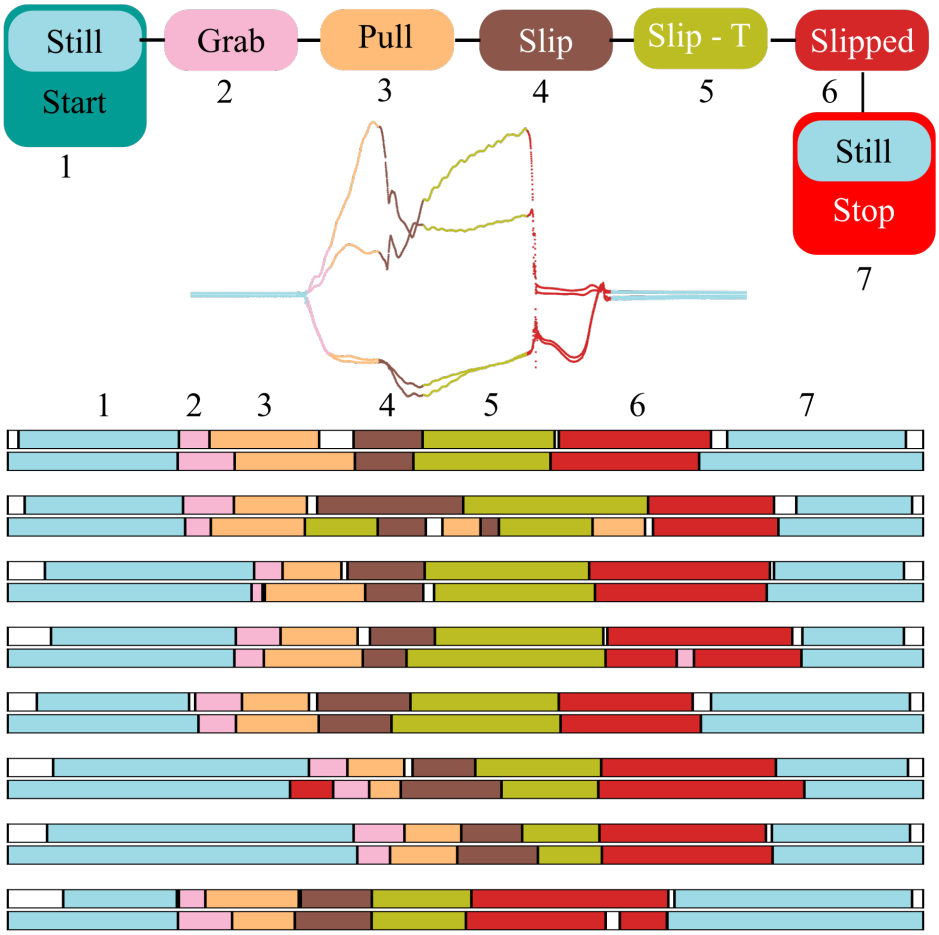

Figure A.8: **Slip detection task with stylus. The predicted labels clearly align with the human coded labels. The predicted labels have very close arrangements when compared to the human coded labels. TIMELINES MUST BE VIEWED AS PAIRS AND IN COLOR, see Sec. A.4**

Slip and Friction Detection - Detecting Slip During Grasp

In this scenario, we attempt to recognize different force actions that occur during slip. We simulate this by grasping a pen between the force sensors and pulling until the pen is released from its grasp. In this case, we chose an peculiar oblong object (i.e. a writing stylus) to allow examination of the algorithm's performance at various

stages of slip. The variable diameter of the stylus causes changes in the friction force. The labels describing the slipping included (1) *slip*, indicating onset of motion between the stylus and the tongs, (2) *slip transition* indicating motion between the tongs and the stylus. The end of the slip action was labeled as *slipped* indicating the completion of the slip action, identified during the experiment through the clap of the instrumented tongs. Fig. A.8 shows the choreography of the action. We see good time alignment and order of actions for most demonstrations.

Slip and Friction Detection - Detecting Friction Force

To evaluate the algorithm's ability to detect friction actions, which contain significant high frequency vibrations, we used our instrumented tongs to apply torque to a dry erase marker cap. The cap has a significant amount of static friction which results in sticky, discontinuous motion when twisted. Fig. A.9 shows the action choreography and the results. The technique does well but has difficulty during the hold action. This may be due in part to the difficulty that the human coders have in understanding and identifying the force actions. For example, holding the marker cap after the first *twist* action can be overlooked while coding.

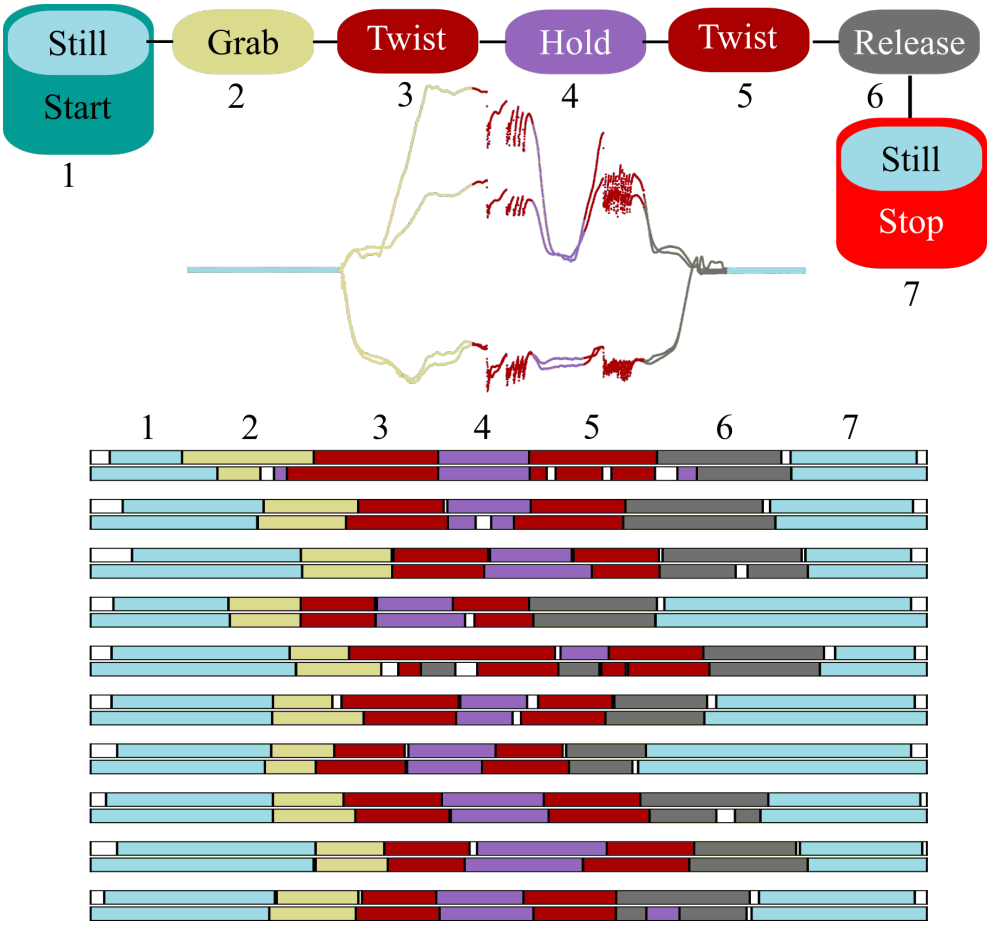

Figure A.9: **Friction detection task with the dry erase Expo® marker. The predicted labels clearly align with the human coded labels. The predicted labels have very close arrangements when compared to the human coded labels.**

Distinguishing between Hard and Soft objects

In this experiment we evaluate the algorithm's ability to detect objects of different hardness. A foam block and a Nintendo 64® cartridge were used as soft and hard objects respectively. We trained and tested using concatenated datasets of the foam block and the Nintendo 64® cartridge. The foam block task was simple pickup and drop of the foam block. Fig. A.10 shows the result in a confusion matrix.

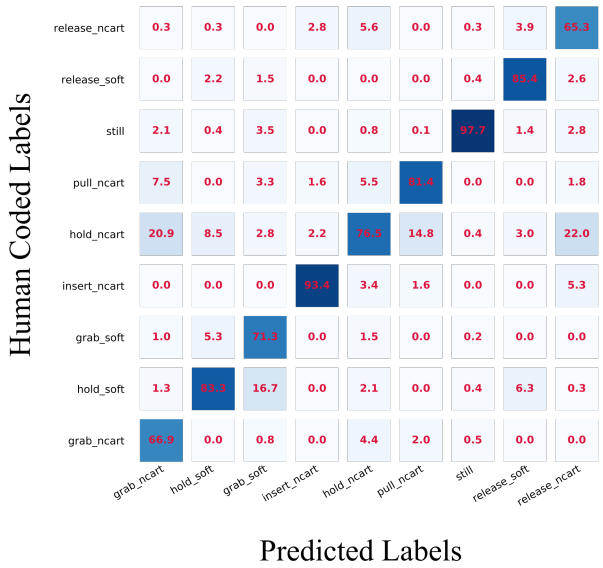

Figure A.10: **Confusion Matrix for training set containing both hard and soft object manipulations. Larger numbers along the diagonal correspond to better results. The soft object was the foam block (labelled with suffix _soft) while the hard object was the Nintendo 64 Cartridge (labelled with suffix _ncart). The algorithm can clearly distinguish between actions performed on the hard object compared to actions performed on the soft object.**

Detecting similar force actions in different tasks

In the previous section, we show examples of our recognition algorithm where the training and test sets use demonstrations from the same task. The purpose of this section is to demonstrate how well our algorithm can recognize learned force actions in a new task. In this experiment the training set contains demonstrations from the LEGO® assembly task. The test set contains examples from the Nintendo 64® cartridge insertion task. To evaluate our results we make use of a confusion matrix shown in Fig. A.11 described in A.4.

Note that the human coder does not code a *search* label for the Nintendo 64® task. We see that the algorithm misclassifies the *insert* label with the *search* label but classifies all other labels correctly. The *search* label corresponds to the fumbling

of the LEGO® block. The cartridge was not always inserted smoothly into the Nintendo 64® and sometimes the user tried to feel for the opening. The human coder coded all these interactions along with the insertion of the cartridge with an *insert* label. It is reasonable for the algorithm to misclassify some of the *insert* labels as *search* labels. For the other labels, the algorithm appears to do well. Overall, the algorithm finds correspondences between similar force actions even when the training set contains no demonstrations from the tested task.

Figure A.11: **Confusion Matrix: Trained on the Lego task and tested on the Nintendo 64 task. There are no search actions in the human coded labels. Since the Lego task contains search actions, the classifier finds them close to the insert actions. While inserting a cartridge the user feels for the opening which corresponds to the search action.**

A.6 Conclusions and Discussion

We proposed a feature representation technique which characterizes the local behavior of a force signal with a set of features at each time sample. We use the CWT with the Complex Morlet analyzing wavelet as a way to transform a time domain signal into feature vectors describing each sample instant.

We show that this feature representation provides a good way to describe a time domain force signal through a series of experiments and example supervised learning tasks. We intentionally provided little post-processing to the estimated test signals to study the limitations and advantages of using this representation.

At boundaries between different force actions we see chattering of the predicted labeled actions (before any post-processing). Such chattering indicates imprecision in localizing the beginning and end of actions as they often do not have hard boundaries. In some scenarios where correct ordering and approximate timing are sufficient it may be acceptable to use filtering techniques during post-processing.

The post-processing methods used in this work require further improvement and merely act as a starting point for filtering techniques. These methods are insufficient to correct classification errors like labeling a *hold* action for a very brief period of time between two *still* actions. Hence the CWT method requires good post-processing techniques to correct such problems.

As a supervised learning approach, our method is limited by and may not generalize beyond the training data. For example, in this paper we show how our algorithm can detect slip between the stylus and tongs. However, it may not perform well on a different object using the training examples from the stylus experiments as they do not contain examples of different objects. Another major draw back with this method is that it is offline. Many robotics applications require realtime recognition.

Possible future work would be to apply probabilistic models which could use labels occurring before or after as priors to correct predicted labels. It would be advantageous to develop methods to make use of the CWT based models for realtime or approximately realtime recognition.

In chapter 8 this method is used to infer grasp and contact actions to segment demonstrations. This application may only scratch the surface to other ways inferred actions can be used for robot programming.

B Recognizing constraints from position and force signals

Many applications may not require full rigid body descriptions of constraint interactions as described in chapter 3. A simple point contact model may be sufficient. In this chapter, constraints are represented as a point $\in \mathbb{R}^3$ constrained to geometry defined in space. Three commonly occurring geometric primitives, arcs, lines and planes are inferred using recorded trajectories of position and force signals.

Use of only position and force measurements to infer constraints is advantageous when orientation information is unavailable or is unreliable. Tracking a point in space is significantly easier than tracking the pose of a 6 degree of freedom rigid body. This results in cleaner signals with less noise. These cleaner signals afford the use of algorithms that can perform better constraint inference albeit only for simple models. Figure B.1 shows an example of the developed constraint inference method (of this appendix) operating on a demonstration. This demonstration is tricky, all three detected constraints, the line, the plane and the arc are all consistent with the detected plane constraint. In addition, the plane constraint is visited twice but the constraint inference approach correctly detects the constraint only once. The tool used is also different from the instrumented tongs described in 2, it precisely measures the motion (position and applied forces) of a constrained point in space. The approach in this appendix describes an effective way to infer constraints when only position and force information are available.

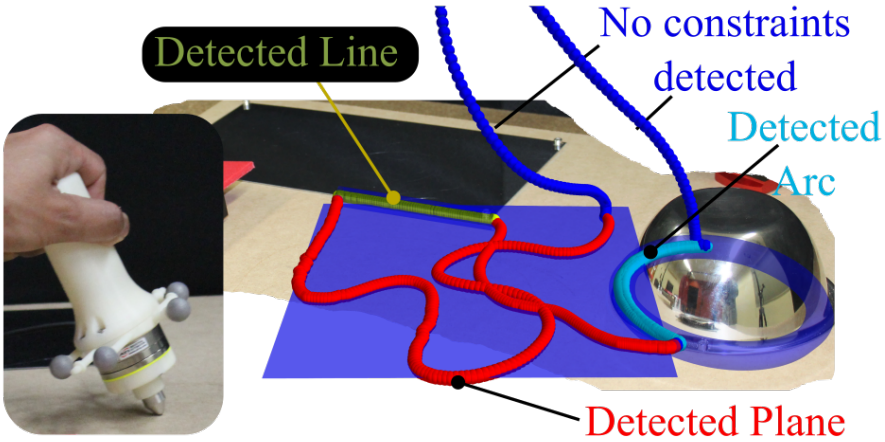

Figure B.1: **Demonstration containing all constraints in the same plane. The demonstration involves moving the Constraint Saber along the edge of the steel sheet, the planar surface and the edge of the bowl consecutively. Notice that in this demonstration, the demonstrator transitions between constraints without lifting the Constraint Saber off of the test bed. Our approach correctly associates samples to their respective constraint. Left - Shows the Constraint Saber demonstration tool.**

B.1 Point model of a constrained object

Let us consider a point object constrained to a subset of $\mathbb{R}$ due to it's interaction with a constraint. When forces are applied to the point object, it experiences reaction forces from the constraint. The principle of virtual work resolves forces into constraint reaction forces and forces along the direction of motion such as frictional forces. Constraint reaction forces must be orthogonal to the motion of the object. While conceptualizing how point contact reaction forces act on a body is intuitive, it is not trivial to explain why a particular constraint produces reaction forces in particular directions. We develop this idea below.

Principle of Virtual Work for Constraint Interactions

Consider an object constrained at a point $\mathbf{p} = (x, y, z) \in \mathbb{R}^3$ by a constraint $\Phi : \mathbb{R}^3 \Rightarrow \mathbb{R}^k$ such that $\Phi(\mathbf{p}) = 0$.[1]

Let an applied force $\mathbf{f} \in \mathbb{R}^3$ act on the body. If we assume that inertial forces are negligible, the reaction force $\mathbf{f_r}$ and friction force $\mathbf{f}$- are given as:

$$\mathbf{f} = \mathbf{f_r} + \mathbf{f}_\mu \tag{B.1}$$

If the velocity of the point $\mathbf{p}$ moving under constraint equations $\Phi\mathbf{(p)} = 0$ is $\mathbf{v} \in \mathbb{R}^3$, then from the principle of virtual work (37):

$$\mathbf{f_r} \cdot \mathbf{v} = 0 \tag{B.2}$$

Assuming that the friction force opposes the contact velocity, and thus is collinear to the velocity vector, force $\mathbf{f}$ can be resolved as friction force $\mathbf{f}_\mu$ and reaction force $\mathbf{f_r}$:

$$\mathbf{f}_\mu = (\mathbf{f} \cdot \hat{\mathbf{v}})\hat{\mathbf{v}} \tag{B.3}$$

$$\mathbf{f_r} = \mathbf{f} - \mathbf{f}_\mu \tag{B.4}$$

where $(\hat{\cdot})$ represents a unit vector. Using the velocity, force and position, a constraint frame F_p is defined, with basis vectors $\hat{\mathbf{v}}$, $\hat{\mathbf{f}}_\mathbf{r}$ and $\hat{\mathbf{v}} \times \hat{\mathbf{f}}_\mathbf{r}$ at position $\mathbf{p}$ of the point contact.

If the variation of $\mathbf{p}$ is $\partial\mathbf{p}$ then the admissible variations of $\mathbf{p}$ under $\Phi(\mathbf{p}) = 0$ must satisfy:

$$\partial\mathbf{p}^T \frac{\partial\Phi}{\partial\mathbf{p}} = 0 \tag{B.5}$$

Equation B.2 can be rewritten in variational form:

$$\partial\mathbf{p}^T \mathbf{f_r} = 0 \tag{B.6}$$

[1]We denote vectors and and matrices with bold symbols.

Using Lagrange multiplier theorem, similar to the approach used by Huag(23), Combining equations B.5 and B.6:

$$\frac{\partial \Phi}{\partial \mathbf{p}}^{T} \lambda + \mathbf{f_r} = 0 \tag{B.7}$$

where $\lambda \in \mathbb{R}^k$ are the Lagrange multipliers. Note: $\frac{\partial \Phi}{\partial \mathbf{p}}$ is size $k \times 3$ and λ is size $k \times 1$.

Equation B.7 is important because it provides a direct relationship between the constraint reaction force $\mathbf{f_r}$ and the constraint equations $\Phi(\mathbf{p}) = 0$. As shown in the following section, any set of constraint equations can essentially be 'plugged in' equation B.7 to retrieve the permissible reaction forces. Note that the Lagrange multipliers λ take any value that satisfy equation B.7.

Directly resolving friction forces from measured forces is not possible when considering rigid body models as described in section . By dropping the orientation information, friction forces can be assumed to act in the direction of motion. This assumption is not possible when reaction moments are involved.

Reaction forces for Geometric Constraints

The point contact constraints considered are line, arc and plane constraints. When a point body is constrained to one of these constraints, it experiences reaction forces and a reduction in its degrees of freedom.

Plane Constraint

The equation of a plane can be represented by:

$$\Phi(p) = ax + by + cz + d = 0 \tag{B.8}$$

where $a, b, c, d \in \mathbb{R}$

Using equation B.7 and B.8, the relationship between the constraint equations

and its corresponding reaction forces are:

$$\frac{\partial \Phi}{\partial \mathbf{p}} = \begin{bmatrix} a & b & c \end{bmatrix}$$

$$\mathbf{f_r} = \begin{bmatrix} a & b & c \end{bmatrix}^T \begin{bmatrix} \lambda_1 \end{bmatrix}$$

The normal $\mathbf{n}$ of the plane defined by equation B.8 is
$< a, b, c >$ This immediately leads to the result:

$$\hat{\mathbf{f}}_\mathbf{r} = \hat{\mathbf{n}} \tag{B.9}$$

Equation B.9 shows that *the constraint reaction force of the plane is always directed along the normal to the plane*.

Line Constraint

The line constraint can be represented by the symmetric equations of a line:

$$\frac{x - x_0}{l_1} = \frac{y - y_0}{l_2} = \frac{z - z_0}{l_3} \tag{B.10}$$

where the tangent vector $\mathbf{t} = (l_1, l_2, l_3) \in \mathbb{R}^3$ is the direction of the line and $p = (x_0, y_0, z_0) \in \mathbb{R}^3$ is a point on the line. Rearranging equation B.10:

$$\Phi(p)_1 = (x - x_0)\, l_2 - (y - y_0)\, l_1 = 0, \tag{B.11}$$

$$\Phi(p)_2 = (z - z_0)\, l_2 - (y - y_0)\, l_3 = 0 \tag{B.12}$$

Applying equation B.7 to the line equations B.11 and B.12, the constrained reaction force contributed by this constraint is given by:

$$\mathbf{f_r} = \begin{bmatrix} l_2 & -l_1 & 0 \\ 0 & -l_3 & l_2 \end{bmatrix}^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \tag{B.13}$$

Equation B.13 provides the permissible values of $\mathbf{f_r}$. As $\mathbf{f_r}$ is a sum of two linear independent vectors, *the constraint reaction force spans the plane perpendicular to the tangent of the line*.

Arc Constraint

The arc constraint is represented by the intersection of the sphere and plane that contain the arc.

$$\Phi_1(p) = (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 - r^2 = 0 \tag{B.14}$$

$$\Phi_2(p) = ax + by + cz + d = 0 \tag{B.15}$$

where the center and radius of the sphere are $C = (x_0, y_0, z_0) \in \mathbb{R}^3$ and $r \in \mathbb{R}$ respectively. Using the same development:

$$\frac{\partial \Phi}{\partial \mathbf{p}} = \begin{bmatrix} 2(x - x_0) & 2(y - y_0) & 2(z - z_0) \\ a & b & c \end{bmatrix}$$

$$\mathbf{f_r} = \begin{bmatrix} 2(x - x_0) & 2(y - y_0) & 2(z - z_0) \\ a & b & c \end{bmatrix}^T \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \tag{B.16}$$

Similarly to the line case, *the constraint reaction force spans a plane perpendicular to* $\mathbf{n} = \langle a, b, c \rangle$ *passing through the center of the circle.* Note: This plane containing the reaction force vector has a normal vector produced by the cross product: $\frac{\partial \Phi_1}{\partial \mathbf{p}}$ and $\frac{\partial \Phi_2}{\partial \mathbf{p}}$.

Parameterizations of Planes, Lines and Arcs

For a representation of a constraint to be useful, a uniquely identifiable one must be selected. We list the parameterizations chosen for each constraint type below:

Plane Constraint

The definition of the plane, given by equation B.8 is parameterized by (a, b, c, d). This is inconvenient as multiple (a, b, c, d) can represent the same plane. Instead, the unit normal vector $\hat{n}$ and the distance of the plane to the origin d are used as the plane constraint representation.

Line Constraint

The line is parameterized with the unit tangent vector $\hat{t}$ and the point on the line $\mathbf{c_p} \in \mathbb{R}^3$ closest to the origin. This is given by:

$$\mathbf{c_p} = \mathbf{q} - \frac{\mathbf{q} \cdot \hat{\mathbf{t}}}{\hat{\mathbf{t}} \cdot \hat{\mathbf{t}}} \hat{\mathbf{t}} \tag{B.17}$$

where $\mathbf{q} \in \mathbb{R}^3$ is any point on the line.

Arc Constraint

The arc is parameterized with the center C, radius r and the normal vector $\hat{n}$ of the plane containing the arc.

B.2 Detailed Overview of Approach

Our approach uses both position and force information to identify local descriptions of the signals and cluster these over the complete task. An overview of our approach is shown in figure B.2.

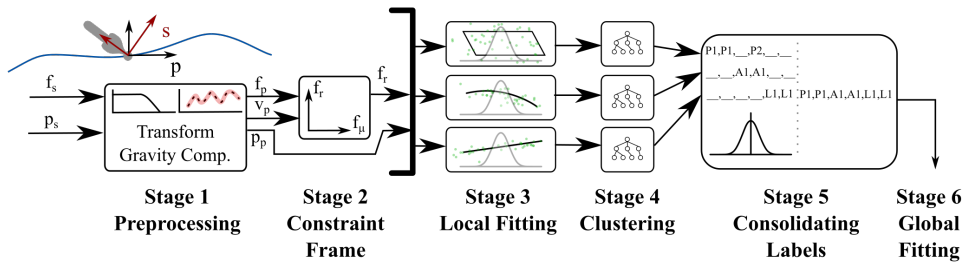

Figure B.2: **Schematic representation of our method. Stage 1 - Signals are preprocessed, Stage 2 - Constraint frames are defined where applicable over the entire signal, Stage 3 - Local geometry is fit at every constraint frame, Stage 4 - Local geomery is clustered for each constraint type, Stage 5 - Cluster labels are consolidated, Stage 6 - Geometry is fit globally.**

Stage 1: Preliminary signal preprocessing and calibration

To mitigate the effects of sensor noise and to prevent signal aliasing, the position and force measurements are resampled and filtered. Gravitational effects are removed by compensating for tool weight. Finally, the tool tip forces are transformed to the global coordinate frame using the orientation measurements.

Stage 2: Determining Constraint Frames across the Demonstration

The purpose of this stage is to determine where the constraints occur in the demonstration. The reaction and frictional forces are computed by using an estimate of the tool tip velocity and force measurements using equations B.3 and B.4. The reaction force and friction force combine to form the constraint frame F_p, a local description of the constraint at the point of contact. There are two situations where the constraint frame is not well defined:

- At locations where the velocity is approximately zero - the tool tip must be moving and its position measurements free of excessive noise.

- At locations where reaction forces do not exist - the computed reaction force is smaller than sensor noise.

Samples where the reaction force is zero correspond to the absence of a constraint and are ignored. Locations where the velocity is effectively zero are unusable because it is impossible to resolve the measured forces into reaction forces and friction forces.

Stage 3: Determining local constraint geometry

In this stage, all defined constraint types (i.e., plane, line and arc) are fit to the measured position and force data locally at each well defined constraint frame. This provides a local description of the constraint. The proposed fitting methods are all least squares approximations as the errors in our measurements are Gaussian distributed.

Fitting Planes

A plane is represented uniquely by a plane normal and a point on the plane. To fit a plane, the normal reaction force and the point of contact are used as the normal and the point on the plane respectively. Over a local window about the sample of interest, gaussian weighted average of the reaction force $\hat{f}_r$ represents the plane normal vector. Note that this is equivalent to applying a gaussian kernel to a signal as done in low pass filtering. Using the local estimate of the normal reaction force and a similar weighted average of the position of the tool tip, a local plane is constructed. For a unique representation of this plane, the distance of the plane d and the unit normal of the plane $\hat{f}_r$ are used.

Fitting lines

To fit a line, position measurements need only be used. A gaussian windowed SVD performs the fit described in section B.2. The final parameterization of the line is the tangent vector and a point on the line.

Weighted SVD for plane and line estimates

Lines or planes can be fit using the weighted SVD technique elaborated by Shakarji's et al. (64) . Consider a set of N points $p_1, p_2 ... p_N$ where each $p_i = (x_i, y_i, z_i) \in \mathbb{R}^3$, with corresponding positive weights $w_1, w_2 ... w_i ... w_N \in \mathbb{R}$. A point on the plane (line) is given by the weighted centroid

$$\overline{p} = \frac{\sum_{i=1}^{N} w_i p_i}{\sum_{i=1}^{N} w_i} \tag{B.18}$$

and the normal (tangent) vector to the plane (line) is the right singular vector that corresponds to the smallest (largest) singular value in the singular value decomposition of the $N \times 3$ matrix given by:

$$\mathbf{M} = \begin{bmatrix} \sqrt{w_1}(\mathbf{p_1}^T - \overline{p}^T) \\ \sqrt{w_2}(\mathbf{p_2} +^T - \overline{p}^T) \\ \vdots \\ \sqrt{w_N}(\mathbf{p_N}^T - \overline{p}^T) \end{bmatrix} \tag{B.19}$$

Fitting Arcs

The arc fitting procedure is accomplished using a nonlinear least squares fit. The arc parameters are the center C, the radius r and the normal of the plane $\hat{n}$ containing the arc. The least squares regression algorithm is as follows:

Consider a set of N points as in the previous section. The procedure to fit an arc is to first find the best fit plane containing the points then to use nonlinear least squares to fit the arc in that plane and finally transform back. The best fit plane is obtained using SVD and a corresponding inverse rotation matrix $R^T, R \in SO3$ is found. The points are transformed into the plane by:

$$\tilde{\mathbf{p}} = R^T (\mathbf{p} - \overline{\mathbf{p}})$$

Consider $\tilde{\mathbf{p}}$ as a 2×1 matrix ignoring the z component. The radius r and center

$C = R\tilde{C}$ are given by:

$$\arg\min_{\tilde{C},r} \sum_{i=1}^{N} w_i \left(\left\| (\tilde{\mathbf{p}}_i - \tilde{C}) \right\|^2 - r^2 \right)^2$$

The result from this stage is a stream of labels containing fit parameters for each constraint type for all well defined constraint frame samples over the complete demonstration.

Stage 4: Clustering

Local parameters of each constraint type are clustered over the entire task. The procedure clusters local constraints into larger groups and removes outliers and noise. These larger groups of similar parameters represent global constraints.

Our dataset contains unit vectors, euclidean points and simple scalars called features. Like all clustering methods, if features used to cluster are of vastly different units or type, clustering will provide poor results. Ideally providing a different distance metric for each feature type is preferable. Also, the number of clusters is unknown before hand.

To meet these clustering requirements, a custom method referred to as the *Tiered DBSCAN* clustering algorithm was developed. The conventional DBSCAN(18) algorithm is augmented with a tiered approach so it is suitable for clustering datasets with features of different types including unit vectors, scalars or euclidian coordinates. Alternatively, one might try to weight these features and develop a common metric but determining effective weights for each feature type is challenging.

Tiered DBSCAN clusters in stages, using a new feature at every stage. The clusters from the previous stage are subdivided using a new feature to create sub-clusters. As DBSCAN can remove outliers effectively, many of the sub-clusters from higher tiers are rejected after successive stages of clustering. The algorithm is described in Algorithm 2 and depicted in Figure B.4 :

The two parameters for DBSCAN are *min-points* - the number of points required to form a cluster and *eps* - the minimum distance points need to be to successfully

Algorithm 2 Tiered DBSCAN

DBSCAN parameters: *min-points*, *eps* and distance *metric*.

```
clusters := [data]                                  ▷ Single cluster containing all data
for each feature in features
current_cluster_set := [ ]
for each cluster in clusters
sub_clusters := DBSCAN(cluster, feature.metric, feature.eps,
    feature.min_samples)
append sub_clusters to current_cluster_set
clusters := current_cluster_set
clusters is the list of required clusters.
```

cluster. DBSCAN also allows specification of custom distance metrics apart from the L2 norm. Min-points is set considering the smallest possible number of points that would be considered a cluster. Eps is set depending on the measurement noise and associated fitting errors. The order of clustering is chosen empirically but follows intuition: features that cluster more reliably and reject noise should be preferred first. For example, clustering plane normal vectors before plane distance is better because the normal force measurement is more reliable than the plane distance estimate.

	Feature	Metric	eps	min_samples
Plane	Unit Normal Force	Magnitude of cross product	0.02	10
	Plane Distance	L2 norm distance	0.05(m)	10
Line	Unit Tangent Vector	Magnitude of cross product	0.02	50
	Closest Point to Origin	L2 norm distance	0.02(m)	10
Arc	Center	L2 norm distance	0.01(m)	10
	Radius			
	Unit Normal of Plane	Magnitude of cross product	0.02	10

Figure B.3: **Various metrics for each feature type corresponding to each constraint. The features are clustered in decreasing order of reliability to DBSCAN clustering.**

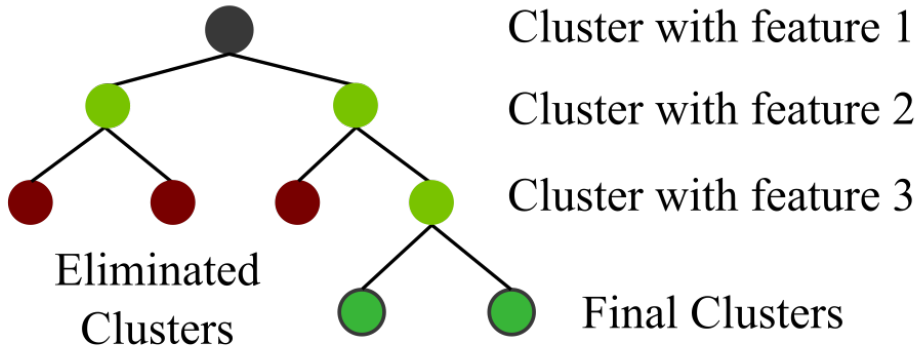

Figure B.4: **Shows the clustering procedure by Tiered DBSCAN. Clustering is performed in stages where each level represents clustering with a specific feature and its corresponding metric. This is useful for datasets with different types of features, like unit vectors and euclidian points.**

Figure B.3 shows the features, corresponding metric and DBSCAN parameters at each tier for each type of geometric constraint. The result of the clustering procedure is a signal/stream of labels for each constraint type. Note that clustering for each constraint type is performed individually.

Each cluster recognized by Tiered DBSCAN represents an individual constraint. The clustering procedure creates constraint label streams over time for each constraint type. At locations where constraints were not detected, the stream labels take no value.

Stage 5: Consolidating Plane, Line and Arc labels

The clustering procedure creates three labeled data streams/signals, each corresponding to a constraint type. To consolidate these streams into a single stream, disambiguation heuristics and the *Proximity Weighted Selection Filter* are applied. The heuristics are as follows:

H1 Plane Check: For each plane cluster, the positions and forces are checked for consistency with the recognized plane. A plane using the points contained in

the cluster are fit to recover a normal vector. This normal vector is checked for consistency with the average constraint reaction force, using the magnitude of the cross product between the two. This heuristic helps eliminate line constraints detected as planes, as a very straight line would most likely provide a plane inconsistent with the reaction force vector.

H2 Arc Check: Arc labels where a plane and arc are simultaneously detected are ignored. As an arc constraint will most probably have different constraint reaction forces than a plane, the plane is preferred when both are detected.

These heuristics deal with degenerate cases where the constraint is ambiguous. Note that in both these cases either constraint is physically possible but in practice the heuristic captures the real constraint.

After applying the heuristics, the Proximity Weighted Selection Filter combines these three label streams to form one stream of labels. The filter assumes that at a particular sample the probability of a label depends on the density of that label about the sample of interest. The filter uses a gaussian to weight points closer to the sample more than those further away shown in algorithm 3.

Stage 6: Fitting constraints

The points corresponding to each constraint are used to fit their respective constraint. At this point, the constraint type, the parameters associated with the constraints and the associated samples of each constraint are identified.

Algorithm 3 Proximity Weighted Selection Filter

Consider m discrete sequences of length n that take values from an un-ordered set $labels$ with l elements.

```
sig_mat is an n × m matrix of input sequences
prob_mat := 0_{n×l}
consolidated_signal := empty length n sequence
for k ∈ 1, 2...l do
    for i ∈ 1, 2...n do
        for j ∈ 1, 2...m do
            if labels[k] == sig_mat[i, j] then
                prob_mat[i, k] := prob_mat[i, k] + 1
    prob_mat[:, k] := gaussian_filter(prob_mat[:, k])
for i ∈ 1, 2...n do
    label_id := max_index(prob_mat[i, :])
    consolidated_signal[i] := labels[label_id]
consolidated_signal is the required filtered signal
```

	Experiment	TP	Missed	FP					Total	Recall	Precision
		#	#	planes #	lines #	arcs #	total #	total %	#		
Force and Position	**1.** Only plane constraints	116	1	1	1	3	5	4.27	117	**0.99**	**0.96**
	2. Only line constraints	117	0	2	0	0	2	1.71	117	**1.00**	**0.98**
	3. Only arc constraints	103	5	12	1	8	21	19.44	108	**0.95**	**0.83**
	4. Random	208	23	15	0	0	15	6.49	231	**0.90**	**0.93**
	5. Flat Tool	100	0	1	0	2	3	3.00	100	**1.00**	**0.97**
Only Position	**6.** Random	201	30	125	1	15	141	61.04	231	**0.87**	**0.59**

Figure B.5: **Experimental results shown. TP - True Positives, detects the right constraint at the correct location, FP - False Positives, FPs are situations when the method detects a constraint but none exists at the detected location, Precision indicates how well the method is robust against detecting false positives (Larger is better). Recall indicates how well the method identifies the constraints. Missed indicates the number of constraints that were not detected. Total indicates the total number of constraint in the demonstrations. Notice the position only method has a smaller precision when compared to the position and force method.**

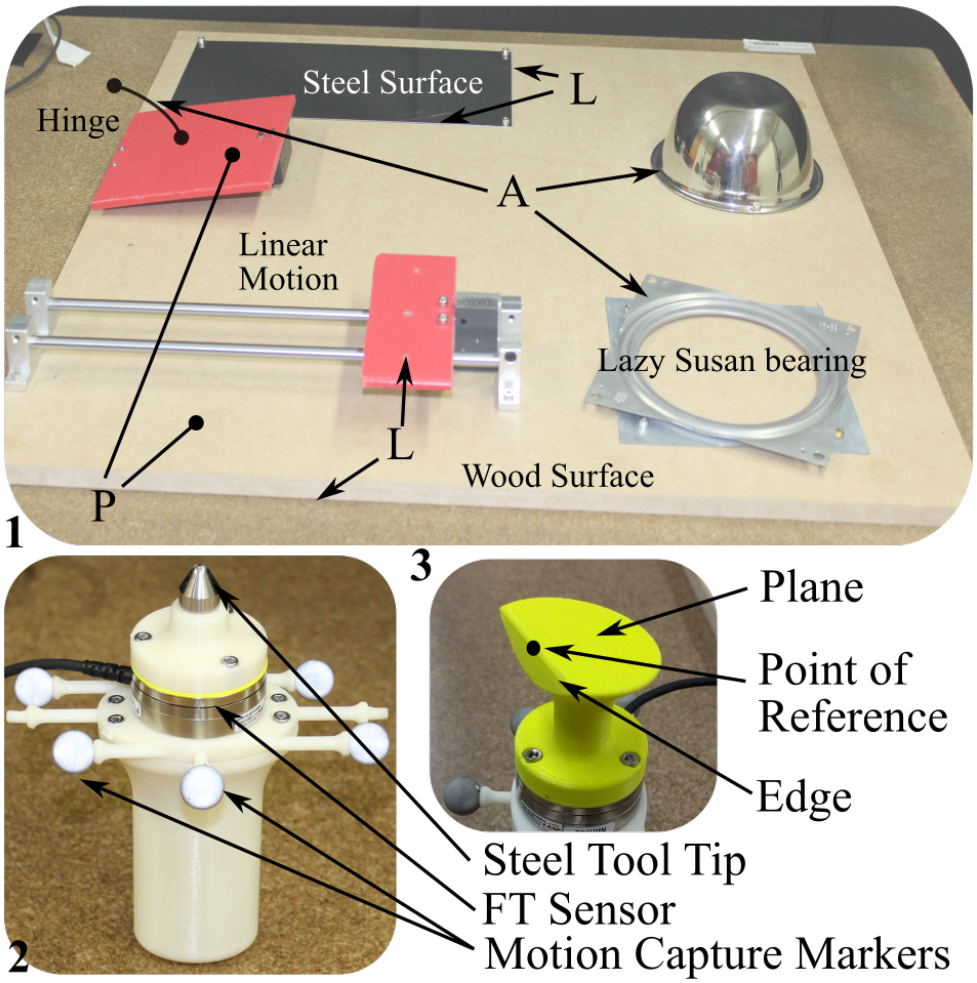

Figure B.6: **1. Shows the test bed containing plane, line and arc constraints. 2. Shows the Constraint Saber. The constraint Saber has motion capture markers to track its position and a FT sensor to measure forces. 3. An alternative tool for edge and plane constraint interaction (Flat Tool).**

B.3 Experimental Evaluation

The performance of the developed constraint recognition method was evaluated using a dedicated apparatus containing a constraint recognition instrument, the Constraint Saber and a custom testbed. The Constraint Saber described in chapter 2

has motion capture markers and an ATI Mini40 force torque sensor. While the force torque sensor can measure forces and moments, force measurements are only used. The testbed was designed to include multiple examples of each constraint type (i.e., plane, line and arc) with various levels of frictional resistance varied through the use of different materials and rolling element bearings.

The Constraint Saber, shown in Figure B.6 contains a force torque sensor to measure forces and motion capture markers to track its position in space. The Constraint Saber interacts with the environment through a steel tool tip to guarantee point contacts with the environment and ensure that moments are not applied to the environment. Note that all items that move with the constraint saber tool tip have a notch to guarantee consistent contact. Other tools may be attached to the end of the Constraint Saber.

In a typical demonstration, a demonstrator uses the constraint saber to interact with the available constraints on the testbed. Such demonstrations contain multiple constraint interactions of different types where transitions can occur between constraint types or between a constraint type and free space.

Experimental evaluations are performed with two demonstrators. For each evaluation, (an approximately) equal number of trials are performed by each demonstrator. An expert catalogues the number of true and false positive constraint detections: a true positive occurs when a constraint is detected correctly, i.e. the location and type of constraint are correctly identified, a false positive occurs when a constraint is detected in a location where that particular constraint does not exist. Examples include a plane constraint detected where a line constraint should be or a plane constraint detected in a location where no constraint exists. In our experiments we use four metrics:

- TP: Number of correctly detected constraints - *True Positives*
- FP: Number of incorrectly detected constraints - *False Positives*
- PPV: = TP/(TP + FP) - *Precision, Positive Predictive Value*
- Recall: = TP/(Total number of actual constraints)

Precision is an efficient metric to evaluate this kind of recognition as it describes how well the method is robust against detecting false positives. Recall provides a measure of how well constraints are identified. As we will see, detection is not as difficult as preventing a false detection.

Single Constraint Type Recognition

The algorithm's performance is evaluated in situations where only one type of constraint appears in a demonstration. In such a demonstration, the demonstrator uses the constraint saber to interact with a particular type of constraint such as only line constraints. This experiment contains 39, 39 and 36 trials of 3 plane, 3 line and 3 arc constraints respectively.

A summary of the results are given in figure (table) B.5, row 1-3. The precision of the identified plane, line and arc constraints are 0.96, 0.98 and 0.83 respectively. The recall numbers for the same are 0.99, 1 and 0.95. The arc is the most difficult to recognize, particularly in cases where the ratio of the arc-length to radius is small or when the radial forces applied are low.

Multiple Constraint Type Recognition

Our approach is evaluated in situations where multiple constraint types occur in a single demonstration. See Figure B.1. The demonstrators interact with constraints on the test bed and are directed to perform 2 planar, 2 arc and 2 line constraints in a demonstration. Our method's performance is evaluated by recording 39 trials containing 6 constraints each from two different demonstrators(20 and 19 each).

A summary of the results are given in figure (table) B.5, row 4. The precision was 0.93 while the recall was 0.90 for this experiment. In these experiments the majority of false positives were due to arc constraints falsely detected as plane constraints.

Position only approach evaluation

From the experimental results described above, it is clear that the performance of our method is reasonably good. However it is important to assess its performance relative to a method that uses only position (kinematic) information. To make a fair comparison, our approach is adapted to function with only position measurements making minimal changes.

The position only approach uses the gaussian weighted windowed SVD to estimate the local plane normal vector. (Instead of the normal force vector - which is not available.) The most significant difference is due to the absence of the reaction force measurement and its use in identifying active constraints. In the position-only approach, this step is not possible and thus all samples must be considered for constraint detection. Other than these differences, the other elements of the position only approach are identical.

Using the position-only approach we perform demonstrations described in section B.3. While the recall value (0.87) is similar to the force and position method, the precision is low (0.59). The position only method correctly labels 201 constraints out of 231 constraints, while also detecting an additional 141 false positives. This is captured by the low precision (PPV: 0.59) which is significantly less than the forces and position approach (PPV: 0.93). A summary of the results is provided in figure (table)B.5, row 6.

The position only approach detects plane constraints in locations where constraints do not exist. Figure B.7 shows an example of incorrect detection of a planar constraint in free space by the position only approach.

In conclusion, the large false positive rate is contributed by the lack of sufficient information i.e., the absence of forces. While the position only method correctly identifies motions contained in fictitious planes in free space, these planes are not constraints as a constraint physically requires force interaction.

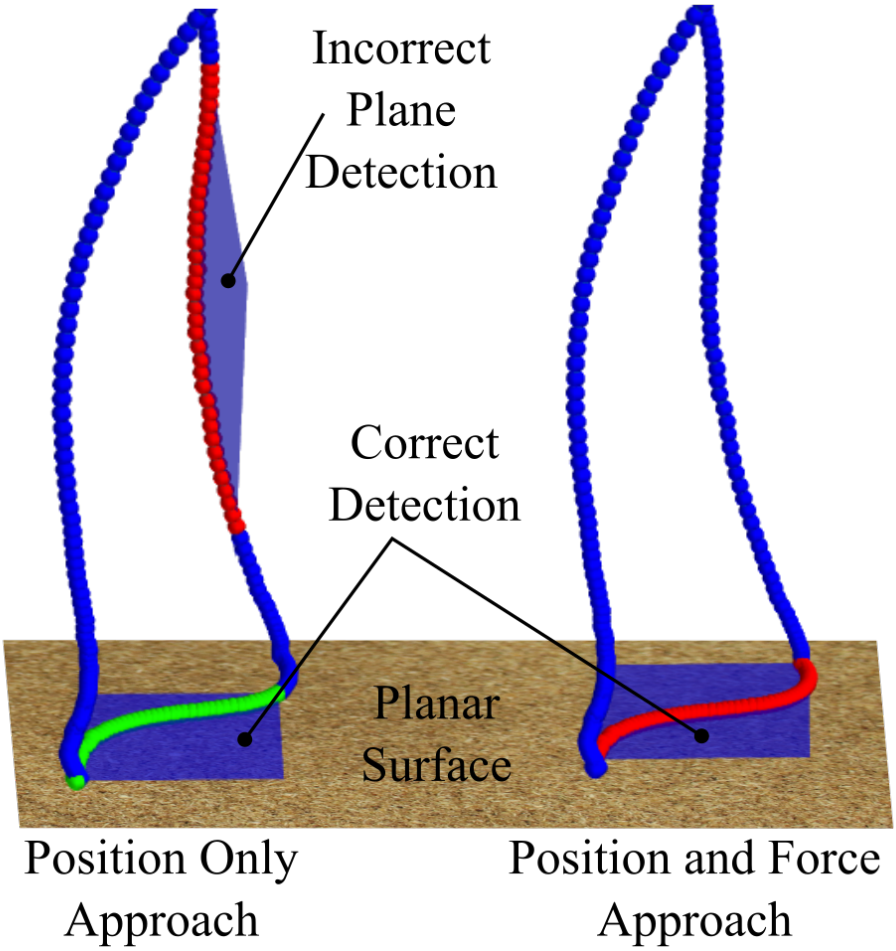

Figure B.7: **The position only approach detects planes in free space while the force and position approach does not.**

Bibliography

[1] E. Aertbeliën, J. D. Schutter, E. Aertbelien, and J. De Schutter, "ETaSL/eTC: A constraint-based task specification language and robot controller using expression graphs," in *IEEE International Conference on Intelligent Robots and Systems*, 2014, pp. 1540–1546. [Online]. Available: https://ieeexplore.ieee.org/document/6942760

[2] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 12 1974.

[3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009. [Online]. Available: https://doi.org/10.1016/j.robot.2008.10.024

[4] L. Armesto, J. Bosga, V. Ivan, and S. Vijayakumar, "Efficient learning of constraints and generic null space policies," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1520–1526, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7989181

[5] L. Armesto, J. Moura, V. Ivan, M. S. Erden, A. Sala, and S. Vijayakumar, "Constraint-aware learning of policies by demonstration," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1673–1689, 2018. [Online]. Available: https://doi.org/10.1177/0278364918784354

[6] L. Armesto, J. Moura, V. Ivan, A. Sala, and S. Vijayakumar, "Learning constrained generalizable policies by demonstration," *Robotics: Science and Systems*, vol. 13, 2017.

[7] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation planning on constraint manifolds," *2009 IEEE International Conference on Robotics and Automation*, vol. i, pp. 625–632, 2009. [Online]. Available: https://ieeexplore.ieee.org/document/5152399

[8] S. Bing Kang and S. B. Kang, "Robot Instruction by Human Demonstration," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 12 1994.

[9] H. Bruyninckx and J. D. Schutter, “Specification of force-controlled actions in the "task frame formalism"-a synthesis,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 581–589, 8 1996. [Online]. Available: https://ieeexplore.ieee.org/document/508440

[10] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal, “Learning variable impedance control,” *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 820–833, 2011.

[11] C. S. Burrus, “Iterative reweighted least squares,” *OpenStax CNX. Available online: http://cnx. org/contents/92b90377-2b34-49e4-b26f-7fe572db78a1*, vol. 12, 2012.

[12] S. Calinon, I. Sardellitti, and D. G. Caldwell, “Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies,” in *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2010, pp. 249–254.

[13] L. G. Chambers and R. Fletcher, “Practical Methods of Optimization,” *The Mathematical Gazette*, vol. 85, no. 504, p. 562, 2007.

[14] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[15] J. de Schutter, H. Bruyninckx, S. Dutré, J. de Geeter, J. Katupitiya, S. Demey, and T. Lefebvre, “Estimating First-Order Geometric Parameters and Monitoring Contact Transitions during Force-Controlled Compliant Motion,” *The International Journal of Robotics Research*, vol. 18, no. 12, pp. 1161–1184, 1999. [Online]. Available: https://doi.org/10.1177/02783649922067780

[16] K. L. Doty, C. Melchiorri, and C. Bonivento, “A theory of generalized inverses applied to robotics,” *The International Journal of Robotics Research*, vol. 12, no. 1, pp. 1–19, 1993.

[17] P. Englert, N. A. Vien, and M. Toussaint, “Inverse KKT: learning cost functions of manipulation tasks from demonstrations,” *The International Journal of Robotics Research*, p. 027836491774598, 2017. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364917745980

[18] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996. [Online]. Available: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[19] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941.

[20] S. C. Gholamhosein Sheikholeslami and A. Zhang, "Wavecluster: A multi-Resolution Clustering Approach for Very Large Spatial Databases," *International Journal on Very Large Data Bases (VLDB98)*, vol. 8, no. 3-4, pp. 428–429, 1998.

[21] H. Goldstein, C. P. Poole, and J. L. Safko, *Classical Mechanics*. Addison Wesley, 2002. [Online]. Available: https://books.google.com/books?id=tJCuQgAACAAJ

[22] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous Deep Q-Learning with Model-based Acceleration," pp. 122–148, 2016. [Online]. Available: http://arxiv.org/abs/1603.00748

[23] E. J. Haug, "Computer aided kinematics and dynamics of mechanical systems, volume 1: Basic methods," *Journal of Manufacturing Systems*, vol. 11, no. 2, p. 146, 2008.

[24] I. Havoutis and S. Ramamoorthy, "Motion planning and reactive control on learnt skill manifolds," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1120–1150, 2013.

[25] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance," *2009 IEEE International Conference on Robotics and Automation*, pp. 2587–2592, 2009.

[26] J. Huh, B. Lee, and D. D. Lee, "Constrained Sampling-Based Planning for Grasping and Manipulation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 5 2018, pp. 223–230. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8461265

[27] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Trajectory formation for imitation with nonlinear dynamical systems," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2, pp. 752–757, 2001.

[28] I. F. Jasim and P. W. Plapper, "Contact-state monitoring of force-guided robotic assembly tasks using expectation maximization-based Gaussian mixtures models," *International Journal of Advanced Manufacturing Technology*, vol. 73, no. 5-8, pp. 623–633, 2014.

[29] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[30] G. Kaiser, *A friendly guide to wavelets*. Springer Science & Business Media, 2010.

[31] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning force control policies for compliant manipulation," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4639–4644, 2011.

[32] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[33] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[34] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1163/016918611X558261%5Cnhttp://www.tandfonline.com/doi/pdf/10.1163/016918611X558261

[35] V. Krüger, D. Kragic, A. Ude, and C. Geib, "The meaning of action: A review on action recognition and mapping," *Advanced robotics*, vol. 21, no. 13, pp. 1473–1501, 2007.

[36] D. Kulic, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 330–345, 2012.

[37] C. Lanczos, *The variational principles of mechanics*. Courier Corporation, 2012.

[38] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. April 2016, pp. 255–258, 1995.

[39] A. X. Lee, H. Lu, A. Gupta, S. Levine, and P. Abbeel, "Learning force-based manipulation of deformable objects from multiple demonstrations," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June. IEEE, 2015, pp. 177–184.

[40] S. Levine, N. Wagener, and P. Abbeel, "Learning Contact-Rich Manipulation Skills with Guided Policy Search," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 156–163, 2015. [Online]. Available: https://arxiv.org/pdf/1501.05611.pdf

[41] M. Li, K. Tahara, and A. Billard, "Learning task manifolds for constrained object manipulation," *Autonomous Robots*, vol. 42, no. 1, pp. 1–16, 2017.

[42] H. Liu, Y. Yu, F. Sun, and J. Gu, "Visual – Tactile Fusion for Object Recognition," *IEEE Trans. on Automation Science and Engineering*, no. January, pp. 1–13, 2016.

[43] J. Luo and K. Hauser, "Interactive generation of dynamically feasible robot trajectories from sketches using temporal mimicking," in *2012 IEEE International Conference on Robotics and Automation*, 5 2012, pp. 3665–3670. [Online]. Available: https://ieeexplore.ieee.org/document/6225341

[44] S. A. Mallat, "Time-Frequency Dictionaries," *A Wavelet Tour of Signal Processing*, pp. 11–14, 1999.

[45] W. Meeussen, J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter, "Contact state segmentation using particle filters for programming by human demonstration in compliant motion tasks," *Springer Tracts in Advanced Robotics*, vol. 39, pp. 3–12, 2008.

[46] C. Melchiorri, "Slip detection and control using tactile and force sensors," *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 3, pp. 235–243, 2000.

[47] K. Muelling, J. Kober, and J. Peters, "Learning table tennis with a mixture of motor primitives," *2010 10th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2010*, pp. 411–416, 2010.

[48] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[49] B. Nemec, M. Zorko, and L. Žlajpah, "Learning of a ball-in-a-cup playing robot," *19th International Workshop on Robotics in Alpe-Adria-Danube Region, RAAD 2010 - Proceedings*, pp. 297–301, 2010.

[50] S. Niekum, S. Osentoski, C. G. Atkeson, and A. G. Barto, "Online Bayesian changepoint detection for articulated motion models," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 1468–1475, 2015.

[51] S. Niekum, S. Osentoski, G. Konidaris, and A. G. Barto, "Learning and generalization of complex tasks from unstructured demonstrations," *IEEE International Conference on Intelligent Robots and Systems*, no. August, pp. 5239–5246, 2012.

[52] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning Dexterous In-Hand Manipulation," *CoRR*, 2018. [Online]. Available: http://arxiv.org/abs/1808.00177

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in {P}ython," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[54] C. Perez-D'Arpino and J. A. Shah, "C-LEARN: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2017, pp. 4058–4065.

[55] R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing*, vol. 28, no. 6, pp. 976–990, 2010.

[56] P. Praveena, G. Subramani, B. Mutlu, and M. Gleicher, "Characterizing Input Methods for Human-to-robot Demonstrations," in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 3 2019, pp. 344–353. [Online]. Available: http://graphics.cs.wisc.edu/Papers/2019/PSMG19

[57] M. H. Raibert and J. J. Craig, "Hybrid Position/Force Control of Manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, p. 126, 2009. [Online]. Available: http://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1403159

[58] D. Rakita and M. Gleicher, "RelaxedIK: Real-time Synthesis of Accurate and Feasible Robot Arm Motion," in *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, 6 2018. [Online]. Available: http://www.roboticsproceedings.org/rss14/p43.html

[59] D. Rakita, B. Mutlu, and M. Gleicher, "An Autonomous Dynamic Camera Method for Effective Remote Teleoperation," in *2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 3 2018. [Online]. Available: http://graphics.cs.wisc.edu/Papers/2018/RMG18

[60] J. Rojas, N. Yamanobe, E. Yoshida, and K. Nagata, "Towards snap sensing," no. September 2013, 2017.

[61] L. Rozo, P. Jiménez, and C. Torras, "A robot learning from demonstration framework to perform force-based manipulation tasks," *Intelligent Service Robotics*, vol. 6, no. 1, pp. 33–51, 1 2013. [Online]. Available: https://doi.org/10.1007/s11370-012-0128-9

[62] S. J. Schiff, A. Aldroubi, M. Unser, and S. Sato, "Fast wavelet transformation of EEG," *Electroencephalography and Clinical Neurophysiology*, vol. 91, no. 6, pp. 442–455, 1994.

[63] L. Sentis and O. Khatib, "A whole-body control framework for humanoids operating in human environments," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 5 2006, pp. 2641–2648.

[64] C. M. Shakarji and V. Srinivasan, "Theory and Algorithms for Weighted Total Least-Squares Fitting of Lines, Planes, and Parallel Planes to Support Tolerancing Standards," *Journal of Computing and Information Science in Engineering*, vol. 13, no. 3, pp. 31 008–31 011, 8 2013. [Online]. Available: http://dx.doi.org/10.1115/1.4024854

[65] J. Silvério, Y. Huang, L. Rozo, S. Calinon, and D. G. Caldwell, "Probabilistic Learning of Torque Controllers from Kinematic and Force Constraints," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10 2018, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8594103

[66] M. Stilman, "Task constrained motion planning in robot joint space," *IEEE International Conference on Intelligent Robots and Systems*, pp. 3074–3081, 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4399305

[67] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *Journal of Artificial Intelligence Research*, vol. 41, pp. 477–526, 2011.

[68] Z. Su, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Slip Detection and Classification for Grip Control using Multiple Sensory Modalities on a Biomimetic Tactile Sensor," *IROS 2015 workshop*, pp. 2–3, 2015.

[69] G. Subramani, D. Rakita, H. Wang, J. Black, M. Zinn, and M. Gleicher, "Recognizing actions during tactile manipulations through force sensing," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4386–4393. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8206302

[70] G. Subramani, M. Gleicher, and M. Zinn, "Recognizing Geometric Constraints in Human Demonstrations using Force and Position Signals," *IEEE Robotics and Automation Letters*, vol. 3766, no. c, pp. 1–1, 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8264714/

[71] G. Subramani, M. Zinn, and M. Gleicher, "Inferring geometric constraints in human demonstrations," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 2018, pp. 223–236. [Online]. Available: http://proceedings.mlr.press/v87/subramani18a.html

[72] M. Tamosiunaite, B. Nemec, A. s. Ude, and F. Wörgötter, "Learning to pour with a robot arm combining goal and shape learning for dynamic movement primitives," *Robotics and Autonomous Systems*, vol. 59, no. 11, pp. 910–922, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2011.07.004

[73] C. Torrence and G. P. Compo, "A Practical Guide to Wavelet Analysis," *Bulletin of the American Meteorological Society*, vol. 79, no. 1, pp. 61–78, 1998.

[74] Y. Yang, Y. Li, C. Fermuller, and Y. Aloimonos, "Robot Learning Manipulation Action Plans by "Watching" Unconstrained Videos from the World Wide Web," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 3686–3692. [Online]. Available: http://dl.acm.org/citation.cfm?id=2888116.2888228

[75] T. Yoshikawa, "Force control of robot manipulators," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, 4 2000, pp. 220–226. [Online]. Available: https://ieeexplore.ieee.org/document/844062

[76] R. Zhang, "Spectral decomposition of seismic data with CWPT," *The Leading Edge*, vol. 27, no. 3, p. 326, 2008.