MONASH University

Learning Manipulation Affordances via Intrinsically Motivated Exploration

Benjamin Border

Bachelor of Mechatronics Engineering

A thesis submitted for the degree of *Master* at

Monash University in 2015

Master of Engineering Science (Research)

Copyright notice

Abstract

In this thesis, the problem of unsupervised discovery and learning of manipulations for an object is addressed. The concept of manipulation affordances is introduced, which characterise a naive model of an object's expected motion due to interaction, given various conditions. Interaction with an object is performed via pushing, sliding or dragging motions. In order to learn the motion of an object, the agent observes how the object itself moves as a result of the agent's own interactions with it, then generalises and learns the necessary characteristics and conditions to recreate the object motion. Generalisation requires knowledge of the environment the object resides in. This thesis describes the environment via constraints, which represent impenetrable barriers for the object. Generalisation is then performed by referencing affordances from constraints, recognising symmetry and parametrisation. It is also necessary to be able to use affordances to complete tasks. This is addressed by generating a discrete sequence of moves from affordances using a recursive tree algorithm. It is shown that affordances of an object can be generalised and learned, then used to perform simple manipulation tasks for an object.

Discovery plays an important role in building a repertoire of manipulation affordances. The ability to discover and learn new affordances can be useful for unfamiliar environments or objects and enables a measure of adaptability. Verifying and revisiting discovered, but unfamiliar affordances, can also benefit the robotic agent by refining observed characteristics. Random exploration or a specified strategy can be used for discovery and refining of affordances, however these methods can be sub-optimal, prone to unnecessary repetitions or suffer from poor adaptability. Another approach, is to use intrinsic motivation in the form of interest to direct the robotic agent's actions. In this thesis, the agent uses intrinsic motivation to prioritise interaction in areas of interest, where interest is generated from the novelty of known affordances of the object. It is shown that intrinsic motivation can be used to facilitate exploration of affordances, that it is adaptable and improves over random exploration.

This thesis experimentally evaluates the process of affordance discovery, learning and use in tasks. Evaluation is conducted in 2D environments via a real robotic system consisting of a fixed 3 degrees of freedom finger with a compliant 1D tactile sensor and a vision system for tracking the object. Simulations are also conducted, where some limitations of the fixed 3 DOF finger are addressed. The performance of motivated exploration approaches in this thesis to that of random exploration is also experimentally compared.

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Acknowledgements

I would like to thank my supervisor, R. Andrew Russell for his assistance and advice. He provided invaluable insight into this project and without whom this thesis would not be possible. I would also like to thank my family and friends for their support and motivation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

This thesis is about learning to manipulate an object by exploring how the object itself moves when interacted with via an agent, namely a robotic system. The agent explores the object using pushing, sliding or dragging interactions at various points to find a variety of possible moves. Exploration is motivated by what the agent finds interesting, which influences the probability of where and how the agent interacts with an object's face. Interest is derived from the novelty of a move during exploration and is correlated to the number of times moves are observed. Interest can also be derived from the potential of moves to put the object in a different state which can lead to other known interesting moves. In this thesis, the concept of *manipulation affordance* is used to describe and generalise a move's characteristics. When a sufficient number of manipulation affordances are discovered, a task can be given to the agent to enact. The agent generates a plan to complete the task by sequencing together moves generated from discovered manipulation affordances. For this thesis, the object is constrained to a 2D environment (3 degrees of freedom) for simplicity. Despite some simplifications, it is suggested that the general concept can be applicable to more complex environments and interactions.

Learning manipulation affordances is useful as the various outcomes from exploratory interaction can both be classified and quantified based on actual observation. Generalising these affordances is advantageous as they can be used in a wider variety of situations. While this can increase the complexity of an individual affordance, it reduces the overall complexity of the system, as fewer affordances need to be learned. For example, if an affordance is generalised, altering the agent's orientation perspective should not affect how the affordance is learned, despite the apparent differences in object movement relative to the agent's perspective. Similarly, identifying symmetry in an object can likewise reduce the overall complexity. It is also beneficial for agents to learn how to manipulate objects themselves. In

doing so, this can not only remove the complexity in defining and programming an agent, but enable a potentially unexpected and perhaps improved performance over that which humans themselves are capable of conceiving. In an environment which is intricate, involved and continually changing, there can arise situations where an agent is not familiar with an object and how it should be manipulated. The environment and the effects it has on the object may also be unknown. It is not practical for the programmer to consider and factor every possible scenario which the agent may encounter. Therefore, an agent which is capable of exploring and learning 'on-site' not only removes the complexity for the programmer, but also the problems associated with re-programming an agent which is already in operation. The agent is simply introduced to the new object, environment or task and solves the problem itself, learning anything new which is required in the process.

There are two predominant types of learning techniques that have been developed for manipulation in robotics: motor primitive learning, where the motion of the manipulator joints are learned in order to perform the required action, and object affordance or characteristic learning, where the motions, or characteristics of object motions are learned to determine how to perform a manipulation operation.

The motor primitive based approach, such as [1–3], focuses on learning how to move the manipulator to performs tasks. Various classical learning methods such as reinforcement learning [4] or neural networks [5] are predominately used. However, these approaches suffer from a difficulty with generalisation due to the specificity of the task, and the non-linearity and sometimes erratic nature of joint motion. As such, this approach is largely unable to be effectively adapted beyond the task that has been given, unless the agent is retrained. Neural networks can be used to generalise, but are impractical to train due to the complexity of the tasks. Hence, this type of approach is better suited to 'muscle memory' applications such as playing table tennis [6], or the ball-in-a-cup scenario [7]. In such applications, the scope of the task is very narrow, though still complex. Hence, parametrising and learning minor changes in those tasks, becomes practical. However, moving beyond a limited range of tasks remains time consuming and therefore impractical with this approach. The work of Fuentes and Nelson [8] partially address this by learning a library of grasping and in-hand manipulation 'types' using motor primitives. However, as they use a supervised genetic algorithm for learning, the system is not adaptable as it cannot discover and learn new types of manipulations.

The affordance based approaches, focus on learning how the object itself moves in order to reproduce the required action. There are two main approaches to object affordance or characteristic learning; namely simulation or prediction based, and affordance based. Simulation based systems [9] typically rely heavily on physics simulation techniques for predicting and planning motion, requiring a well defined object model and physics simulator to predict the next state of the object for planning. Interestingly, Kopicki [10] addresses this problem via probabilistic modelling from observations of interactions with an object. The model is then used to predict the behaviour when interaction takes place. While this

approach provides great versatility, a large amount of time and repetition is also required to learn and characterise the model. Furthermore, when generating a plan for a task, every possible predictable outcome must be considered after every move in order to find a valid plan. While this provides exhaustive possibilities, it also requires a large amount of processing time.

Affordance based approaches [11, 12] learn object characteristics or the object model through observations of different motions either autonomously or via demonstration. In this case, the specific type of motion is learned. For instance, a push in the middle of an object results in a straight motion, whereas a push on the side of the object results in a rotation. Typically, neural networks or probabilistic models are employed to learn the characteristics of the object. However, the planning capability for these implementations is limited or non-existent due to the lack of quantitative learning. They do not factor how far an object is moved, but simply the type of move that is performed.

The majority of these learning systems use pre-determined interaction sequences, or employ a simple random selection method for discovery. While Kraft [13] has utilised autonomous learning for discovering affordance relations for grasping, there is limited literature on exploratory methods employed in agents to intrinsically motivate and direct the next interaction, based on the characteristics of the object in its current observed state. Early literature in the psychology field defined and improved understanding of intrinsic motivation in humans [14, 15]. More recent literature discusses how intrinsic motivation can be defined for agents and the different approaches for implementation [16, 17]. There are some robotic systems which implement various forms of intrinsic motivation [18–23]. However, very few systems look at effective and efficient methods of exploration to reduce the time spent exploring an object for manipulation. One such example, is the work of Hart and Grupen [23], which use intrinsic motivation to motivate learning of generalised control policies which characterise affordances.

When exploring objects, pushing interactions have been the main focus outside of grasping in the field of robotic manipulation [24–28]. This is primarily due to the complexity of interacting with objects. Pushing is therefore considered the simplest form of interaction as either no control is required, or basic open loop control is used. The majority of robotic manipulation is primarily focused on grasping and in-hand manipulation [29–35], where in general, more useful outcomes can be produced. Limited work has been done looking at other forms of interaction when learning to manipulate objects [36–38]. These typically ignore the effect of the environment, using simple flat and empty workspaces. Recent work has investigated learning multi-object interactions in simulations [39], but do not produce any real world data.

In this thesis, affordance based learning is further explored, by expanding on the concept of affordances in a manipulation context. A different approach to ascertaining affordances is used to those discussed above, where both qualitative and quantitative characteristics of object movement are learned. Specifically:

- Affordances can be parametrised so as to generalise and use them to form plans.
- The agent is intrinsically motivated to explore the faces of an object using pushing, dragging or sliding interactions.
- A simple planning system which uses affordance information is implemented.
- Real world and simulation based experiments are presented.

The remainder of this chapter covers; a description of the main hypothesis and contributions that are made in Section 1.2, an outline of the approach taken for each of the main concepts in Section 1.3, the domain in which results are tested in Section 1.4, and finally, an outline of the remaining chapters of the thesis in Section 1.5.

1.2 Hypothesis and contributions

The fundamental objective in this thesis is to design an agent which can interact with and explore an object so as to learn its affordances. Upon learning affordances, a simple task can be designated and based on this, affordances can be sequenced together to form a plan in order to complete the task.

This thesis therefore poses the questions:

- Can an object's movement characteristics be learned and generalised for different contexts using the concept of affordances?
- Can motivated exploration of an object's characteristics improve discovery and learning of affordances over random exploration?
- Can manipulation affordances be used to form simple plans to complete tasks?

Hence, the following contributions are made:

- A proposed generalised manipulation affordance based learning system, which utilises both pushing, sliding or dragging motions to interact with objects.
- A proposed constraints concept to generalise the context of affordances in any simple environment.
- A proposed intrinsically motivated exploration algorithm to improve discovery and learning efficiency of manipulation affordances.

- A proposed path planning technique based on the nested hierarchical controller method, which uses learned affordances to plan a sequence of moves to complete a task.

1.3 Approach

1.3.1 Representations

In this thesis, a frame $\boldsymbol{A}$, represents the object's current state, containing the position and orientation of the object as well as the current coordinates of its vertices (see Figure 1.1). The object's position is defined by its centre of mass, and the orientation is obtained by determining the axis of minimum moment of inertia. The manipulator frame is given by $\boldsymbol{M}$, which describes the current position, orientation and force acting on the manipulator's end point. A face $\boldsymbol{f}$ of an object is defined by 2 adjacent vertices in an anti-clockwise link. A constraint $\boldsymbol{C}$ is described by a static line segment that restricts movement of the object in some manner. For example, the ground, or a wall. These constraints remains fixed and constant, and are known to the agent prior to operation. It is assumed there is only one object within the workspace at any given time and there are no external unaccounted forces or restrictions acting on the object or manipulator. It is also assumed that the agent's vision is not changed by some means over the duration of operation.

An affordance, denoted $\boldsymbol{P}$, is defined as a function of the initial and final frames and the manipulator state $\boldsymbol{M}$ as shown in Equation 1.1. The starting or initial frame is given by $\boldsymbol{A}_I$, and the end or final frame is given by $\boldsymbol{A}_E$. A specific goal frame is denoted by $\boldsymbol{A}_G$.

$$\boldsymbol{P} = f(\boldsymbol{A}_I, \boldsymbol{A}_E, \boldsymbol{M}) \tag{1.1}$$

where $\boldsymbol{P}$ is the affordance.

1.3.2 Learning and exploration

An affordance is learned by observing the transformation of the object between 2 frames (see Figure 1.2). The manipulator frame $\boldsymbol{M}$ is also used to identify where interaction takes place and the direction of force application. The agent chooses where to interact based on what is interesting by identifying the validity of possible outcomes of currently known affordances when performed in the current state. The higher the interest the more likely the agent will explore the area around the interaction point of the affordance. The interest can be increased for an affordance when it is likely to produce a known novel transformation or a different state which can thereby produce a known novel transformation.

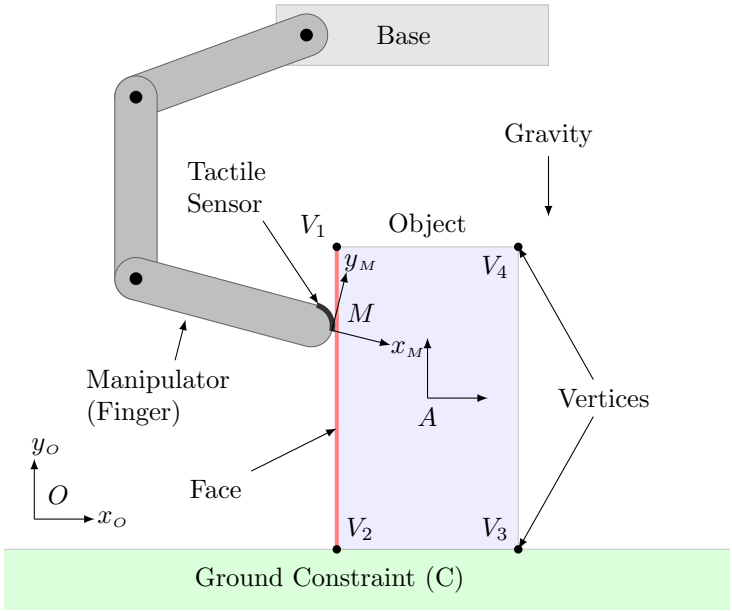


Figure 1.1: A representation of the workspace and system in this thesis, where $\boldsymbol{A}$ is a frame representing the centre of mass position and state of the object, $\boldsymbol{M}$ is the state of the finger and O is the origin of the visual reference frame. Gravity is shown for reference. The base and ground constraints are fixed in position throughout testing operations.

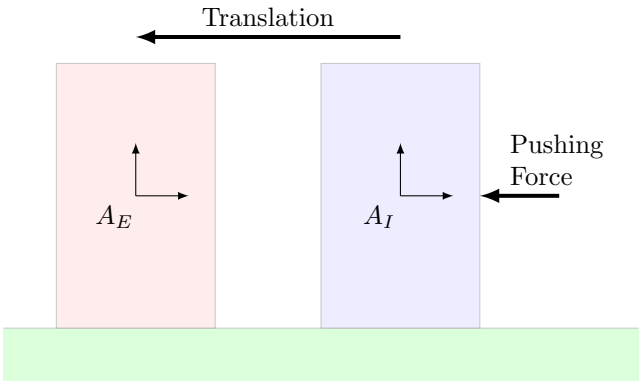


Figure 1.2: Initial $\boldsymbol{A}_I$ and final $\boldsymbol{A}_E$ frame states due to a pushing interaction. The difference between the two frames describes the affordance.

1.3.3 Planning

To use learned affordances, a task can be given to the agent which requires the object be moved from one state and/or position to another using one or more moves. The agent can form a plan by sequencing multiple affordances together, generating a series of moves. The object is then moved through a series of states, via interaction, until the desired state and position is attained. A modified version of the nested hierarchical controller is used to plan a path for the object to be moved through. This involves generating a tree of moves from discovered affordances and using the moves which result in the object eventually matching the goal state. After each move is executed the agent re-plans to factor slight differences or failures. To set a task, the start and end states are shown or given to the agent.

1.4 Domain of testing

A robotic system has been designed for experimentation where a typical setup would consist of the following:

- A 3R robotic manipulator operating on a 2D plane. The manipulator end point houses a tactile sensor for force feedback.
- Experimental "2D objects" which are geometrically simple, for example variations of rectangular or triangular shapes. These objects are assumed to be rigid, of uniform density and having constant coefficient of friction.
- A pseudo 2D environment containing the robotic manipulator and object. The workspace is defined as the area reachable by the manipulator.
- The environment is observed by a camera facing the object and manipulator. Visual tracking algorithms are used to track the object.

Due to the complex nature of manipulation, it is difficult to define all the expected outcomes. In fact, unexpected outcomes are just as (if not more) interesting than achieving expected outcomes. Some commonly expected outcomes are shown in Figure 1.3, where both pushing, dragging, or sliding motions are used to interact with the object. It is also expected that some interactions may fail. For simplicity such failures are ignored, as determining the cause of failure is beyond the scope of this thesis.

Planned manipulation sequences are desired to work as expected. Unexpected outcomes during execution of plans are considered a failure. However, if a new plan can be formed and completed after failure in order to recover, it is not considered a failure. Some typical plans are shown in Figure 1.4,

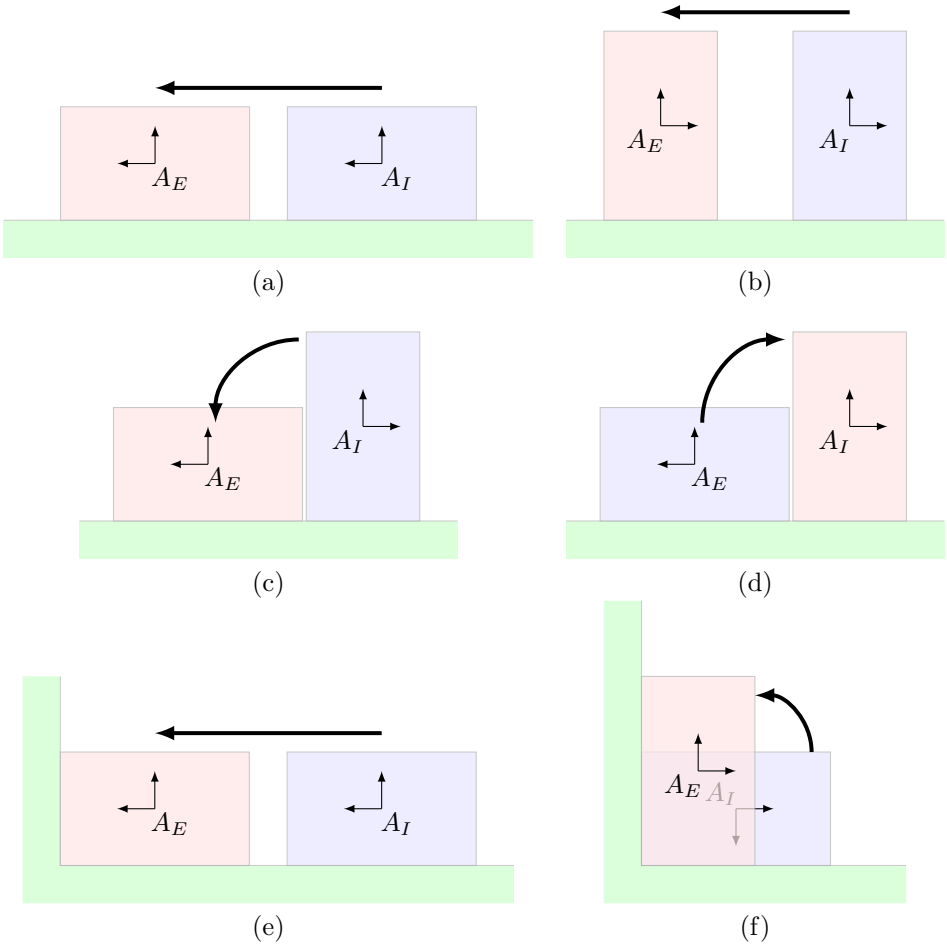


Figure 1.3: Example of affordances that the agent may encounter. There may be multiple means to perform the same affordance, hence no specific interaction is shown.

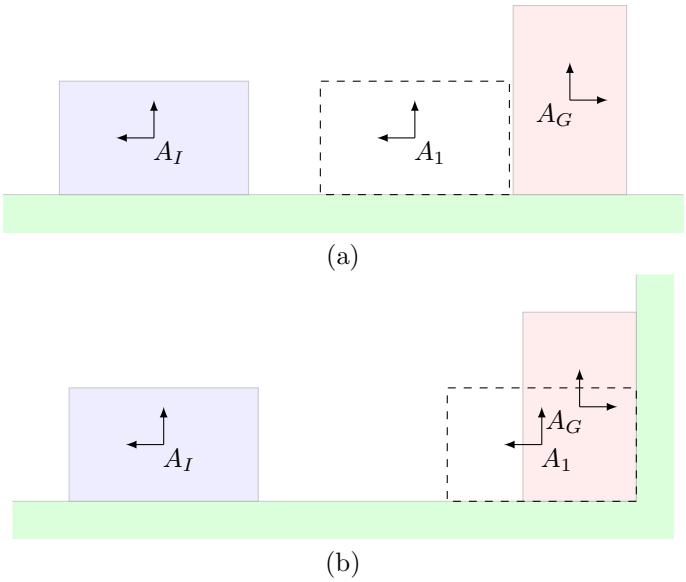


Figure 1.4: Example of alternate plans that can be formed to complete a particular task. The starting state of the object is given by $\boldsymbol{A}_I$, the goal is given by $\boldsymbol{A}_G$ in red and the intermediate steps are shown with dotted lines and their sequence number, for example $\boldsymbol{A}_1$.

which use some of the outcomes from Figure 1.3. It is expected plans will typically sequence between 2 to 5 transformations to complete simple tasks.

A simulation system has also been developed to explore more complex scenarios which are beyond the scope and capability of the physical robotic system. For simulations, control of the manipulator is forgone in favour of simply applying a point force on the object analogous to the manipulator of the robotic system.

1.5 Outline

The following outlines the subsequent chapters of this thesis:

Chapter 2: Environment and Agent

Details the environment and agent framework used in this thesis.

Chapter 3: Manipulation Affordances

Presents the concept of manipulation affordances and provides a basic overview of the fundamental concepts presented in this thesis.

Chapter 4: Generalising Affordances via Constraints and Symmetry

Introduces constraints for manipulation affordances and details how affordance are constructed to factor them. The issue of symmetry is also discussed.

Chapter 5: Intrinsically Motivated Exploration and Learning

Presents the method by which motivated exploration is conducted.

Chapter 6: Results

Presents the robotic and simulation systems used for experimentation. The results of both the robotic and simulation systems experimentation are also presented.

Chapter 7: Conclusion

Summarises the findings of the thesis and discusses future work.

Chapter 2

Environment and Agent

This chapter describes the environment scope and agent framework used in the following chapters of this thesis. Section 2.1 describes the environment, Section 2.2 describes the agent behaviour and lastly Section 2.3 summarises the chapter.

2.1 Environment Description

The typical environments which humans reside in are extremely complex as Kemp discusses [40]. Even when considering an essentially 2D environment, there are many factors which can influence the movement of an object. For example; surface compliance, friction, orientation, static or non static state. Each of these lead to countless possibilities and require complex solutions. Presently, in complex unstructured environments, systems are generally limited to identifying and grasping objects [41–43]. The computational requirements to identify, construct plans for, and manipulate objects without grasping them in a complex environment are considerable, despite the high accessibility in computing power. The design requirements for such systems are extensive, making a fully realised system that is capable of performing complex manipulations unaided, extremely difficult to attain. As such, the afore mentioned requirements are beyond the scope of this thesis. Therefore, the environment must be simplified to make the problem more approachable. This involves removing as many complications from the environment and object as possible. Hence, the objects used are simple rigid convex geometrical shapes and the environment is described by simple straight line segments acting as constraints that restrict movement of the object.

Many scenarios in a 3D environment can be reduced to scenarios analogous to a 2D environment. This is due to a reduction of the degrees of freedom caused by various physical constraints acting on the object, restricting motion. It can be observed that the majority of object states in a 3D environment have

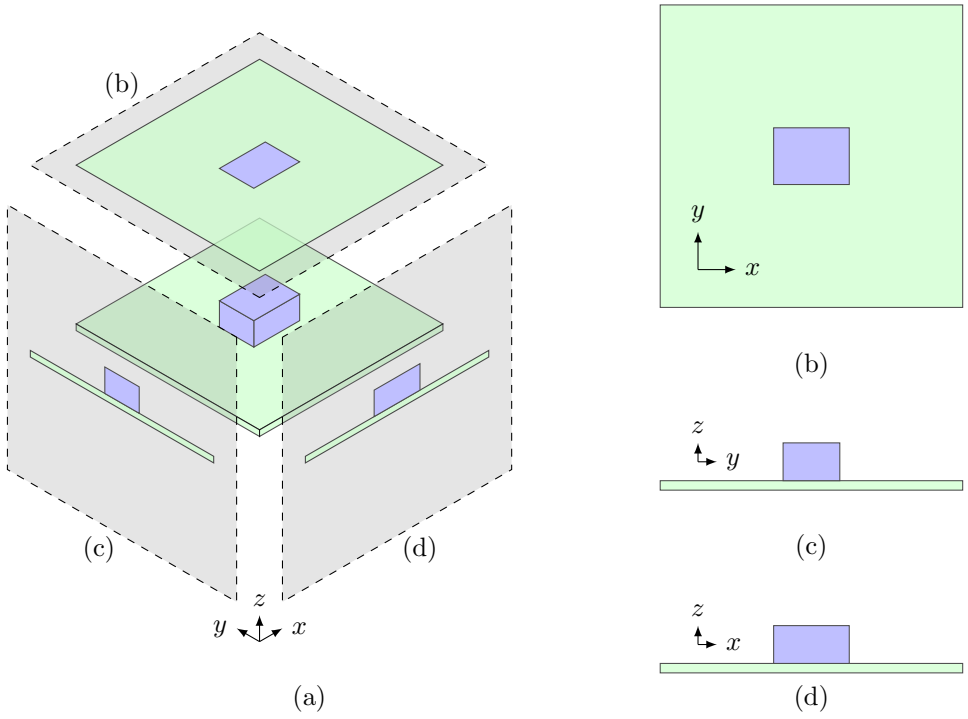


Figure 2.1: Projecting a 3D world into 3 separate orthographic 2D views. (a) shows how the projections are formed and relate to the 3D world. (b) shows the z projection, (c) shows the x projection and (d) shows the y projection.

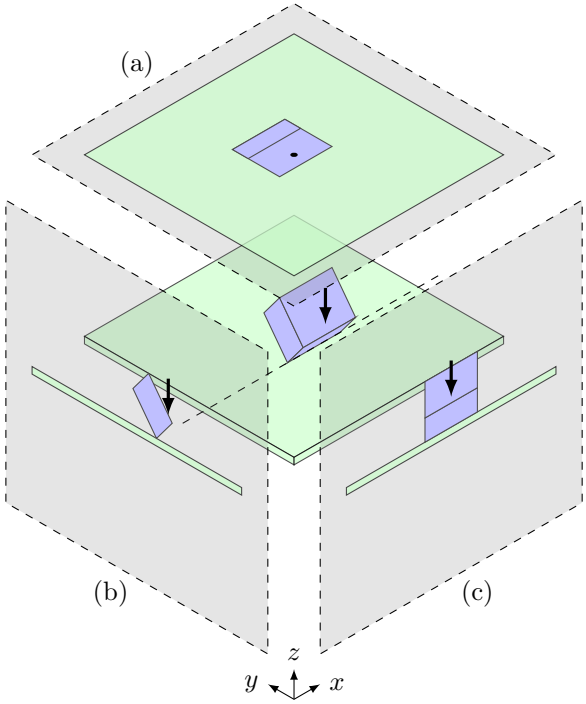
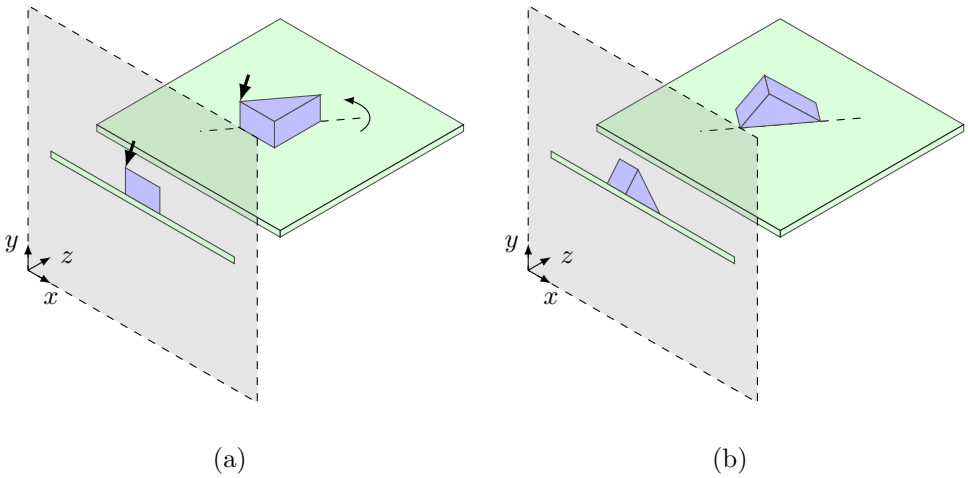


Figure 2.2: Problems with rotation in projections. Rotating the rectangle in any axis will result in at least one invalid projection. In this case projections (a) and (c) become invalid.

at least one degree of freedom restricted. This is primarily due to the effect of gravity and a surface in the environment restricting movement from the force of gravity. For example, consider when a simple geometrical shape is placed on a flat level table. The 3D environment can be broken down to three 2D orthogonal projections based on the orientation of the frame of reference of the object, as shown in Figure 2.1. Each projection represents a 2D scenario where the agent can interact with the object, learn affordances and perform tasks. This breaks the problem down a step further, enabling complex problems to be further simplified and better understood in a 2D environment, before attempting to solve them in a 3D environment.

However, creating orthogonal projections in an environment which does not sufficiently reduce the degrees of freedom can lead to problems. In the case of Figure 2.1, a rotation about any projection's z-axis will invalidate the other projections. For example, consider Figure 2.2. When working from side projection (b), applying a force to rotate the object upright produces a valid result. However, in the other projections, the object's shape has changed, invalidating them. While it is possible to produce ideal forces which do not cause the object to rotate, rotation is a major part of moving an object. Therefore, when using orthographic projections, only environments which reduce the degrees of freedom to 3 for objects are considered for use in this thesis.



(a) (b)

Figure 2.3: Problems with object uniformity in projections. Attempting to manipulate a triangle object that is not uniform along the projections z-axis. While (a) may initially look like a valid interaction, the result in (b) demonstrates that the transformation produces an invalid 2D state.

Furthermore, when considering a 3D object in a 2D projection, unless the object is uniform along the projection's z-axis (the axis normal to the projection plane), when rotated about the z-axis the result is invalid. For example, consider the triangle in Figure 2.3a. A rectangle is produced from a projection, however the motion of the object does not reflect that of a rectangle. The projection is not valid as the object is not uniform along the projections z-axis. While the agent could potentially attempt interaction, the object shape would appear to change as shown in Figure 2.3b.

In this thesis, only scenarios where an object rests on a flat, uniform surface are considered. Of the three 2D projections created from a 3D environment, they can be classified into two distinct types; Surface projections and Side projections. Surface projections only include those whose z-axis are parallel to gravity (See Figure 2.1b). Side projections involves those where gravity is perpendicular to the projection's z-axis (see Figures 2.1c and d as well as Figure 2.4b and c). For simplicity, this thesis only considers side projections where the environment restricts an object's motion to 3 degrees of freedom and the object is uniform along the projection's z-axis. Therefore, in Figure 2.4, projection (c) is the only projection that will be considered in this thesis.

2.2 Agent

The primary task of the agent is to explore and learn manipulation affordances. It has been designed such that it can be situated in either a physical or simulated world. Due to the specificity of the task, that is manipulating an object, the agent's behaviour is functionally simple.

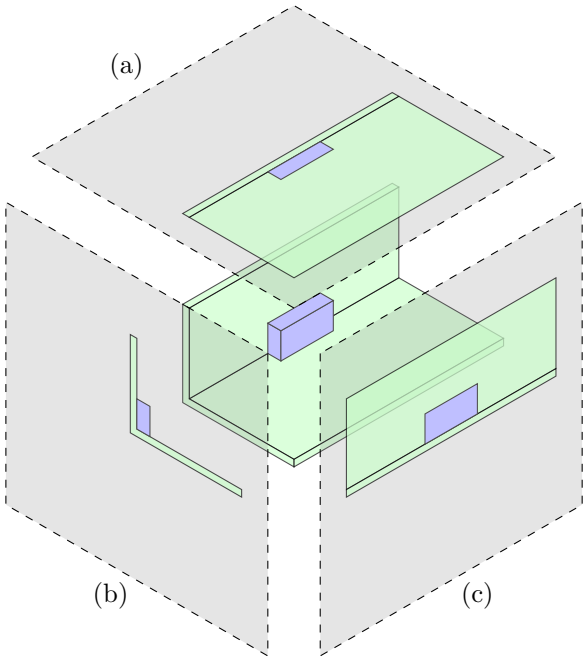


Figure 2.4: Projections produced when an edge is encountered. (c) is the projection scenario typically used for side projection scenarios as the DOF is restricted to 3.

2.2.1 Environment and Object

Robotic manipulation is complex. An object's own complexity can significantly increase manipulation complexity. Therefore, geometrically simple shapes are used for experimentation. The specific objects used in this thesis include a rectangle and isosceles triangle. Despite the simplification, these shapes can represent approximations of a wide range of shapes encountered in everyday human life. For example, a phone, container or plate on a table can be approximated with a rectangle. For this thesis, circular objects are omitted due to limitations later discussed in Chapter 4.

As discussed in Section 2.1, the environment is designed such that an object is restricted to 3DOF, on a 2D plane. It is represented by straight line segments for the agent, which are assumed static and impenetrable.

2.2.2 Interaction

In this thesis, single discrete interaction points are implemented by describing only the position and direction of force of the manipulator end point. This generalises the interaction by abstracting the manipulator interaction information. Hence, ideally any manipulator configuration can be used, as long as

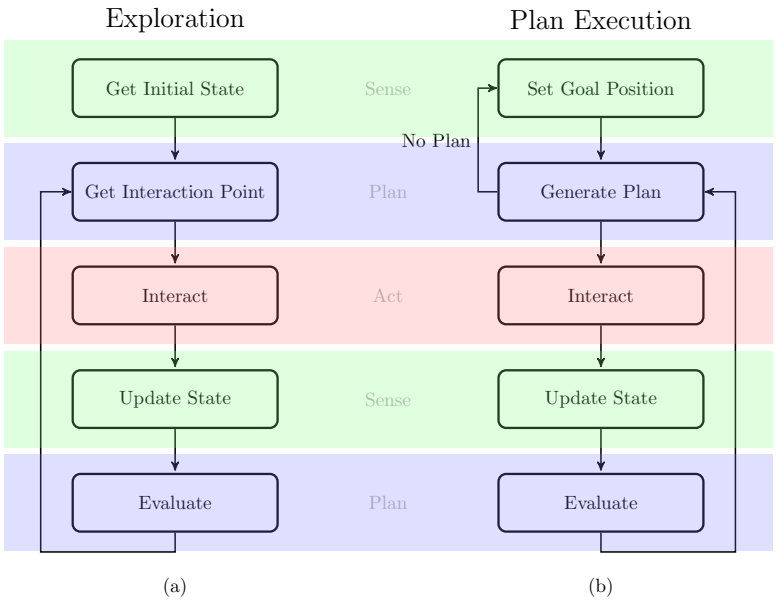


Figure 2.5: Control flow of the agent for exploration in (a) and plan execution in (b).

it is capable of applying the necessary forces at the required points. Chapter 3 discusses this in further detail. For simplicity, the directions of force are limited to normal or sheer, that is pushing or sliding and dragging actions. To apply a pushing force, the agent applies a normal force at the desired interaction point. To apply a sheer force, the finger applies a normal force at the interaction point in the same manner as pushing, to maintain traction. A tangential force is then added to the fingers total force vector, potentially sliding or dragging the object. The agent continues to maintain the normal force on the object as best as possible.

2.2.3 Operation

The basic behaviour of the agent remains the same in both the physical and simulated world. A simple state based system is used, where at the highest level the agent is either exploring, or planning and executing a plan. In both cases, the agent broadly follows the “Sense Plan Act” methodology [44]. During the Act phase the agent is continually updating the object state and sensors to maintain closed loop control, again following the “Sense Plan Act” methodology.

Exploration

For exploration, the object frame is continually updated. When an interaction is desired, the initial frame is captured and saved. The agent then determines the point on the object to interact with (detailed further in Chapter 3). When interaction ends, the start and end frames are evaluated and an affordance is either matched or added. The system then waits for any movement of the object to cease and again determines a new point to interact with, continually repeating the process (see Figure 2.5a).

Planning

For planning, once the agent has been shown the final goal position and the object is placed in its initial position, the system starts generating a plan. If a plan is found, the agent performs the first move in the plan, applying a force in the position and direction indicated by the plan. The agent continues interaction until the expected outcome is achieved or a period of time has expired. The agent then re-plans, updating with the object's new position and state, repeating the process until the goal is attained (see Figure 2.5b).

2.3 Summary

This chapter discussed how environments can be complex and thereby require simplification to make the manipulation problem more tractable. It detailed how orthogonal 2D projections can be used to describe some 3D scenarios. While not all 3D scenarios can produce projections which are valid, they cover a limited variety of scenarios which are valid and worth while exploring. There are two main types of projections:

- Surface Projection - View point which looks down onto the object, following the direction of gravity.
- Side Projection - View point which looks side on to the object.

This thesis focus's on the Side Projection manipulation problems and the objects are assumed to be geometrically simple, using shapes such as a rectangle and isosceles triangle. In order to explore the environment an agent has been designed with a behaviour that follows the basic methodology of "Sense Plan Act" and is common to both the physical and simulation systems. Point based interaction is assumed as it decouples manipulator information, allowing solely force position and direction to characterise the interaction.

The agent has two primary modes of operation:

- Exploration - Involves discovering and learning new affordances
- Plan execution - Involves using the affordances to formulate and execute a plan to complete a task.

These concepts underpin the proceeding chapters.

Chapter 3

Manipulation Affordances

As discussed in the introduction chapter, manipulation can be broken down into two main areas; joint motion and object motion. This thesis focuses on object motion, or specifically *manipulation affordances*. This chapter introduces the concept of *affordances* used throughout this thesis. Due to the dependencies of many concepts in this thesis, primary focus is given to detailing *manipulation affordances*. Related concepts, such as planning and basic exploration of objects is detailed to add context to the usage of *affordances*. Subsequent chapters expand upon the basic concepts presented in this chapter, providing further detail and addressing some limitations that are encountered.

In this chapter; Section 3.1 discusses perception and interaction, Section 3.2 introduces and defines manipulation affordances, Section 3.3 describes the approach to learning affordances in this thesis, Section 3.4 details the basic planning method implemented for utilising affordances in user set goals or tasks and finally, Section 3.5 summarises the chapter.

3.1 Perception and Interaction

Humans have an extraordinary capacity to perceive and utilise complex manipulation skills. That is, humans can see a manipulation performed and understand how it was done simply by observing it. The manipulation can then be reproduced by the observer without having previously performed the specific manipulation. Part of this ability, involves advanced motor control capability of joints and the fine tactile resolution and sensitivity of the skin for closed loop control. However, and arguably more importantly, the ability to perceive, learn and generalise object motion due to interaction forms a significant part of human manipulation prowess.

Humans can reproduce observed manipulations using a variety of tools, effectively augmenting the

configuration of the manipulator from what was used when originally learned [45]. Consider therefore, a study [46], where a computer simulation which displays and simulates real physics of planar objects in 2D, on a screen. A single point manipulator represented by a mouse pointer can be used to push objects around on the screen and is controlled by an electroencephalography (EEG) device connected to a human. Humans are able to use their prior knowledge of manipulations and move objects in the simulation by simply using their mind. As discussed in the introduction chapter, there are two main fields of manipulation, joint motion and object motion. The predominant focus in research has been on the joint motion approach. However, learned information can not be directly transferred to other system configurations as the information is highly dependent on the system's own configuration. In the EEG study, as manipulator control is effectively simplified down to a single direct interaction point of force, it can be seen that learned object motion is being utilised by the test subjects. This suggests that there is some detachment between learning object motion and motor skills in the context of manipulation. Humans can envisage how an object must be moved to complete a task. An appropriate method of control to perform that move can then be used based on the current manipulator capabilities. As the EEG simulation indicates, it is sufficient to simply identify the necessary point and direction of interaction which caused the motion in order to characterise it. Once manipulator control skills are sufficiently mastered (or programmed), the correct interaction can be produced. The control capabilities can also help to compensate for small differences in identified force directions and magnitudes. Therefore, as alluded to above, systems such as Hart and Grupen [23], while highly generalisable and capable, are dependent on maintaining the exact manipulator configuration in order to function. Unlike the humans using the EEG, the system must relearn its previous knowledge when the manipulator is augmented.

Visual information plays a significant role in perceiving and learning manipulations. In many cases the tactile sensing ability of the manipulator may not be necessary. Visual information may supply sufficient feedback for control of the manipulator. However, there are some forms of manipulation which require very fine and/or fast control skills. These typically involve muscle memory applications due to their complexity, where actual control of the manipulator is learned [6, 7]. Another is in-hand manipulation, where the object is fully constrained and in a well known controllable state [6, 35]. In these applications, complex actions can be performed quickly and accurately from learned motions [2]. For manipulations which are less constrained, the exact necessary forces required may be vastly different depending on circumstances. As Sikka demonstrates [47, 48], it is highly advantageous to characterise contact forces via tactile sensing for manipulating objects. Therefore, without tactile sensing of some type, learning the exact motions and forces required becomes increasingly more difficult due to the wide variety of possibilities. Some interactions, such as pushing may have sufficient feedback simply from visual information. Other interactions such as dragging or sliding, require tactile feedback to maintain traction, or prevent too much force being applied.

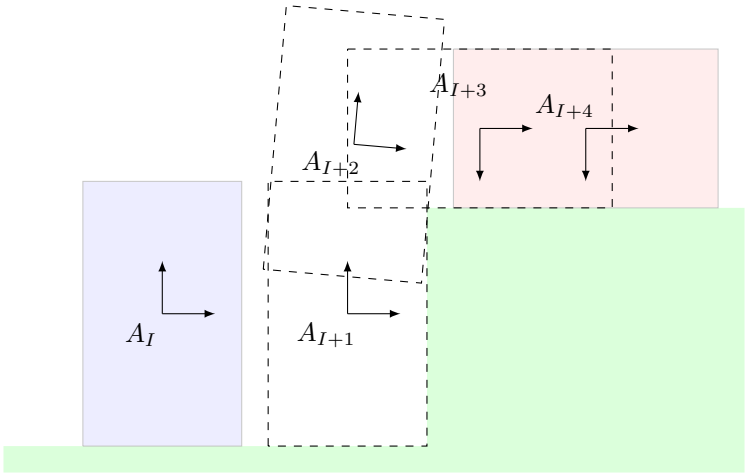


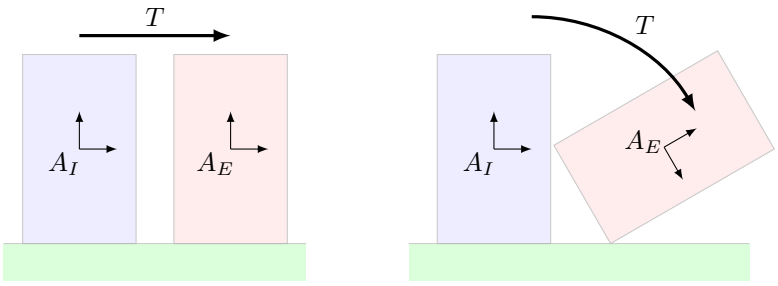
Figure 3.1: Moving an object through a complex path from start $\boldsymbol{A}_I$ to finish $\boldsymbol{A}_{I+4}$. The transformation consists of several simple translations and rotations which form the single complex task.

The remainder of this chapter focuses on defining a method by which object motion is learned such that only the position and direction of force interaction information is required, rather than learning joint motions.

3.2 Defining Manipulation Affordances

Affordance is defined as the quality of an object or environment which allows an agent (human or robot) to perform an action [49]. For example, a rigid object may afford pushing, grasping, sliding or even cutting and breaking, but does not afford bending. Various works [12,50,51] use this concept to describe manipulation techniques such as basic grasping and pushing actions. In the context of this thesis, manipulation affordances describe the quality of an object which allows a specific type of transformation due to manipulation interaction such as pushing, sliding or falling. That is, objects in this thesis are assumed to afford pushing, sliding and falling. However, the availability of each can be dependent on the context of the object within the environment. Some affordances may not be possible due to the configuration of the environment or object itself.

When observing an object undergoing some form of complex transformation due to interaction with an external force, the transformation as a whole can be quite involved. However, when this transformation is broken down into smaller elementary ones, each can be fairly simple. Often, complex transformations can be approximated by a series of simple ones, where each is the result of an affordance being performed. Figure 3.1 illustrates a complex transformation. The path of the object is not simple enough to solely consider the start and end points and interpolate between them to obtain the path. In



(a) (b)

Figure 3.2: Basic affordance concept. The object is displaced from $\boldsymbol{A}_I$ to $\boldsymbol{A}_E$ and a comparison is made between the two frames to determine the transformation $\boldsymbol{T}$. a) is a simple translational transformation and b) contains both a translational and rotational transformation.

this case, breaking the transformation down into a simple sequence of translations and/or rotations can greatly reduce the complexity. In doing so, each component in the sequence can be considered as a simple point to point transformation. At each point in time the object affords certain transformations which can be performed. In this thesis, describing the type of transformation, or what the object affords so as to perform a transformation is defined as a *manipulation affordance*. For convenience manipulation affordances are sometimes simply referred to as *affordances*.

3.2.1 Transformations

Formally, a manipulation affordance, is defined as characterising the rigid body transformation of an object from a frame $\boldsymbol{A}_I$, to $\boldsymbol{A}_E$ as depicted in Figure 3.2. A frame describes the object's position and orientation within the environment. They are situated at the centre of mass of the object and aligned with the object's orientation so that $\{x, y, \alpha\} \subseteq \boldsymbol{A}$. The x and y components are the position of the frames axis and the α component is the orientation of the frame relative to the agent. $\boldsymbol{A}_I$ and $\boldsymbol{A}_E$ describe the start and end frames of a transformation respectively. The time between frames depends on the time taken for the transformation to occur. In the case of Figure 3.1, there may be a different interval of time between $\boldsymbol{A}_{I+1}$ to $\boldsymbol{A}_{I+2}$ and $\boldsymbol{A}_{I+3}$ to $\boldsymbol{A}_{I+4}$. An affordance, denoted $\boldsymbol{P}$, can therefore be represented as the co-domain of a function g of two frames, characterising the change between them:

$$\boldsymbol{P} = g(\boldsymbol{A}_I, \boldsymbol{A}_E) \tag{3.1}$$

The frame position is with respect to the agent's visual frame $\boldsymbol{O}$ which remains static during object motion, hence the $\boldsymbol{A}$ frames themselves are with respect to $\boldsymbol{O}$. For convenience this is assumed for all

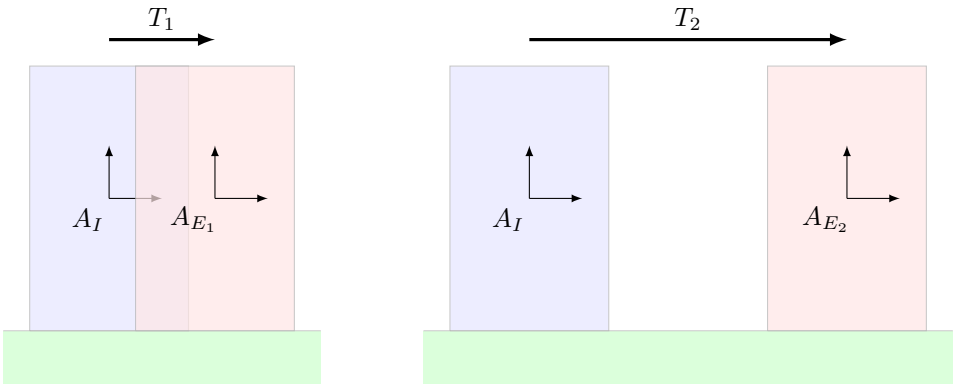


Figure 3.3: Two different transformations $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ with identical transformation descriptors $\hat{\boldsymbol{T}}$ describing the same types of motion in both (a) and (b), but of different magnitudes, $|T_1|$ and $|T_2|$.

figures in this thesis.

To illustrate the concept further, consider Figure 3.2a, where the object is displaced from its initial state of frame $\boldsymbol{A}_I$ to frame $\boldsymbol{A}_E$. The motion can be characterised by a transformation vector $\boldsymbol{T}$ describing the difference in $\{x, y, \alpha\}$ between 2 frames:

$$h = g|_{\grave{\boldsymbol{A}}} \tag{3.2}$$

where, $\grave{\boldsymbol{A}}$ is a set of the (x, y, α) components from both $\boldsymbol{A}_I$ and $\boldsymbol{A}_E$ frames. Hence:

$$\boldsymbol{T} = h(\boldsymbol{A}_I, \boldsymbol{A}_E) \tag{3.3}$$

where, $\boldsymbol{T} \subseteq \boldsymbol{P}$

$\boldsymbol{T}$ describes the difference of the object's position and orientation, indicating both the direction and magnitude of motion. This is useful for describing specific distances the object must be moved. Ideally, for identical types of motion such as those in Figure 3.3, generalising the transformation such that all variations in distance can be described is desirable. To achieve this, the effective direction of $\boldsymbol{T}$ is determined by normalising it to obtain a unit vector (see Equation 3.4), denoted the transformation descriptor $\hat{\boldsymbol{T}}$. A transformation can then be produced with any desired magnitude while maintaining its type. The actual observed magnitude of a transformation is denoted T^d. Both the $\hat{\boldsymbol{T}}$ and T^d form part of an affordance and as such are a subset of $\boldsymbol{P}$ ($\hat{\boldsymbol{T}} \subseteq \boldsymbol{P}$, and $T^d \subseteq \boldsymbol{P}$). A scaling factor, s_f is used to scale the orientation component (T_α) when calculating $\hat{\boldsymbol{T}}$ to balance it against the x, y components of the transformation descriptor. Hence, $\hat{\boldsymbol{T}}$ is defined as:

$$\hat{\boldsymbol{T}} = \frac{\boldsymbol{T}}{|\boldsymbol{T}|} \cdot \begin{bmatrix} 1 \\ 1 \\ s_f \end{bmatrix} \tag{3.4}$$

The value of s_f is dependent on the object's size with respect to the agent's perspective (see Appendix A for derivation of s_f). This is necessary as normalising variables with different units can cause one to be rendered insignificant compared to another due to differences in unit magnitude. To ensure that the transformation remains valid, the following conditions are assumed when describing affordances:

- The object must be stationary and stable at the initial frame $\boldsymbol{A}_I$ of interaction.
- The object must have a small rate of change in the direction of $\hat{\boldsymbol{T}}$ at the final frame $\boldsymbol{A}_E$ of interaction.

If these are not met, the result may lead to an unstable state in which it is physically impossible for the object to remain in.

3.2.2 Rotation

A transformation is classified as a rotation when the dominant component of $\hat{\boldsymbol{T}}$ is the rotational component α. To determine the dominant component, a relationship can be established where the rotational (α) and translational (x, y) components must have a combined magnitude of 1.

$$1 = \hat{T}_\alpha^2 + d_{xy}^2 \tag{3.5}$$

where,

$$d_{xy} = \sqrt{\hat{T}_x^2 + \hat{T}_y^2} \tag{3.6}$$

Given this relationship, a threshold of $\frac{1}{\sqrt{2}}$ can be used in Equation 3.7 to classify the transformation as a rotation or translation.

$$\boldsymbol{T}_{type} = \begin{cases} rotation, & \text{if } \hat{T}_\alpha \geq \frac{1}{\sqrt{2}} \\ translation, & \text{if } \hat{T}_\alpha < \frac{1}{\sqrt{2}} \end{cases} \tag{3.7}$$

Parametrising the magnitude of a transformation works well with translations as the states between initial and final frames of the object remain stable, and the final state generally remains the same as the initial. This is possible because gravity and the ground constraint restrict the motion of the object along

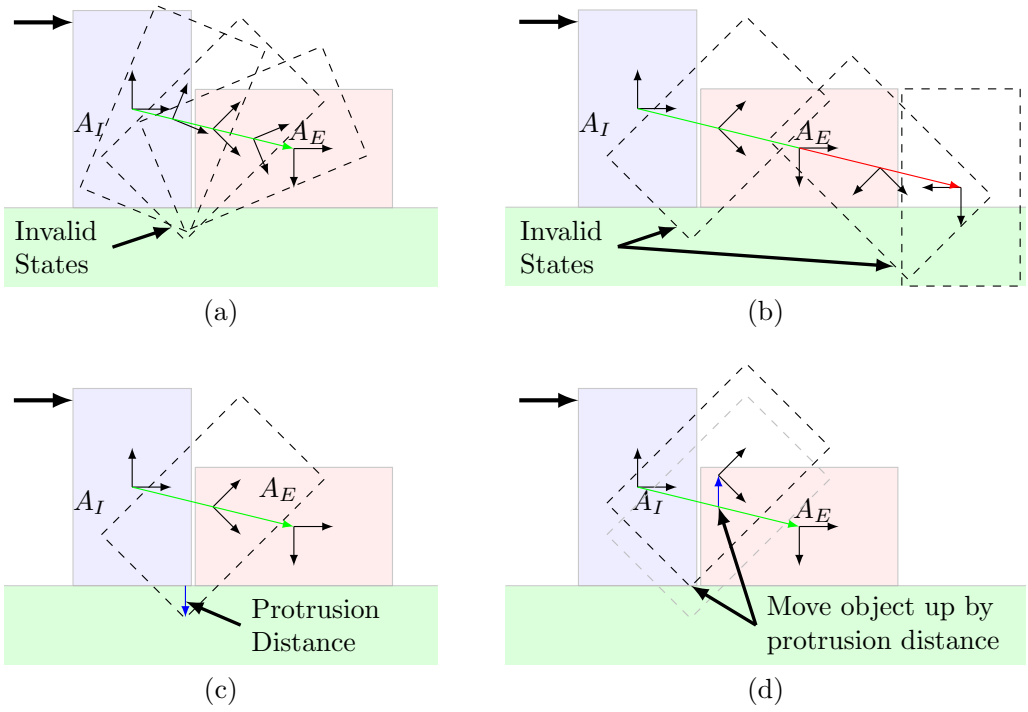


Figure 3.4: Demonstrates how invalid states are produced when parametrising rotational motion in both (a) and (d). In (b) and (c) the invalid state is compensated for by offsetting the object. Both (b) and (c) still have the issue of having an unstable state.

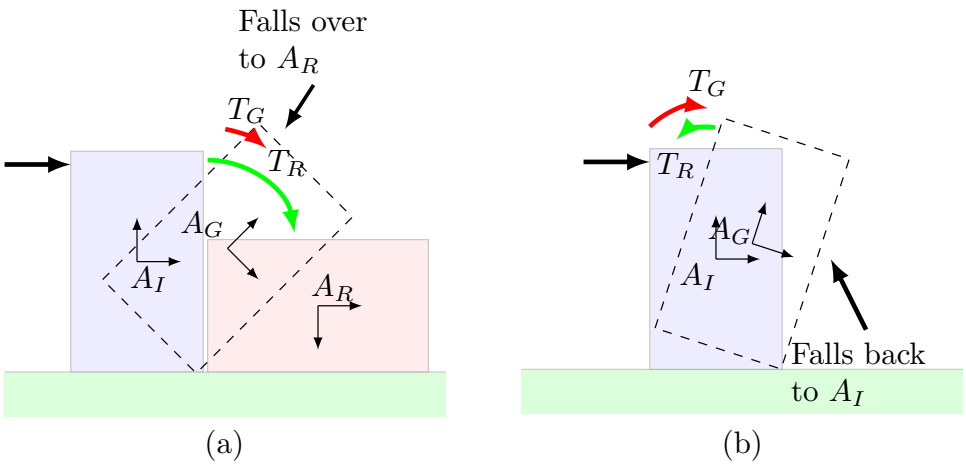


Figure 3.5: The effect of parametrising rotational motion by changing the magnitude (T^d) of a transformation $\boldsymbol{T}$. Each instance demonstrates the desired transformation $\boldsymbol{T}_G$ with a red arrow. The actual resulting transformation $\boldsymbol{T}_R$ is shown with a green arrow. The initial frame is given by $\boldsymbol{A}_I$, and the desired frame state is given by $\boldsymbol{A}_G$, and the resulting frame is given by $\boldsymbol{A}_R$ where applicable. In (a), the $\boldsymbol{A}_G$ frame is in a state where the object is beyond the tipping point. Hence, it will result in the object falling over to $\boldsymbol{A}_R$. In (b), $\boldsymbol{A}_G$ frame is in a state where the object is not beyond the tipping point. Hence, it will result in the object falling back to $\boldsymbol{A}_I$.

and about certain axis. When considering transformations which have a dominant rotational component, parametrising the magnitude as shown in Figures 3.4a and b, leads to complexities. Often vertices of the object protrude the ground when scaling the magnitude of a rotational transformation. These protrusions render the states of the object invalid. For example, in Figure 3.4c, the goal frame's $\boldsymbol{T}_G$ is set to half of the real $\boldsymbol{T}$ by scaling the magnitude by a factor of 0.5. This however, causes the frame $\boldsymbol{A}_G$ to protrude the ground constraint, creating an invalid state. The protrusions can be compensated for by offsetting the object normal to the distance it protrudes the ground constraint, as shown in Figure 3.4d. However, adjusting the height of $\boldsymbol{A}_G$ does not have the same affect for scenarios such as (b) in cases where the magnitude is scaled by a factor greater than 1. The state is not possible to attain without adjusting the direction force is applied to the object, potentially creating a different affordance. Hence, for rotational motion, the magnitude should only be scaled in the range (0,1].

However, even though protrusions can be compensated for, due to the effects of gravity and the ground constraint, there are no states that are stable between the initial and final frames of the transformation. The final object state has two possible results. Either, the rotational transformation is not performed sufficiently and the object falls back to its initial state (see Figure 3.5a), or the transformation causes the object to move past its 'tipping' point, resulting in the object falling over to its final state (see Figure 3.5b). Though it may be possible for multiple points of contact to be used to stabilise the object,

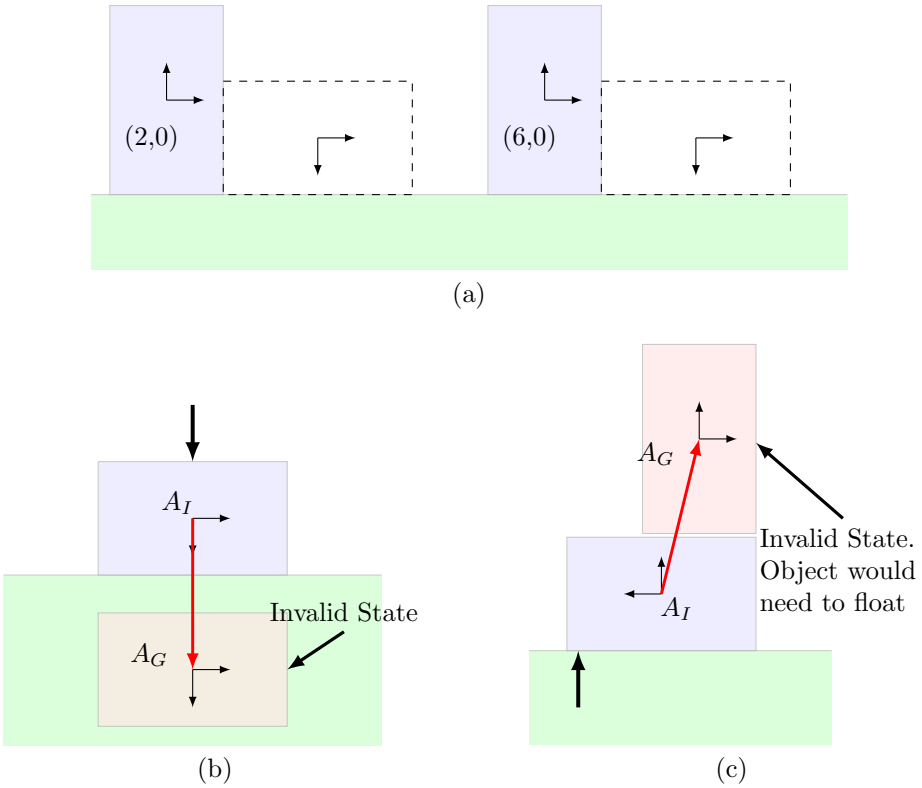


Figure 3.6: Demonstrating the affects of initial position and orientation and why initial conditions are essential. In (a) the initial position does not change what the object affords. In the case of (b) the object is expected to be transformed into the ground. In the case of (c) the object is expected to be transformed such that it is effectively floating above the ground.

which would be useful for alignment tasks, the practical use for such rotational motion is unlikely to be of much value. For example, when releasing an object which has undergone partial rotation, it would revert to a stable state. Therefore, pursuing this avenue further was deemed beyond the scope of this thesis and can be explored in future work. Hence, when gravity is affecting the object, rotational motion can not be parametrised. That is, the magnitude of the transformation must be constant and equal to T^d.

3.2.3 Initial State

When considering transformations, the conditions necessary for them to occur are as equally important as the transformation itself. One of these conditions is the initial state of the object, which is obtained from the initial frame of the transformation, $\boldsymbol{A}_I$. The initial Cartesian position of the object has no effect on the type of transformation that can be produced in uniform environments. For example, in Figure 3.6a, the two different positions of the object do not change what it affords. However, if the object moves to a

section of the environment which is non-uniform, the object's affordances can change. This is addressed in chapter 4, hence for the purposes of this chapter it is assumed the environment is uniform. It can be observed that the initial orientation of the object can drastically affect the type of transformation that can be produced. Consider Figure 3.6b and c, which demonstrates the effect of ignoring initial conditions when performing transformations. The resulting transformations consequently produce unrealistic or invalid object states. However, if the initial orientation condition is adhered to, invalid states are no longer produced.

To formally define this, the state of a frame is defined by $\boldsymbol{S}$, where $\boldsymbol{S} \subseteq \boldsymbol{A}$. Hence, the initial state can be obtained from the initial frame ($\boldsymbol{A}_I$) in a transformation and is denoted $\boldsymbol{I}$. Presently, $\boldsymbol{I}$ only consists of the orientation of the initial frame, however subsequent chapters will expand the definition of $\boldsymbol{I}$.

3.2.4 Agent Interaction

Thus far, only the transformation and initial conditions of the object have been defined for the affordance. It is difficult to accurately reproduce any transformation without some knowledge of the interaction that was performed to produce it. A method to relate the cause (action performed) with the effect (reaction outcome) is required.

It is typical for some systems, such as the work of Montesano *et al* [12], to relate object features to set actions, either learned or programmed. These actions are classified into distinct classes such as push, tap or grasp, where object features consist of size, shape and colour. Other work, such as Ridge *et al* [52], use three distinct pushing modes; top, middle and bottom, to explore the different motion that can be obtained from pushes applied to different positions on an object. While this is good, it lacks the ability to encode specific interaction locations which may, for example, be located between the top and middle points of an object. Given that the exact location of interaction on an object surface can affect the outcome of the movement generated [53, 54], it is of benefit to capture and utilise this information in order to more completely define affordances.

Faces and Contact Points

To properly characterise the type of interaction applied to the object, the concept of *face* segmentation is introduced. The shape contour is obtained and a polygon approximation is used to generate a set of vertices describing the object. Each line segment created from two sequentially linked vertices of the polygon, represents a *face* $\boldsymbol{f}$, which can be interacted with by the agent. These faces facilitate easy classification of similar transformations when a different face is involved. Consider Figure 3.7, which

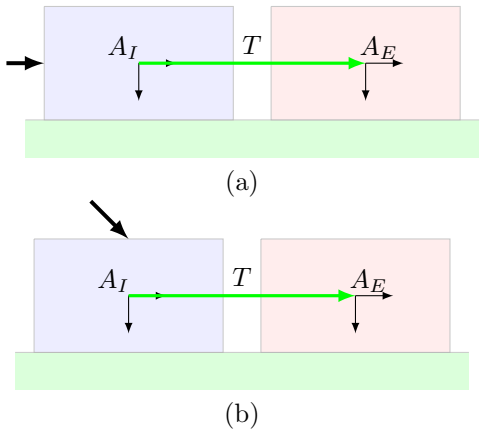


Figure 3.7: Identical transformations $\boldsymbol{T}$, with difference faces used to perform the transformation.

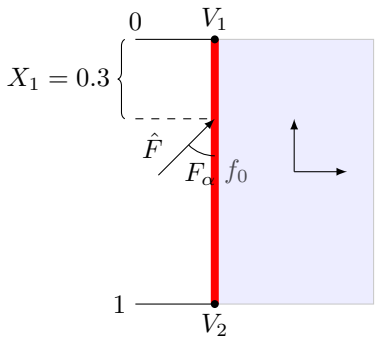


Figure 3.8: Face representation. Demonstrates how the interaction point X_1 and direction of the force $\hat{\boldsymbol{F}}$ is referenced from a face.

contains two transformations with identical $\boldsymbol{I}$ and $\hat{\boldsymbol{T}}$. The sole difference between them is that the interaction was performed on a *different* face. Without knowing which face was involved it is impossible to distinguish the difference between both transformations, using the current definition of an affordance.

To represent an interaction point X_i, a relative position from the centre of mass could be used. However, this can encounter problems as differences in the agent's perspective and scale will cause the position to become invalid. Instead, the interaction point can be represented by a ratio of the distance between the two vertices forming the face (see Figure 3.8).

Multiple separate contact points on a face can produce similar results. For example, all the points in Figure 3.9a produce a pure translation in the x-axis. These points are therefore added to the list of valid interaction points $\boldsymbol{X}$ for the associated affordance, where $\boldsymbol{X} = \{X_1, X_2, ..\}$. The mean of the list X_μ, is used to obtain the optimal point to interact with the object face in order to perform the transformation

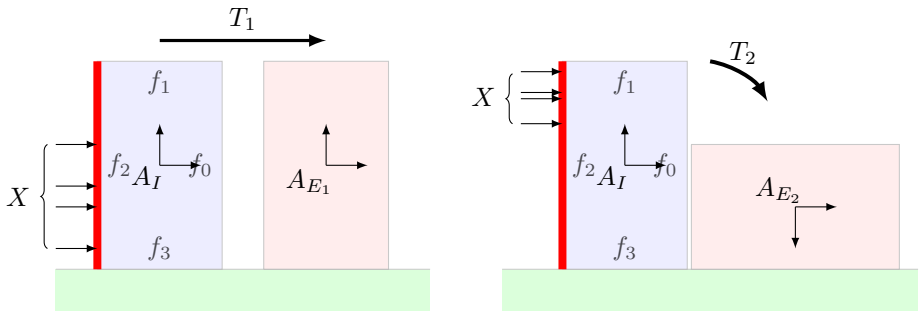


Figure 3.9: Interaction on face $\boldsymbol{f}_2$ with a total of 8 points, causing a different effect depending on where they are located. They are separated out based on the transformations they produce - In this example a pure translation (a) and a translation and rotation (b).

the affordance describes. The interaction point can be obtained from the manipulator frame $\boldsymbol{M}$, which is controlled by the agent. The manipulator frame contains the position and orientation of the manipulator end point. Hence, the definition of an affordance can then be updated as follows:

$$\boldsymbol{P} = g(\boldsymbol{A}_I, \boldsymbol{A}_E, \boldsymbol{M}_I) \tag{3.8}$$

where $\boldsymbol{M}_I$ is the initial manipulator frame.

As interaction points are added to an affordance, the distance can increase between X_μ and the X_i which produces the initially observed $\hat{\boldsymbol{T}}$ used to describe the affordance transformation. In some cases the transformation produced at X_μ can change slightly from what is produced at X_i. As the agent will use the interaction point at X_μ, the $\hat{\boldsymbol{T}}$ and T^d from the interaction point closest to X_μ is therefore used for the affordance in order to more accurately reproduce transformations. A Local learner could be used in future work to produce transformations over the range of points in $\boldsymbol{X}$. This would facilitate flexibility and robustness for transformations from interaction at X_μ.

Force Application

When interacting with an object, a wide range of force magnitudes and directions can be applied, leading to complex information regarding the interaction. For simplicity, force orientations are defined relative to the orientation of the face and are obtained from the frame $\boldsymbol{M}$. As there are a wide variety of interaction orientations, force interaction is limited to normal or sheer force applied by the agent. Furthermore, the magnitude is not necessary to be included in the affordance definition as the agent itself actively determines how much force is required at any given time, in real time. The agent is however, given a

target force to maintain which is common across all affordances. Therefore, the direction of force relative to the face it is applied to, is denoted $\hat{\boldsymbol{F}}$, as shown in Figure 3.8.

3.2.5 Formal Definition

A complete affordance is characterised by the following inputs, $\boldsymbol{A}_I$, $\boldsymbol{M}_I$, $\boldsymbol{A}_E$ (as Equation 3.8 described) and is summarised by the following equation:

$$\boldsymbol{P} = \{\hat{\boldsymbol{T}}, T^d, \boldsymbol{I}, \boldsymbol{f}, \boldsymbol{X}, \hat{\boldsymbol{F}}\} = g(\boldsymbol{A}_I, \boldsymbol{A}_E, \boldsymbol{M}_I) \tag{3.9}$$

3.2.6 Limitations of affordances

An affordance is only classified as complete if the whole transformation the affordance describes is performed, otherwise the object is in an unknown state. This approach allows the affordance information to be stored independent of the current position of the object, enabling the affordance to be generalised and used in any position that meets the initial state conditions $\boldsymbol{I}$. Motion during the initial frame of a transformation is not factored into this thesis as it was deemed beyond the scope of the research objective. Therefore, an object must be static when the agent starts interaction and it must wait for motion of the object to cease or the direction of motion to be constant when interaction has stopped. Objects which continue their movement after waiting a period of time for them to come to rest, are considered invalid and are ignored. Due to this, and given shapes with circular segments can not be easily approximated by straight line segments, round objects are not considered in this thesis and can be addressed in future work.

3.3 Learning Affordances

As discussed in the introduction chapter, some current systems which learn affordances utilise neural networks or probabilistic modelling. These methods, while potentially robust against small changes in conditions and noise, are slow, requiring many trials to properly learn affordances. For practicality and convenience, it is desirable to require only a few trials to learn and sufficiently characterise an affordance for use. Work such as Peshkin and Sanderson's [25] specifically looks at the geometrical properties of the object and use an energy minimisation principal to optimally determine where to push the object and how far. This approach can only deal with quasi-static motion, meaning that falling objects can not be considered. Therefore, instead of obtaining a comprehensive characterisation model of affordances, so as to effectively predict the outcome in every possible state, an alternative for learning affordances

is presented to address this problem. In order to obtain a desired outcome, which has previously been observed, the approach in this thesis focuses solely on identifying and learning the optimal point of interaction for observed object transformations through experimentation. Essentially, the agent learns what it perceives to be the result and what it perceives to be the conditions for the result to occur. This approach is somewhat similar in principal to the concept of naive physics as it does not require the agent to distinguish exactly why the result is attained, or the exact process behind it. It simply learns cause and effect at a basic level, or rather, what manipulations an object affords.

3.3.1 Basic Exploration

To autonomously discover affordances of an object, an exploration method which can facilitate discovery of a range of agent object interactions is necessary. A simplified method is discussed in this chapter, however Chapter 5 builds upon the approach outlined below, by introducing intrinsic motivation. As normal and sheer forces are used in this thesis, exploration is split into two corresponding modes.

- Normal Force Interaction - Utilises a pushing force normal to the face of the object
- Sheer Force Interaction - Maintains a pressure normal to the face of the object for grip, while moving the agent's manipulator parallel to the object surface so as to slide or drag.

In this thesis, pushing forces normal to the surface of the object are used as the primary form of interaction. Pushing is used for exploring and discovering affordances as it is simpler and easier to perform. As such, the agent is designed so that when interaction is desired, it moves the manipulator towards the object to apply a pushing action. If the agent determines that no visible transformation resulted from the application of a normal force, it then switches to attempting sheer force interaction. If neither normal nor sheer forces produce a visible transformation, and force is able to be maintained on the object, a new point is chosen and the process is repeated.

To determine the interaction point, a simple randomised exploratory approach is taken. An object face which is reachable by the agent's manipulator is randomly selected. A point on the face is also randomly selected as the target for the agent to interact with. There are several approaches for determining how far to push the object so that the action is adequately captured:

- Displacement Based - A predefined time or distance to push the object
- Motion Based - Continues to apply a force until the object stops moving
- Descriptor Based - Continues to apply a force while there is a difference in the $\hat{\boldsymbol{T}}$ between the current and previous object frames.

All these methods track the frame $\boldsymbol{A}$ of the object and set the initial frame $\boldsymbol{A}_I$ upon contact with the object. When tracking is ceased, the final frame is set $\boldsymbol{A}_E$.

The descriptor based approach was chosen because of to its ability to handle minor pauses in motion due to force disturbances on the finger tip during interaction. Interaction is only performed for as long as is necessary and is independent of object shape and configuration within the environment. This approach, involves obtaining the transformation from the initial frame to the current frame. The descriptor is calculated using Equation 3.4. Equation 3.10 is then used to determine the similarity between the current and previous frames descriptors.

$$\upsilon(t) = \hat{\boldsymbol{T}}_{t-1} \bullet \hat{\boldsymbol{T}}_t \tag{3.10}$$

The derivative is then approximated and summed using the current and previous frames:

$$b_r = \sum_{t=0,\ldots,n-1} \frac{\|\upsilon(t+1) - \upsilon(t)\|}{t_f} \tag{3.11}$$

where b_r is the summed derivative of υ and t_f is the time between frames $\boldsymbol{A}_t$ and $\boldsymbol{A}_{t+1}$.

The rate at which υ changes reduces while the current motion of the object is maintained. When b_r falls below a threshold (see Equation 3.12), agent interaction is stopped and $\hat{\boldsymbol{T}}$ is created from the resultant transformation from the initial frame $\boldsymbol{A}_I$ to the final frame $\boldsymbol{A}_E$.

$$\begin{aligned} continue, &\quad \text{if } b_r \leq k_m \\ stop, &\quad \text{if } b_r > k_m \end{aligned} \tag{3.12}$$

A small minimum movement is required before a descriptor can be created. The basic flow of the affordance exploration algorithm is shown in Figure 3.10a. This exploration process is continued until stopped by the user. Exploration is improved upon and discussed further in Chapter 5.

3.3.2 Evaluation

After each exploration operation has ended, the system evaluates the transformation descriptor and either creates a new affordance, or adds the contact point to the $\boldsymbol{X}$ of a matching affordance. A match is determined by the transformation descriptor $\hat{\boldsymbol{T}}$, the initial state $\boldsymbol{I}$, the face $\boldsymbol{f}$ and the direction force was applied $\hat{\boldsymbol{F}}$. These components must match for the current transformation to be considered a match to the affordance. Figure 3.10b shows this evaluation process. To determine if transformation descriptors are similar (Equation 3.14), the dot product is used as follows in Equations 3.13:

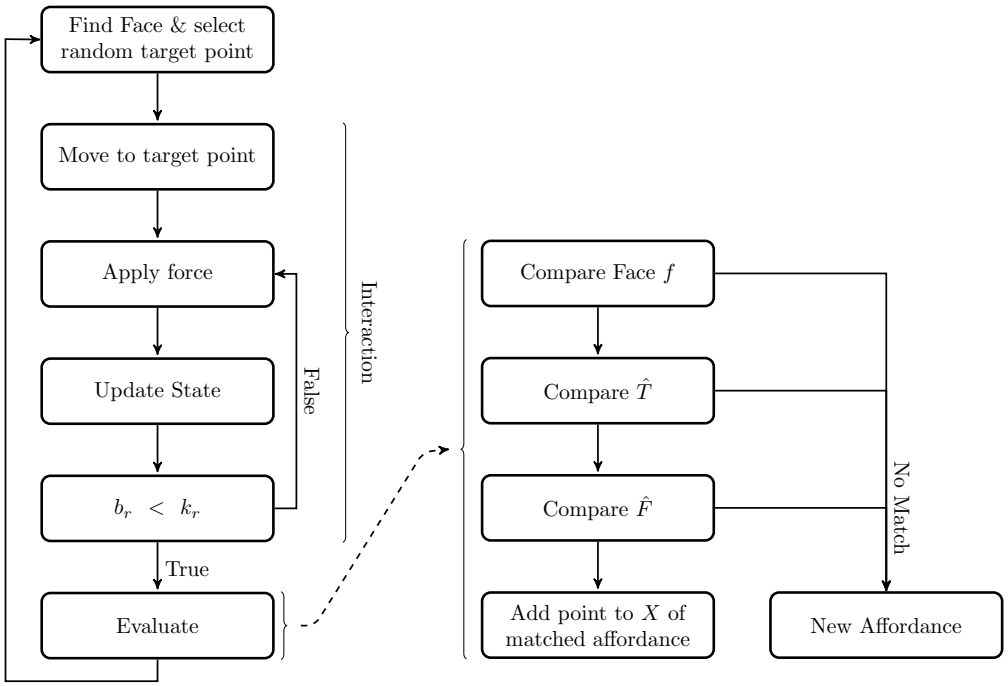


Figure 3.10: Exploration flow chart. (a) Is the flow of the exploration algorithm, which loops continually. (b) Is the flow of the evaluation process, which follows two paths. It the face $\boldsymbol{f}$, transformation descriptor $\hat{\boldsymbol{T}}$ and force direction $\hat{\boldsymbol{F}}$ all match, the current point is added to the contact point list $\boldsymbol{X}$. If any of those conditions are not met, a new affordance is created.

$$d_T = \hat{\boldsymbol{T}} \bullet \hat{\boldsymbol{T}}_c \tag{3.13}$$

where $\hat{\boldsymbol{T}}$ is the current descriptor to be matched, $\hat{\boldsymbol{T}}_c$ is the descriptor to be compared and:

$$\begin{aligned} \hat{\boldsymbol{T}} &\approx \hat{\boldsymbol{T}}_c, \quad \text{if } d_T \geq k_s \\ \hat{\boldsymbol{T}} &\neq \hat{\boldsymbol{T}}_c, \quad \text{if } d_T < k_s \end{aligned} \tag{3.14}$$

where k_s is the matching threshold and typically $k_s = 0.9$. This approach is also used for force comparison.

3.4 Planning with Affordances

Planning is a well defined field in robotics, with many methods to choose from. In this thesis a simple approach is selected in order to focus on learning manipulation affordances. Typically, planning for robotic manipulation tasks involves manipulator motion planning, where learned manipulator motions are used to form the plan. Such approaches generally utilise a learn by demonstration methodology or supervised learning, as more complex manipulations can be learnt faster. Work such as Montesano *et al* [12] and Fuentes and Nelson [8] are examples where motor primitives [1] (joint motion) in the form of discrete motion of the agent's manipulator, are used to form building blocks for plans. As discussed above, this essentially couples the two processes together, making the manipulation techniques dependent on the ability of the manipulator. As a consequence, the agent's motor primitives can not be improved with further learning, without affecting the prior learnt manipulations. Similarly, the agent's manipulator can not be physically altered in any way without effecting the motor primitives, which in turn affect the learned manipulations. For example, consider an agent whose motor primitives were altered to produce a smoother action. When the agent later uses this motor primitive for a manipulation, the result may be that of failure. This is due to the manipulation technique requiring a close replica of motion, to successfully reproduce the manipulation. It may be that the agent utilised a particular characteristic of a motor primitive that is no longer present to perform a manipulation.

The approach taken in this thesis, instead utilises affordances to generate discrete motion of the object, rather than the manipulator. This removes the dependencies of manipulation techniques from the agent's manipulator ability. The agent can therefore utilise any method desired to control the manipulator. Furthermore, this approach ideally allows the agent to interchange with any manipulator configuration, provided the new configuration can perform the necessary actions. The process can therefore be thought of as analogous to the path planning paradigm of mobile robots. However, instead of the robot moving itself, the agent moves the object with its manipulator and the planner instructs the agent how to do so.

3.4.1 A Modified Nested Hierarchical Controller

Given the planning problem has shifted from that of manipulator control, to that akin to mobile navigation, a planner that is conducive to using affordances is desired. As Murphy discusses [44], many path planning methodologies perform various optimisations, so as to avoid obstacles, or smooth the path. These methods typically treat the mobile robot as a single point in the workspace, with the orientation of the robot having no relevance to the path. This however, poses a problem as the orientation of the object can greatly affect how the object moves as discussed in Section 3.2.2. Optimising the position of the object may also produce invalid states, causing the object to float above ground or move into the ground as discussed in Section 3.2.3. Furthermore, these methods do not factor the effects of gravity, meaning certain types of movement are not possible. The planner would need to somehow know about gravity and how it affects the object's movement. Therefore, typical path planning methods are unsuitable for creating plans with affordances. A navigation algorithm could however provide high-level goals for larger tasks that have many stages. The low-level planner would simply endeavour to move the object between two high level goals supplied by the high-level planner. For this thesis, only the low-level planning aspect will be considered, as there is already significant research in the field of high-level path planning.

When considering how to formulate a plan, going back to simple or core principals provides a good starting point when considering using affordances to formulate plans. As such, the principal of Means-Ends Analysis (MEA) [44] which involves minimising the value or distance between the current frame and the end goal frame, provides an appropriate starting point. MEA is performed after every move and thereby used to determine the next move. However, simply moving the object towards the goal frame as quickly as possible, can potentially put the system in an unsolvable state. There may be cases where there are obstacles in the path which later restrict the choice of move. Consider Figure 3.11, where the object is pushed over to the state indicated by frame $\boldsymbol{A}_1$. If the system has no ability to flip the object back upright, the problem can never be solved. Yet according to MEA, this was the best possible move to make, as it provided the greatest reduction in distance to the goal frame.

Therefore, to properly form a plan using affordances, a modified version of the Nested Hierarchical Controller (NHC) [44] is used. The NHC can be broken down into three functional systems; Sense, Plan and Act. In this section we consider the Plan system, which involves three subsystems; Mission Planner, Navigator and the Pilot. The Mission Planner involves assigning a plan to the agent, either by manual user assignment or autonomous means. In this thesis, when a task is requested, the goal is set by the user by showing the final state of the object and then placing the object in the desired starting position. The Navigator then generates a tree of moves and forms a plan. The Pilot uses the affordances, which define the type of moves available, and translates the moves into interaction points for the agent's manipulator to interact with.

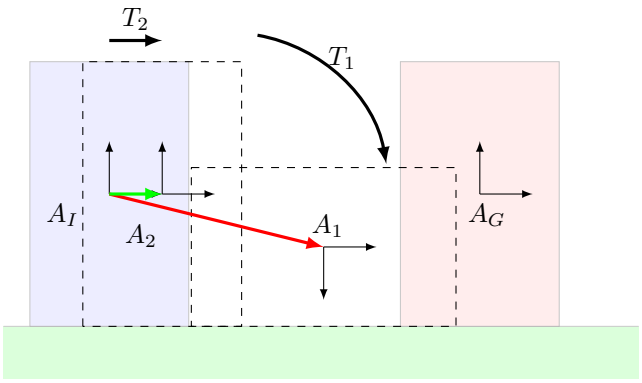


Figure 3.11: Moving from $\boldsymbol{A}_I$ to $\boldsymbol{A}_G$ using MEA. Given only two possible moves, the best move according to MEA is the one which results in $\boldsymbol{A}_1$ (red arrows). However, in this scenario there is no ability to flip the object back into the correct orientation. Therefore the $\boldsymbol{A}_2$ (green arrows) is the correct choice despite its lower reduction of goal distance compared to $\boldsymbol{A}_1$.

The NHC Navigator uses the principal of MEA, however instead of performing a single move, the Navigator explores the outcome of multiple moves, recursively generating a tree of all possible moves. When the goal is reached, a sequence of moves is created that forms a path to the goal. This solves the issues with MEA as discussed above. Every possible move is considered and explored to determine if the goal is reached. However, the Navigator only uses moves which result in either one or both a reduction in translational or rotational distance to the goal. This allows rotational moves to be utilised which may require the object to be translated further away to reach the required orientation.

When using affordances, it can be difficult for the agent to determine the mode of failure and how to correct it. Typically, the Pilot corrects discrepancies by interpolating between two local waypoints and uses small adjustments. However, the Pilot does not know how to move the object and some modes of failure can put the object into a completely different state, making interpolation impossible. Therefore, a simple, but effective approach is used which requires the Navigator to regenerate a plan after each move. Hence, the agent does not have to consider failure, but simply update the current position of the object and attempt a new move determined from an updated plan. When considering high-level planning the planner implemented here could be thought of as the Pilot of the high-level system, which instructs the agent how to move between two local waypoints.

3.4.2 Generating Moves from Affordances

When forming a plan using the NHC, a tree of nodes is generated. Each node $\boldsymbol{N}$ contains:

- an object frame $\boldsymbol{A}$, which represents the current position, orientation and state $\boldsymbol{S}$ of the object.

- a list of discrete moves (transformations) $\boldsymbol{Y}$ derived from valid affordances.
- a list of child nodes $\boldsymbol{Q}$ generated from valid discrete moves.

Therefore, $\{\boldsymbol{A}, \boldsymbol{Y}, \boldsymbol{Q}\} \subseteq \boldsymbol{N}$. To generate a transformation, all affordances are checked for initial states $\boldsymbol{I}$ which match the current state $\boldsymbol{S}$ of the nodes frame $\boldsymbol{A}$. The transformation descriptor $\hat{\boldsymbol{T}}$ of each affordance which matches the initial state is then used to form a transformation, creating a move. These moves are then added to the list of moves $\boldsymbol{Y}$ for the node. However, as using a tree is limited to discrete moves, it is necessary to generate a series of transformation with a range of different magnitudes to cover a range of distances as shown:

$$\boldsymbol{T}_i = \hat{\boldsymbol{T}} \cdot e^{i-q}, \quad \forall i \in \{0, \ldots, q\} \tag{3.15}$$

where $\boldsymbol{T}_i$ is the calculated transformation and is added to $\boldsymbol{Y}$. q is the number of transformations to create from the affordance's $\hat{\boldsymbol{T}}$ and is related to the level of precision desired in plan generation. The effectiveness of increasing q reduces due to a logarithmic relationship. For this thesis, typically $q = 5$.

If an affordance is classified as rotational, the distance variable T^d is included as discussed in Section 3.2.2. Hence, there is only a single transformation generated and Equation 3.15 becomes:

$$\boldsymbol{T}_i = \begin{cases} \hat{\boldsymbol{T}} \cdot k_a \cdot e^{i-q}, & \text{if } \hat{T}_\alpha > \frac{1}{\sqrt{2}}, \quad \forall i \in \{0, \ldots, q\} \\ \hat{\boldsymbol{T}} \cdot T^d, & \text{if } \hat{T}_\alpha \geq \frac{1}{\sqrt{2}}, \quad \text{where } i = 0 \end{cases} \tag{3.16}$$

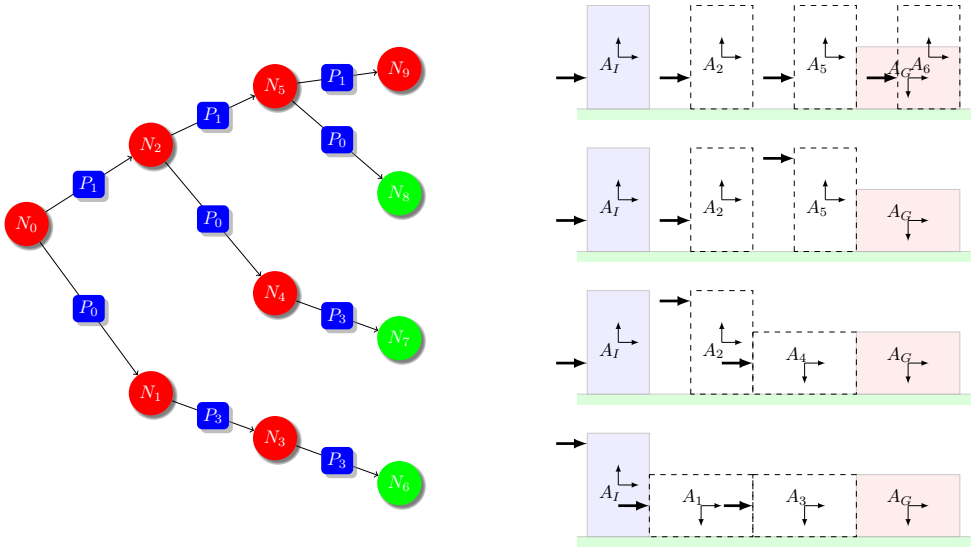
where k_a is a scaling value typically set to $k_a = 0.2$ in this thesis.

To create a new node from a transformation, the current node's ($\boldsymbol{N}_c$) frame $\boldsymbol{A}_c$ is added to one of the transformations from the list of moves $\boldsymbol{Y}_c$ able to be produced from node $\boldsymbol{N}_c$.

$$\boldsymbol{A}_n = \boldsymbol{A}_c + \boldsymbol{T}, \quad \forall T \in \boldsymbol{Y}_c \tag{3.17}$$

where $\boldsymbol{A}_n$ is the frame of a new node $\boldsymbol{N}_n$, $\boldsymbol{A}_c$ is the frame of the current node $\boldsymbol{N}_c$ and $\boldsymbol{T}$ is a transformation used to create the new frame.

To generate a full tree of moves, the agent is required to calculate every possible transformation and the resulting new frame for every node as shown in a simplified manner in Figure 3.12. This however, can create an exponentially large tree, being computationally expensive to generate, particularly for more complex shapes and environments. Therefore, a branch trimming process is utilised to invalidate transformations which are too small. Transformations which result in motion with a reduction of the current goal distance by a ratio less than k_h, are omitted. The value of k_h is calculated based on the number of transformations desired to be created, namely q:



(a) (b)

Figure 3.12: Illustrates a simplified tree generation process. The tree starts with the initial Node $\boldsymbol{N}_0$ and after a move, described by affordances $\boldsymbol{P}$ (shown by Blue blocks) is performed, a new Node is created. The node contains both the frame information $\boldsymbol{A}$ and the affordance $\boldsymbol{P}$ used to create the frame. Each of the frame states in (a) are represented in (b). The nodes $\boldsymbol{N}_6$, $\boldsymbol{N}_7$ and $\boldsymbol{N}_8$ have frame states that match $\boldsymbol{A}_G$ and are indicated by Green Circles.

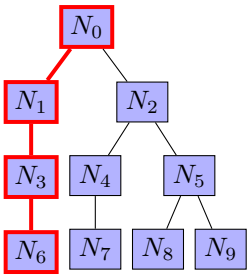
$$k_h = \psi \cdot e^{-\zeta \cdot q^2} \tag{3.18}$$

where ψ is the maximum desired ratio, typically $\psi = 0.5$ and ζ limits the effective range of q, typically $\zeta = 0.1$.

This removes many unnecessary small transformations, which could be completed by fewer or even a single transformation with a larger magnitude. As a further method to keep the size of the tree manageable, the maximum branch depth of the tree is limited to 6 moves. However, this limitation is dependent on the complexity of the task given to the system as more than 6 moves may be necessary. Nevertheless, for the scope of the tasks considered in this thesis, 6 moves is deemed sufficient to produce many viable viable plans.

3.4.3 Generating a Plan

To start generating the tree, the root node's frame is set as the current frame of the object and is the initial frame ($\boldsymbol{A}_I$) of the plan. The desired goal frame $\boldsymbol{A}_G$ can be set by either the agent, or manually. When the tree is generated, if there are no frames of nodes in the tree which matches $\boldsymbol{A}_G$, the planning process



Nodes	Frame	Affordance	Face	X_μ	$\hat{F}_\alpha$
N_0	A_I	-	-	-	-
N_1	A_1	P_0	2	0.2	π
N_3	A_5	P_3	3	0.5	π
N_6	A_G	P_3	3	0.5	π

Figure 3.13: Example of constructing a plan from the tree of Figure 3.12. The first encountered frame which matches the goal frame $\boldsymbol{A}_G$ is used. In this tree $\boldsymbol{N}_6$ contains the first encountered frame which matches $\boldsymbol{A}_G$. The agent then works back up the tree adding the affordance from each node to a plan list. The table in (b) shows the list of nodes from the plan and their associated frames and affordances. The face, interaction point (X_μ) and force orientation $\hat{\boldsymbol{F}}_\alpha$ are obtained from the affordance. In this case, nodes $\boldsymbol{N}_3$ and $\boldsymbol{N}_6$ would be merged as they contain the same affordance.

has failed and the agent indicates that it cannot solve the problem. If $\boldsymbol{A}_G$ has been matched, any further generation of child nodes beyond the current, is stopped and the node is flagged as a valid end point. To ensure that an optimal plan can be found, generation along branches of nodes whose frames have not yet matched $\boldsymbol{A}_G$ is continued. In this chapter, when there are multiple solutions to reach $\boldsymbol{A}_G$, the first successful plan with the lowest number of nodes is chosen out of all possible plans. An improved approach is discussed in Chapter 4.

When the tree is generated, the list of transformations required to enact the plan, can be obtained by recursively working back up the tree towards the root node from the node which matched $\boldsymbol{A}_G$ (see Figure 3.13). For each node parsed, the affordance used for each transformation is added to the plan list $\boldsymbol{E}$. The agent only needs to know the face, point and direction to apply a force on an object in order to enact the desired transformation. The frame information of each node is used to determine how far the interaction must be continued to attain the desired frame state at each point in the plan. After each transformation is performed, the agent generates a new plan with the newly updated frame of the object, rather than performing all the listed transformations blindly. This accounts for differences in position or even failed transformations, without requiring the capability to recognise that a transformation failed, or has a slight offset from the expected frame position. The new plan is generated by setting the current frame of the object, as the frame of the root node of the tree. The originally set goal $\boldsymbol{A}_G$ is kept the same. This process is repeated until $\boldsymbol{A}_G$ has been matched by the current frame of the object.

3.4.4 Move Merging

In many cases, moves based on the same affordance must be performed sequentially multiple times. If these moves transformations have the same direction, they can be combined to form a larger single move.

When moves are combined, the transformation of each move is added to form a new transformation. The state of the last move is used for the final state of the new single move. This can occur multiple times and with moves of different magnitudes, until the next move's affordance or direction no longer matches the previous. For example, in Figure 3.12, to move the object to the frame of $\boldsymbol{N}_6$, affordance $\boldsymbol{P}_3$ is performed twice. The moves transformations from $\boldsymbol{N}_1$ to $\boldsymbol{N}_3$ to $\boldsymbol{N}_6$ can be be combined together. The state of $\boldsymbol{N}_1$ forms the initial state of the move and the state of $\boldsymbol{N}_6$ forms the final state. The single larger move can then be used instead of the two smaller moves.

3.5 Summary

In this chapter, the concept of manipulation affordances was introduced and used to describe how simple interactions such as pushing or sliding can be parameterised and learned by an agent. A simple planning system can then utilise affordances to perform basic tasks.

Affordances are defined as the quality of an object, which induces a transformation due to point contact interaction such as pushing or sliding. To characterise this the affordance consists of the following:

- Transformation descriptor ($\hat{\boldsymbol{T}}$) is the effective direction of the transformation which was observed between two frames. It is used to reproduce transformations for use in performing tasks.
- Transformation distance variable (T^d) is the magnitude of the transformation and used to reproduce rotational motion, or scale translational motion, to produce multiple transformations of different distances.
- Initial state ($\boldsymbol{I}$) describes the conditions which the affordance must meet to be successfully reproduced. It ensures the transformation descriptor produces a valid new state.
- Face ($\boldsymbol{f}$) is formed from two vertices of the object, forming a segment. Faces are used as a reference point for the force direction and interaction point.
- A list of contact points $\boldsymbol{X}$ produces the transformation that $\hat{\boldsymbol{T}}$ describes. It is the point at which the agent's manipulator makes contact with the object face to apply a force. The average of these points X_μ is used as the optimal interaction point when reproducing the transformation in plans for tasks.
- Direction of force $\hat{\boldsymbol{F}}$ describes the angle at which the force was applied on the face. For this thesis forces are limited to normal or sheer.

Two primary assumptions are made to ensure affordances remain in stable known states:

- The object must be stationary and stable at the initial frame $\boldsymbol{A}_I$ of interaction.
- The object must have negligible rate of change in $\hat{\boldsymbol{T}}$ at the final frame $\boldsymbol{A}_E$ of interaction

Exploration enables the agent to expand its knowledge of the object through interaction. The following facilitates this process:

- The Agent is given a tendency to move towards and contact a face at a random point to apply a force.
- The Agent determines when to use normal or sheer force, by monitoring the motion of the object. When no motion occurs when attempting normal force interaction, sheer force is attempted.
- The derivative of the similarity between the current and previous frames transformation descriptor ($\hat{\boldsymbol{T}}$), is used to determine when to stop the application of force on the object. The agent continues to apply a force while the similarity is changing.
- When force application is ceased and the object comes to a rest, the initial and final frames, $\boldsymbol{A}_I$ and $\boldsymbol{A}_E$ respectively, are evaluated to form either a new affordance, or update an existing one.

To use affordances to perform a simple task, such as moving an object from one position to another, a simple planner based on mobile robot navigation techniques is used. The planner can be summarised as follows:

- The Nested Hierarchical Controller (NHC) is used which is based on the principal of means ends analysis.
- Only moves which result in a reduction of the rotational or translational goal distance are used.
- A tree of nodes, containing frames of object states created by transformations, forms the basis for generating a plan.
- Nodes consist of a frame $\boldsymbol{A}$ describing the current object state and a list of possible transformations $\boldsymbol{Y}$.
- The list of transformations ($\boldsymbol{Y}$) are generated by using the transformation descriptor $\hat{\boldsymbol{T}}$ of affordances whose initial state $\boldsymbol{I}$, match the current state of the nodes frame $\boldsymbol{A}$ (Equation 3.15).

- A range of transformations with different magnitudes, are generated based on the desired number q.
- To create a new child node, the current frame $\boldsymbol{A}$ of the node is added to one of the listed transformations $\boldsymbol{T}$ (Equation 3.17).
- Transformations which do not result in a reduction of the rotational or translational distance from the current frame to the goal frame, are pruned from the tree.
- Transformations which are too small are also pruned (Equation 3.18).
- Tasks can be set by demonstrating the desired end goal frame $\boldsymbol{A}_G$ and placing the object in the desired starting frame $\boldsymbol{A}_I$.
- If $\boldsymbol{A}_G$ is matched to a nodes frame $\boldsymbol{A}$, the node is flagged as a valid end point. Hence, further generation of child nodes from that frame are stopped.
- For multiple plan solutions, the first plan encountered with a minimum number of nodes is chosen.
- To reproduce a transformation, the agent only needs to know the face $\boldsymbol{f}$, point X_μ and direction to apply a force $\hat{f}$.
- After each transformation is performed, instead of performing the next in the plan, a new plan is generated, based on the current state of the system, so as to factor failed or unexpected transformations.
- Moves which use the same affordance can be combined together when performed sequentially, forming a single move instead of multiple repeated moves.

Chapter 4

Generalising with Constraints and Symmetry

In the previous chapter, the environment considered is extremely simple. It consists only of the ground with no obstructions, which can limit what an object affords. The agent was also configured such that the effect of the environment, in this case the ground, is inherently coupled into affordance characteristics. This effectively prevents the agent from generalising affordances, as changes in the environment would invalidate them. Furthermore, the agent is unable to factor various static objects or boundaries within the environment, such as walls or raised sections. Therefore, the concept of constraints is introduced in this chapter to address these issues. Constraints represent static objects within the environment and are an essential part in describing what an object affords. Specifically, they more clearly define the state of the object within the environment. They also provide a means to generalise affordances such that changes or additions in the environment do not affect the transformation an affordance describes.

To further generalise affordances, utilising and exploiting characteristics of objects themselves, is essential. While much of this is accounted for when learning affordances, features such as symmetry, are not. Many objects in the world exhibit symmetry of some form. Many more objects can simply be approximated to have symmetry for convenience. In the previous chapter, issues of ambiguity for symmetrical objects hinders affordance creation and usage. This chapter therefore aims to demonstrate how symmetry can be used to simplify the learning problem and generalise affordances. The way in which affordances are used to perform a task, is also important. Simply choosing the plan with the least number of moves does not provide a solid foundation for optimal use of affordances. Therefore, an approach to minimising the uncertainty in performing a task is introduced.

In this chapter; Section 4.1 details the implementation of constraints, Section 4.2 covers the ap-

proach taken to deal with symmetry, Section 4.3 discusses some improvements to planning and finally, Section 4.4 summarises the chapter.

4.1 Constraints

This section introduces the concept of a constraint and describes how they are used with affordances and is broken up into the following subsections; Gravity (Section 4.1.1), Defining constraints for use in affordances (Section 4.1.2), Constraint effect on initial conditions (Section 4.1.3), Generalising constraints (Section 4.1.4), Defining levels of constraint interaction (Section 4.1.5), Further generalising affordances (Section 4.1.6) and Using constraints (Section 4.1.7).

4.1.1 Gravity

Gravity is perhaps the most influential force acting on objects in a typical environment encountered by humans and animals. Knowing its direction and magnitude can greatly assist in simplifying the manipulation problem. As Angelaki discusses [55], the Vestibular (inner ear) contributes to many brain functions, including spatial perception, motor coordination and spatial orientation. These functions help the brain keep track of and adjust executed motor plans, despite changes in head position and movement. Hence, while knowing the direction of gravity is not essential, it can help with understanding the results of manipulations. It can also be used as a reference for object movement and give a sense of absolute orientation. For the purposes of this thesis, the gravity vector $\boldsymbol{G}$ is given to the agent. However, robotic agents can obtain the gravity vector $\boldsymbol{G}$ themselves in a similar manner to humans by using accelerometers. Hence, providing the agent with the gravity vector is primarily done for convenience in this thesis.

4.1.2 Defining constraints for use in affordances

A constraint is identified when the object's vertices contact the face of a static object in the environment. The object's degree of freedom (DOF) is then reduced, constraining motion normal to the static object's face. In this thesis, the ground is a common static object depicted in many figures to restrict motion of the object. When contact is made between a static object and a face of an object the agent is manipulating, the associated constraint is considered to be *in effect*.

Formally, a constraint, denoted $\boldsymbol{C}$, is defined by the two vertices of the static object's face, V_1^C and V_2^C, where $\{V_1^C, V_2^C\} \subseteq \boldsymbol{C}$. To fully describe the current environment with an object frame $\boldsymbol{A}$, a list of constraints *in effect* $\boldsymbol{L}$, is included such that $\boldsymbol{L} \subseteq \boldsymbol{A}$. $\boldsymbol{L} = \{\hat{\boldsymbol{C}}_1, \hat{\boldsymbol{C}}_2, ..., \hat{\boldsymbol{C}}_\chi\}$, where χ is the number

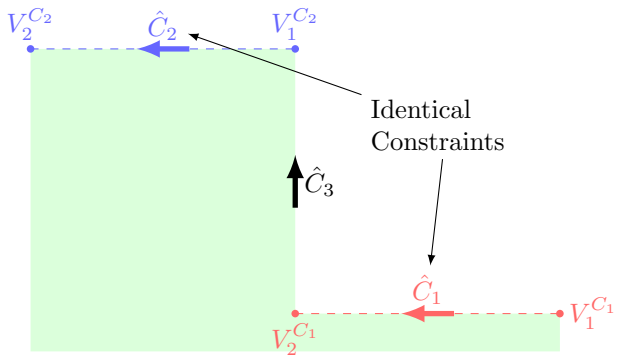


Figure 4.1: Illustrates how constraints are represented. Although constraint $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ are located differently they can be considered the same constraints as objects afford the same transformations when in contact.

of constraints *in effect*. Hence, a constraint is only included as a characteristic of an affordance when *in effect*.

4.1.3 Constraint effect on initial state

When a constraint is *in effect*, the transformations the object affords can change drastically. Some transformations are only possible when a certain configuration of constraints are *in effect*. Consider Figure 4.2a. The object has a set of affordances (X_1 and X_2) from face $\boldsymbol{f}_3$ which the agent can utilise. If another constraint were introduced to the environment such that two constraints were *in effect* (see Figure 4.2b), the available transformations possible for the object are completely different. The object no longer affords the previous transformations. It is therefore necessary to describe the initial constraints *in effect* for an affordance. Hence, the initial list of constraints $\boldsymbol{L}_I$, is used as pre-conditions for an affordance to be considered valid ($\boldsymbol{L}_I \subseteq \boldsymbol{I}$). Using constraints in such a manner can open up a wide variety of transformations and is a first step in introducing more complex environments and object interactions.

Constraints can also be used to provide alternative means to achieve the same or similar transformation as other affordances. In some cases, due to the presence of the constraint, the transformation may be more stable or reliable to use. Consider Figure 4.3a, where the object is 'flipped' upright by using a sheer interaction on the edge of the face. This can be prone to the object slipping or bouncing over too far and falling over on the other side. Alternatively, the same object orientation can be attained by using a constraint as a wall to provide stability and restrict unwanted movement (see in Figure 4.3b). The agent simply applies an 'upward' sheer force to the middle of the object face, essentially 'flipping' the object upright. The wall consequently prevents the object from falling over too far. This behaviour is important when characterising affordances.

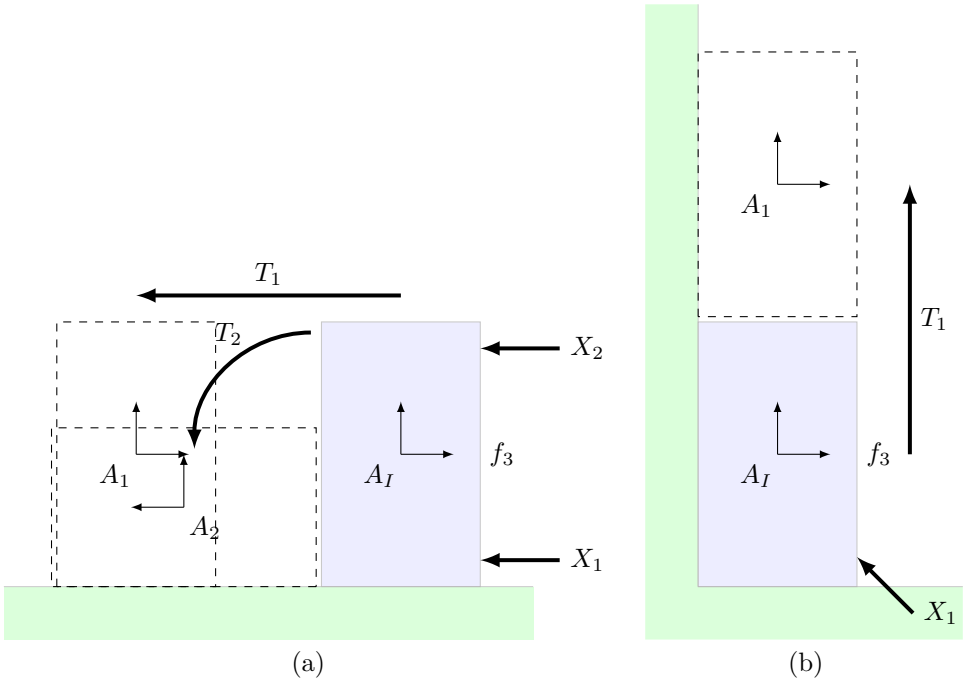


Figure 4.2: Shows the transformations possible under different constraints. Only forces on face f_3 are considered. (a) only has a single constraint where two possible transformations are possible. (b) has two constraints, changing the transformations possible despite the initial conditions being identical to (a).

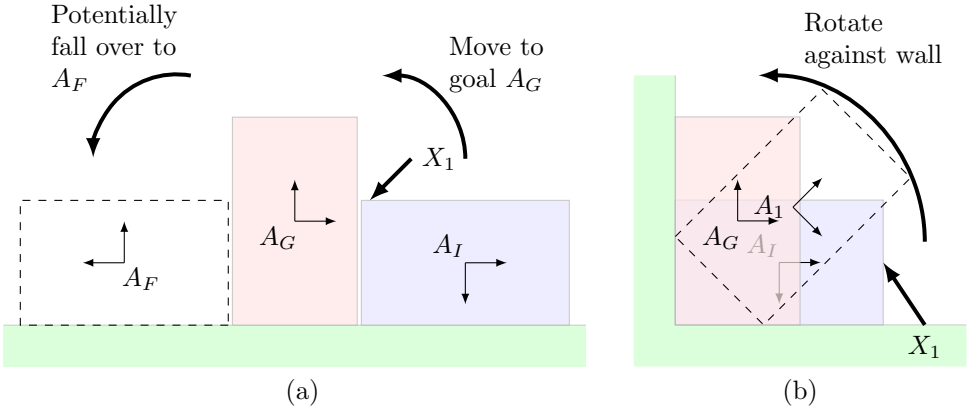


Figure 4.3: Demonstrates how constraints can be used to provide a more reliable transformation. In (a), the sheer interaction could either slip, allowing the object to fall back to its initial frame, or fall over to frame $\boldsymbol{A}_F$, missing the goal frame $\boldsymbol{A}_G$. The interaction is more stable in (b) due to the use of the wall while applying the sheer force. The object can not fall over, missing the goal frame $\boldsymbol{A}_G$.

4.1.4 Generalising constraints

Describing constraints purely based on their vertices does not allow for them to be generalised. For example, consider the two constraints ($\boldsymbol{C}_1$ and $\boldsymbol{C}_2$) in Figure 4.1. Both effectively produce the same type of constraint, yet have different vertex locations. If the vertices are used to describe the constraint, the affordance is not able to be generalised across similar constraints, as the vertices of each are different. Therefore, when describing a constraint for an affordance, the description must be such that the constraint's absolute position in the environment is not used to describe it. Given that constraints are only included in affordances when *in effect*, the position of the vertices becomes irrelevant. Thus, to generalise across differently located constraints, each is defined by a unit vector $\hat{\boldsymbol{C}}$ (see Equation 4.1) as shown by the thick arrows in Figure 4.1.

$$\hat{\boldsymbol{C}} = \frac{\boldsymbol{V}_2^C - \boldsymbol{V}_1^C}{\|\boldsymbol{V}_2^C - \boldsymbol{V}_1^C\|} \tag{4.1}$$

where $\hat{\boldsymbol{C}} \subseteq \boldsymbol{C}$

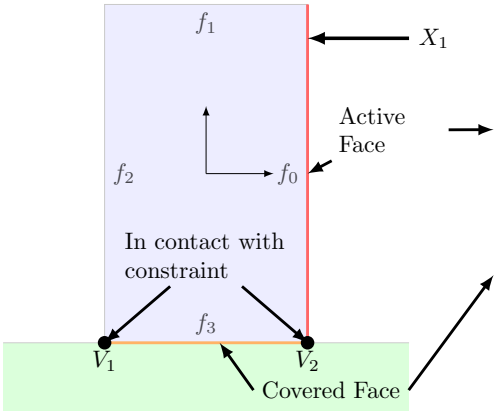
$\hat{\boldsymbol{C}}$ effectively describes the direction of the constraint, allowing others with similarly oriented unit vectors to be easily compared, based on their orientation and not their position. In the case of Figure 4.1, both constraints ($\boldsymbol{C}_1$ and $\boldsymbol{C}_2$) would be considered identical, affording the same transformations. The position of the vertices is still required for the agent to ascertain if the constraint is *in effect*, despite being unnecessary for affordances themselves. Therefore, the agent identifies if a constraint is *in effect* before determining the type of constraint (see Section 4.1.5 below) for use with an affordance. When comparing constraints to determine similarity, the same approach for determining the similarity of transformations (see Section 3.2.1) is used:

$$d_C = \hat{\boldsymbol{C}}_1 \bullet \hat{\boldsymbol{C}}_2 \tag{4.2}$$

$$\begin{aligned} \boldsymbol{C}_1 &= \boldsymbol{C}_2, \quad \text{if } d_C \geq k_c \\ \boldsymbol{C}_1 &\neq \boldsymbol{C}_2, \quad \text{if } d_C < k_c \end{aligned} \tag{4.3}$$

Where $\hat{\boldsymbol{C}}_1$ and $\hat{\boldsymbol{C}}_2$ are constraint descriptors of constraints $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ respectively and k_c is the threshold used to determine a match and typically $k_c = 0.9$.

Therefore, an affordance is considered different despite all other aspects being identical, excepting the constraints. If this is the case during learning and exploration, a new affordance is created. In the case of planning, it means the affordance cannot be used as it is not valid.



Face	V_1	V_2	Contact Level
f_0	1	0	1
f_1	0	0	0
f_2	0	1	1
f_3	1	1	3

Figure 4.4: Constraint contact with object faces. As both vertices $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ are within a threshold distance to the ground constraint, the contact level of $\boldsymbol{f}_3$ is 3, making the face inaccessible. As only a single vertex is in contact with a constraint on faces $\boldsymbol{f}_0$ and $\boldsymbol{f}_2$, the contact level is only 1, allowing interaction on the face. As interaction is taking place on $\boldsymbol{f}_0$ at X_1, $\boldsymbol{f}_0$ is considered the active face.

4.1.5 Defining levels of constraint interaction

To characterise the effect a constraint has with an object, due to contact with a static object, a simple approach consisting of four levels of interaction are defined:

- Level 0 - No vertices interacting with a constraint (constraint not *in effect*)
- Level 1 - A single vertex interacting with a constraint
- Level 2 - 2 vertices of a face interacting with 2 different constraints
- Level 3 - Both vertices of a face interacting with a single constraint

For this thesis, an object is considered to be interacting with a constraint (hence the constraint is *in effect*) if any vertex defining the object face is within a threshold distance to the static object the constraint is derived from. Figure 4.4 illustrates how interaction for each face is determined. The contact state of the object is included in the frame state information $\boldsymbol{S}$, such that $\boldsymbol{f}^c = \{f_1^c, f_2^c, ...f_{n_v}^c\}$, where $\boldsymbol{f}^c \subset \boldsymbol{S}$ and n_v is the number of faces or vertices of the object.

4.1.6 Further generalising affordances

Affordances, as described previously in Chapter 3, are only valid in the environment in which they were initially learnt. Changing the environment and agent view point invalidates the learned affordances.

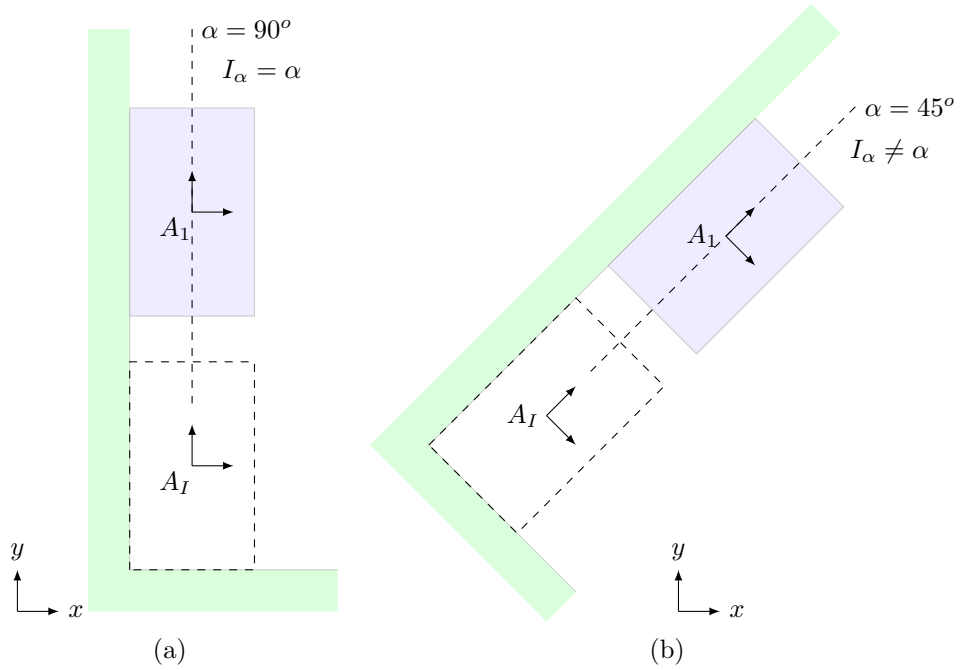


Figure 4.5: Problems with rotating constraints. A transformation is performed in (a). In (b) the agent's view is rotated by 45^o, which changes the orientation α, of the object. As the initial orientation was set as 90^o, the current absolute orientation does not match.

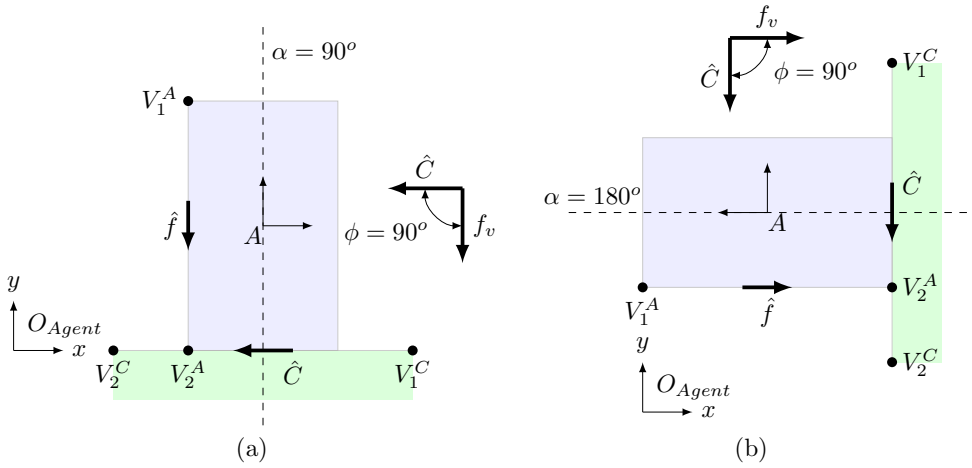


Figure 4.6: Generalising initial orientation by using the angle between the active face and the primary constraint *in effect* as the initial condition. (a) and (b) depict different agent view orientations of the same scenario. However ϕ remains the same for both orientations.

While the addition of constraints partially solves this problem, changing the orientation of the agent's view point is not factored. This is due to the initial orientation of the affordance remaining unchanged relative to a ground reference constraint, despite the change in orientation of the agent, creating a disparity (see Figure 4.5). Generalising affordances such that the initial conditions are not dependent on the orientation of the agent, is therefore desirable.

Orientation generalisation

To achieve further generalisation of affordances, instead of using the absolute orientation of the object for the initial condition (I_α), the angle between the active face and the primary constraint *in effect* ϕ, is used (see Equation 4.4) as shown in Figure 4.6. Therefore, the initial angle between the face and constraint ϕ_I forms part of the initial conditions of an affordance $\phi_I \subseteq \boldsymbol{I}$.

$$\phi = atan2(\hat{\boldsymbol{C}} \times \hat{\boldsymbol{f}}, \hat{\boldsymbol{C}} \bullet \hat{\boldsymbol{f}}) \tag{4.4}$$

where $\hat{\boldsymbol{f}}$ is the vector formed from the two vertices of the active face such that

$$\hat{\boldsymbol{f}} = \frac{\boldsymbol{S}_{V_2} - \boldsymbol{S}_{V_1}}{\|\boldsymbol{S}_{V_2} - \boldsymbol{S}_{V_1}\|} \tag{4.5}$$

where $\boldsymbol{V}$ contains the vertices of the object such that $\boldsymbol{V} \subseteq \boldsymbol{S}$. If there are no constraints, the following is used instead:

$$\phi = \lambda = atan2(\hat{f}_y, \hat{f}_x) \tag{4.6}$$

where λ is the absolute orientation of the face from the agent's reference frame O.

A constraint is considered primary if it is entirely covering a face and a face is considered active if it is undergoing interaction (see Figure 4.4). This ensures that regardless of the orientation of the agent view point, the affordance will remain valid as it is referenced relative to the orientation of the primary constraint. In cases where there is a conflict in determining the primary constraint, due to more than one face being in contact with a constraint, the constraint closest in orientation to $\frac{\pi}{2}$ from gravity is used. When there are no constraints, such as the object on a flat surface, or the object is floating in free space, the absolute orientation of the face is used (see Equation 4.6). In these cases the initial orientation has no effect on the resulting transformation. Hence referencing from the agent viewpoint allows any orientation to be used.

Transformation generalisation

While the previous solution solves the problem for initial conditions, the transformation descriptor $\hat{\boldsymbol{T}}$ remains relative to the initial orientation of the agent. Hence, though the initial conditions are met where required, the resulting transformations would remain invalid as shown in Figure 4.7. To this end, the x and y components of $\hat{\boldsymbol{T}}$ are rotated about the z-axis of the projection by the absolute orientation λ, of the active face (see Figure 4.8) as Equations 4.7 and 4.8 show:

$$\boldsymbol{R}_\lambda = \begin{bmatrix} cos(\lambda) & -sin(\lambda) & 0 \\ sin(\lambda) & cos(\lambda) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

where $\boldsymbol{R}_\lambda$ is a transformation descriptor rotation matrix. Hence,

$$\hat{\boldsymbol{T}}^r = \hat{\boldsymbol{T}} \cdot \boldsymbol{R}_\lambda^\intercal \tag{4.8}$$

where $\hat{\boldsymbol{T}}^r$ is the reference transformation descriptor for the affordance and $\boldsymbol{R}_\lambda^\intercal$ is the transpose of $\boldsymbol{R}_\lambda$.

The rotational component α of the transformation descriptor $\hat{\boldsymbol{T}}$, is unaffected as it is solely dependent on the initial orientation of the object, which is now factored as discussed in the previous section. By rotating the $\hat{\boldsymbol{T}}$, the affordance is made relative to the agent's view point. Furthermore, in order to

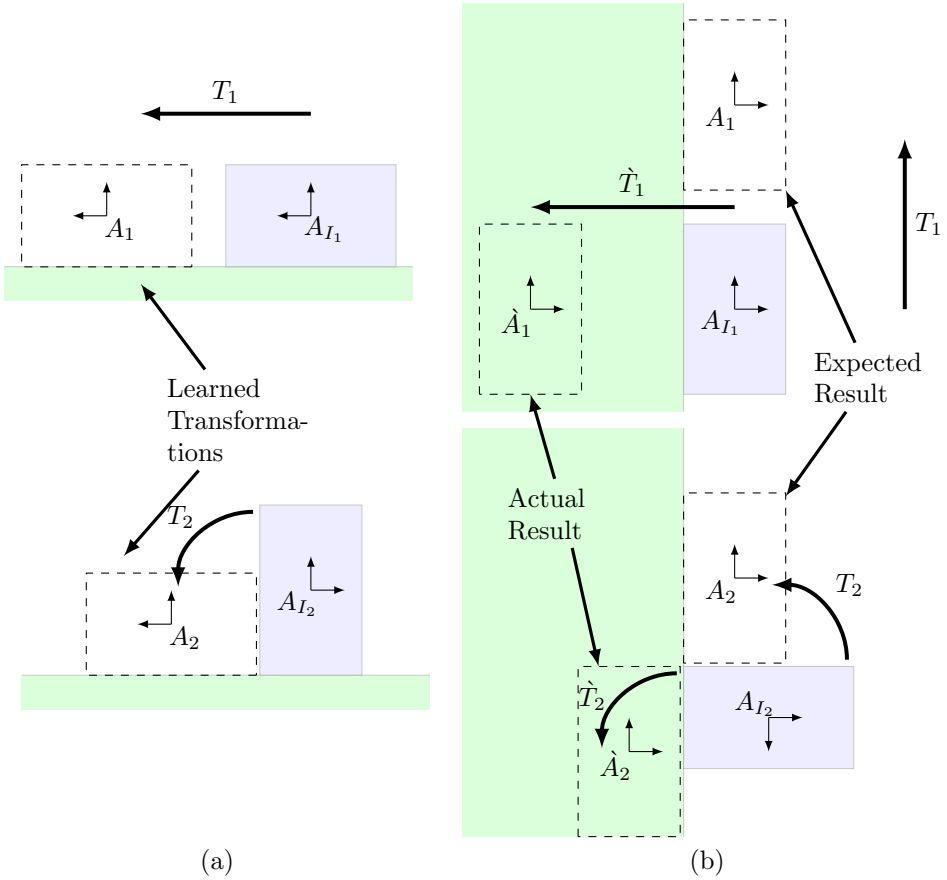


Figure 4.7: Demonstrates how transformation descriptors become invalid when the orientation of the agent changes. (a) shows some transformations that are learned in one orientation. (b) shows how those transformations ($\grave{\boldsymbol{T}}_1$ and $\grave{\boldsymbol{T}}_2$) are invalid when attempted to be used at a different orientation to that which they were learned. The resulting frames are $\grave{\boldsymbol{A}}_1$ and $\grave{\boldsymbol{A}}_2$. The expected frames are $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$.

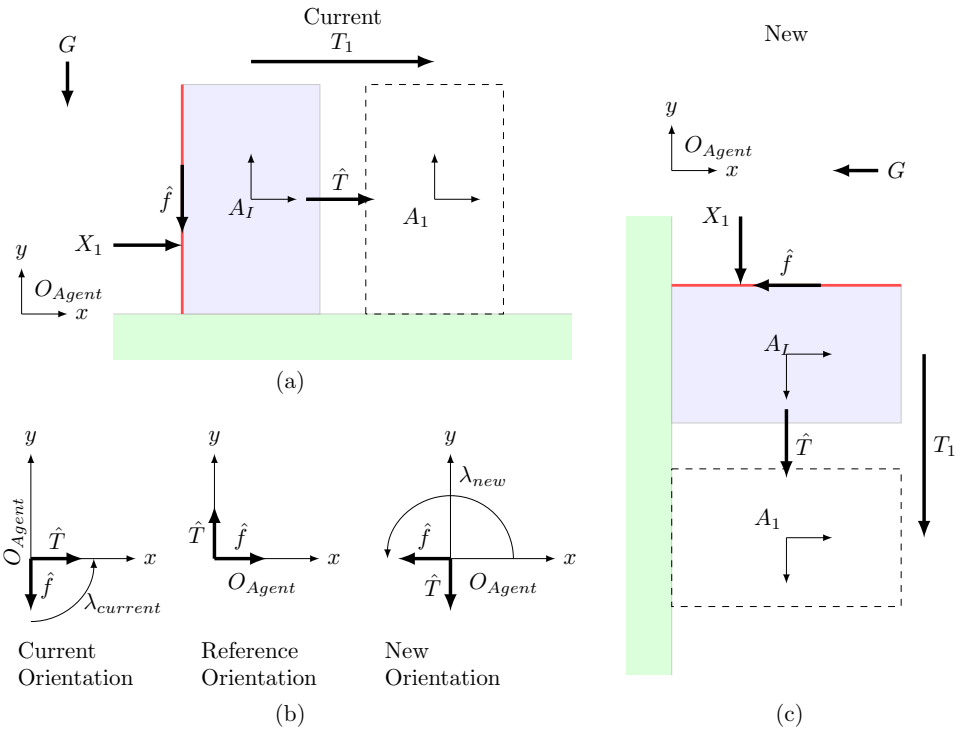


Figure 4.8: Generalising the transformation descriptor. (a) shows a simple transformation that has been learned. In (b) this transformation descriptor $\hat{\boldsymbol{T}}$ is rotated by the absolute orientation of the face $\lambda_{current}$ to the agent reference orientation. To use the $\hat{\boldsymbol{T}}$ in a different agent orientation as shown in (c), the $\hat{\boldsymbol{T}}$ is rotated by the angle of the orientation of the face λ_{new}.

properly reference the constraints, they are also rotated by the absolute orientation of the active face λ, to the agent's zero, or reference orientation.

$$\hat{\boldsymbol{C}}^r = \hat{\boldsymbol{C}} \cdot \boldsymbol{R}_\lambda^\intercal \tag{4.9}$$

where $\hat{\boldsymbol{C}}^r$ is the reference constraint descriptor for the affordance.

In doing so, no matter what orientation the agent has, the affordance remains valid. The affordance can be converted to the current object frame simply by rotating the constraints and $\hat{\boldsymbol{T}}$ back to the current orientation of the active face, based on the primary constraint orientation (see Figure 4.8b and c).

Factoring and generalising gravity

The assumptions made thus far, assume that gravity will follow changes in agent or constraint orientation when it is acting normal to the surface constraint of the environment. However, this is generally not the case as gravity remains constant regardless of the orientation of the agent or constraints. Therefore, to account for gravity, it must be considered separately. The gravity vector $\boldsymbol{G}$, is used to describe the direction and magnitude of gravity in the environment. $\boldsymbol{G}$ must remain constant over the duration of a transformation, otherwise it may not be possible to reproduce the transformation. Therefore, frame $\boldsymbol{A}$ representing the current state of the object contains the current gravity such that $\boldsymbol{G} \subseteq \boldsymbol{A}$. When comparing affordances, the associated $\boldsymbol{G}$ must be within a threshold for the affordances to be considered a match. Similarly to $\hat{\boldsymbol{T}}$, in order to generalise gravity such that agent and constraint orientation are factored, $\boldsymbol{G}$ is rotated by ϕ to the reference point:

$$\boldsymbol{R}_\phi = \begin{bmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

where $\boldsymbol{R}_\phi$ is a constraint descriptor rotation matrix. Hence,

$$\boldsymbol{G}^r = \boldsymbol{G} \cdot \boldsymbol{R}_\phi^\intercal \tag{4.11}$$

where $\boldsymbol{G}^r$ is the reference gravity descriptor and $\boldsymbol{R}_\phi^\intercal$ is the transpose of $\boldsymbol{R}_\phi$.

An affordance also contains the gravity vector at the time it was first learned, such that $\boldsymbol{G}_I^r \subseteq \boldsymbol{I}$. In the surface representation cases when gravity is parallel to the agent view, $\boldsymbol{G}$ will be zero and therefore match any agent orientation.

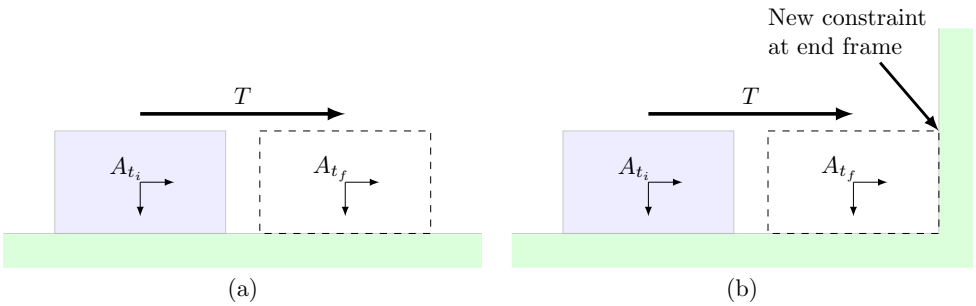


Figure 4.9: The difference between the final frames of transformations when a constraint is present. This can significantly change the outcome of a transformation. In the case of (b), the object cannot be pushed any further.

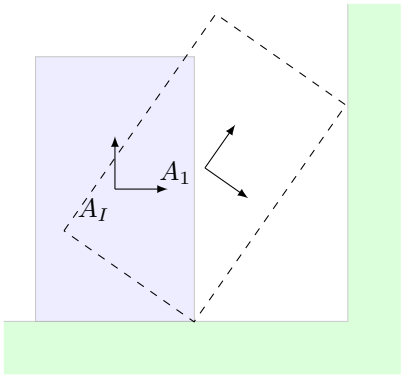


Figure 4.10: A transformation which cannot be described without the final frame. The final frame describes the constraints acting on the object which allow the object to remain in its final position. Without the final frame the transformation could never be reliably reproduced

Initial and final constraints

When considering the initial and final frames of a transformation, the constraints *in effect* can change based on the position of the object within the environment. Although transformations and initial conditions are identical for some affordances, the end frame can have a significant effect on the outcome of an interaction. Consider Figure 4.9, where transformations with identical $\hat{\boldsymbol{T}}$ occur, and the only difference is the end frame constraint. Due to the added constraint, the distance the object is translated is limited. When using affordances to complete a task, the planner must somehow factor in the constraint. Typically, this has been done by simulating the physics of the object in the environment to predict the outcome. However, this can lead to poor assumptions if the simulation parameters which describe the actual environment, are imprecise. Furthermore, relying on simulations can be computationally expensive. For certain affordances, defining the end state is essential to properly defining the affordance. Consider

Figure 4.10, where the object is pushed over to rest on a wall. It is impossible for this to be properly defined without a final state describing the constraint used. Therefore, in this thesis, the constraints are learned in conjunction with the transformations. Where both the initial and final frame constraints are used to characterise the affordance. This approach is preferable as it can potentially be more reliable as the affordance is based on what was learned, or actual outcomes, rather than simulating the outcome. The definition of an affordance can therefore be updated to include the list of final constraints such that $\boldsymbol{L}_E \subseteq \boldsymbol{P}$. $\boldsymbol{L}_E$ is a list containing all the constraint descriptors which are *in effect* at the end frame $\boldsymbol{A}_E$.

The definition of an affordance can therefore be redefined from Equation 3.9 as follows:

$$\boldsymbol{P} = \{\hat{\boldsymbol{T}}^r, T^d, \boldsymbol{I}, \boldsymbol{L}_E, \boldsymbol{f}, \boldsymbol{X}, \hat{\boldsymbol{F}}\} = g(\boldsymbol{A}_I, \boldsymbol{A}_E, \boldsymbol{M}_I) \tag{4.12}$$

4.1.7 Using constraints

Constraints can either be artificially defined by a user, or detected and interpreted by a visual system. In this thesis, the former approach is taken for simplicity. The latter approach can be addressed in future work.

Determining if interaction is possible

When learning affordances, constraints can be used to determine if a face can be accessed by the agent. If the face has a level 2 or higher contact, then it is considered unreachable for the agent. In the interest of simplicity, when a face has a level 1 contact it is assumed that the entire face is reachable, though this is not always the case as shown in Figure 4.11a. It should be noted that in this thesis it is assumed that the polygon the object forms, is a convex shape. Each face is also assumed to be a straight line segment of the polygon. This means that the contact state of the object can never be in a state where faces are blocked from access from a manipulator as shown in Figure 4.11b.

Planning

In order to include constraints in the planning process, the method described in Chapter 3 essentially remains the same. However, as there are now constraints, the face contact information (f^c) from the state $\boldsymbol{S}$ of frame $\boldsymbol{A}$ of node $\boldsymbol{N}$, can be used to determine if the face is accessible for the desired move the affordance describes. Hence, no child node is generated if f^c is level 2 or higher. If a face $\boldsymbol{f}$ in $\boldsymbol{P}$ is valid (level 0 or 1), the constraints in $\boldsymbol{I}$ and $\boldsymbol{L}$ are then rotated by the orientation of the face.

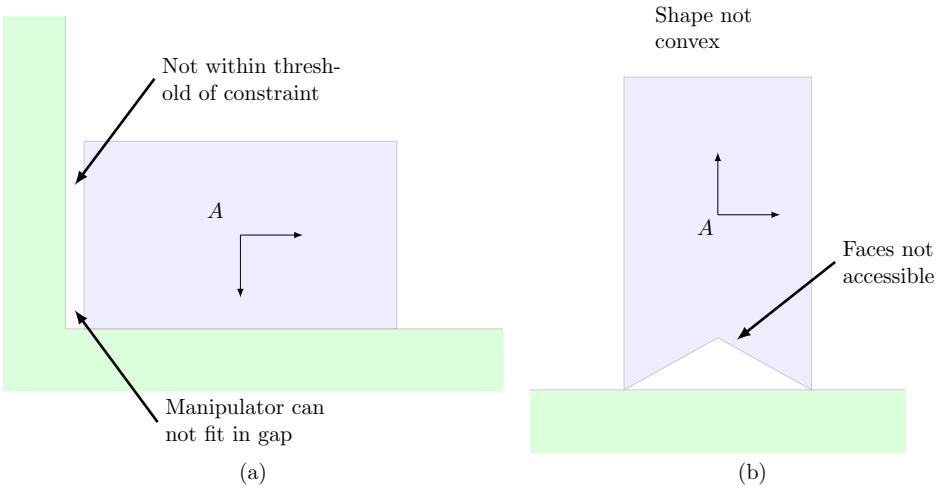


Figure 4.11: Depicts scenarios where interaction by the agent's manipulator are not possible. (a) demonstrates how the gap between the object face and the constraint is large enough to not be considered *in effect*, but too small for the agent's manipulator to fit. (b) shows a concave shape which only has faces with a level 1 contact, however due to the shape, the agent's manipulator cannot interact with these faces without disturbing the object unintentionally.

$$\hat{\boldsymbol{C}} = \hat{\boldsymbol{C}}^r \cdot \boldsymbol{R}_\phi \tag{4.13}$$

ϕ is then calculated for the current node frame state $\boldsymbol{S}$. The initial conditions, $\boldsymbol{I}$, of the affordance are then compared to $\boldsymbol{S}$. If all conditions are met, the transformation can be generated. However, as the $\hat{\boldsymbol{T}}$ has previously been rotated to the agent's reference orientation, it must be rotated back to the current orientation of the face $\boldsymbol{f}$ of $\boldsymbol{P}$ as shown in Equation 4.14:

$$\hat{\boldsymbol{T}} = \hat{\boldsymbol{T}}^r \cdot \boldsymbol{R}_\lambda \tag{4.14}$$

Combining Equation 3.16 and 4.14 we therefore get:

$$\boldsymbol{T}_i = \begin{cases} \hat{\boldsymbol{T}}^r \cdot \boldsymbol{R}_\lambda \cdot e^{i-q}, & \text{if } \hat{T}_\alpha < \frac{1}{\sqrt{2}}, \quad \forall i \in \{0, \ldots, q\} \\ \hat{\boldsymbol{T}}^r \cdot \boldsymbol{R}_\lambda \cdot T^d, & \text{if } \hat{T}_\alpha \geq \frac{1}{\sqrt{2}}, \quad \text{where } i = 0 \end{cases} \tag{4.15}$$

When evaluating each move of a node frame, the agent must check to ensure that the object has not passed through a boundary a constraint describes. The difference between the node parent frame (or initial) and the current node frame (or final) is checked for an intersection with a constraint boundary. If an intersection is detected, the transformation is adjusted by Δd, which is the largest distance between

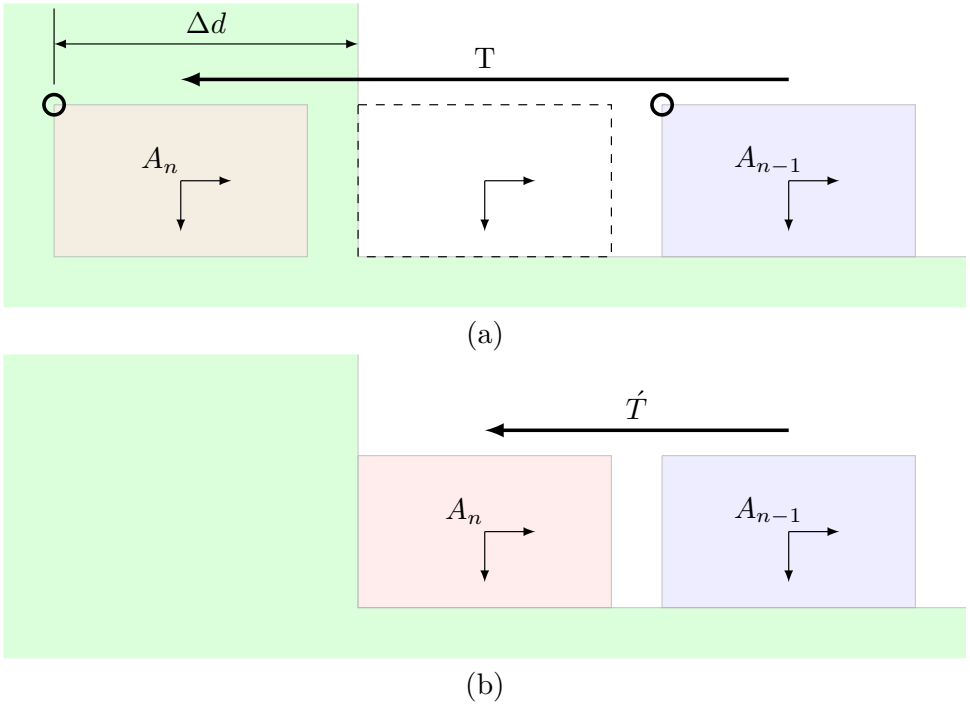


Figure 4.12: Constraint intersection compensation. Constraint intersection occurs in (a) and the largest vertex to constraint distance is labelled Δd. The transformation is then adjusted to obtain $\acute{\boldsymbol{T}}$ in (b).

the constraint boundary and any vertex of the object (see Figure 4.12). Hence, equation 4.16 is used to generate a correct $\boldsymbol{T}$:

$$\acute{\boldsymbol{T}}_i = \boldsymbol{T}_i \cdot (\Delta d - T^d) \tag{4.16}$$

where $\acute{\boldsymbol{T}}_i$ is the adjusted transformation used to produce a new frame $\boldsymbol{A}_n$.

If the transformation is classified as a rotation, due to the difficulties with rotational motion discussed in chapter 3, and rather than working back to determine where the object is likely to contact the constraint, the move is simply flagged as invalid and pruned from further tree generation. The constraints *in effect* for the new child node are also compared to the final constraints $\boldsymbol{L}_E$ of the affordance, which the move is based on. If they do not match, the move is pruned from further tree generation.

Reconstruction noise

In some cases an affordance may have noise introduced into the transformation descriptor. If the magnitude component of the transformation is small and it is later used for a transformation with a large magnitude, the noise can be amplified, resulting in an invalid state. This can be compensated for in the case of pure translational moves by identifying constraints which are parallel to the direction of motion. It is then assumed that motion will follow that of the constraint. Therefore, instead of using the transformation descriptor of the affordance, the descriptor of the constraint is used. The constraint must be in effect for both the initial and final frames of the affordance. This method is only valid for affordances where the gravity vector $\boldsymbol{G}$ is not parallel to a constraint.

4.2 Symmetry

Symmetry is present in many aspects of the real world and presents unique opportunities for exploitation in manipulation contexts. It has been used to varying degrees for classification and identification [56–58]. For the purpose of this thesis, symmetry is not learned by the agent as it is beyond the scope of the research objective. Rather, it is detected and recognised by the agent, at which point the appropriate action is taken.

4.2.1 Defining symmetry for affordances

Objects with symmetry have a redundancy of information as Attneave discusses [59], which is also applicable in the context of learning affordances. In this thesis, this redundancy is represented by two

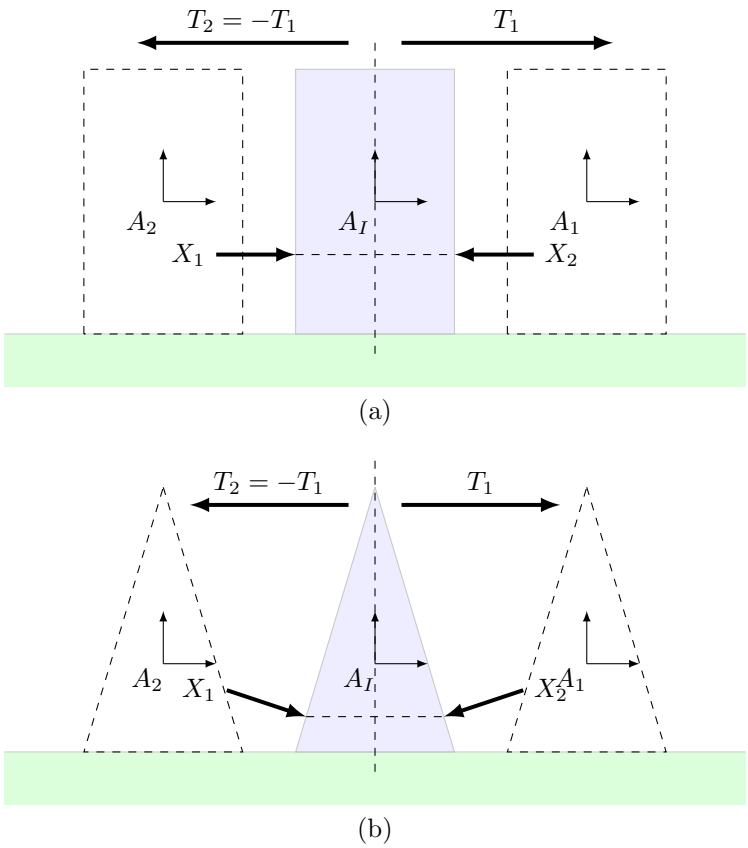


Figure 4.13: Demonstrates symmetrical transformations. A simple relationship can be observed which implies that symmetrical transformations are simply the reverse of the original.

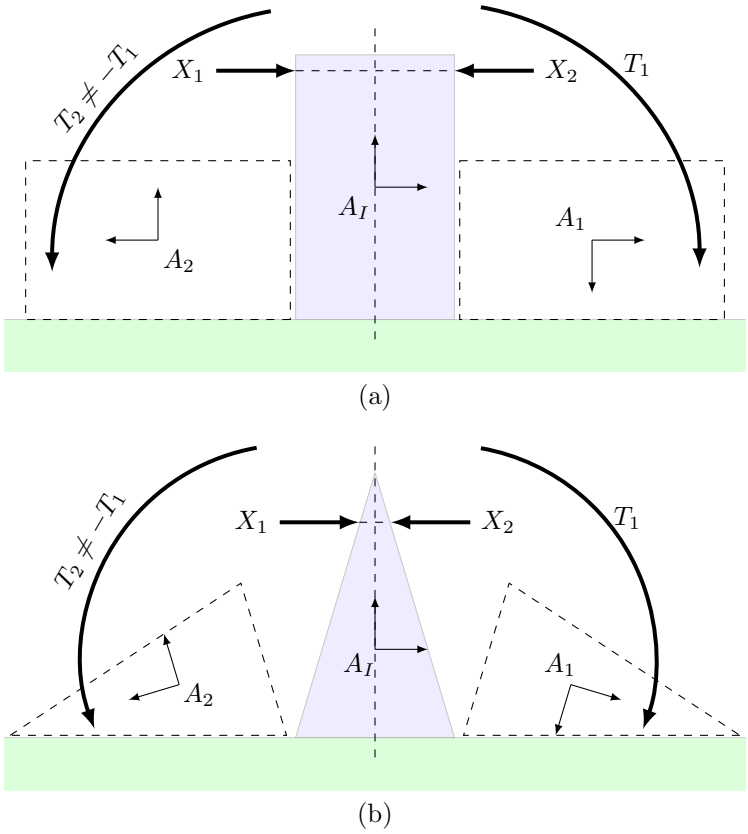


Figure 4.14: Rotational transformations demonstrate how the symmetry relationship can not simply be the inverse of the original transformation. In this case, the transformation is only reversed along the x axis. The y axis remains the same.

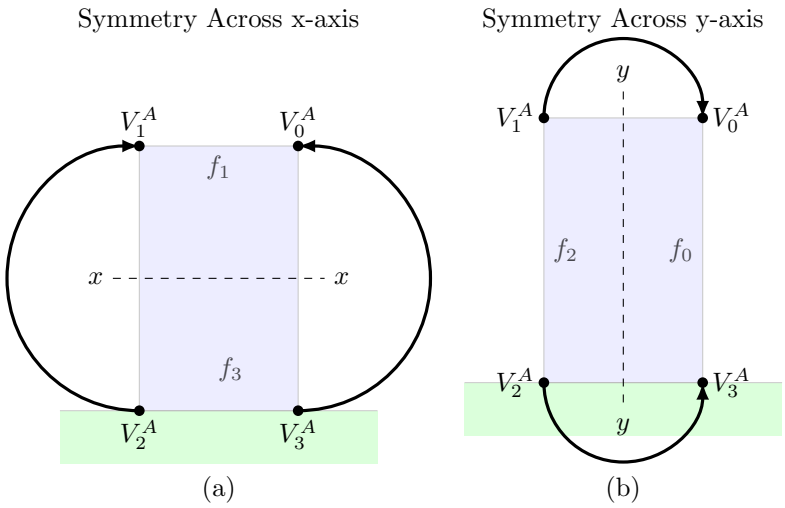


Figure 4.15: Detection of symmetry. (a) compares the vertices across the x-axis to find matches. (b) compares the vertices across the y-axis to find matches.

faces of an object which are symmetrical across and/or perpendicularly across the axis of minimum moment of inertia. For example, shapes such as the rectangle and isosceles triangle shown in Figure 4.13, have two faces which are mirrored across their minimum moment of inertia axis and are effectively identical from a manipulation context. Consider the rectangle (Figure 4.13a) which is pushed on one face, and the same action is mirrored on the opposite. The same is true for rotational transformations as shown in Figure 4.14. In both cases in Figures 4.13 and 4.14 the transformations from interaction at X_2 result in a transformation in a mirrored direction to the interaction from X_1. Hence, for object faces with symmetry, for all transformations there exists a mirror transformation. It is therefore unnecessary to consider symmetrical faces as being mutually exclusive, requiring separate exploration. Linking the faces together and factoring the mirrored direction in the transformation descriptor $\hat{\boldsymbol{T}}$, simplifies learning of the object's affordances. In the case of the rectangle, there is also symmetry perpendicularly across the axis of minimum moment of inertia, reducing the number of faces to explore from 4 to 2. As a result, the exploration and learning time is effectively halved. Hence, the complexity of the shape is reduced.

As the agent relies purely on visual information to detect symmetry, the assumption is made that the object texture and friction are also identical. If symmetrical faces were to have significantly different friction coefficients, the object may behave unexpectedly, producing different affordances. The agent could further explore the object to determine the coefficient of friction for each face. However, it was deemed beyond the scope of this thesis to pursue as it is not directly relevant to the objective.

4.2.2 Identifying symmetry

Symmetry detection is performed when the agent first encounters the object and is identified by comparing the vertices of each face. The object's frame axis orientation is aligned with the axis of minimum moment of inertia. The y-axis is parallel to the axis of minimum moment of inertia and the x-axis is perpendicular. Each face is compared by mirroring the vertices along the y-axis, for symmetry across the x-axis, and along the x-axis, for symmetry across the y-axis (see Figure 4.15). If the vertices match up within a threshold then the faces are considered symmetrical and subsequently linked. This process is repeated on each of the remaining unlinked faces, until all faces have been checked for symmetry (see Appendix A).

When transformations are performed using a face with symmetry, there are two outcomes which change how the $\boldsymbol{T}$ is generated from the $\hat{\boldsymbol{T}}$. The first, is when there are constraints *in effect* on the object. As shown in Figure 4.13, the result when a constraint is *in effect*, is a mirror in direction. This can be factored by rotating the $\hat{\boldsymbol{T}}$ from either of the symmetrical faces current orientation, to the reference orientation to produce $\hat{\boldsymbol{T}}^r$, as previously described in Section 4.1.6. When generating a transformation for symmetrical faces, $\hat{\boldsymbol{T}}^r$ can be rotated to the orientation of either of the faces to produce the correct $\hat{\boldsymbol{T}}$. This results in the transformations shown in Figure 4.13. However, when considering Figure 4.14, there is a translation along both x-axis and y-axis, as well as a rotation. Therefore, simply rotating $\hat{\boldsymbol{T}}^r$ to the orientation of the current face, does not produce the desired result.

To illustrate this, consider Figure 4.16, where faces $\boldsymbol{f}_0$ and $\boldsymbol{f}_2$ are symmetrical and therefore linked. In (a), a transformation is performed on the active face ($\boldsymbol{f}_2$), and subsequently learned. This involves the $\hat{\boldsymbol{T}}$ being rotated to the agent's reference orientation by the angle of $\boldsymbol{f}_2$ (λ_2), as shown in (c). In (b), $\hat{\boldsymbol{T}}^r$ is rotated to the orientation of $\boldsymbol{f}_0$ (by λ_0) as shown in (d) to attempt to produce a symmetrical transformation. The outcome is a transformation which does not produce the desired result. This puts the object in an invalid state, effectively floating above the ground. To account for this effect, it can be seen that the transformation in (b) simply needs to be inverted for both the y and α components of $\hat{\boldsymbol{T}}$. However, when the orientation of a face is not aligned with an axis as is the case in (b), the required transformation can be difficult to ascertain. Hence, in Figure 4.17(a), after rotating the $\hat{\boldsymbol{T}}$ by λ_2 to the agent's reference axis, the $\hat{\boldsymbol{T}}$ x and α components are inverted (see Equation 4.17) as shown in (c). The interaction point X_μ and force direction $\hat{F}$ x component are also inverted to account for symmetry (see Equations 4.18 and 4.19). Now when $\hat{\boldsymbol{T}}^r$ is rotated to $\boldsymbol{f}_0$ by λ_0 as shown in (d), the result is the desired transformation, symmetrical to that of $\boldsymbol{f}_2$ as shown in (b).

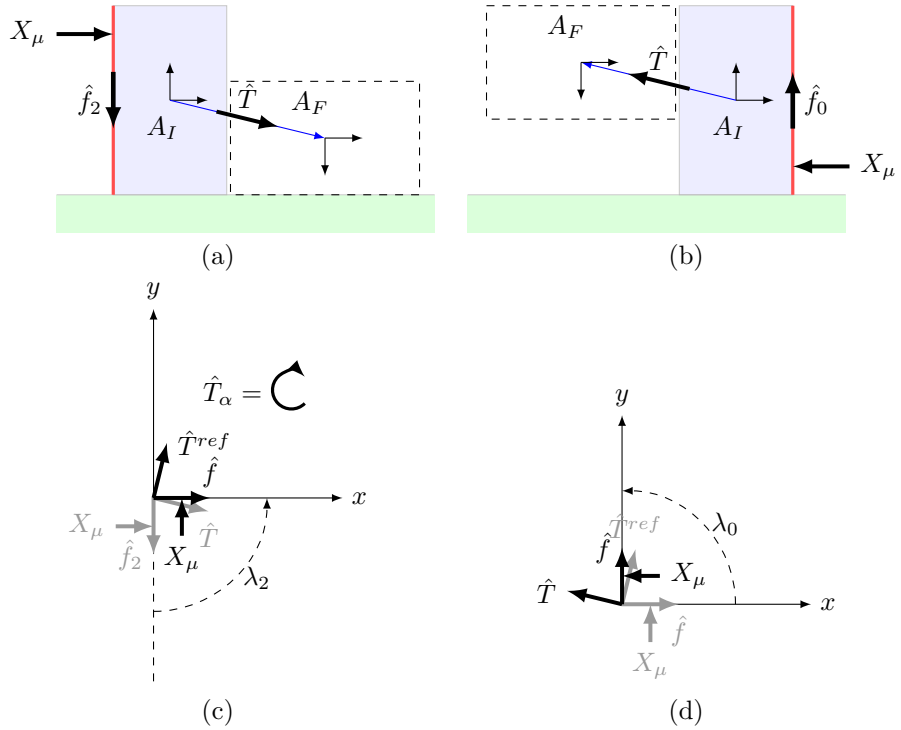


Figure 4.16: Problems with symmetrical faces. A transformation is performed and learned from an interaction on $\boldsymbol{f}_2$ in (a) and rotated to the agent's reference axis as shown in (c). The transformation descriptor $\hat{\boldsymbol{T}}$ is attempted to be used on a symmetrical face $\boldsymbol{f}_0$ in (b) by by rotating the $\hat{\boldsymbol{T}}$ to the orientation of $\boldsymbol{f}_0$ as shown in (c). This produces an invalid object state.

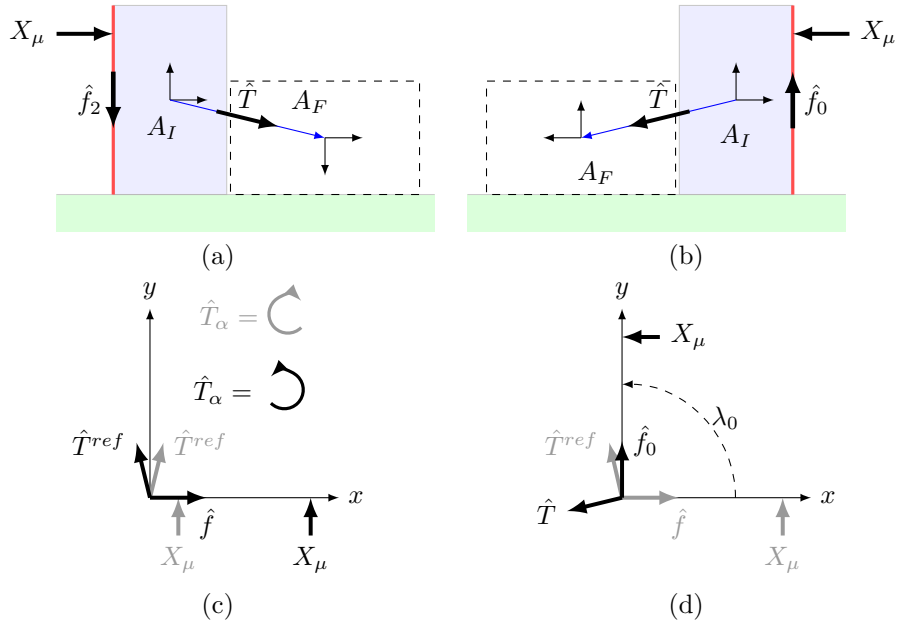


Figure 4.17: Generalising symmetrical faces transformations. A transformation is performed and learned from an interaction on $\boldsymbol{f}_2$ in (a) and rotated to the agent's reference axis. However, as face f_2 has a negative orientation, the orientation component $\hat{T}_\alpha$, x component $\hat{T}_x$, interaction point X_μ and force direction x component F_x are inverted as shown in (c). When the $\hat{\boldsymbol{T}}$ is rotated to $\boldsymbol{f}_0$ as shown in (d), the correct transformation is produced.

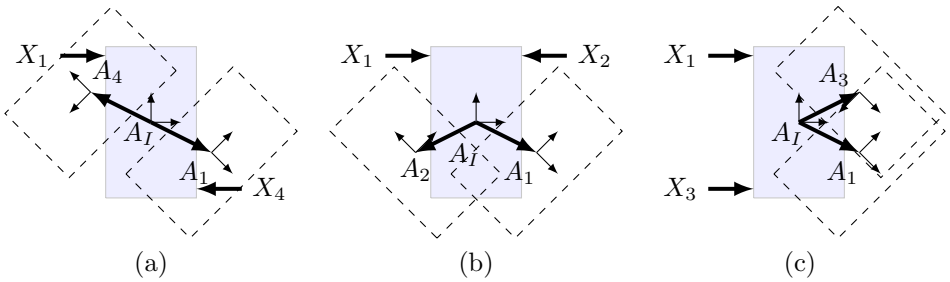


Figure 4.18: Symmetry when using Top Projection environment. The object behaves differently to side Projections. The object effectively has symmetry on all 4 faces as there are no constraints *in effect*.

$$\hat{\boldsymbol{T}} = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \hat{\boldsymbol{T}} \tag{4.17}$$

$$X_\mu = -X_\mu \tag{4.18}$$

$$\hat{F}_x = -\hat{F}_x \tag{4.19}$$

Only faces which have a negative orientation, such as $\boldsymbol{f}_2$ has in Figure 4.16, are inverted. Likewise, faces which are at orientations of approximately π, are also inverted. This produces a correct $\hat{\boldsymbol{T}}^r$ regardless of the orientation of each symmetrical face at any given time. Therefore, even if a face was originally in a positive orientation when an affordance was learned, if currently in a negative orientation, the correct transformation can be generated.

The second outcome, is primarily when there are no constraints *in effect* and is typically applicable to surface projection environments. As shown in Figure 4.18, the behaviour of the object with symmetrical faces is different to that of those in Figure 4.14. An interaction on any part of a face can be replicated on a symmetrical face by rotating the $\hat{\boldsymbol{T}}^r$ to the desired face's orientation. It is unnecessary to perform any kind of inversion on the symmetrical face to produce a valid transformation.

This approach therefore reduces the complexity of objects with symmetry and facilitates a more generalised approach to learning affordances.

4.2.3 Adjacent faces

When a face that is adjacent to two faces which are identified as having symmetry, the face is also treated as if it has symmetry itself. This holds true, even if the face itself has not been identified as having a

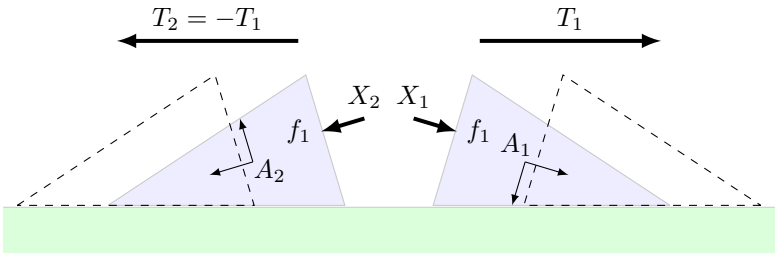


Figure 4.19: Symmetry due to adjacent faces. A face which has two adjacent faces with symmetry behaves as if it has symmetry itself. Hence, for each transformation there exists a mirror transformation. In this case $T_2 = -T_1$.

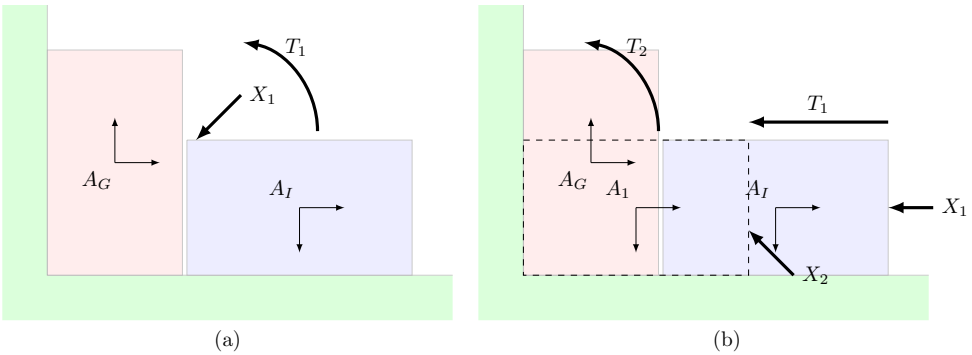


Figure 4.20: Improving the reliability of plans. In (a) the object undergoes a sheer operation to flip the object over to the goal frame $\boldsymbol{A}_G$. The can be prone to slip, causing failure. A more reliable plan is shown in (b) which utilises the wall constraint. Although more moves occur, each has a high probability of success, thereby representing the most reliable plan.

symmetrical face. For example, in Figure 4.19, the face $\boldsymbol{f}_1$ does not have a symmetrical face. However, it behaves as if it does due to the faces adjacent to it.

4.3 Improving planning

In Chapter 3, when multiple possible plans were generated, the first encountered plan with the least number of transformations was selected. While this is effective for many scenarios, it fails to consider the case when there are multiple plans with the same minimum number of transformations. It further does not consider the reliability of affordances. Consider the plan shown in Figure 4.20a. While this has the minimum number of transformations, it is potentially far less reliable than the plan in Figure 4.20b. This is due to a smaller area of interaction in (a) to that of (b), in which the agent can be confident the transformation will be successful. Furthermore, if other transformations are possible, whose interaction

point is in close proximity to the area of interaction of the desired transformation, the certainty that the interaction will produce the desired result may be reduced. Therefore, defining and using the concept of uncertainty, will help to produce a plan with the minimum number of transformations and the highest certainty of success. The planner is therefore simply required to find the sequence of transformations with the lowest uncertainty value.

4.3.1 Obtaining uncertainty

To calculate the uncertainty of any transformation, only affordances whose initial conditions meet the current state $\boldsymbol{S}$ of the object are considered. The probability of each valid move over an area of the object face $X_\mu - \eta < X_\mu < X_\mu + \eta$ is calculated and subsequently averaged. Where η is a threshold typically set to 0.1.

$$\xi = \frac{1}{n} \sum_{p=P_0,..,P_{n_p}} \operatorname{erf}\left(\frac{X_\mu - \eta - X_\mu^p}{\sqrt{2 \cdot (X_\sigma^p)^2}}\right) - \operatorname{erf}\left(\frac{X_\mu + \eta - X_\mu^p}{\sqrt{2 \cdot (X_\sigma^p)^2}}\right) \tag{4.20}$$

where ξ is the uncertainty, p is an affordance of n_p affordances, erf is the error function, X_μ is the interaction point of the desired affordance and X_μ^p and X_σ^p are the average interaction point and standard deviation of the interaction point list $\boldsymbol{X}$ respectively, of affordance p whose initial conditions meet the current state.

ξ represents the probability of performing a transformation other than the intended one. While this is in essence an approximation of the probability based on the standard deviation of the interaction points, it can be used as a reasonable approximation.

4.3.2 Manipulator considerations for uncertainty

The manipulator is a factor when calculating the uncertainty which mostly cannot be accounted for. If the manipulator is highly accurate or repeatable, the uncertainty probability is likely to be lower. If the manipulator is sub par such that accuracy or repeatability are in question, the uncertainty probability is likely to be higher. While this is factored to a small extent when learning affordances, if the manipulator was switched post-learning, the uncertainty may produce incorrect optimisations. The agent would need to go through a calibration process to check the various affordances that it has learned. This could be conducted during normal operation while the agent is using the manipulator. This would involve the agent progressively making small adjustments to the affordances to factor the new manipulator configuration. In this thesis, the manipulator configuration is not changed and any effect is assumed to be negligible.

4.4 Summary

In this chapter, constraints were introduced as a means to increase the complexity of environments so as to generalise and facilitate more complex affordances and can be summarised as follows:

- Constraints are the result of a static object restricting motion normal to the face and are defined by 2 vertices, describing a line.
- A constraint is described to be *in effect* when a vertex of the object is within a threshold of the line segment of the constraint.
- The object frame contains a list of constraints *in effect* ($\boldsymbol{L} \subset \boldsymbol{A}$).
- A constraint has 4 levels of interaction, each describing the type of interaction the object's faces have with constraints.
- A constraint is considered primary if it has a level 3 interaction.
- The constraint closest in orientation to $\frac{\pi}{2}$ from gravity is used when multiple constraints have a level 3 interaction.
- A face is active if it is undergoing interaction.
- To generalise a constraint, a unit vector is used to describe it ($\hat{\boldsymbol{C}}$).
- The initial conditions, of an affordance contain a list of constraints *in effect* ($\boldsymbol{L}_I \subset \boldsymbol{I}$).
- The angle between the active face and the primary constraint (ϕ), is used to describe the initial orientation of the object face.
- The transformation descriptor $\hat{\boldsymbol{T}}$ is rotated by $\boldsymbol{R}_\lambda$ so as to allow the transformation to compensate for any orientation the object is in, provided I is met.
- Gravity, $\boldsymbol{G}$, is used as a reference point in the environment to ensure no matter what orientation the agent is currently in, the transformation will be properly reproduced.
- Similarly to $\hat{\boldsymbol{T}}$, $\boldsymbol{G}$ is rotated by $\boldsymbol{R}_\phi$.
- The constraint in the final frame can greatly change the result of an affordance, hence a list of constraints is included in an affordance ($\boldsymbol{L}_E \subset \boldsymbol{P}$).
- Constraints can be defined by a user, or located by a visual system.

- A level 2 or higher constraint implies that the face cannot be interacted with.
- An object must have a convex shape, otherwise problems can occur with face interaction.

When using constraints in a plan the following must occur:

- The constraint descriptor ($\hat{\boldsymbol{C}}$) must be rotated to the current orientation by $\boldsymbol{R}_\phi$ (see Equation 4.14).
- The reference transformation descriptor ($\hat{\boldsymbol{T}}^r$ must be used along with the rotation matrix $\boldsymbol{R}_\lambda$ to produce a valid transformation (see Equation 4.15).
- The agent must check after each transformation if the object results in the object passing through a constraint.
- Transformations which result in passing through a constraint are offset by the distance they protrude the constraint.
- The agent must check if the final frame matches the affordance's final constraint state $\boldsymbol{L}_E$ to ensure the planned move is valid.

Objects with symmetry can be used as a means to reduce the complexity of affordances and is summarised as follows:

- An object is defined as symmetrical if it is mirrored either across or perpendicular to the axis of minimum moment of inertia.
- Vertices are compared across both the x and y axis to identify symmetrical faces.
- Constraints affect how symmetrical objects behave, therefore there are two main approaches to deal with symmetry.
- When constraints are *in effect* and $\lambda < 0$, the x and α components of $\hat{\boldsymbol{T}}$ are inverted.
- When there are no constraints (typically in surface projections) the inversion is unnecessary as simply rotating the $\hat{\boldsymbol{T}}$ to the desired face is sufficient.

Planning uncertainty was introduced so as to optimise the plans that are generated based on the probability of affordances being successful. This involved finding the uncertainty of each affordance based on the probability of a transformation being performed on the same face. This is affected by the area and position of interaction for each valid affordance. The manipulator configuration can also affect the uncertainty, however for the setup in this thesis, the effect is negligible.

Chapter 5

Intrinsically Motivated Exploration and Learning

The basic random exploration approach outlined in Chapter 3 (Section 3.3.2), may work reasonably well for simple environments. However, when more complex environments are introduced (Chapter 4), the effectiveness of random exploration is reduced. Not only the environment, but the object itself presents an interesting challenge when considering how to explore and learn its affordances. Traditionally, the approach has been to use supervised controlled environments which allow the agent to focus on either; learning a single manipulation task, learning each manipulation component individually, or manually introducing the necessary information by programming the agent [3, 8, 10, 11]. This typically involves using a controlled environmental setup, where the object is then placed in a favourable position. A human created process then directs the manipulator, allowing the agent to interact and learn via the chosen method. This approach allows the agent to ignore many of the complexities of the object as well as the environment, improving learning time and reducing difficulty.

With increasingly more complex environments and objects, manually demonstrating, assisting or programming an agent becomes a tedious process. It is preferable that the agent explore and learn without assistance. However, this requires the agent to know what and where to explore. This chapter therefore introduces and examines the problem of autonomous exploration in terms of learning affordances. Specifically it looks at methods of intrinsic motivation (self motivation) [60] to direct the agent's actions and improve upon random exploration.

In this chapter; Section 5.1 discusses the issues with object geometry in exploration, Section 5.2 explores methods of intrinsic motivation, Section 5.3 describes how intrinsic motivation is used for exploration., Section 5.4 discuss further methods to improve exploration and learning, and finally Section

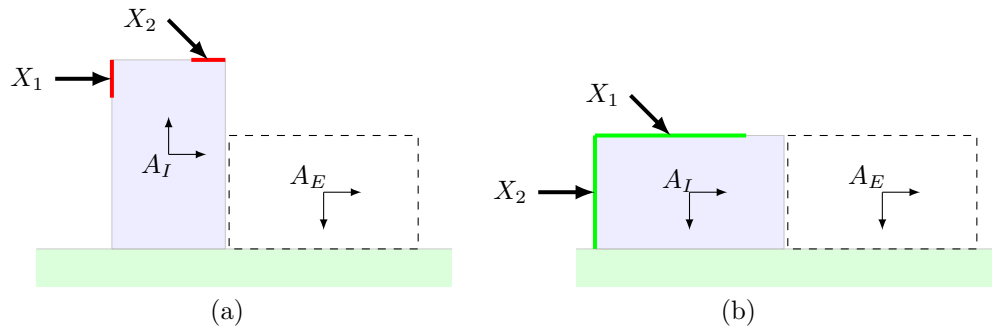


Figure 5.1: Differences in interaction area size. Demonstrates how each affordance can have a different area of interaction based on the initial object state.

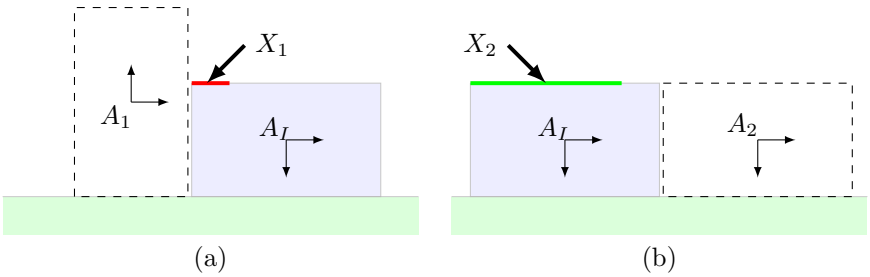


Figure 5.2: Probability of performing a transformation. It is assumed that each face has an equal probability of selection and the probability of choosing any point on a face is uniformly distributed. Hence, the probability due to interaction in the green region is approximately a 25% chance of being selected at the current orientation, while the red region is about a 7% chance.

5.5 summarises the chapter.

5.1 Exploration and factoring geometry of an object

When considering exploration of an object, it is possible, indeed common in some environments, for a limited set of the object's faces to be inaccessible in certain states. This affects what the agent can explore at any given time regardless of what the agent may intend to explore. The state the object is in can also affect the type of transformation it can undergo as discussed in previous chapters. Consider Figure 5.1, where the object is in a common initial state. There are many transformations the agent can not perform due to the current state and constraints placed on the object by the environment. A

further physical restriction on exploration ability involves the size of the area which the agent must interact with to perform the desired transformation. Certain transformations, such as those shown in Figure 5.1a, have a very small area of interaction. Other transformations, such as the one shown in Figure 5.1b, have large area's of interaction. This can make it very difficult for some transformations to be discovered and explored under a uniformly distributed random exploration process. The geometry of the object naturally weights the probability of performing a transformation in each state the object is in. Consider the rectangle in Figure 5.2. If each face has an equal probability of selection and the probability of choosing any point on a face is uniformly distributed, the probability of performing certain transformations can be approximated. For example, interaction in the green region (Figure 5.2b) has a 25% chance of being selected at the current orientation, while the red region (Figure 5.2a) has about a 7% chance. Both red and green regions are only possible in the current orientation, hence the overall probability of selection is half (3.5% and 12.5% respectively, when factoring symmetry). In this example, the assumption is made that there are only two stable states the object can be in. The probability will change in different environments and likely reduce in more complex environments as more varied stable states become possible.

When the state of the object is changed and a transformation is no longer possible, alternative transformations must be used to continue exploring. This geometrical restriction on exploration can have positive effects, as it prevents the agent from repeating some of the same actions multiple times. However, these restrictions can also lead to many unnecessary trials, due to temporary restrictions. Returning to the correct state, selecting the correct face and region randomly, can be difficult, especially for small regions. While attempting this, it is likely that many other transformations will be performed in the meantime. Hence, a significantly longer time is required before the object's affordances have been sufficiently explored and learned. Depending on the number of trials, some transformations may not be discovered at all by virtue of their small probability of selection. It is also statistically possible that all affordances are discovered with minimal trials, this however, is very unlikely to occur.

A random exploration process may work well in uniformly distributed objects and environments, such as a circle on a flat surface. However, there are problems when exploring affordances of objects in non uniform environments with multiple possible states. The remainder of this chapter explores approaches to; improve upon random exploration, counteract the naturally weighted affect an object's geometry has on exploration and provides a method to focus on discovering and learning affordances efficiently.

5.2 Intrinsically motivating an agent

Automating an agent to explore is a commonly researched topic in robotics. Typically, human derived processes defined by simple rules are utilised to perform exploration. However, in the field of artificial intelligence, intrinsically motivating agents is a growing area of interest. A wide array of approaches have been explored. These approaches can generally be classified as either knowledge based, where the agent is motivated to obtain information, or competence based, where the agent is motivated to refine its internal representation of information. In robotics, and specifically in the field of robotic manipulation, there are many examples of competence based methods for manipulation. Various forms of reinforcement learning are typically implemented [22, 61–63]. For example, Oudeyer et al, look at various error minimisation techniques to reward the reduction of error in a task as a means to motivate an agent [64]. While competence based methods may prove useful for evaluation and planning, this thesis focuses on knowledge based methods as they directly pertain to exploratory behaviour. Whilst knowledge based approaches have much less literature in the robotics field, there is much promising work. For example; Merrick and Maher motivate an agent from interesting events [65], Huang and Weng use novelty to motivate an agent [66], Baranes and Oudeyer use curiosity and interest to direct learning endeavours based on learning progress [67] and Sungmoon et al utilise visual attention shifts to generate motor commands for a robotic system [68]. However, there is limited literature which specifically address the topic of exploratory manipulation of objects.

The remainder of this chapter proposes an approach to motivate an agent to explore objects in order to learn manipulation affordances. Interest [69] is used as the motivating force for the agent, where its actions are based on the level of interest of an affordance. Each action involves the basic exploration process described in Chapter 3, which involved simple pushing or sheer actions. To simplify the problem, interest is generated purely based on manipulation information, which is discussed in further detail in the section below. When improving exploration from that of random interaction, the approach should not hinder learning of affordances, but ideally improve the learning capability.

5.2.1 Affordance interest

Determining how an affordance may be interesting in order to induce an exploratory behaviour in the agent, presents an intriguing challenge. There are many ways that an affordance may be interesting. This can range from features such as object motion, frequency of trials and newness to the agent, to context based information such as state information and necessary environment configurations. To simplify the problem, interest is generated purely based on manipulation information. It is defined by a single value, denoted ϵ, over the domain [0,1] where 0 is uninteresting and 1 is highly interesting. This section seeks

to answer the question: what should be classified interesting to an agent, in the context of learning affordances and how can it be used to improve learning?

Object movement

It was initially theorised that motivating the agent to simply cause some kind of movement of the object could be interesting, and thereby motivating the agent to explore. Interest was therefore defined to be proportional to the magnitude of the transformation of the object (T^d). Such movement would assist in discovering a range of affordances, allowing the agent to focus on certain areas of the object for a time, then moving to others. However, transformations which are effectively the same type, such as a translation along the x-axis, have different levels of interest due to a different T^d. Simply moving objects further would achieve a higher interest value and as a consequence the agent would tend to focus on the interaction which produced larger movements. Objects which involved a rotation would also appear more interesting compared to translations, due to the greater combined translational and rotational magnitude.

The agent, would therefore be motivated to maximise its discovery of larger movements, ignoring potentially valuable smaller movements. The currently discovered largest possible transformation would then become the focus of the agent and repeated indefinitely. A time factor could be introduced to reduce ϵ over time. However, this would result in the agent simply moving the object even further to increase ϵ. The environment boundaries, constraints, or the limit of manipulator reach, could then become an issue. Hence, the object's current location in the environment would greatly affect how interesting it appears to the agent, despite it having no influence on the transformation itself. There is also a point where it serves no further purpose when learning an affordance to continue moving the object, as discussed in Section 3.3.1.

Even if a time factor is used, such an approach would simply mean that larger transformations would be explored before smaller ones. While this may be advantageous, as initially learning larger transformations may be desirable in some circumstances, it does not help the agent explore and learn different types of transformations. Hence, by itself, a displacement approach is unlikely to provide any improvement over a random exploration process. While displacement may still be a factor in classifying something as interesting, the core reason why a transformation would be interesting to the agent, should involve other metrics. It may not be the distance itself, but rather the reason why the distance was larger or smaller, that should be interesting.

Patterns and goals

Work has been conducted that looks at finding patterns or sequences in learning data and intrinsically motivating the agent to find new patterns and utilise them [65]. Such an approach could potentially provide a method to generate interest to direct the agent's actions. Unique patterns of sequenced transformations could be used to identify a solution to change the object's state in ways which require the use of multiple affordances. An alternative, but similar approach, could be to use goal achievements as a means to motivate the agent. Interest can be generated based on completion of a goal, where the affordances involved in completing the goal have their interest modified. This would tend to cause the agent to perform affordances which produce transformations that are more likely to help it complete goals. The problem with the latter approach is that goals would need to be identified in some manner, either requiring external input or a system to self generate the goals. Without understanding anything about the affordances themselves, it would be difficult to generate meaningful goals that improve the exploration ability of the agent.

While these approaches allow the agent to learn how to complete specific tasks, the process has been significantly complicated before achieving any form of learning improvement over random exploration. In both the afore mentioned cases, the agent would still have to perform random interactions to discover the various possible affordances in order to perform a task or achieve a goal. In addition, the agent would have a delayed response from performing the first transformation to completing the last. This could lead to large periods where nothing is discovered, due to the randomness of the transformations performed. The agent may continually perform the wrong transformations and never reach a goal, or only find a small subset of sequences until finally discovering a new affordance and consequently a new range of sequences. These approaches also raise the question of how the agent switches between performing a task and learning a new task. Furthermore, these approaches do not address specific object transformations themselves, rather, multiple transformations that are grouped together to perform a task. They only improve the agent's exploration when sufficient affordances have been discovered to allow the agent to reach the majority of states required to complete a task. Therefore, the agent must be motivated to first discover fundamental affordances to reliably reach these states.

Novelty

It was then theorised that novelty of individual affordances could be used to motivate the agent towards exploring newly discovered affordances. The dictionary generally defines novelty as an entity or concept being unique or new. Hence, as the entity or concept becomes more familiar and exposure is increased, novelty is reduced. In robotics, novelty is typically represented by the number of trials in relation to an entity. In this case, the entity is an affordance.

Novelty directly relates to exploration and has a strong relationship to interest [70, 71] and is what motivates many forms of learning and exploration [72,73]. The more an affordance is performed the less likely it will be performed again. Consequently the more likely other affordances will be performed or discovered. The agent will seek to only perform or find affordances which are considered novel. When a new affordance is discovered, the agent is compelled to explore the new affordance. Problems arise with this approach as a new affordance may be discovered which is unrepeatable. For example, consider a particular interaction which has a 99% change of falling over and a 1% chance (due to slip) of a pure translation. Once the pure translation is discovered, it becomes very hard to repeat and has a low chance of success. The agent would become stuck trying to attempt interaction in the same area, unable to perform the novel affordance for some time. A time factor could be introduced to reduce the novelty of all affordances over time. However, due to the difficulty of reproducing some valid affordances, those that are valid may essentially become forgotten due to the novelty reducing to nothing. This effectively prevents such affordances from being further verified, harming the exploration ability of the agent. An individual time factor could be used, however, this would still mean that a significant amount of time is spent unnecessarily repeating interactions that will produce an undesired outcome.

In the psychology field, significant research has been conducted into the concepts of curiosity, novelty, exploration and learning [14,15,74,75]. Of particular note, the 'mere-exposure' effect [76] involves repeatedly exposing an individual to a stimulus object to enhance the individuals attitude towards it. For example, controlled observational studies [77, 78] have found that animals prefer exploring unfamiliar environments when they are located nearby familiar ones. When animals were placed in completely foreign environments they are less likely to explore in order to discover novel occurrences. The animals were also less likely to explore when they were placed in well known environments. It was noted that the exploratory behaviour of animals peaked when environments were not too unfamiliar or familiar. Sreenivasan [79] considers peoples interest of novel genres and themes in movies. When a movie with a novel theme or concept was produced, there was an initial delay of interest. It was reasoned that initially the ideas presented in the movie were too foreign and strange. However, after repeated exposure to movies with slightly similar, but more familiar genres or themes, the interest increased, resulting in the subsequent acceptance of the new genres and themes.

Huang and Weng [66] were aware of this concept and proposed using a delayed interest response to novelty. Merrick [17] also utilises an interest based approach, correlating moderate novelty with high interest and high or low novelty with low interest. This delayed interest to novelty can prevent the agent from producing erratic behaviour, constantly switching to the newest affordance and ignoring other affordances. It allows the agent to focus on exploring and verifying specific affordances while in the appropriate state, potentially leading to more methodical and thorough exploration in a shorter timespan.

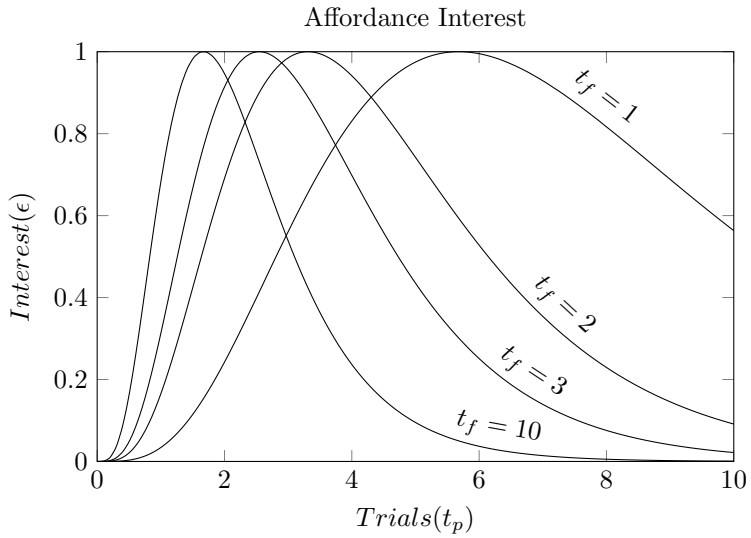


Figure 5.3: The initial interest curves of a new affordance with different levels of prior exploration (t_f) with the same face and state.

5.2.2 Generating interest based on novelty

To replicate the mere exposure effect and generate delayed interest based on the novelty of an affordance, Equation 5.1 is used. Unlike Merrick's approach [17], in this thesis, interest is defined in the domain [0,1]. Rather than using an aversion mechanic (negative interest) via generating a negative feedback to low or high novelty, a function that can be parametrised to change how the agent responds to novelty is used.

$$\epsilon = s_a \cdot \beta \cdot (1 - \beta)^4 \tag{5.1}$$

where ϵ is interest, and

$$\beta = e^{-M \cdot t_p} \tag{5.2}$$

$$M = 1 - e^{-s_b \cdot t_f} \tag{5.3}$$

Where $s_a = 5^5/4^4$ and is a scaling factor to keep the curve in the domain [0,1]. $s_b = 1/3$ and is a scaling factor which determines how quickly the agent 'warms up' to a new occurrence. t_p is the number of trials the affordance has undergone and t_f is the number of trials that are performed in the same state and face. s_a is derived by setting Equation 5.1 to $\epsilon = 1$, solving for the roots of the derivative and subsequently using a root to solve for s_a. s_b was chosen through experimentation.

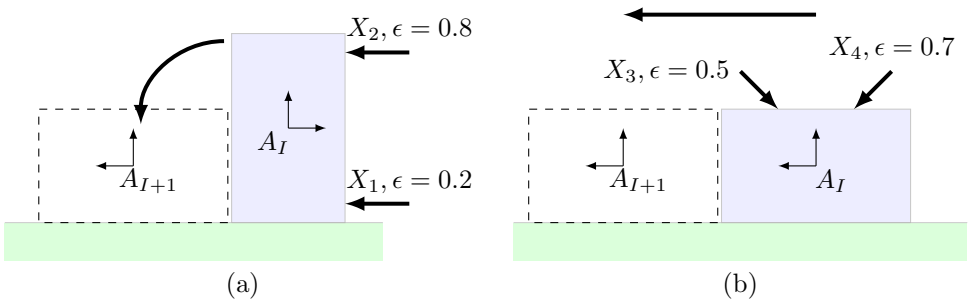


Figure 5.4: Move selection based on interest. The object is pushed over to $\boldsymbol{A}_{I+1}$ as X_2 was more interesting. The object no longer affords the same transformations, hence the agent must choose a new affordance to explore, despite others from X_2 being more interesting. In this case the agent would seek to perform the move shown in (b).

Figure 5.3 shows the interest curve of a new affordance over a range of different numbers of prior exploration trials (t_f) that have been conducted on the same face and initial state. Initially, when only a few trials are performed on a face in a specific state ($t_f < 3$), the interest is slow to increase for new affordances. As more trials are performed on the face in a specific state (increasing t_f), the interest for new affordances is higher. This ensures that initially undue focus is not maintained on the first affordance that is discovered on the face in a specific state. It potentially allows other affordances to be found so that the agent can explore over a range of affordances, rather than focusing on learning a single affordance at a time. As a result, this helps to prevent initial unnecessary repetition, while still allowing focus when required.

As the face and state become more familiar to the agent, subsequent affordances which are discovered for the same face and state, become interesting to the agent sooner. The agent therefore spends less time focusing on exploring affordances which have familiar initial states. Once sufficient trials have been performed to explore the affordance, focus is either brought to the next highest interest affordance, or random exploration is performed in many cases. Hence, for Equation 5.3, if s_b is too large, the interest for novel affordances increases too quickly, reducing exploratory behaviour. If it is too small, it delays focus, discouraging refining of potentially useful affordances. A correctly chosen s_b allows the agent to start focusing on prior discovered affordances and learn their characteristics when a useful number of affordances have been discovered.

It should be noted that due to the nature of manipulating objects, repeating operations may not be possible as the object is no longer in the required state (assuming the agent can not pick up the object). This has somewhat of an unintentional positive effect, which can help to prevent the agent from overly focusing on a single task. It also helps to speed up the rate at which new affordances are learned by

stopping the agent from constantly being exposed to the same state. Consider Figure 5.4. The object is pushed over and as a consequence, the transformation that was just performed cannot be repeated again in the new state of the object. A different transformation must be performed to return the object to its prior state so as to perform the operation again. By initially discouraging focus on novel affordances, the agent can potentially discover affordances which allow the agent to quickly return to the previous state. Further detail is covered in Section 5.4.1.

5.3 Motivated exploration

As discussed in section 5.1, the geometry of an object can affect how affordances are discovered. Causing the agent to be curious, that is interested in moderately novel occurrences, provides it with a means to focus on areas of the object that may be small and difficult to reproduce associated transformations. Hence, even though the object may not always be in the correct state, when the correct state is attained, the agent will focus on the affordance it is curious about, and by definition, interested in. This section discusses the approach to counteract the effect of the object's geometry when randomly exploring it. Interest is used to direct focus of the agent towards moderately novel occurrences.

5.3.1 Face exploration

To create a focused exploration behaviour, based on interest, it is first necessary to identify which face the agent should interact with and explore. To select the face to explore, the Boltzmann distribution is used with a similar approach to the work of Huang and Weng [66]. This involves determining the probability of selecting a face based on the interest of each face:

$$f_{sel}(u) = \frac{e^{\frac{-f_\epsilon^u}{1-\epsilon_{max}}}}{\sum_{i=0}^{n_p} e^{\frac{-f_\epsilon^i}{1-\epsilon_{max}}}} \tag{5.4}$$

Where $f_{sel}(u)$ is the probability of selecting face u, f_ϵ^i is the interest value of face i, in the current state, ϵ_{max} is the maximum interest of the object (the highest interest face) and f_ϵ^u is the interest value of face u.

To obtain the interest of a face (f_ϵ), the highest interest value, from affordances which utilise interactions with the specific face in its current state, is used. The maximum object interest (ϵ_{max}) is the value of the highest interest affordance of the object in its current state. The average was not used for both these variables as this can lead to problems with outliers. For example, consider a scenario where the average ϵ of affordances utilising the face is low, due to exhaustive exploration, but one of the affor-

dances becomes interesting. The interest of the face f_ϵ is marginally increased, causing the face to be rarely selected for exploration. In the case when there are no affordances that are valid for the selected face in its current state, a uniformly distributed random number is used to select a face.

The higher the interest of a face relative to other faces, the more likely it will be chosen. This means that regardless of the absolute magnitude, the most interesting face will have the highest probability of selection. For example, consider when the interest value of a specific face is small, $\epsilon = 0.1$, but for all others $\epsilon = 0$. The agent will likely select the face with the interest value of 0.1 as there is a much higher probability of being selected. This however, does not allow for random exploration to be introduced into the distribution unless all interest values are equal, which is likely only to occur when they are all 0. Hence, ϵ_{max} is used to control the level of randomness in the distribution. If ϵ_{max} is low, the general interest is also low. This leads the agent to tend towards focusing on random exploration of faces, rather than repeatedly exploring reasonably well explored faces. If the ϵ_{max} is high, the agent will tend towards exploring the most interesting face. This lets the agent explore, while bringing focus to sources of interest and reducing focus proportionally with interest when applicable.

5.3.2 Interaction point exploration

Once a face is selected, a specific area on the face must also be selected for exploration. The Boltzmann distribution can be utilised again, this time using the interest of affordances that are valid on the selected face:

$$P_{sel}(w) = \frac{e^{\frac{-P_\epsilon^w}{1-f_\epsilon}}}{\sum_{i=0}^{n_p} e^{\frac{-P_\epsilon^i}{1-f_\epsilon}}} \tag{5.5}$$

Where $P_{sel}(w)$ is the probability of selecting affordance w of n_p affordances, P_ϵ^w is the interest of affordance w, P_ϵ^i is the interest of affordance i and f_ϵ is the interest of the selected face.

In this thesis, affordances identify a single discrete point (the interaction point X_μ), and a valid range about the point on the face (X_σ), which the agent can explore. Similarly to the distribution of faces in Equation 5.4, Equation 5.5 produces a higher probability of selection of high interest affordances, over lower interest affordances. While this causes the agent to focus on specific affordances for exploration, there may be a range of unexplored areas on the face outside the range which any affordance describes, as shown in Figure 5.5. As the probability function, from Equation 5.5, only produces a distribution for a discrete selection of affordances and only factors affordances which have been discovered, this immediately restricts exploration to already discovered affordances. Hence, discovering new affordances

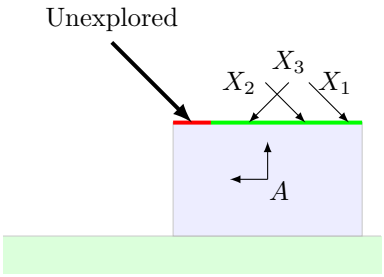


Figure 5.5: Demonstrates the problem with unexplored areas when using the discrete point approach. The unexplored area will never be reached as there are no interaction points to choose from to explore.

becomes problematic, as the Boltzmann distribution is only defined for a known number of discrete points. If there are an unknown number of points such that the distribution becomes continuous, as is the case, Equation 5.5 can not be used.

In order to produce a continuous distribution for exploration of interaction points, a different method is proposed. This involves calculating the normal distribution to create a continuous Probability Density Function (PDF) for each affordance that is relevant to the face and state of the object. The distribution is based on the interaction points $\boldsymbol{X}$, which produce the transformation the affordance describes. To calculate the probability for a single affordance, the normal distribution is used:

$$P_{pdf}(u,i) = \frac{1}{\sqrt{2 \cdot \pi \cdot (X^i_\sigma)^2}} e^{\frac{-(u-X^i_\mu)^2}{2 \cdot (X^i_\sigma)^2}} \tag{5.6}$$

Where $P_{pdf}(u,i)$ is the PDF of selecting a point at u which produces affordance i on the selected face, X^i_σ and X^i_μ are the variance and mean of the list of contact points of affordance i respectively. If only a single point is currently explored in X^i, $(X^i_\sigma)^2$ is set to 0.02.

In order to use the distribution to obtain a point, a Uniformly Distributed Random Number (UDRN) is generated and the Cumulative Distribution Function (CDF) is used:

$$P_{cdf}(u,i) = 0.5 \cdot (1 + erf(\frac{(u - X^i_\mu)}{\sqrt{2 \cdot (X^i_\sigma)^2}})) \tag{5.7}$$

Where $P_{cdf}(u,i)$ is the CDF of selecting a point at u which produces affordance i on the selected face.

Equation 5.6 only represents the exploration PDF for a single affordance. To create a continuous distribution for every affordance across the whole face, the average could be used as shown:

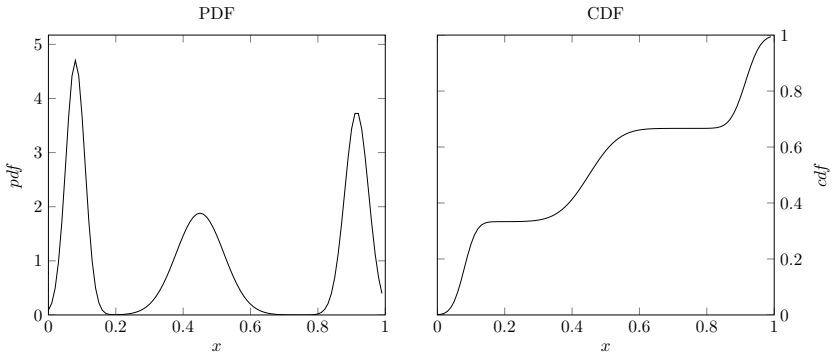


Figure 5.6: The PDF in (a) and the CDF in (b) of a possible curve that can be produced using the average based on Equations 5.8 and 5.9.

$$f_{pdf}(u) = \frac{1}{n_p} \cdot \sum_{i=0}^{n_p} P_{pdf}(u, i) \tag{5.8}$$

where $f_{pdf}(u)$ produces the PDF for exploring a point u on the face and n_p is the number of valid affordances on the face.

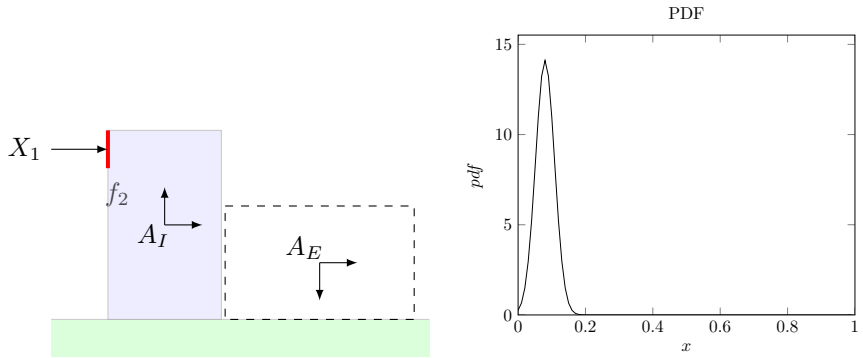
$$f_{cdf}(u) = \frac{1}{n_p} \cdot \sum_{i=0}^{n_p} P_{cdf}(u, i) \tag{5.9}$$

where $f_{cdf}(u)$ produces the CDF of selecting a point u on the face.

This can produce a curve as shown in Figure 5.6, where, given 3 affordances that are valid for the face, each has an equal probability to be explored. While this allows the agent to identify and focus on areas of interest for exploration, it does not enable the agent to explore foreign areas on the face. It further provides no mechanism to cause focus to be reduced after thorough exploration. Focus on areas of high density of affordances is maintained at the expense of areas on a face with low density of affordances. This can lead to unbalanced and ineffective exploration which is likely to be less effective than random exploration. It is therefore desired to have a distribution which both covers the whole range of the face and dynamically adjusts to the changing interest of areas the agent has explored.

The interest from each affordance could be used to scale its distribution to represent the level of exploration:

$$f_{pdf}(u) = \sum_{i=0}^{n_p} P_{\epsilon}^{i} \cdot P_{pdf}(u, i) \tag{5.10}$$



(a) (b)

Figure 5.7: Generating interest from each affordances own PDF and interest. A new affordance is discovered on face f_2 in (a), with the corresponding distribution for f_2 shown in (b). The agent will solely focus on the newly discovered area until it reaches 0 interest.

Where f_{pdf} is the total PDF of selecting an interaction point on the face and P_ϵ^i is the interest of affordance i.

This enables the agent to focus on the most interesting areas at any specific time. However, in the example in Figure 5.7a, after the first affordance is discovered, it is highly unlikely that other affordances would be discovered until the first has been thoroughly explored. This is due to the distribution being limited to the region of the only discovered affordance as shown in Figure 5.7b. In this case, the agent would continue exploring the area defined by the affordance until it becomes uninteresting. Other areas with affordances would then become more interesting, thereby more likely to be explored. However, in Equation 5.10 the distribution is no longer valid as the integral over the domain [0,1] does not always equal 1. As such, it does not satisfy Equation 5.11:

$$\int_0^1 f_{pdf}(u) \cdot du = 1 \tag{5.11}$$

In order to satisfy Equation 5.11, the Boltzmann distribution from Equation 5.5, can be used. This scales the distribution of each affordance based on their probability of selection, which is representative of their level of interest, hence level of exploration:

$$f_{pdf}(u) = \sum_{i=0}^{n_p} P_{sel}(i) \cdot P_{pdf}(u, i) \tag{5.12}$$

$$f_{cdf}(u) = \sum_{i=0}^{n_p} P_{sel}(i) \cdot P_{cdf}(u, i) \tag{5.13}$$

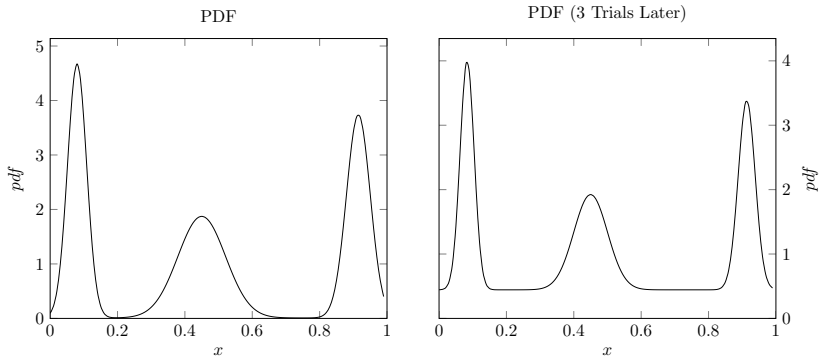


Figure 5.8: Adjusting PDF based on maximum face interest. The PDF in (a) of a possible curve that can be produced from Equations 5.14 and 5.15. In (a), the curve consists of 3 affordances with a single trial. In (b), another trial has been added to each affordance.

Where f_{cdf} is the total CDF of selecting an interaction point on the face.

However, this approach still does not facilitate exploration of unexplored or low interest areas on the face. Ideally the agent should explore the regions of affordances for only a few trials before seeking alternate areas on the face to explore. Therefore, to enable unexplored and low interest areas to be factored into the distribution, Equations 5.12 and 5.13 are changed to the following:

$$f_{pdf}(u) = (1 - f_\epsilon) + f_\epsilon \cdot \sum_{i=0}^{n_p} P_{sel}(i) \cdot P_{pdf}(u, i) \tag{5.14}$$

$$f_{cdf}(u) = (1 - f_\epsilon) \cdot u + f_\epsilon \cdot \sum_{i=0}^{n_p} P_{sel}(i) \cdot P_{cdf}(u, i) \tag{5.15}$$

Where f_ϵ is the interest of the face.

Under the same conditions as Figure 5.6, Figure 5.8a shows how the probability is represented. This has not changed much, however by using Equation 5.14, the probability of choosing a point is affected by the interest of each affordance. The higher the interest, the more likely the point will be chosen. As specific affordances are explored and become less novel, and as a consequence less interesting, the interest in unexplored and areas previously of low interest, increases. Hence, if the interest level of the face is low, the agent will tend to lose focus and choose points more randomly. Conversely, if the interest level of the face is high, the agent will gain focus, choosing points in areas that are more interesting. When an area becomes the focus of the agent, due to its interest level, the agent will maintain focus until its interest lowers to the extent that another area is more interesting. The result is that after a few more

points have been explored the PDF will change to Figure 5.8b.

While exploring, discovering new affordances and repeating known ones, the face interest distribution will fluctuate up and down. Eventually, the agent will no longer discover new affordances. When all affordances are sufficiently explored, such that there is no interest in any specific one, the f_{pdf} of the face becomes uniformly distributed ($f_{pdf}(u) = 1$). Hence, the face is equally explorable by the agent. At this stage, it may be a good time to stop exploration. However, it is impossible to know if a new affordance is yet to be discovered. Therefore, exploration should be continued until the agent is given a task or requested to stop.

Using the probability distribution in such a manner drives the agent towards a behaviour, but does not specifically dictate exactly what to do. This allows for varied behaviour based on the current circumstances. By including an element of randomness, which inherently helps the discovery of unexplored areas, the likelihood of the agent following a specific process that can lead to repetitions or getting stuck between two states is significantly reduced.

5.3.3 Boredom

In some cases, as discussed in Section 5.2.1, the agent may not be able to reproduce the desired transformation an affordance describes. This can potentially cause the agent to get stuck, with repeated attempts to interact at the specific point on the object face. To prevent the agent from getting stuck, when the same affordance is repeated, ϵ_{max} is scaled based on the following Equations:

$$\epsilon_{max} = \epsilon_{max}^{raw} \cdot k_a \tag{5.16}$$

where ϵ_{max}^{raw} is the unscaled max interest of the object and k_a is a scaling factor determined by:

$$k_a = e^{\frac{-(1+n_a)^2}{2 \cdot \pi^2}} \tag{5.17}$$

where n_a is the current number of repetitions that have occurred of an affordance.

Each time the same affordance is repeated, regardless of its actual interest, the interest of all affordances and therefore faces temporarily reduces due to the change of ϵ_{max} in Equations 5.4 and 5.14. As the interest reduces, the agent increasingly generates more random interaction points. This can be likened to the agent getting bored.

Rather than immediately returning k_a to 1 when a different affordance is produced, it is slowly adjusted by taking the square root of k_a (such that $k_a = \sqrt{k_a}$) to approach 1 each time a different affordance is produced and no repetition occurs. The agent is therefore more likely to produce more

random interactions for a limited time after interactions which produce the same affordance. This can help to prevent the agent from returning back to the same state after a different affordance is finally produced.

5.3.4 Force exploration

In order to improve learning time, the force direction can be influenced by the interest of the agent. Similarly to the selection of interaction points, the probability of selecting a force direction is influenced by the probability of selecting an affordance. In order to factor the position of the force, the CDF of the affordance is used to determine the probability of the affordance being interacted with at the selected point:

$$F_{pdf}(w) = \frac{e^{\frac{P_{sel}(w) \cdot P_{cdf}(u,w)}{1-f_\epsilon}}}{\sum_{j=0}^{n_p} e^{\frac{P_{sel}(j) \cdot P_{cdf}(u,j)}{1-f_\epsilon}}} \tag{5.18}$$

where $F_{pdf}(w)$ is the PDF of selecting a force from affordance w on the face. u is the prior selected interaction point.

This enables the agent to select a force that is likely to produce the most interesting result when the agent is focused. The basic exploration strategy remains the same as discussed in Chapter 3. When a force produces no motion, alternate sheer forces are attempted. If no motion is produced, another point is chosen. The approach taken here reduces the need to always start with a normal force when a different force is desired. A continuous distribution can be used in order to provide a wider range of force orientations to explore. However, for simplicity a discrete approach is used, which limits forces to ranges of known affordances, specifically normal and sheer.

5.4 Further Methods of Generating Interest for Exploration

Basing exploration on interest, allows other factors than that of novelty of affordances themselves, to influence how to explore an object for its affordances. As discussed above, there are many ways in which an agent may find manipulation contexts interesting. In this section, implementation of transitory goals are detailed. This builds upon the basic interest model to provide alternate sources of interest to motivate the agent and potentially improve learning after a few fundamental affordances are learned.

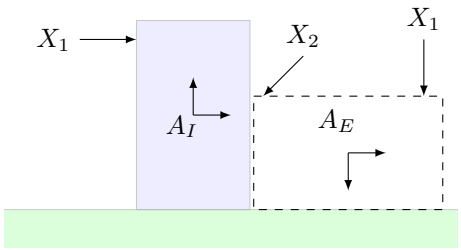


Figure 5.9: Using transitory goals to return to previous state. The interaction at X_1 from an affordance in frame $\boldsymbol{A}_I$ pushes the object over to frame $\boldsymbol{A}_E$. The same interaction (X_1) is no longer possible. The interaction at X_2 transforms the object back to frame $\boldsymbol{A}_I$, allowing the affordance to be trialled again X_1.

5.4.1 Transitory goals

As previously discussed, the availability of affordances to explore at any given time depends on the current state of the object. Consider Figure 5.9, where the state of $\boldsymbol{A}_I$ has a highly interesting affordance $\boldsymbol{P}_I$, with an interaction point X_1. When the agent performs a transformation, causing the object to be transformed to frame $\boldsymbol{A}_E$, the state required for the execution of $\boldsymbol{P}_I$ is no longer possible. This can lead to circumstances where the agent has thoroughly explored nearly all the object's faces, such that current affordances are no longer novel and consequently uninteresting. The exploration behaviour then becomes solely that of random exploration. If a new affordance, $\boldsymbol{P}_{new}$, is then discovered such that it becomes the focus of the agent, it can be some time before the object returns to the necessary state to perform $\boldsymbol{P}_{new}$. This can also significantly increase the time required to perform sufficient trials to identify a good X_μ for the affordance.

To minimise this effect and improve the agents ability to focus on interesting areas, transitory goals can be generated based on desired states. This allows the agent to expand it's exploration motivation outside of its immediate circumstances by considering the state that each affordance will produce. These transitory goals are set based on determining if any affordance $\boldsymbol{P}^i$ is valid in the current state, and can change the object state to match the initial conditions of another affordance $\boldsymbol{P}^{des}$. If this is possible, Equation 5.19 is used to modify the interest of $\boldsymbol{P}^i$, based on the interest of $\boldsymbol{P}^{des}$ and $\boldsymbol{P}^i$ itself:

$$P^i_\epsilon = P^i_\epsilon + s_m \cdot ((1 - P^i_\epsilon) \cdot P^{des}_\epsilon) \tag{5.19}$$

where P^i_ϵ is the interest of affordance $\boldsymbol{P}^i$, P^{des}_ϵ is the interest of the desired affordance and s_m is a scaling factor where typically $s_m = 0.25$.

This ensures that the interest is restricted to the domain [0,1], while also providing a proportional increase in interest based on P^{des}_ϵ. If P^{des}_ϵ is low or 0, then very little or no change is made to P^i_ϵ.

To determine if the initial conditions of $\boldsymbol{P}^{des}$ are met, the agent's planner, as described in chapters 3 & 4, is modified. The goal is set to match the initial conditions of $\boldsymbol{P}^{des}$, which are $\boldsymbol{I}^{des}$. When using the modified planner, the position of the object is not considered when evaluating the transformation. Rather, a single move is performed and evaluated to determine if the goal state is attained. This process is conducted after updating the interest of each affordance. Therefore, when calculating $f_{pdf}(u)$, which has an affordance that can change the state of the object to match $\boldsymbol{I}^{des}$, the agent will tend to choose the area on the face which produces the result the affordance describes.

5.5 Summary

In this chapter the difficulty of exploring an object when considering its geometry was highlighted. Some areas of the object are harder to explore than others, leading to unbalanced and potentially incomplete exploration. Intrinsic motivation in the form of interest is therefore used to improve upon random exploration and is denoted by ϵ. It is defined as a value over the domain [0,1] where 0 is completely uninteresting and 1 is extremely interesting. Three methods of generating interest for exploration and learning were discussed:

- Object movement
- Patterns and Goals
- Novelty

Object movement was found to be an unsuitable means to generate interest to motivate the agent to explore. While goals and sequences were identified as being useful for further motivation, in order to promote exploration of individual affordances, novelty was found to be a better fundamental approach.

Novelty was defined as the newness of an affordance. As such, affordances which have been thoroughly explored, in that they have been observed by the agent numerous times, are not considered novel. When generating interest based on novelty, a delayed response (Equation 5.1) can produce a more methodical and stable exploration behaviour.

In order to use interest to motivate the agent, the Boltzmann distribution is used (see Equation 5.4) to first select a face. To select the interaction point on a face, a PDF function is produced which is based on each affordance that is valid in the current state (see Equations 5.5, 5.6 and 5.14). The CDF (Equation 5.15) is then used with a uniformly distributed random number to select a point on the face. When there are no longer interesting areas to explore, the agent randomly explores in an unfocused manner with the aim to 'stumble across' something interesting.

Transitory Goals were used as a means to motivate the agent towards performing moves that would produce a more interesting result on the *next* move. The agent searches for affordances which return the object to a state with high interest and increases those affordance's interest based on Equation 5.19. This helps the agent return to difficult to reach object states in order to verify affordances produced in those states.

Chapter 6

Results

This chapter presents the results from experimental trials using both the simulation system and robotic system. Section 6.1 presents and discusses the robotic system's results, Section 6.2 presents and discusses the simulation system's results and Section 6.3 summarises the chapter.

6.1 Robotic system

This section describes each component of the physical robotic system implemented to embody the agent and enact its output. The following sections are included; the environment and object in Section 6.1.1, the robotic manipulator in Secton 6.1.2, the vision system in Section 6.1.3, the tactile sensor in Section 6.1.4, the control methodology in Section 6.1.5, the system setup in 6.1.6, the experimentation method is detailed in 6.1.7, the results are presented in 6.1.8 and finally the results are discussed in 6.1.9.

6.1.1 Environment and Object

As discussed in Section 2.1, the environment is designed such that an object is restricted to 3DOF, on a 2D plane. To facilitate this in the real physical 3D world, an environment akin to that shown in Figure 6.1 is used. The object's shape is extruded along the z-axis. Coupled with the ground and wall constraints, this effectively restricts motion to 3DOF (as discussed in Chapter 4). The orthogonal side projection is effectively created by using a camera situated as shown.

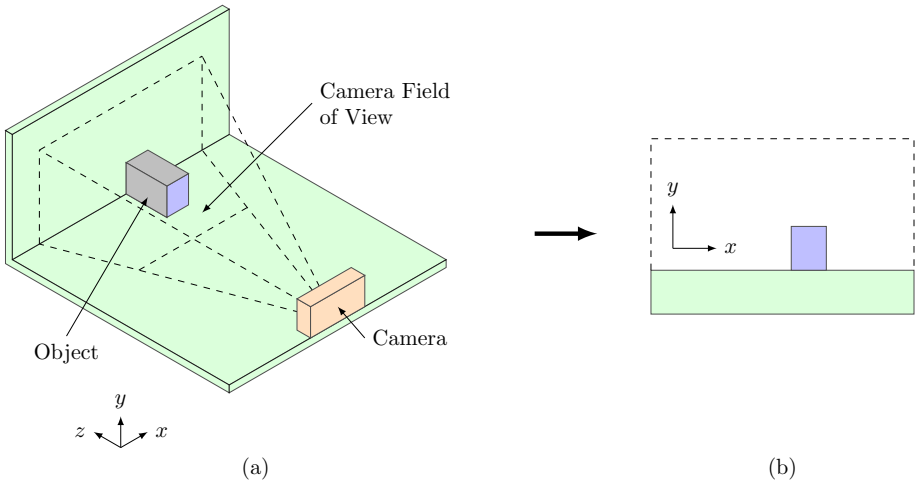


Figure 6.1: Environment used for robotic system. The camera in (a) looks directly at the extruded object, producing the image in (b).

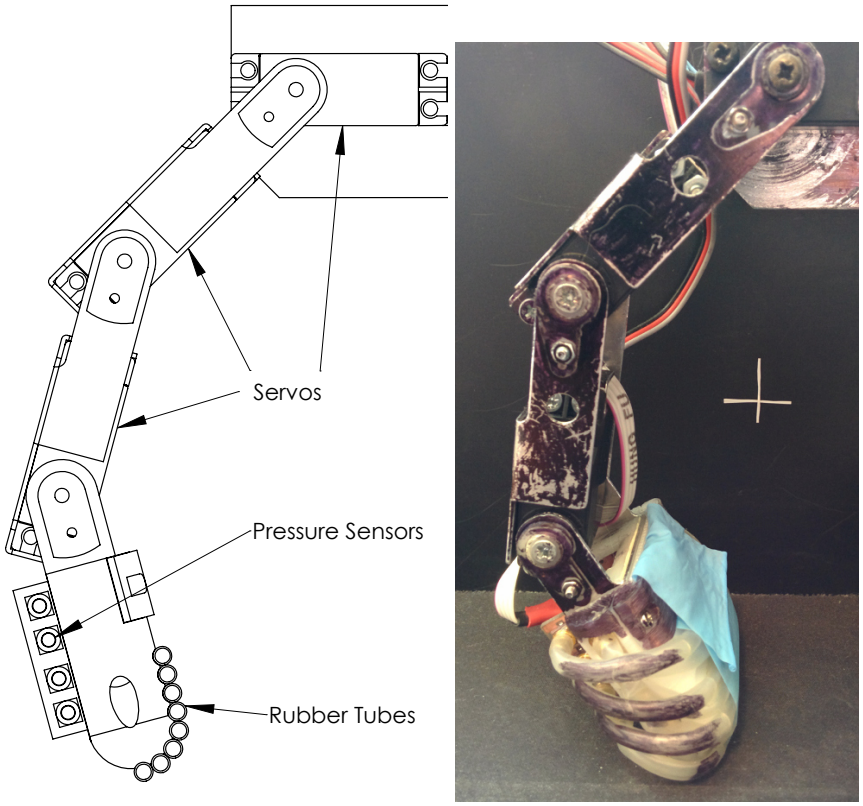


Figure 6.2: Manipulator schematic with tactile sensor attached. The manipulator is akin to a human finger and is attached to a fixed base.

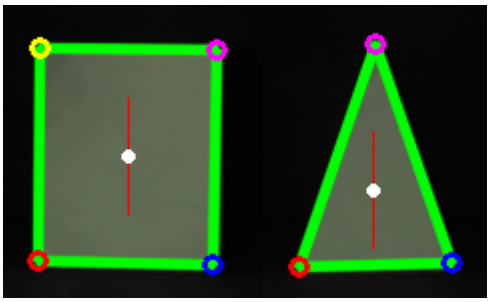


Figure 6.3: Detecting and tracking an object's centre of mass, orientation and faces. The polygonal approximation is obtained (shown by a green outline) to identify face vertices (coloured circles).

6.1.2 Robotic Manipulator

Interacting with objects is an important capability for robotic agents. It enables them to influence and learn about their surroundings. How the robot achieves this can be quite variable and is typically dependent on the specific task at hand. Manipulation, though a specific task, can be very generalised and is therefore equally variable, requiring a large variety of capabilities. As the required capabilities in this thesis have been reduced down to simple point contact forces of either normal or sheer nature, and the environment is limited to 2 dimensions, a planar 3R serial linkage manipulator akin to a human finger was chosen (see Figure 6.2).

This provides a good balance between complexity and workspace capability. Hence, in a large portion of the workspace, the manipulator is capable of providing a range of different fingertip orientations. This allows the system to adjust the orientation of the fingertip in order to apply a force to an object's surface in the desired direction. When the system is working at the extremities of the manipulator's workspace, the fingertip may not be able to attain the desired orientation or apply the desired force. Therefore, the majority of operations are endeavoured to be conducted sufficiently within the workspace where the extremities are not an issue, however, this is often not possible.

The base of the manipulator is fixed to a vertical wall and situated approximately 14cm above the ground. The first two links of the manipulator are 50mm in length between joints, while the last has a joint to fingertip length of 60mm. A tactile sensor is attached at the fingertip to provide force feedback (see Section 6.1.4).

6.1.3 Vision System

To identify the object shape, as well as track the position and orientation, images from a camera situated face-on to the object are processed using the OpenCV library, as shown in Figure 6.3. The largest

detected contour is used to determine the moments of the object. The moments are then used to obtain the centre of mass for object position and the axis of minimum moment of inertia for the orientation. The vision coordinates are then mapped onto the world coordinates via calibration as described in Appendix A. The camera was operating at a resolution of 640x480. Appendix A describes the method for tracking the object.

Matching Objects

To allow the agent to deal with different object shapes, it is necessary to classify them in some manner. There is much work on classifying objects based on appearance, mass distribution, texture etc. [80]. It might be advantageous to classify the type of shape, such as round, rectangular, triangular etc. as it would provide insight into the type of transformations that could potentially be performed. However, in the implementation in this thesis, simply classifying the shape is not indicative of how the object may be transformed. If the object has not been encountered before and has no learned affordances, classification provides no information on how the object might behave. The agent is only required to differentiate between shapes it has previously encountered in order to identify which object's affordances it should use. Hence, the Hu invariants [81] can be used to obtain a comparison between two contours describing the shapes (see Appendix A). The agent can then determine if the shape presented is a match to one previously encountered, or if no match, then a new shape.

6.1.4 Tactile Sensor

In some cases, visual feedback may be sufficient to interact with an object and perform a manipulation. This is supported by work on sensor-less manipulation [36, 82], where properties of the environment can be used as constant known control points to move the object to desired states. However, this does not imply physical feedback of some form (such as tactile sensing) is unnecessary for manipulation. As Russell discusses [83], there are many circumstances where tactile sensors are of major benefit and even a necessity for manipulations. Systems which implement tactile sensors can generally perform significantly more complex manipulations, or are simply more reliable [38, 84, 85]. For example, for controlled cyclic manipulation, or rolling [35, 86] it is necessary that systems be capable of gauging how much force is being applied, and therefore how to adjust their contact forces. Without any form of feedback other than visual, it becomes increasingly difficult to perform mediated tasks, particularly when the view of the object is obstructed by the manipulator.

Pushing does not necessarily require tactile sensing to be successfully performed [25, 26, 87, 88]. However, as this thesis also includes sliding and dragging interactions which require force magnitude and direction feedback, tactile sensing becomes critical. A tactile sensor was therefore designed for a 2D

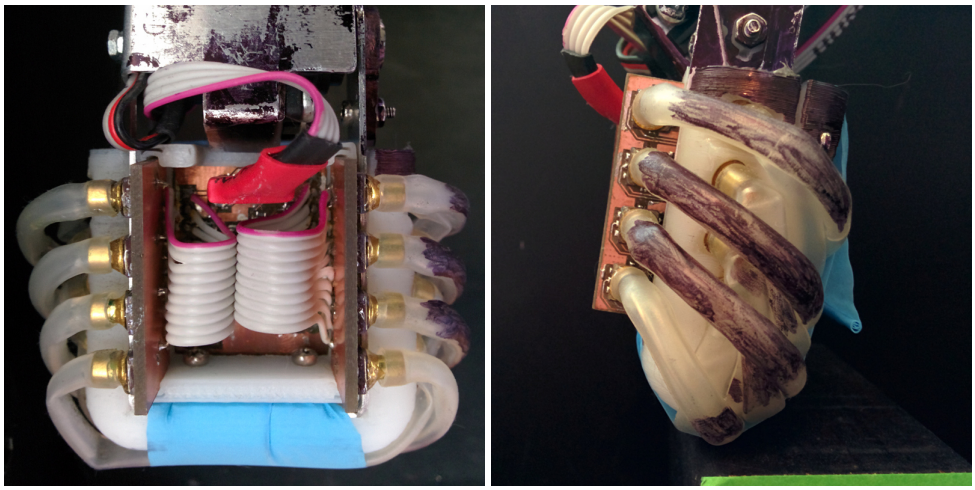


Figure 6.4: The tactile sensor. Left: Top view of tactile sensor, Right: Side view of tactile sensor. The rubber skin is blue and drawn taught to ensure the silicon tubes remain in place.

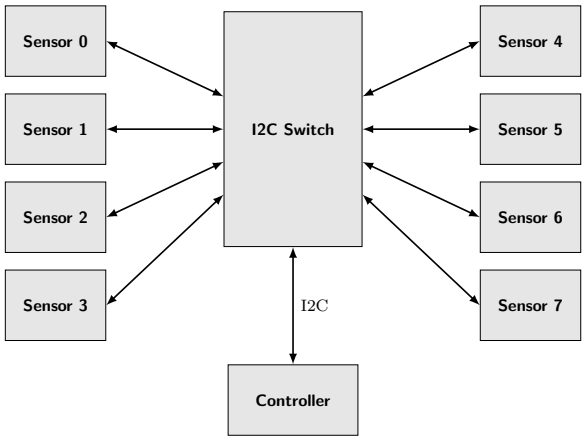


Figure 6.5: Tactile sensor schematic. Each sensor can be individually addressed from a single I2C interface from the controller using the 8-way I2C switch.

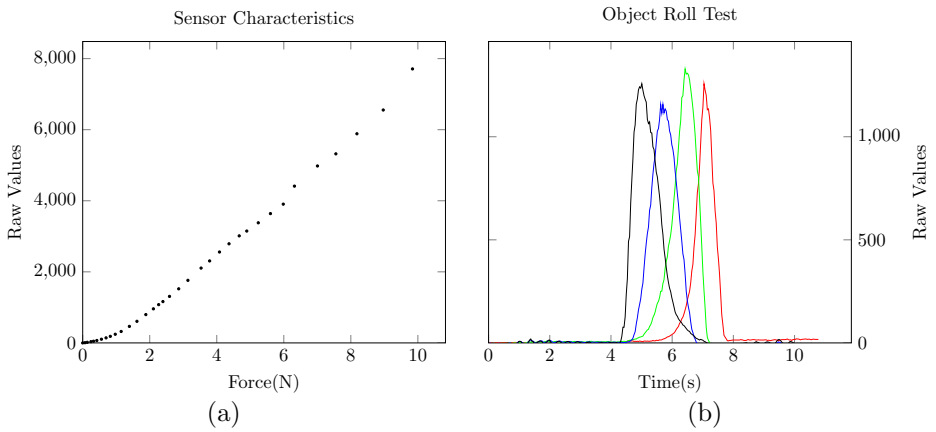


Figure 6.6: Tactile sensor force characteristics and response. (a) Tactile Sensor Response. Testing was conducted using the entire working surface of a single tactel. (b) Results of 4 sensors from a roll test, where an object with a flat surface is rolled across the tactile sensor surface.

environment (see Figures 6.2 and 6.4). The sensor consists of 8 silicon rubber tubes wrapped around a 3D printed frame and is similar in design to the work of Chun and Wise [89]. The rubber tubes have an outer diameter of 4mm, with a wall thickness of 0.4mm. A thin rubber skin is used to distribute force amongst adjacent tubes as well as prevent objects from catching between them. Each tube is sealed and connected to a pressure sensor, which is in turn connected to an 8-way I2C switch as shown in Figure 6.5. For this sensor, the use of silicone rubber tubes provides compliance. As Russell states [90], "sensor compliance accommodates a degree of inaccuracy in robot positioning, aids stable grasping and resists damage". For application in this thesis, compliance allows the fingertip to perform small manoeuvres, supplying the correct force at any given time, with reduced risk of losing traction.

Figure 6.6a shows the sensor response of a single tactel, which is capable of detecting forces down to 0.1N. However, as each rubber tube has a different length containing different volumes of air, the measured pressure values differ for each sensor, some quite significantly. Therefore, the sensors are calibrated by rolling an object across the working surface several times (see Figure 6.6b). The assumption is made that the force applied to the sensor is approximately the same over each object roll. The average is obtained from multiple tests and the resulting data is used to scale each tactel so that its output is approximately the same when the same force is applied. A unit vector normal to each tactel's surface is used to create a force vector. Each force vector is summed to obtain the resultant force acting on the sensor.

$$force_{res} = \sum_{i=0..8} F^{i}_{sen} \hat{\boldsymbol{U}}^{i}_{sen} \tag{6.1}$$

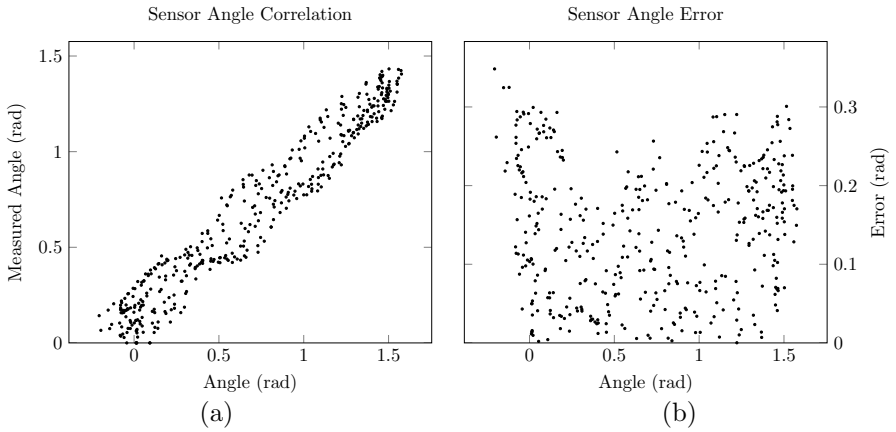


Figure 6.7: Tactile sensor correlation and error. (a) Tactile sensor force direction correlation, which shows how closely measured force direction matches the actual force direction. (b) Sensor error, indicating the range of error.

where F^i_{sen} is the magnitude of the tactel i and $\hat{\boldsymbol{U}}^i_{sen}$ is the unit vector describing the normal force direction of tactel i.

The correlation of the resultant force direction and the actual force direction is shown in Figure 6.7. This was measured by applying forces at a range of known directions by means of rolling an object face of known orientation across the working surface of the tactile sensor. The average error is 0.143 radians, the minimum is $2 \cdot 10^{-4}$ radians, and the maximum is 0.348 radians.

It can be noted that over the initial 0 to 1N force range (see Figure 6.6a) the output is non-linear. It is difficult to distinguish the exact cause of this and how much each affects the sensor reading as a number of factors are involved. The first, is the compressible nature of air, which, as predicted by the ideal gas law, indicates a non-linear relationship between volume and pressure. The second factor, is the characteristics of the silicone tubing, which does not follow HookâĂŹs law when compressed. The third factor is the curvature of the silicone tube, where initially there is a small contact area and consequently a small change in pressure. As the tube further deforms due to higher forces, the contact area increases, subsequently increasing the ratio of tube deflection and volume. Other minor factors which may affect readings include the outside air temperature and reference or outside air pressure. Fortunately, these generally change slowly and can be compensated for by zeroing the sensor when it is known there is no contact force.

In testing, hysteresis was negligible as a result of a brief deformation of the tactile tube. When the sensor was subjected to prolonged or significant deformation, creep became an issue, lasting up to several minutes before the sensor's readings returned to approximately 0. It was noted in early experimentation that creep is related to the wall thickness and diameter of the silicone tube. A thicker wall and/or smaller

tube diameter will shorten the creep effect, but sacrifice sensitivity, while a thinner wall and/or larger tube diameter will lengthen the time where creep is in effect, but gain sensitivity. The tube diameter and wall thickness were selected based on testing available tube samples. The characteristics which best fit the design size of the fingertip and provided high sensitivity were prioritised for selection.

6.1.5 Control

As discussed in Section 2.2.2, a point contact approach is used to interact with objects. This not only simplifies the problem, but also allows different manipulator configurations to be used. However, the fingertip must be capable of point contact and able to apply the range of necessary forces to maintain a normal force and apply sheer forces.

In this thesis, a 3R manipulator was chosen due to its versatility. Inverse Kinematics is used to obtain the correct joint orientations based on a fingertip x-y coordinate. However, the versatility of the 3R manipulator comes with complexity in the form of ambiguity, where there may be multiple solutions to a desired fingertip position. This is solved by using an optimisation algorithm (see Appendix A).

There are 2 control stages used when interacting with an object:

- Position based control
- Force based control

Each is discussed in the following sections.

Position Control

To prevent the manipulator from moving too quickly, potentially damaging its servos, a velocity limit is placed on the system. The speed of each joint is set so that they arrive at the desired orientation simultaneously. This serves to smooth the motion of the manipulator from noise as well as filter sudden large changes to the desired position (see Appendix A). Upon initially identifying an interaction point, the agent moves the fingertip towards the object surface until contact is made. If the object position changes while the fingertip is approaching, the agent compensates by adjusting the interaction point's position.

Force Control

Once contact is made with the object, the system switches to a force control method, where a consistent force is endeavoured to be maintained normal to the desired contact surface of the object. The visual

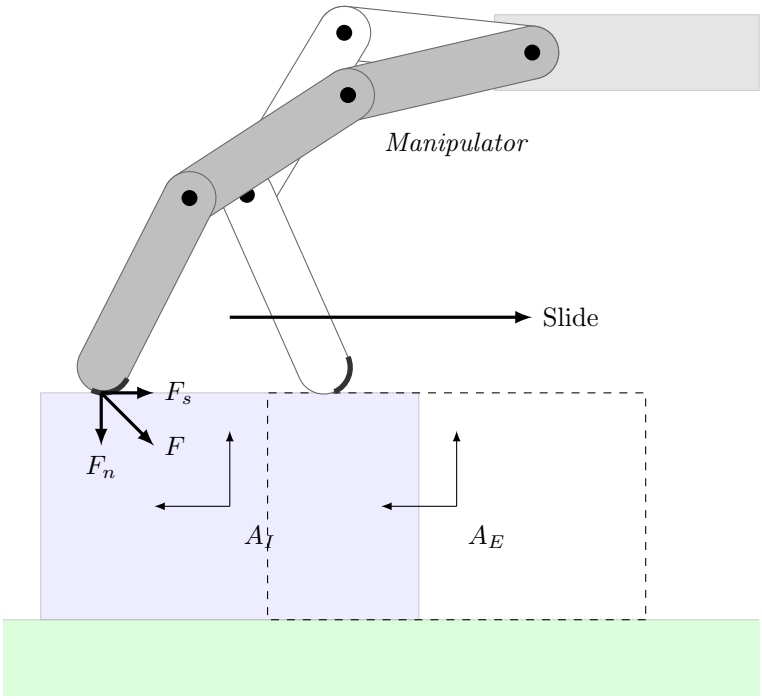


Figure 6.8: Implementing dragging or sliding interaction by applying a force comprising of a normal force to maintain grip F_n and a sheer force F_s to drag or slide the object.

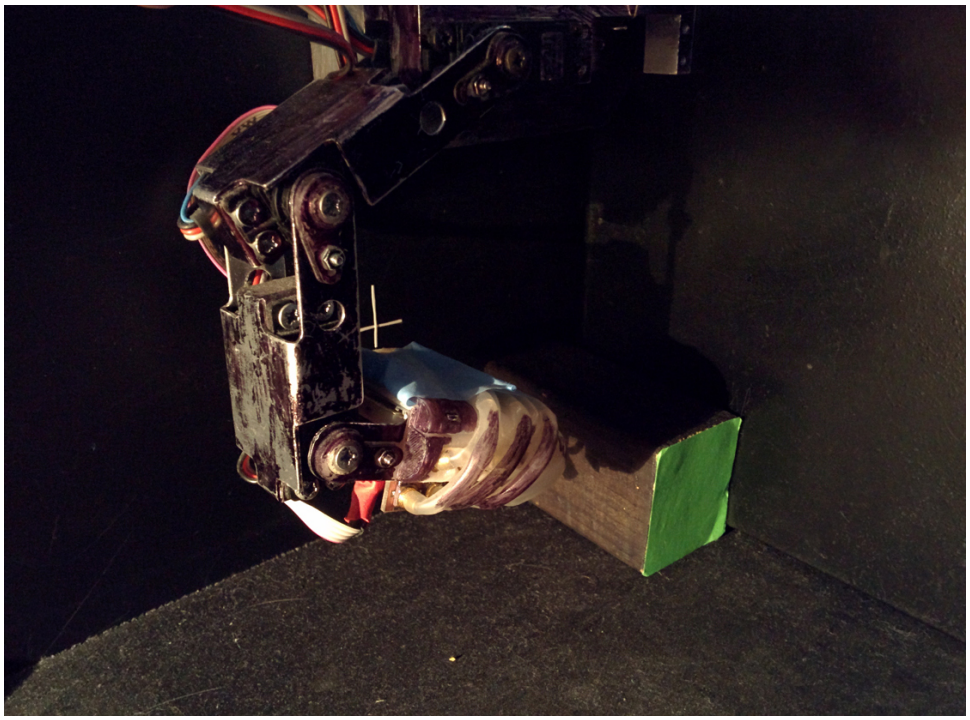


Figure 6.9: Environment and setup of the robotic system.

system is used to determine the orientation of the object surface and the force is obtained from the tactile sensor. The force normal to the object face is used for feedback to a PID controller to maintain sufficient traction to prevent slip when performing sheer operations. The controller therefore moves the fingertip based on the orientation of the object's surface and the force from the tactile sensor. When sheer force is required, the fingertip continues to maintain a force normal to the object surface, moving the fingertip across the surface as shown in Figure 6.8. For simplicity, the required normal force to maintain is manually set by the user prior to operation. This normal force remains constant over the entire range of trials.

6.1.6 Setup

Due to the limitations of the manipulator's reach, only a single environment was used. It consisted of the ground with a single wall on the right side of the environment, as shown in Figure 6.9. Furthermore, due to the fixed nature of the manipulator and subsequent limit of its reach, the object's position was often reset and moved back to a start location. Where possible, the orientation of the object was kept the same. If the object became stuck and the manipulator was unable to move it after several attempts, it was also reset. Experimentation was performed on both a rectangle and an isosceles triangle.

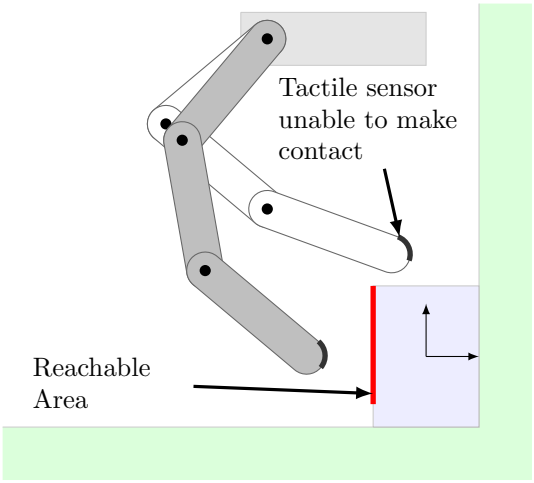


Figure 6.10: The limit of reach of the robotic manipulator generally restricts interaction to a single face.

6.1.7 Method

Experimentation was broken down into two phases; an exploration phase and a planning phase (see Section 2.2.3). The exploration phase involved placing an object in the environment and allowing the agent to interact with the object in various ways. After 50 successful interaction trials were performed, exploration was stopped and the learned affordances saved. In order to gauge the performance of the motivated exploration algorithm proposed in this thesis, a random exploration set of trials was also gathered using the same approach. This involved the agent randomly choosing a reachable face and interaction point for each trial. Each interaction trial normally takes between 10 to 15 seconds and 5 seconds to reposition the finger for the next trial. If an unexpected outcome occurs, the system waits for the user to reset it.

The planning phase involved providing a series of tasks to the agent by showing it the the desired goal position and orientation of an object. The object was then placed in an initial position and the agent generated and displayed a plan. The agent then attempted to perform the plan, as described in Section 2.2.3, until the goal was met or the agent could find no valid affordance to move the object closer to the goal. For all plans generated, the maximum move search depth was 6 and the maximum number of moves to generate from each affordance was 5 ($q = 5$, see Section 3.4.2).

Due to the limited reach of the robotic system's manipulator, the agent could only supply a limited range of pushing or dragging forces on the object's face, reducing the number of possible affordances. Furthermore, the limited reach generally restricted interaction to a single face. For example, in Figure 6.10, the manipulator can only interact with a single face as it cannot reach other faces at an appropriate orientation. This also limited the range of forces applicable near the limits of the manipulator's reach.

Transitory goals were not tested as the agent does not factor the manipulators ability to physically reach an interaction point in specific desired states while filtering initial states of affordances. This limitation can be addressed in future work.

6.1.8 Results

In order to depict the learned affordances (see Figure 6.11), they are reconstructed in an environment derived from their initial and final conditions and states. The transformation is displayed with a constant magnitude for affordances classified as translations. If an affordance is classified as a rotation, the learned magnitude of the affordance is used. For each affordance, two object states are shown. The final state is identified by the object shaded red. The initial state is identified by the object state shaded blue and also has an arrow on a face. The arrow indicates the position of the interaction point and direction of force on the object face required to produce the final state. A red line is also drawn on the object face which indicates the estimated margin of error where the agent can interact to acceptably reproduce the affordance. This is determined by one standard deviation from either side of the interaction point. In some cases, due to image noise, the final and initial states do not match exactly, hence the constraints may be shown at a slightly incorrect orientation and/or the object may either protrude constraints, or not appear to contact the constraint.

In order to identify if an affordance has symmetry, the object must have a symmetrical face and the initial and final state conditions must be identical or mirrored. For example, affordances P_0 and P_6 are symmetrical (See Figure 6.11). When placed up against a wall on the right side of the object, instead of the left, a valid affordance can be produced. In the case of P_4 neither of the object's symmetrical faces are in full contact with a constraint, hence it can produce symmetrical transformations from either face.

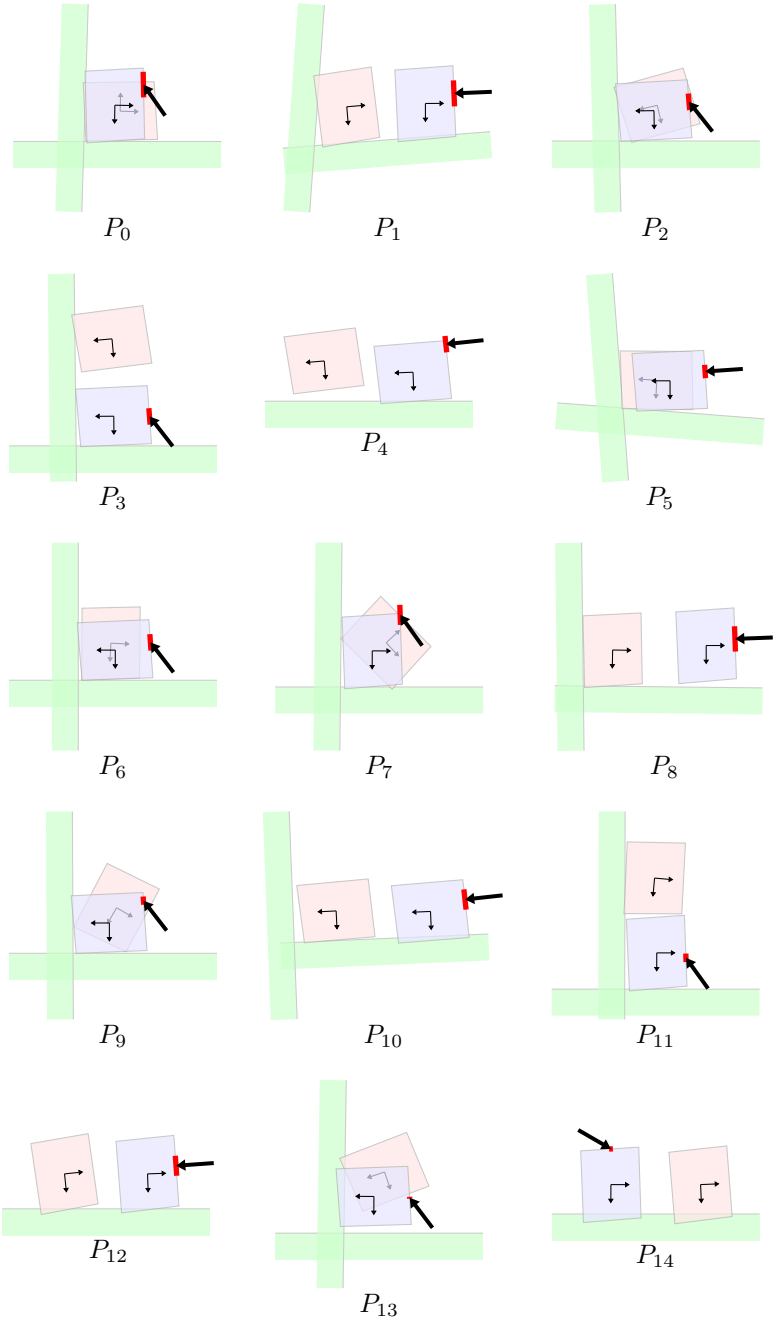


Figure 6.11: All learned affordances for the rectangle ground and wall trials using the robotic system.

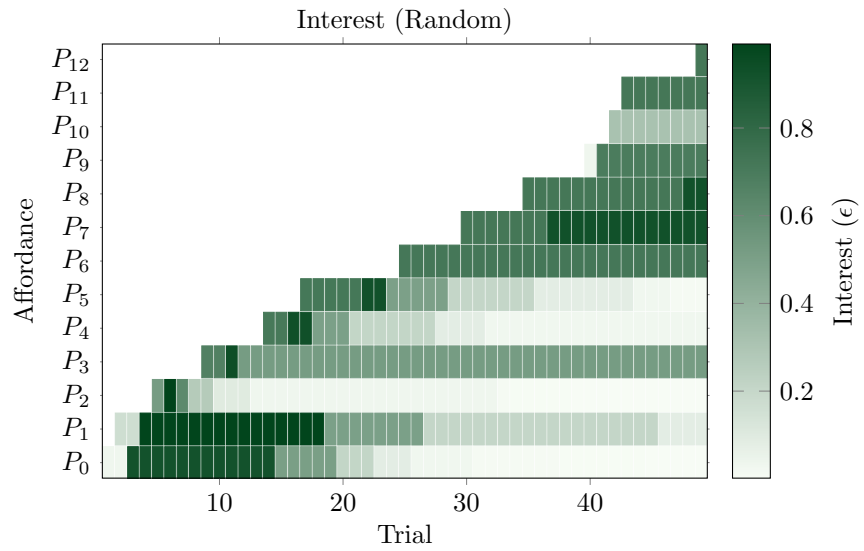


Figure 6.12: The interest for each affordance in a single set run using random exploration.

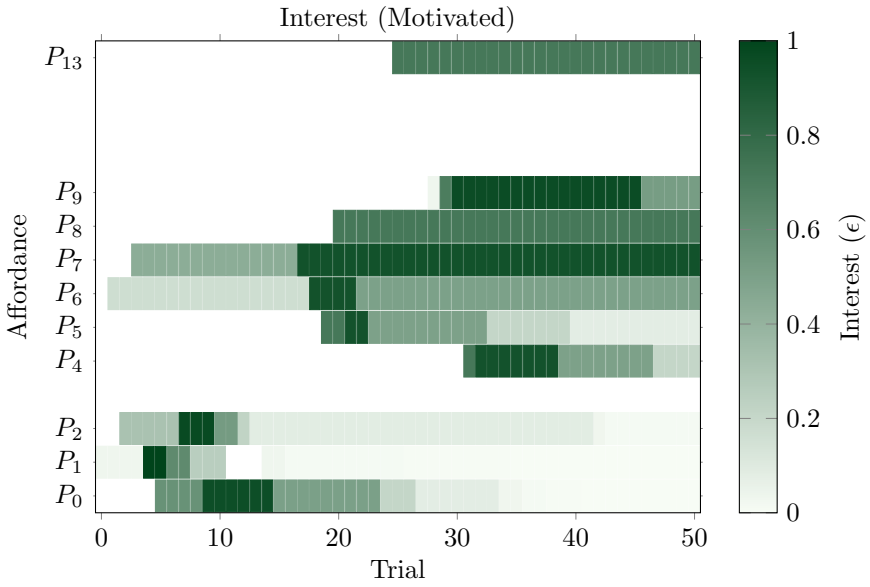


Figure 6.13: The interest for each affordance in a single set run using motivated exploration.

To depict the agents level of interest for each affordance, interest graphs are generated (See Figures 6.12 and 6.13). This shows the interest after each trial is recorded. Upon initial discovery of an affordance, the interest starts low, upon further 1-3 successful trials, the affordance interest increases to maximum. With successive trials the interest reduces down to near zero. The interest the agent has for each affordance is shown in the shade of green in the y-axis for each trial number on the x-axis.

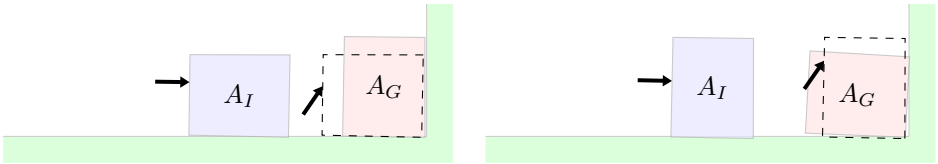
This shows the time and frequency an affordance is being interacted with and indicates how much the affordance is being focused on by the agent. In the case of the random set (Figure 6.12), the interest is calculated solely in order to facilitate visualisation of the quantity and frequency each affordance is explored. The interest value itself has no effect on the agents actions, rather, it can be used as a point of comparison to the motivated graphs. In Figure 6.13, some affordances are missing (P_3 and $P_{10} - P_{12}$) as the agent was not able to discover them within its 50 trial limit for this set.

The plans generated when a goal is set by a user are shown in Figure 6.14. The interaction point and force direction for each stage of the plan are shown via an arrow. The initial frame is indicated by A_I and the required goal frame is indicated by A_G. Intermediary frames used to transform the object through states to reach the goal state are depicted with dotted lines. These transformations are derived from the various affordances learned previously. The specific transformation and affordance information can be found in Appendix B. The isosceles triangle results can also be found in Appendix B.

6.1.9 Discussion

The system was able to discover 13 affordances for the rectangle and 15 for the isosceles triangle. Due to the limited interaction capability and symmetrical nature of the rectangle, only 2 plans were generated and performed. The plans from both the rectangle and isosceles triangle were able to successful sequence together 2 or more learned affordances to achieve a specified goal. The affordances utilised demonstrate the majority of useful affordances discovered and learned.

Although the tests described above are mostly autonomous, there was a small element of human interference. In some cases the agent was unable to reach areas of the object where interaction would result in a change of state. For example, in the isosceles triangle sets the agent was unable to flip the object from its side into an upright position. Therefore, the object was manually moved to different states when the same set of affordances had been performed repeatedly, such that the interest of the accessible



(a) (b)

Figure 6.14: Various goals and generated plans to complete them using learned affordances. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

faces became low. While this was necessary to prevent the agent from getting stuck and unable to explore further, there was potentially some bias in how affordances were explored. However, the primary purpose of this set of trials was to demonstrate learning and use of affordances in a real environment. Although there are many limitations with the system, future work would involve using a manipulator which has significantly better reach. This would facilitate a similar kind of capability as available in the simulation system presented below.

If the configuration of the manipulator was changed such that it could reach the entire uncovered surface of the object, many affordances would still be reproducible due to symmetry. Hence, there is technically no need to learn the opposite side. For example, it is unnecessary for the agent to learn to push the object to the left, as it already knows how to push to the right and can use symmetry to push to the left. Therefore, despite learning affordances in a limited environment, the affordances remain valid in many cases and can be used in more complex environments, without the need for re-learning the new environment.

As the object transformations are learned, not the specific actions of the manipulator, the capability of the manipulator can be changed without relearning affordances. However, in non ideal systems, there are still minor effects the manipulator can have. For example, it was observed that the required force magnitude and direction may be slightly different due to slackness in the joints of the manipulator. The compliance of the finger contact area can also affect the result of an interaction. This would generally only affect affordances which have a small area of interaction, hence require a measure of precision. Affordances which have large areas of interaction would more than likely be unaffected. Future work could further abstract the manipulators characteristics, allowing the agent to recalibrate slight differences with the manipulator without requiring minor relearning of affordances. An improved control system could also compensate for most small variations in manipulator configuration.

The compliant nature of the manipulator finger proved to be crucial for sheer interactions. This was particularly the case for interactions near the edge of a face. The finger was able to maintain traction when half the surface area was past the edge. Due to the unpredictable nature of manipulation, particularly during exploration, the way the object will react is unknown. Hence, finger compliance afforded the control system a measure of tolerance while moving the manipulator during force application on the object. It would be difficult to repeat many of the sheer interactions reliably without a compliant or high friction finger.

Due to the limited reach of the manipulator, the advantages of motivated exploration for the robotic system are not as significant as it could be. This is primarily due to the limited area available to explore, hence the smaller the area to explore and learn, the less effective motivated exploration becomes. Conversely, as seen later with the simulation system, the more there is to explore, the more effective motivated exploration becomes. With some improvements, this system could perform well in even more

complex environments.

Interestingly, areas with high potential often fostered great interest as many different affordances can be performed in those areas. This generated a compounding effect, greatly increasing the focus on those areas, which in turn increased the affordances discovered, which in turn increased the interest in that area. The agent would generally focus on these areas for a significant time until either all affordances were successfully explored, reducing interest, or the interest of the object was reduced from boredom (due to repeated outcomes).

There were a number of problems encountered which reduced the performance of the system in various ways. Image noise could sometimes cause instabilities in object and/or face detection. For example, in Figure 6.11 for P_1 and P_5 the face vertices were incorrectly approximated, causing constraints to have a slightly incorrect orientation. Although these errors are within the threshold allowed, for more complex environments, this will likely prove to be problematic. Image noise was also exacerbated by small movements. For example, in Figure 6.11 for P_4, the actual average movement of transformations captured for the affordance is an order of magnitude smaller. Hence, the noise is amplified when scaling the transformation up for larger moves. This was addressed in Section 4.1.7, where the constraint descriptor is used to generate a move instead of the affordance descriptor. Despite this, the further the object is moved during the exploration phase, the less likely noise will adversely effect the affordance. Improving the image quality and using more advanced image tracking techniques would also improve the results, allowing smaller tolerances and better performance.

Due to non-ideal contact, the object would occasionally slip, rotating about the x-axis of the projection, causing the object image to warp and the detected orientation to change despite no actual object rotation in the z-axis (see Appendix Section B.1.1, Figure B.1, P_4). Hence, despite the desire to remove manipulator influence by using point contact interaction, the manipulator can still have a minor impact on the outcome of the manipulation. Factors such as the coefficient of friction between the object and manipulator can affect the outcome for some affordances. This is primarily a limitation of the tactile sensor, which cannot detect slip or estimate friction. Although it is advantageous for a tactile sensor to detect slip, it may not be necessary as precision visual inspection may be sufficient for slip detection. Alternatively, the coefficient of friction could be estimated to provide sufficient feedback for control.

While flipping the object up, it could sometimes get caught on the tactile sensors circuit boards protruding from the back of the finger, preventing further rotation. For example, in Figure 6.11, P_7 and P_9 were the result of this issue. This is purely a limitation of the design of the tactile sensor and finger end point. A more compact design would alleviate these issues.

Plans were generated as shown in Figures 6.14. The data for these plans can be found in Appendix B, Section B.4. Due to the inaccuracies introduced by noise and the imperfection of the real environment,

the affordances which require very specific states can be difficult to successfully generate a plan. For many of these specific tasks the agent fails to generate a plan, or if successful, fails to successfully perform the plan. Furthermore, the error in orientation and position for each affordance in the plan accumulates. Hence, the longer the plan, the greater the chance of an erroneous state being produced and a poor, or no solution being found. The agent can move the object closer to the goal and as it gets closer, the accumulated error reduces. However, this can lead to non optimal plans as the agent may need to backtrack moves, or may become stuck and unable to produce a valid solution. A better performing system and longer learning times would likely reduce these issues and can be addressed in future work.

6.2 Simulation system

The physical system is somewhat limited in its workspace due to the fixed manipulator base. This restricts the possible areas of interaction on the object depending where it is located within the workspace. For example, the agent may only be able to interact with one side of the object. The manipulator itself can also interfere with the object after an operation is performed as it must return to a point where there is a valid approach to the desired interaction point. Furthermore, a user must place the object back into the workspace when moved outside. This can slow down the experimentation process, taking significant time, preventing exploration of a large number of trials.

Therefore, simulations are used to extend experimentation duration, provide a wider range of possible interactions and remove all human influences. To simulate a 2D environment and object, a solid body 2D physics engine is used [91]. In order to simplify the simulation, a manipulator is not used. Rather, a force is applied at the desired interaction point on the object face. Contact is assumed and ideal, hence there is no slip when applying sheer forces.

In this section, the environment and object in Section 6.2.1, the control methodology in Section 6.2.2, the system setup is described in 6.2.3, the experimentation method is detailed in 6.2.4, the results are presented in 6.2.5 and finally the results are discussed in 6.2.6.

6.2.1 Environment and Object

The environment is planar and represented by straight line segments which are assumed static and impenetrable as discussed in Section 2.2.1. The simulation world coordinates are used in place of the vision systems coordinates, hence the agent's workspace is not limited to a small area. The environment is therefore designed in such a way that operations can be continuously performed without the object moving outside the workspace of the agent. The objects used in simulation are a rectangle and an isosceles triangle.

6.2.2 Control

Control is implemented such that force is inverse exponentially proportional to object velocity. Initially when the object is stationary, maximum force is applied. As the object velocity increases due to applied force, force is reduced until velocity is zero. The agent increases or reduces force to maintain an equilibrium such that the object has no acceleration. As the object rotates, to simulate a loss of ability of the fingertip to supply force in the desired direction, the actual applied force is reduced. When the rotation magnitude reaches $\frac{\pi}{2}$, force can no longer be applied, effectively limiting the fingertip to a realistic range of forces. This serves to more accurately represent a manipulator approaching the object from a specific direction.

6.2.3 Setup

As discussed above, the simulation system is based on a solid body 2D physics engine. Static objects are placed into the environment in the form of ground and walls, which produce constraints on the object's movement when in contact. The simulation system provides the agent with the vertices of these objects in order to create constraints. As the physics engine is a solid body simulation, attempting to simulate the malleable properties of a robotic finger does not produce realistic interactions. Hence, simulating a finger akin to that described in Section 6.1.2 can not be done. Future work would involve implementing a soft-body physics engine for realistic finger interaction. To circumvent this problem, a single point force is used. This is acceptable as the agent is already designed to use point interactions as discussed in Section 2.2.2. Therefore, in these simulations, the agent is capable of reaching all unobstructed positions and orientations of object faces. As interaction is reduced to point forces, contact is ideal, hence there is no possibility of slip.

6.2.4 Method

The methodology used was similar to that of the robotic system as described in Section 6.1.7. However, because the simulation system was not limited by physical reach as the robotic system was, a greater range of interactions could be tested. Furthermore, because every face that was not in full contact with a constraint, became accessible to the agent, transitory goals were able to be implemented. In order to account for outliers and to obtain more reliable results, each set of interaction trials were repeated 10 times and the total trials of each affordance were averaged.

For the simulation system, experimentation was further broken up into two environments. The first simply consisted of a ground constraint, where 100 interaction trials per set are performed. The second consisted of a ground constraint with two vertical wall constraints situated either side, where 300

trials per set are performed. For the purpose of clarity and conciseness, only a single set of trials for the rectangle object are presented for a point of comparison with the robotic system. Further detail, including isosceles triangle sets of trials, can be found in the Appendix in Section B.1.2.

To test the affordances, plans are created in the same manner as described with the physical robotic system. A series of tasks are given to the agent by showing the agent the desired goal position and orientation of the object. The object is then placed in an initial position and a plan is calculated and shown.

It should be noted that the accuracy of the simulation reduces for very small or very large forces, due to the limits of floating point precision. All interactions were kept within acceptable ranges to avoid problems at the extremities.

6.2.5 Results

Ground environment

To depict the affordances the same method is used as described in Section 6.1.8.

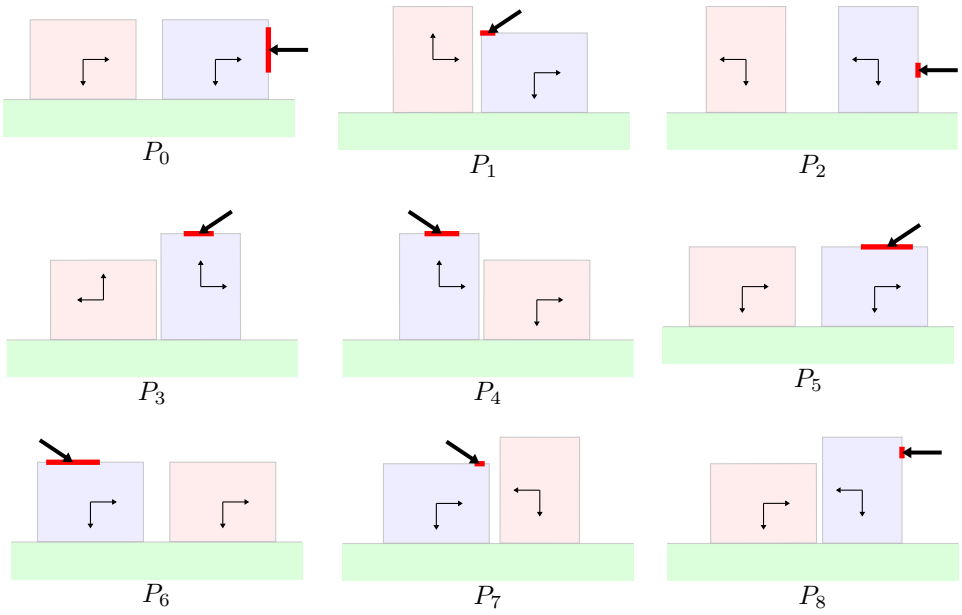


Figure 6.15: Learned affordances for the rectangle ground environment trials. One standard deviation of the interaction point range on the face is drawn in red either side from the interaction point, which is shown as an arrow.

The generated interest graphs only show one of the 10 sets captured. The motivated interest graphs can be seen to be quite different from the random graph. The interest of each affordance generally remains high for a much shorter period as the agent has focused on interactions in the area defined by the affordance. As described previously, the affordance interest rises and falls a single time. In the case of motivated transitory goals (Figure 6.18), the interest can rise and fall multiple times. This is dependent on the possible moves and states that can be attained that have interesting affordances. This can be observed with affordances P_1 and P_7, and to a smaller extent P_3 and P_4, which change the object state via a rotational transformation.

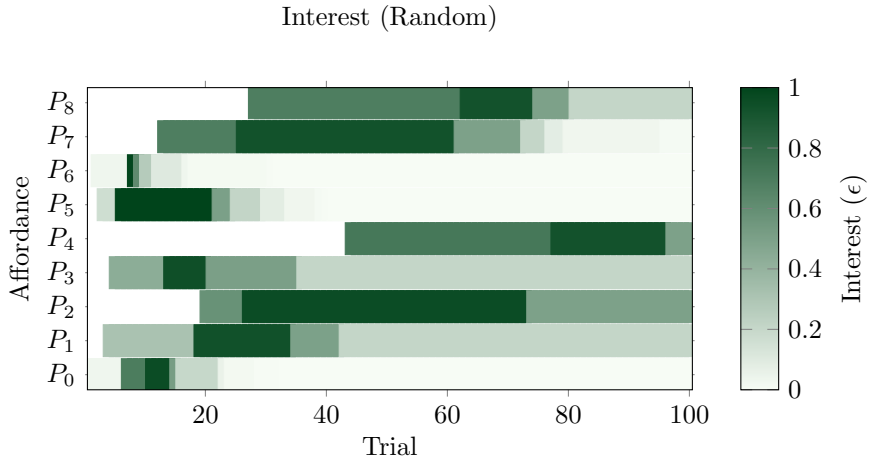


Figure 6.16: The interest for each affordance in a single set run using random exploration of a rectangle with a ground constraint.

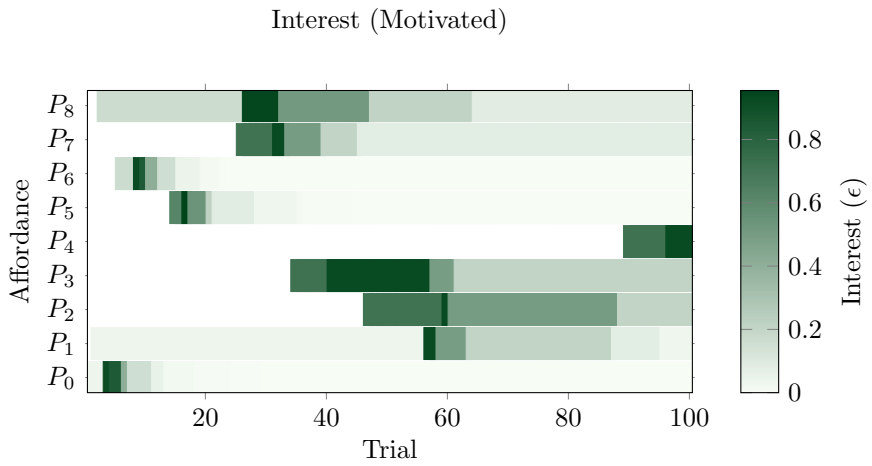


Figure 6.17: The interest for each affordance in a single set run using motivated exploration of a rectangle with a ground constraint.

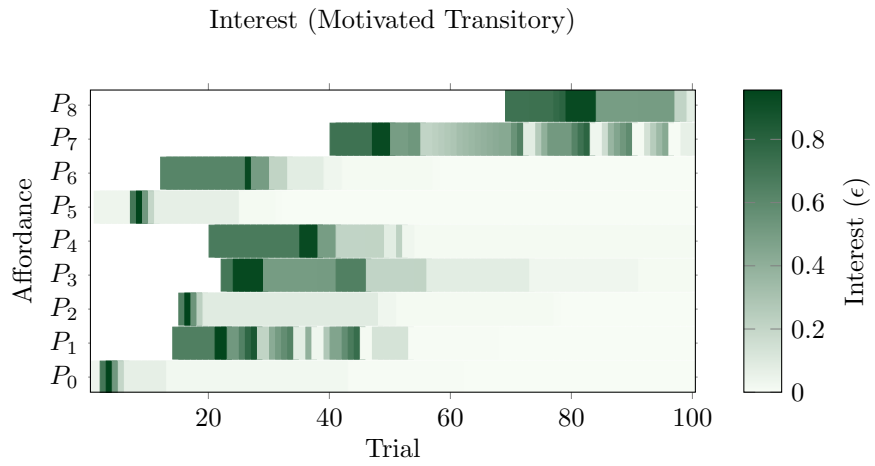


Figure 6.18: The interest for each affordance in a single set run using motivated transitory exploration of a rectangle with a ground constraint.

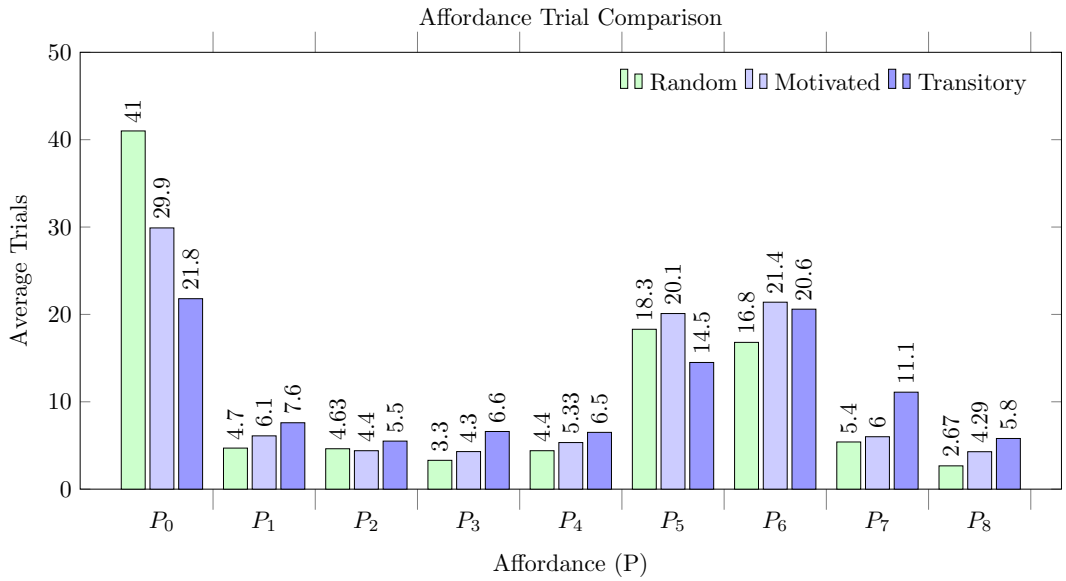


Figure 6.19: Comparing the average number of trials for each affordance in each exploration mode of a rectangle with a ground constraint.

To compare the motivated sets to that of random sets, the average of each affordance for each set is graphed for comparison in Figure 6.19.

Several goal frames were shown to the agent to solve. The initially generated plans for each are shown in Figure 6.20. They are depicted in the same manner as described previously in section 6.1.8. The isosceles triangle results can be found in Appendix B.

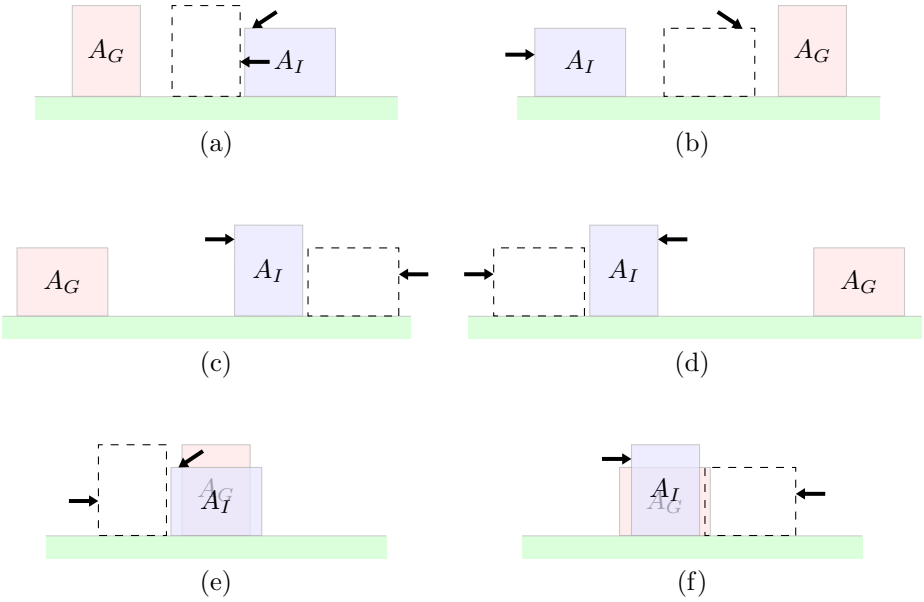


Figure 6.20: Various goals and generated plans to complete them using learned affordances of a rectangle with a ground constraint. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

Ground and walls environment

The number of affordances discovered for the ground and wall sets was 45. Hence, for the purpose of conciseness, the results for the ground and walls environment are shown in Appendix B. Section B.2.1 contains the rectangle results and Section B.1.2 contains the isosceles triangle results. The method of display follows the prior described approach.

Figure 6.21 shows the difference between random exploration, motivated exploration and motivated exploration using transitory goals. Each affordance is shown on the x-axis, with the y-axis showing the average number of trials. As can be seen with transitory goals, affordances such as P_8 and P_{17} are utilised more frequently as they change the object state in more useful ways. Other affordances could also be used, such as P_{16} and P_9, however the uncertainty and order in the tree often result in P_8 and P_{17} becoming the preferable methods. Circumstances where easy to reproduce affordances are repeated excessively, such as P_5 and P_6, are less frequent using motivated exploration. As can be seen for some affordances, due to their difficult requirements for reproduction, motivated exploration has a small effect.

The generated plans for various supplied goals are presented in Figure 6.22. All the goals presented in Figure 6.20 can also be solved in the ground and walls environment, but were omitted in favour of showing plans which utilise the walls in various ways.

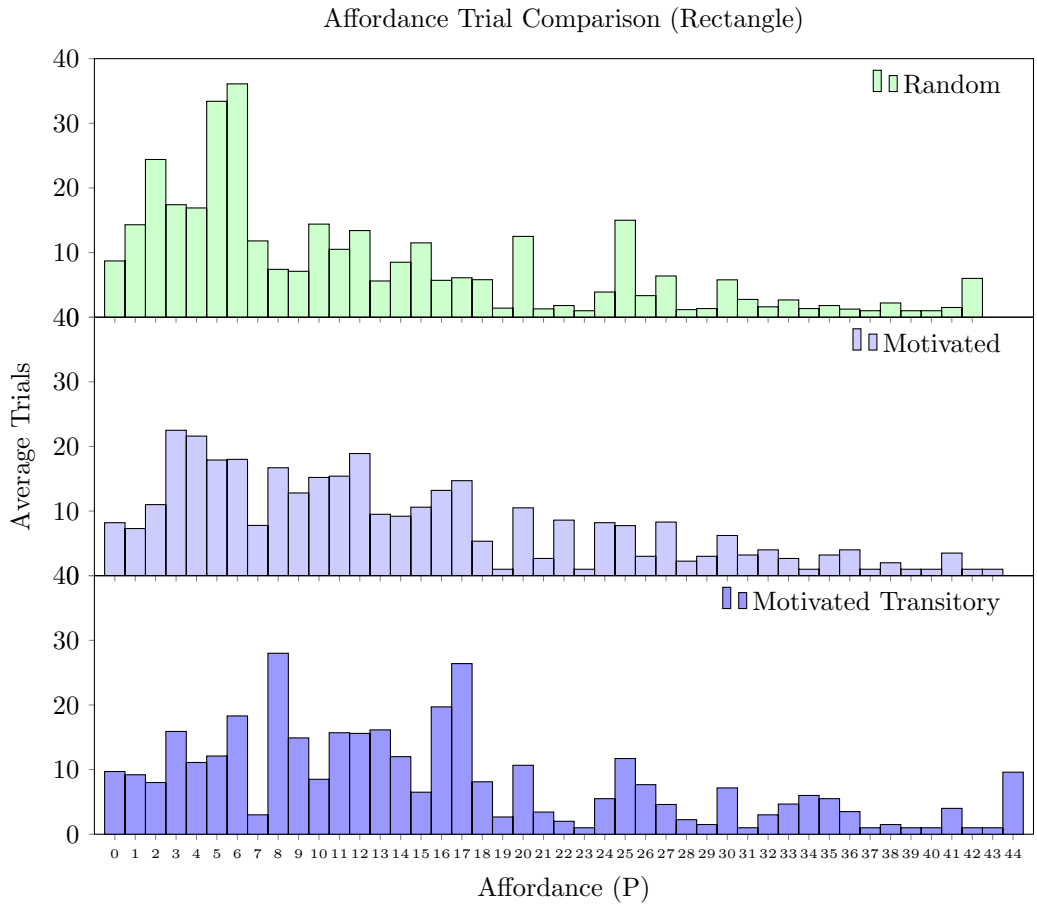


Figure 6.21: Comparing the average number of trials for each affordance in each exploration mode of a rectangle with ground and wall constraints.

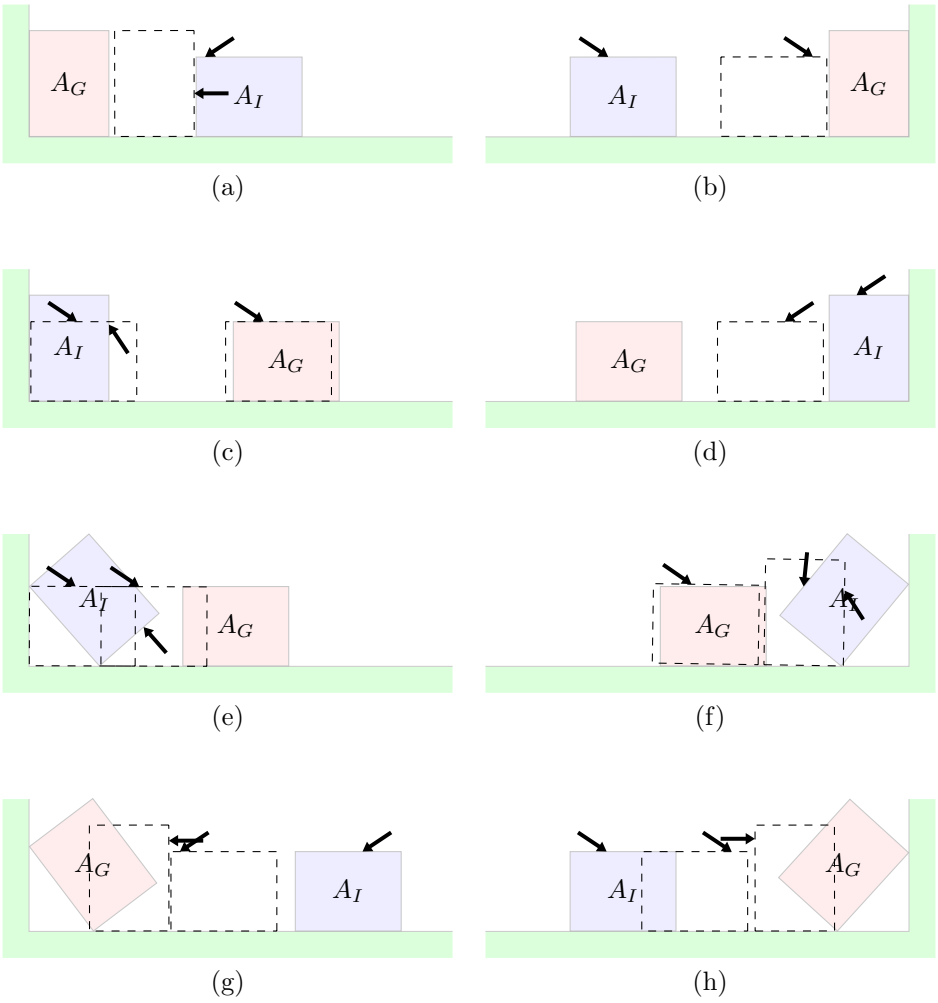


Figure 6.22: Various goals and generated plans to complete them using learned affordances of a rectangle with ground and wall constraints. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

6.2.6 Discussion

Ground environment

For the ground environment, a range of affordances from pushing, sliding or dragging interactions were learned for both the rectangle and triangle (see Figure 6.15). Many of these affordances achieve the same change in state, but via different transformations and/or interactions. Symmetry was successfully detected and utilised for both objects. As can be seen in Figure 6.20 in plans (c) and (d), the same affordances are used as the active face has been detected to have symmetry. However, as can be seen in Figure 6.15 for P_1 and P_7, or P_3 and P_4, or P_5 and P_6, symmetry across the same face is not accounted for, hence separate affordances are produced. This limitation can be addressed in future work.

For planning, to identify which affordance to use, the uncertainty is calculated based on Equation 4.20 (see Appendix B, Section B.3 for uncertainty of each move in the generated plans). The lower the uncertainty, the more preferable a move becomes and thereby the more preferable the plan path in the move tree. As the planning approach in this thesis uses discrete moves, the exact required move distance may not be able to be matched. This can result in the object falling outside the required goal threshold as all possible moves are too large to move the object closer. For goals with only a single solution, valid plans may therefore fail to be formed. This can be alleviated by reducing the size of each discrete move by increasing the precision value q in Equation 3.16 or reducing the scaling value k_a to reduce the move distance. In order to facilitate the increase in the number of move possibilities, the depth of the tree would need to be increased. However, this exponentially increases the processing time for plan generation, particularly for plans which have many valid possibilities. The goal threshold value k_h could alternatively be increased, but at the cost of accuracy. This can lead to plans that do not adequately produce a solution to reach the required goal.

In Figure 6.20, $q = 5$ and the maximum depth of moves is 6. Many of the moves are based on the same affordance and have the same direction and are therefore combined as discussed in Section 3.4.4. Consider plan (b), the depth of the last move is 5 (assuming a zero based depth index, See Section B.3 for depth data), despite only two moves being generated in the final plan. This indicates that several moves were merged as their affordances and directions were the same and therefore combined to form a single move.

In plans (c) and (d), the object is pushed further away in order to rotate it to the appropriate orientation. This behaviour is allowed by design as either the translational or rotational distance must be reduced for a move to be considered valid. However, as the current face has symmetry, this move can also be performed on the opposite face. The alternate plan would therefore have the same uncertainty. Hence, the first valid plan encountered is used, causing plans with the same uncertainty to be ranked

based on their order in the tree. A possible approach to better selecting an optimal plan would be to also consider the distance to the goal. Another possible factor affecting optimal plan selection, is the goal threshold issue as described above. In the case of the difference between (a) and (b) in Figure 6.20, the moves in (b) may put the object in a better position to meet the goal threshold, due to a lower number of successive identical moves. These issues could be addressed in future work where both the uncertainty and traversed distance are factored into ranking plans, in order to produce an optimal plan.

As can be seen for the ground only trials in Figures 6.19, using motivated exploration significantly improves exploration coverage of affordances. When the object interest was sufficiently high, the agent would focus on areas which produced affordances of high interest. As affordances interest were reduced, lowering the object interest, the agent increasingly produced more random trials, widening its exploration to potentially find new affordances. On average, affordances with low exploration coverage were increased by a factor of 2. Affordance P_0 in Figure 6.19, which was explored 41% of the time in random trials, was reduced to 29.9% for motivated trials and 21.8% for motivated trials using transitory goals. The transitory goals were able to move the object into states where more interesting affordances could be produced. They can be seen explicitly taking effect in Figure 6.18 where P_7 is utilised to move the object into the required state to perform P_8.

Ground and walls environment

In simple environments such as the ground only trials, the number of possible affordances to explore is quite small. However, when even a single wall is added to the environment, the number of possible affordances exponentially increases. Hence, large numbers of affordances were discovered and learned for the ground and walls trials. Many of these provide valid approaches to changing the object state in useful ways, where a new state affords multiple different affordances which can further change the state in useful ways. However, many affordances only produce unique states where a single, or no affordance can be learned. This is often due to the manipulator force holding the object in place, preventing it from falling. If the force is removed in order to provide a force at another location, the object falls or moves, changing the state. It is also unknown to the agent if affordances which produce unique states are desirable. Hence, it must be assumed they are all of equal importance and should be explored. A metric is therefore required to identify the usefulness of an affordance, adjust the interest and thereby the focus for exploration. One possible solution could involve motivating the agent by generating interest for unique states, focusing on those which are useful to explore. States which do not lead to other affordances are given low interest and filtered out for later exploration, while states which do, are focused upon and explored. To incorporate this into the exploration algorithm, the state interest would attenuate the interest of associated affordances. Affordances which produce states of high interest have their interest

increased, while those with states of low interest, are reduced.

The unstable states typically required to reach states of low usefulness can often be difficult to reproduce. This is typically a result of the very specific initial states required to produce the necessary transformation. For example, consider P_{37} to P_{43} of Figure B.17 (see Section B.2.1 in the Appendix). These must be moved into nearly the exact same position to reproduce. However, during exploration these positions are often not attained due to their specificity. Furthermore, due to the large number of affordances which are difficult to reach, the interest for the associated faces remains high for most of the exploration time. This is particularly problematic when all other affordances, which have interaction points in the vicinity of the difficult to reach affordance, have low interest due to exhaustive exploration. In order to prevent the agent from getting stuck, each time the same affordance is repeated, the interest is adjusted as described in Section 5.3.3. This is required as there may be affordances which cannot be reproduced reliably, or are extremely difficult to reproduce. However, this does not address exploring these affordances. Therefore, in order to explore affordances which have difficult to match initial conditions, plans could be generated and specifically followed to attempt to reach the required state to perform the move. The agent could therefore implement specific phases where it attempts to reproduce prior discovered affordances that change the object state to ones which have low usefulness. This can be further addressed in future work.

Symmetry was successfully detected and utilised for affordances with multiple constraints (see P_5 to P_7 of Figure B.17 for example). This significantly reduced the number of affordances to explore, generally improving the range of discovered affordances under limited trials. As discussed above, there are numerous affordances which have symmetry on the same face. Factoring these would significantly reduce the number of affordances and the learning time.

As can be seen from Figures 6.21 and B.15, the motivated and transitory goals exploration improved learning for affordances whose initial states were reasonably easy to reach. Affordances whose initial states were difficult to match saw little to no benefit from using interest to motivate exploration. While transitory goals could potentially help, they are limited to a single move. This is because the single move must exactly match the initial state of the desired affordance, otherwise the move will not be considered. The number of moves to search could therefore be increased, however the processing time increases exponentially and as moves are discrete, the same issue as describe above for plans in Section 6.2.6 is encountered. If the object is not within the goal threshold, the move will not be considered. Furthermore, even if a series of moves was identified the agent would have to perform them all exactly as described. However as the agent uses motivated exploration, the exact series of moves is unlikely to be produced. Furthermore, the agent determines interaction duration based on transformation metrics (See Section 3.3.1), not a distance to a specific point as used in plans. Therefore, it would be extremely difficult to produce the desired sequence of transformations for transitory goals.

The plans generated in Figures 6.22 and B.16 demonstrate the capability of the system to; recognise the various states of the object (including symmetry), identify valid affordances to use and execute them to move the object through states to the desired final state. The specific data for these plans can be found in Section B.4. Some states are difficult to achieve and will often fail to be successfully performed. They require significantly more 'practice' to identify the exact interaction point and conditions which produce the state. In some cases there are other factors outside of those which are accounted for which contribute to the interaction being successful or unsuccessful. For example, a specific force may be required at a specific angle and time during an interaction.

As discussed in Section 3.4.3 the agent generates a new plan after every move to account for discrepancies in the desired state and the state resulting from an interaction. If an interaction fails in some manner, causing the object's state to significantly deviate from what is predicted in the plan, it can result in the agent failing to successfully generate a plan to meet the goal. In the majority of cases this is because the agent must first move the object further away from the goal in order to then move the object into a state which can be moved to the goal. This can be addressed by allowing a limited number of valid moves which increase the distance to the goal. However, doing so can significantly increase the number of moves generated in the tree, therefore it should only be done when no valid solutions can be found initially. This can be explored further in future work.

The simulation system itself was not prone to failure, but was limited. This is primarily due to the ideal nature of the simulation, where the force acting on the object does not suffer from contact issues such as slip and has no inaccuracy when attempting to interact at a desired point. Interestingly, although the robotic system suffered from slip, it did not significantly affect the outcome for some affordances. When comparing relevant affordances from the robotic system to the simulation system, excepting the effects of noise, the affordances are generally similar. However, for other affordances, slip drastically changed the transformation, resulting in an entirely different affordance. Some affordances relied on slip to occur, failing without it. Slip therefore had the effect of increasing the uncertainty for those affordances, but also increasing the chance of failure while using them to perform a plan. The simulation system could not reproduce this effect. This limitation could be addressed in future work by using a soft body physics engine to simulate a finger with compliance.

6.3 Summary

Both physical and simulation system's environments were detailed. The physical system includes:

- An environment constraining an object to 3DOF.

- 3R robotic manipulator to interact with objects
- Vision system to detect object location, orientation, shape and outline.
- 1D tactile sensor to detect force magnitude and direction.
- Control system for movement and force control.

Objects are tracked and classified via visual characteristics using a vision system. The following is involved in the process:

- Object is detected and converted to a polygon approximation.
- New objects are classified, by their similarity to current known objects by using the Hu Invariants.

The tactile sensor was designed for a 2D environment and is therefore only 1 dimensional. It consists of 8 silicon rubber tubes wrapped around the fingertip. These tubes are sealed and have pressure sensors attached on one end. A rubber skin distributes force and reduces the effect of dead zones between tubes. Each tactel is combined to generate a resultant force direction and magnitude. The silicone tube design enables the sensor to be compliant, allowing for better traction over a larger range of forces.

The robotic system utilises two modes of control:

- Movement control - The robotic manipulator's velocity is limited to provide smooth and controlled movement.
- Force control - The fingertip maintains a specific force normal to the object face. If sheer force is required, the agent attempts to move the fingertip parallel to the object face whilst maintaining the normal force.

The robotic system:

- Demonstrates learning affordances and using them to execute moves in a real environment for a rectangle and isosceles triangle.
- Demonstrates the agent successfully applying normal and sheer force to produce a variety of transformations.
- Successfully generated and performed simple plans to complete a given task.

Numerous problems were encountered and can be considered in further work:

- The discrete nature of the plan generation algorithm can sometimes be problematic as the object may not fall within any threshold. This can be addressed by reducing the discrete move size, at the cost of tree depth.
- Suboptimal plans are sometimes generated due to the ambiguity from symmetrical moves. This can be addressed by also considered the distance the object moves and not just the uncertainty when determining the optimal plan.
- Motivated exploration provided little improvement for very difficult to reach affordances. Alternate forms of motivation could be used to encourage exploration of difficult to reach states.
- Plans can sometimes fail to be generated as the agent must first move the object further away from the goal. Allowing a certain number of moves which increase the distance would likely address this issue.
- Manipulator reach was restricted due to the system design, generally limiting interaction to a single object face. Using a more capable robotic system would address this issue.

The simulation system is implemented with a 2D physics engine. It is simpler than the physical robot, using direct force application for interaction. The simulation system is however not as limited as the physical system's workspace, hence allowing a wider range of interactions and outcomes. Simple proportional control is used for force application and the simulation's world coordinates are used in place of a vision system.

The simulation system:

- Demonstrated the generation of affordances and use of symmetry to simplify the affordances in various simple environments.
- Compares random exploration to that of the motivated exploration methodologies proposed in this thesis.
- Demonstrated the application of motivated learning to improve affordance exploration and learning optimal areas of interaction.
- Produced plans using affordances to solve simple tasks given to the agent.

A limitation of the simulation system was that it used an idealistic approach for simulating finger contact. This could be addressed in future work by using a soft body physics engine to fully simulate a compliant finger.

Chapter 7

Conclusion

This thesis addressed the problem of learning manipulation affordances of an object in a simple environment via the use of motivated exploratory pushing and sheering interactions. This chapter outlines the key contributions of the thesis in Section 7.1 and discusses further work in Section 7.2.

7.1 Conclusions

Manipulation is a difficult problem. This thesis presented a simple, limited approach for autonomous agents to generalise, learn and explore manipulation affordances. A system was developed that implements this, and is capable of learning manipulation affordances by observing the transformation an object undergoes due to some interaction by the agent. This thesis has specifically shown that:

- It is possible to build a system which can learn what an object affords in the context of manipulations in simple environments. Having learned what an object affords, the system can then use those affordances to manipulate the object into a specific state.
- The proposed manipulation affordances describe the transformation of an object and the necessary interaction required to reproduce the transformation. Affordances can be generalised by using a unit vector to describe the transformation and referencing it from the active face of the object, which is in turn referenced from the agent. Affordances can be discovered and learned via an unsupervised autonomous agent using exploratory behaviour which has a tendency to push the object at various points.
- Both normal and sheer interactions can be used to produce a variety of transformations which change the state of the object in useful ways. Sheer interactions are achieved by maintaining a

normal force on an object shape and moving the manipulator parallel to the object face.

- The proposed constraint concept enables affordances to generalise and characterise the effects of the environment on an object. Constraints are static barriers, described by a straight line, which prevent movement of the object in a specific direction when in contact with the object, thereby reducing the degrees of freedom. They are generalised by using a unit vector describing their direction and are referenced from the active face of the object, which is in turn referenced from the agent. Constraints are supplied to the agent prior to operation.

- Detection of symmetry facilitates identification of redundancy in object faces, thereby reducing the number of unique faces that need to be explored. This enables the agent to recognise a range of states where a single affordance can be utilised on symmetrical faces.

- Affordances can be used to generate plans to perform tasks given to the agent using a modified version of the nested hierarchical controller (NHC) to generate a discrete tree of all possible moves. Uncertainty is used as a means to produce an optimal plan by taking into account the approximate probability a different affordance will be performed at the same interaction point.

- Motivated exploration assists in the prevention of the natural bias of an object's own geometry during exploration for affordances, by identifying those that are interesting. Affordance interest is derived from novelty of an affordance which is based on the number of times the affordance has been observed. Initially, maximum interest is produced for an affordance when a it has been observed a moderate number of times. The more familiar an object's face becomes to the agent, the faster an affordance becomes highly interesting. The use of dynamic probability distribution functions based on the Boltzmann distribution and the interest of affordances, facilitates more precise exploration of object faces and affordances and prevents the agent from focusing too much, or too little on any given affordance. This allows objects to be explored in a more efficient manner, reducing learning time. Transitory goals aid the identification of interesting affordances which are only possible in a different state than that of the current state. The agent is then motivated to use other affordances to change the object state in order to match the required state of the originally interesting affordance. This further improves exploration efficiency and reduces learning time.

7.2 Future work

The work in this thesis is considered a starting point for further work on manipulation algorithms and approaches. Hence, there are many improvements that can be made to the algorithms and systems implemented in this theses. While some were discussed in previous chapters, the following outlines primary

avenues of further work which can be implemented to potentially improve the performance, scope and overall capability of the systems discussed in this thesis:

- Implement learning and exploration of a range of force direction interactions instead of assuming normal or sheer. The optimal force to use for the affordance would be determined by the average force from all observed transformations which match the descriptor.
- Incorporate tactile sensor force magnitude data into affordances, enabling friction and mass to be estimated and accounted for. This would require recognition of the environment to identify if the environment's surface/texture is different or if the object itself is different. The optimal force to perform the transformation an affordance describes, accounting for object mass and friction, could then be learned.
- Implement local experts (local learners) to learn features such as interaction force angle and the effect of local face push deviation from the ideal interaction point (the effect of moving the finger further from the ideal interaction point changes the result of the affordance). As such, the local expert information can be used to further improve the optimal force angle and interaction point which minimises the change in finger orientation for efficient movement when performing manipulations.
- Use a manipulator that can move the finger to any location or orientation to increase the capability of the system. Path and trajectory planning can also be implemented for manipulator movement in order to avoid undesired interaction with an object or environment.
- Implement multiple interaction points for affordances. Interaction points would therefore need to be treated like a form of constraint which restricts the degree of freedom and prevents motion normal to its surface. Optimal interaction points could then be learned in order to enact a specific affordance. This would potentially enable the system to learn how to better stabilise a transformation, or ensure a transformation occurs as described by the affordance.
- Enable the agent to learn sequences of affordances that change the object state. The sequence is analogous to an affordance itself, but transforming the object through a series of states, rather than a single state. Sequences which have valid initial and final states and result in a reduction of the goal distance can be used in planning. Only affordances matching the sequence order are allowed to be performed. Therefore, if the start and end states of a goal match that of a sequence, the agent can easily identify the sequence of affordances required to meet the goal. The agent simply has to determine the distance for each move as all other moves not derived from the sequence can be ignored. This would reduce the possibilities of moves when generating a tree for a plan and potentially reduce the size of the tree significantly, particularly for complex environments.

- Use multiple sources for generation of motivation, instead of solely affordance novelty. For example, novelty in object state, or in sequences of moves. This can potentially improve exploration of affordances which are difficult to reproduce. Interest could also be influenced by the ability to complete a goal or perform a novel sequence of moves.
- In planning, use the end state of the current move to identify failure. If the current state does not match the desired end state, then failure occurred. The number of constraints in effect can also be used to identify affordances which have lower degrees of freedom, hence greater stability. The rate of failure and stability of an affordance can then be used with uncertainty to better select an optimal plan.
- Use vision and tactile information with exploration to identify and learn what constitutes a constraint and what does not. This could be achieved by recognising the object coming into contact with an unknown object. The agent could then apply a force on the object. If no movement is detected and the force does not change, the unknown object could be a constraint. The agent would need to test the assumption several times before classifying the unknown object as a constraint. This would allow an agent to be placed in an unfamiliar environment without any user input to dictate what constitutes the environment constraints.
- Finally, the agent can be implemented for use in a full 3D environment. Instead of line segments representing faces and constraints, a surface is used. This would require the interaction point be a point on a plane and the force a 2D vector intersecting with the plane.

As manipulation moves into ever more complex and unpredictable environments, self-adapting and generalised learning approaches will become increasingly more important. Furthermore, knowing how to learn will become just as important as learning itself. With this in mind, this thesis has examined simple techniques for an agent to learn how to manipulate objects via self-motivated exploration.

Appendices

Appendix A

Vision and Control

A.1 Scaling Rotation Component

To obtain the appropriate scale value for transformations when converting them to a descriptor, the following equation is used:

$$s_f = \sqrt{\frac{2 \cdot Area}{b \cdot \pi^2}} \tag{A.1}$$

where $b = 0.999^{-2} - 1 \ = 0.002$, and $Area$ is the object area

It can be understood as the scaling value required to normalise the alpha component of a transformation descriptor (of an object of area A in m^2) to 0.999 when an object is rotated 180 degrees. This essentially results in a relationship where a factor of 4 increase in object area, requires a factor of 2 increase of scaling value. This however, is an approximation as the maximum object height (along y-axis) and width (along x-axis) are required with respect to its axis of minimum moment of inertia. The equation above uses the area instead. The exact equation is:

$$s_f = \sqrt{\frac{w^2 + h^2}{b \cdot \pi^2}} \tag{A.2}$$

where w and h are the width and height of the object respectively.

This accounts for objects which have small area but are elongated, increasing the distance travelled during rotations. If the object size or shape does not change significantly, the scaling value does not need to change.

A.2 Visual Calibration

To transform the camera coordinates to real world coordinates, the camera is first calibrated by sampling a number of images in order to determine the distortion coefficients and camera focal lengths. This is done by using the OpenCV 'calibrateCamera' function. For each frame, the camera image is undistorted using the 'undistort' function from OpenCV. To convert the image coordinates to real world coordinates, the camera focal length (f_c) obtained in calibration is used as follows:

$$s = \frac{Z}{f_c} \tag{A.3}$$

where Z is the distance to the manipulator and object. In this thesis $Z = 0.24$, where the units are meters. When calculating the position of an object in the real world from camera coordinates, the following is used:

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = s \cdot \begin{bmatrix} x_c \\ y_c \end{bmatrix} \tag{A.4}$$

where x_r and y_r are the real world coordinates and x_c and y_c are the camera coordinates.

A.3 Hu invariant

The Hu invariant [92] is defined as follows:

$$I_3(A, B) = \sum_{i=1...7} \frac{|m_A^i - m_B^i|}{|m_A^i|} \tag{A.5}$$

Where,

$$m_A^i = sign(h_A^i) \cdot log(h_A^i) \tag{A.6}$$

$$m_B^i = sign(h_B^i) \cdot log(h_B^i) \tag{A.7}$$

and h_A^i and h_B^i are the Hu moments of contours A and B respectively.

The Hu invariant enables a comparison independent of size, orientation and position. A perfectly matching object returns a value of 0, hence a threshold is used to account for noise and slight variances. The implementation used in this thesis is from the OpenCV library, using the 'matchShape' function.

A.4 Tracking Faces

To obtain a polygonal representation of the object to identify faces, the Douglas-Peucker algorithm [93] is used. However, the algorithm, as implemented in OpenCV, uses a raster approach. Hence, it always starts at the first encountered pixel in the contour. Therefore, when the object is rotated by the agent, the order of its vertices is lost and as a consequence the order of the object faces is lost. Furthermore, when using the axis of minimum moment of inertia to measure the orientation, there is a π ambiguity. Hence, the orientation may be incorrect by $\pm\pi$, provided it is in the domain $(-\pi,\pi]$.

To solve these problems, the orientation of the object can be tracked outside of the $(-\pi,\pi]$ domain by using the difference of orientation between frames and factoring sudden inversions of orientation as shown in Equation A.8.

$$\alpha = \begin{cases} \alpha_p + \alpha_\delta, & \text{if } |\alpha_\delta| \leq \pi \\ \alpha_p + \alpha_\delta + \pi, & \text{if } |\alpha_\delta| > \pi \wedge \alpha_\delta < 0 \\ \alpha_p + \alpha_\delta - \pi, & \text{if } |\alpha_\delta| > \pi \wedge \alpha_\delta \geq 0 \end{cases} \tag{A.8}$$

where, α and α_p are the current and previous orientations of the object respectively and α_δ is the difference in the detected orientation of the object.

Upon initial detection of the object, it is rotated by $-\alpha$ to a reference orientation and stored. Subsequent detection of object faces use this reference to compare vertices, implementing a naive approach to detect the order of vertices which best match the initial reference. This produces reliable tracking of the object faces. However, for objects with more than one axis of symmetry, such as a rectangle, an ambiguity still exists. Chapter 4 discusses an approach for dealing with symmetrical objects.

In order to track faces when their vertex order changes, a naive tracking algorithm is used:

```
void UpdateVertices(const vector<mVec2>& v, const mVec3& pos)
{
  auto size = v.size();
  if (size == state.vertices.size())
  {
    double min = 100; // initialise with large value
    vector<double> error;
    size_t offset = 0;
    int trackdir = 1;
    for (size_t o = 0; o < size; ++o)
    {
      double derror = 0;
```

```
      for (size_t i = 0, j = o; i < size && o < size; j = ++i + o)
      {
        if (j >= size) { j -= size; }

        double d = mDistance(mRotate(v[i], -pos.a),
                state.shape->vertices[j]);
        derror += d;
      }

      error.push_back(derror);

      if (derror < min) { min = derror; offset = o; trackdir = 1; }

      derror = 0;

      for (size_t i = 0, j = o; i < size && o < size; ++i)
      {
        double d = mDistance(mRotate(v[i], -pos.a),
                state.shape->vertices[j]);
        derror += d;

        --j;
        if (j >= size) { j += size; }
      }

      error.push_back(derror);

      if (derror < min) { min = derror; offset = o; trackdir = -1; }
    }

    double av_error = min / (double)size;

    SetVertices(v, pos, offset, trackdir, size, 0.75);
  }
}

void SetVertices(const vector<mVec2>& v, const mVec3& pos, size_t o,
        int trackdir, size_t size, double ratio)
{
  for (size_t i = 0, j = o; i < size; j = trackdir * (++i) + o)
  {
    if (j >= size)
    {
      if (j > 10 * size)
```

```
        j += size;
      else
        j -= size;
    }

    auto vert = v[i] + pos;
    auto& vo = state.vertices[j];

    vo = ratio * vert + (1 - ratio) * vo;
  }
}
```

A.5 Detecting Symmetry

The symmetry detection algorithm used is shown below. This method creates a face structure for each new object face discovered, storing them in the array 'faces'. If an object face is detected as symmetrical instead of creating a new face, it is linked to the symmetrical face structure.

```
void Shape(vector<mVec2>& v)
{
  for (int i = 0, i_1 = 1, ilen = v.size(); i < ilen; i_1 = (++i) + 1)
  {
    for (int j = 0, jlen = faces.size(); j < jlen; ++j)
    {
      for (int k = 0, klen = faces[j]->index.size(); k < klen; ++k)
      {
        if (faces[j]->index[k] == i)
        {
          faces.push_back(faces[j]);
          i_1 = (++i) + 1;
          j = 0;
          break;
        }
      }
    }

    if (i >= ilen) { return; }
    if (i_1 >= ilen) { i_1 = 0; }

    Face *const face = new Face(i);
```

```
    // faces contains the face pointer for each face of the object
    faces.push_back(face);
    // a list of pointers to unique faces will always <= to
    // the faces array
    flist.push_back(face);

    // mVec2 is a 2 dimensional vector
    mVec2 vpx1(abs(v[i].x), v[i].y);
    mVec2 vpy1(v[i].x, abs(v[i].y));

    mVec2 vpx2(abs(v[i_1].x), v[i_1].y);
    mVec2 vpy2(v[i_1].x, abs(v[i_1].y));

    for (int j = i+1, j_1 = j+1, jlen = v.size(); j < jlen;
                    j_1 = (++j) + 1)
    {
      if (j_1 >= jlen) { j_1 = 0; }

      mVec2 vcx1(abs(v[j].x), v[j].y);
      mVec2 vcy1(v[j].x, abs(v[j].y));

      mVec2 vcx2(abs(v[j_1].x), v[j_1].y);
      mVec2 vcy2(v[j_1].x, abs(v[j_1].y));

      // check for mirroring on x axis - check for mirroring on y axis
      // mDistance is the euclidean distance between 2 supplied points
      if ((mDistance(vpx1,vcx2) < THRESH &&
           mDistance(vpx2,vcx1) < THRESH) ||
          (mDistance(vpy1,vcy2) < THRESH &&
           mDistance(vpy2,vcy1) < THRESH))
      {
        face->index.push_back(j);
      }
    }
  }
}
```

A.6 Robotic System Control

A.6.1 Servo Control

In order to move the servos, a discrete interval based approach is used to periodically update their positions:

$$\nu = [\theta_1, \theta_2, \theta_3] \tag{A.9}$$

where ν is a matrix representing the orientation of each servo $(\theta_1, \theta_2, \theta_3)$

$$\nu_d = \nu_f - \nu_c \tag{A.10}$$

where ν_d is the distance the servos must travel to reach the desired final position ν_f and ν_c is the current position of the servos.

$$\hat{\nu_d} = \frac{\nu_d}{|\nu_d|} \tag{A.11}$$

where $\hat{\nu_d}$ is a unit vector representing the relative speeds of each servo to each other

$$\nu_s = \delta\hat{\nu_d} \tag{A.12}$$

where δ is the maximum servo speed and ν_s is the amount to move the servos in the time interval.

$$\nu_n = \nu_c + \nu_s \tag{A.13}$$

where ν_n is the new position of the servo.

Hence, at each interval the servo position is updated with ν_n. If this new position is outside of the servo's capability, the positional update is ignored.

A.6.2 Inverse Kinematics of a 3R manipulator

Assuming a 3R manipulator with known joint lengths L_1, L_2 and L_3 and a known base position $\boldsymbol{B}$, to obtain the joint angles; θ_1, θ_2, θ_3 which situate the manipulator end-point $\boldsymbol{P}$ in the desired position, inverse kinematics is used. The equations are as follows:

$$\omega_x = (P_x - B_x) - L_3 cos(\vartheta) \tag{A.14}$$

$$\omega_y = (P_y - B_y) - L_3 sin(\vartheta) \tag{A.15}$$

$$\alpha = atan2(\omega_y, \omega_x) \tag{A.16}$$

$$\tilde{\beta} = acos((L_1^2 + L_2^2 - \omega_x^2 - \omega_y^2)/(2L_1L_2)) \tag{A.17}$$

$$\beta = acos(\tilde{\beta}) \tag{A.18}$$

$$\tilde{\gamma} = acos((L_1^2 - L_2^2 - \omega_x^2 + \omega_y^2)/(2L_1 sqrt(\omega_x^2 + \omega_y^2))) \tag{A.19}$$

$$\gamma = acos(\tilde{\gamma}) \tag{A.20}$$

$$\theta_0 = \alpha - \gamma \tag{A.21}$$

$$\theta_1 = \pi - \beta \tag{A.22}$$

$$\theta_2 = \vartheta - \theta_0 - \theta_1 \tag{A.23}$$

$$\acute{\theta_0} = \theta_0 + 2\gamma \tag{A.24}$$

$$\acute{\theta_1} = -\theta_1 \tag{A.25}$$

$$\acute{\theta_2} = \vartheta - \acute{\theta_0} - \acute{\theta_1} \tag{A.26}$$

However this requires that ϑ be known. It is desired that only an x-y coordinate be given and the optimal servo positions found. In order to find the optimal servo positions, a search algorithm is used.

This involves searching over the range $[-\pi, \pi)$ for valid solutions. A solution is considered valid in the domains: $-1 \leq \tilde{\beta} \leq 1$ and $-1 \leq \tilde{\gamma} \leq 1$, and θ and $\acute{\theta}$ are within the specified servo limits. Once a list of valid solutions are found, the average of $\tilde{\beta}$ is calculated. The solution with the closest $\tilde{\beta}$ to the average is used. This ensures that where possible the end-effector has room to adjust its orientation either way as much as possible with minimal adjustments to servo positions.

The joint lengths are measured prior to operation. The position of the manipulator base is measured by manually locating the base from a camera image and converting its coordinates from camera coordinates to real world coordinates. This process is only required to be performed once.

```
// returns true if a solution is found, otherwise returns false
bool InverseKinematics3R(mVec3& pos, vector<double>& theta)
{
  vector<kinematics> klist;

  double  min=1, max=0;
  mVec3 cpos = pos;
  cpos.a = -M_PI;

  while(cpos.a < M_PI)
  {
    // base_offset is the position of the manipulator base
    // l1,l2,l3 are the joint lengths
    kinematics k(cpos,base_offset,l1,l2,l3);

    if((abs(k.b_in) <= 1) || (abs(k.g_in) <= 1))
    {
      k.Calculate(); // calculates inverse kinematics

      if(ServoLimits(k.theta) || ServoLimits(k.theta_alt))
      {
        double ab_in = abs(k.b_in);
        if(ab_in < min) { min = ab_in; }
        if(ab_in > max) { max = ab_in; }
        klist.push_back(k);
      }
    }

    cpos.a += M_PI * KINEMATIC_SEARCH_STEP; // total 200 steps
  }

  if(klist.size() == 0) { return false; }
```

```
  double av  = (min + max) / 2;

  // Find the solution which provides the most minor movement
  // either side of the desired orientation
  kinematics k;
  double d = 1;
  for(auto& kt : klist)
  {
    double ab_in = abs(abs(kt.b_in) - av);
    if(ab_in < d)
    {
      k = kt;
      d = ab_in;
    }
  }

  // determine if theta or theta_ should used based on which is within
  // the servo limits
  if(ServoLimits(k.theta))
    theta = k.theta;
  else
    theta = k.theta_alt;

  pos.a = k.ar;

  return true;
}
```

Appendix B

Additional Results

B.1 Isosceles Triangle Affordance Results

This section contains results for the isosceles triangle for both the robotic and simulation systems from chapter 6.

B.1.1 Robotic System

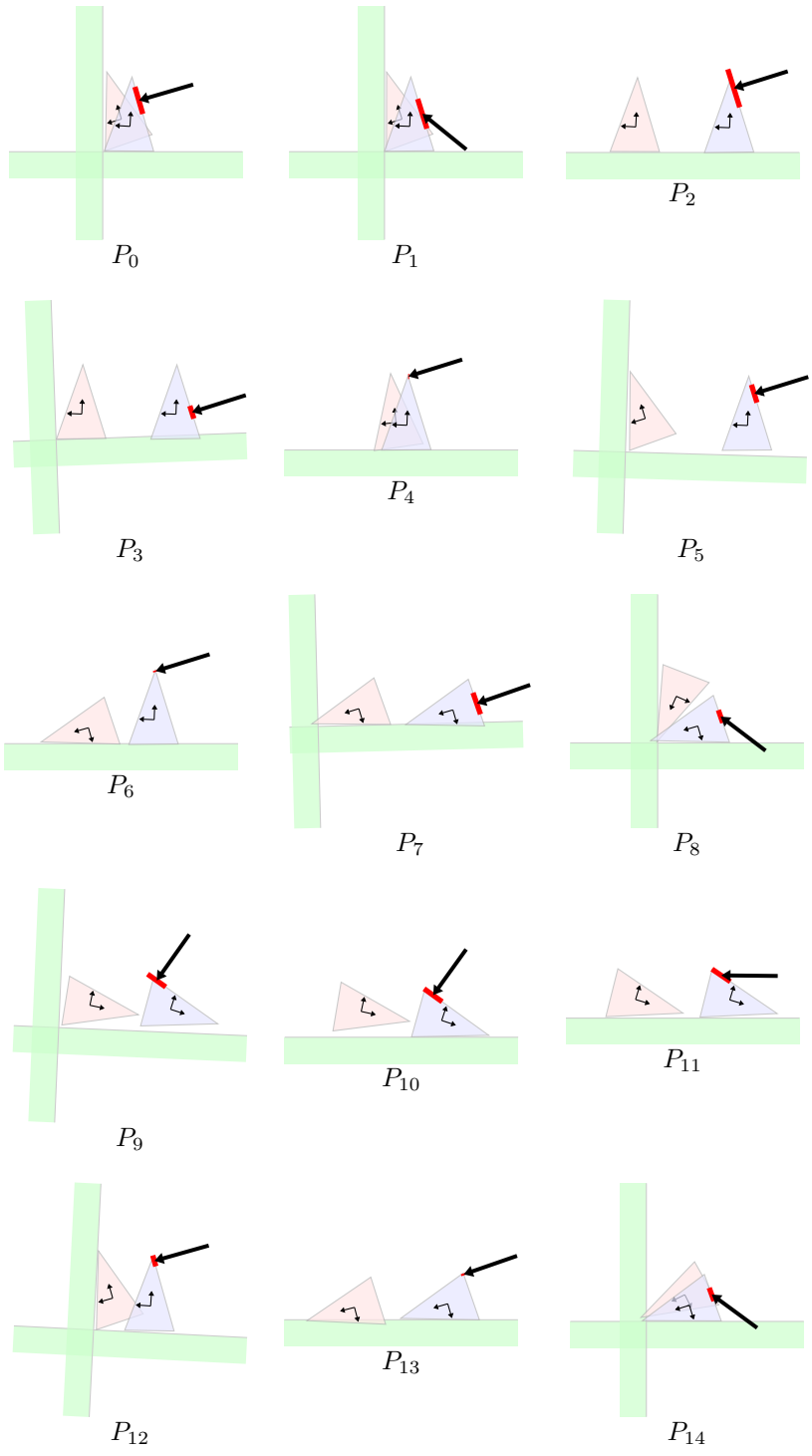


Figure B.1: Learned affordances for the isosceles triangle ground and wall trials.

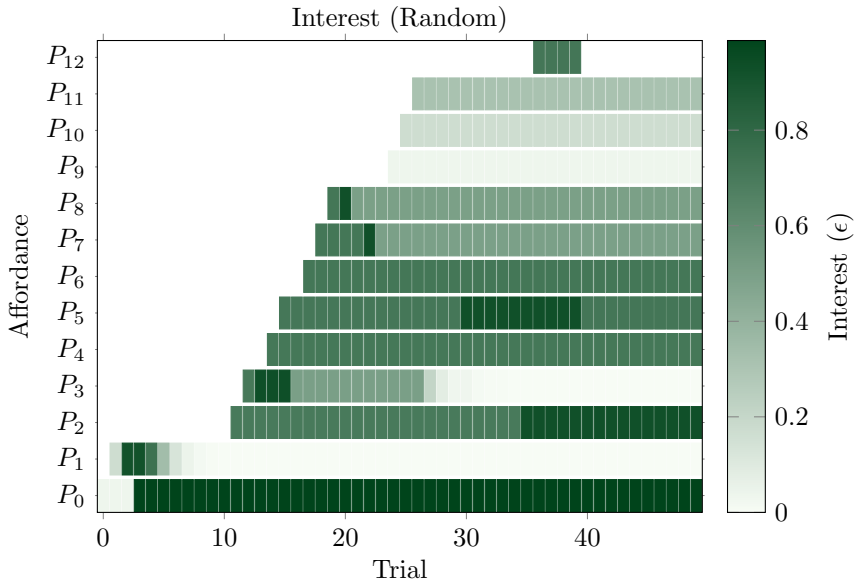


Figure B.2: The interest for each affordance in a single set run using random exploration.

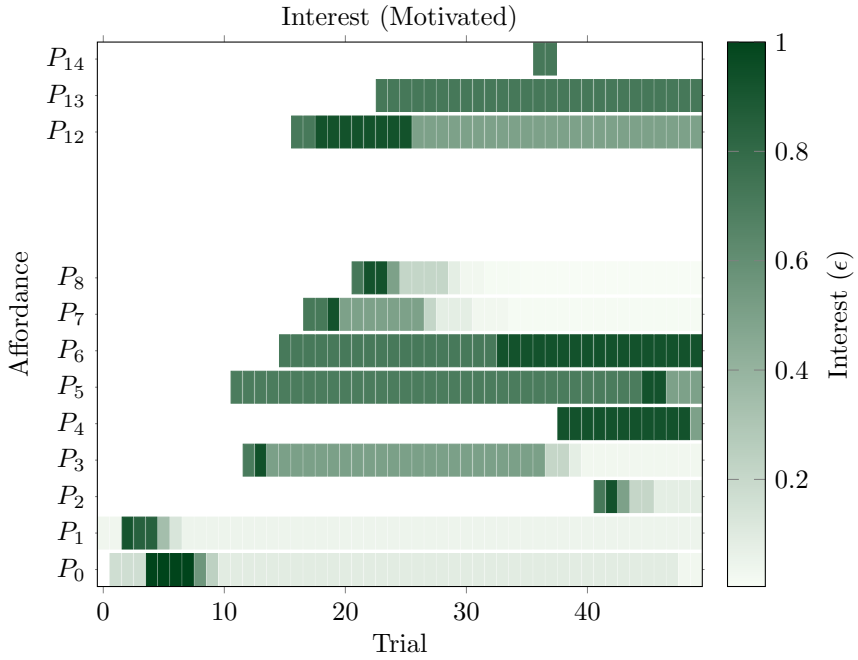


Figure B.3: The interest for each affordance in a single set run using motivated exploration.

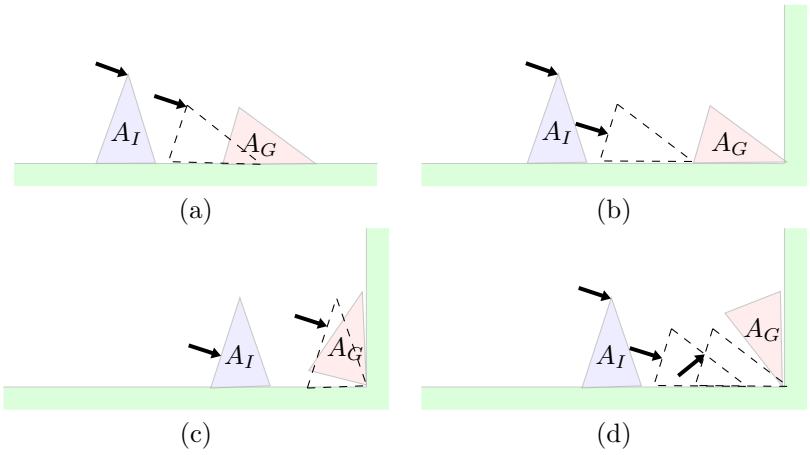


Figure B.4: Various goals and generated plans to complete them using learned affordances. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

B.1.2 Simulation System

Ground Environment

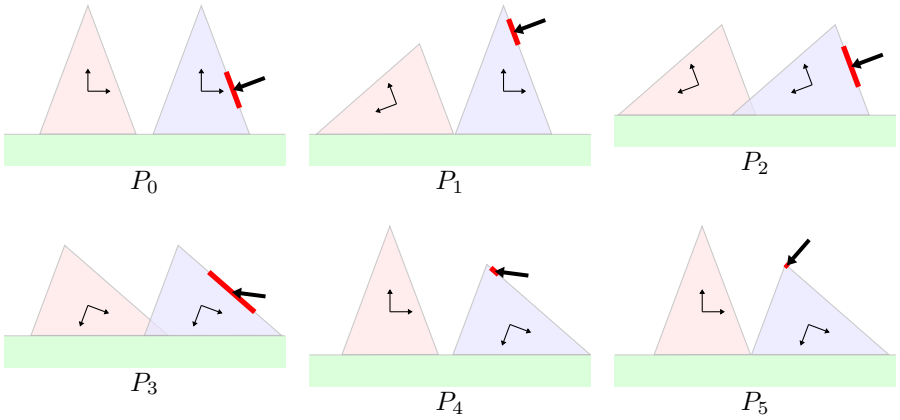


Figure B.5: Learned affordances for the isosceles triangle ground only trials. One standard deviation of the interaction point range on the face is drawn in red either side from the interaction point.

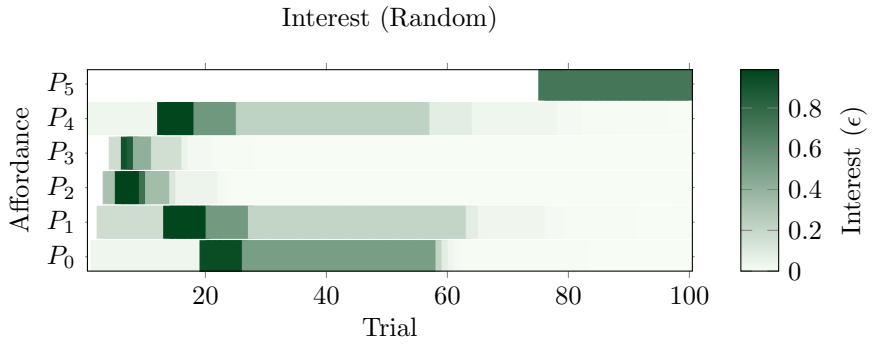


Figure B.6: The interest for each affordance in a single set run using random exploration of an isosceles triangle with a ground constraint.

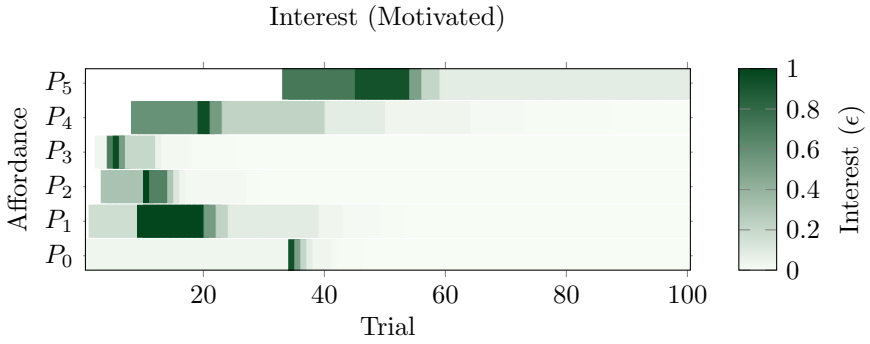


Figure B.7: The interest for each affordance in a single set run using motivated exploration of an isosceles triangle with a ground constraint.

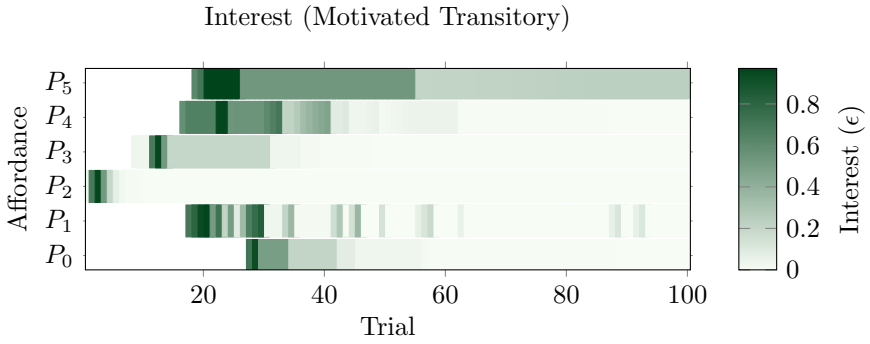


Figure B.8: The interest for each affordance in a single set run using motivated transitory exploration of an isosceles triangle with a ground constraint.

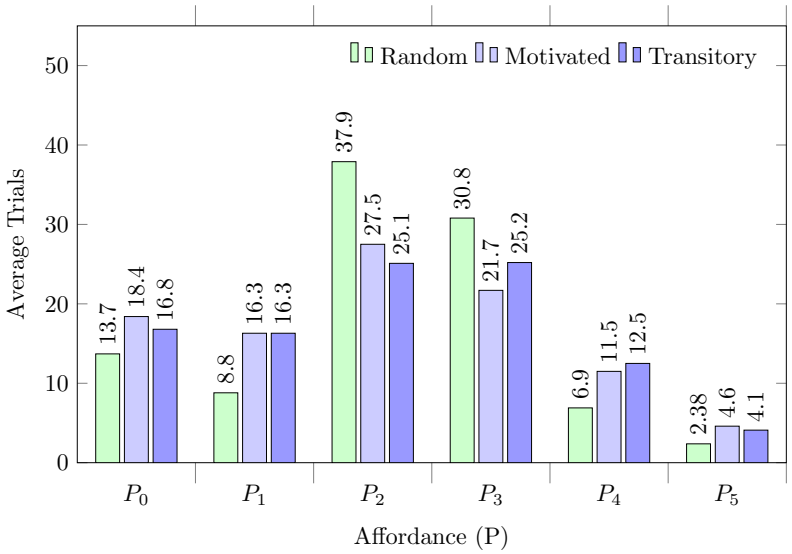


Figure B.9: Comparing the average number of trials for each affordance in each exploration mode of an isosceles triangle with a ground constraint.

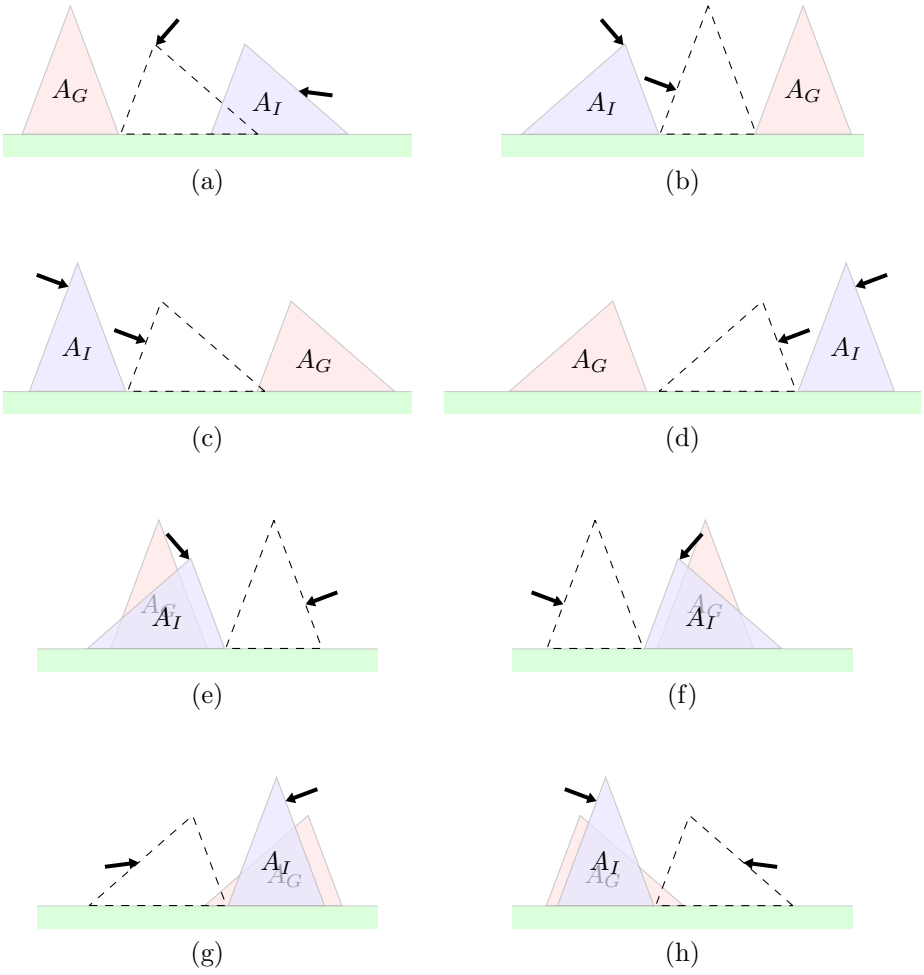


Figure B.10: Various goals and generated plans to complete them using learned affordances of an isosceles triangle with a ground constraint. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

Ground and Walls Environment

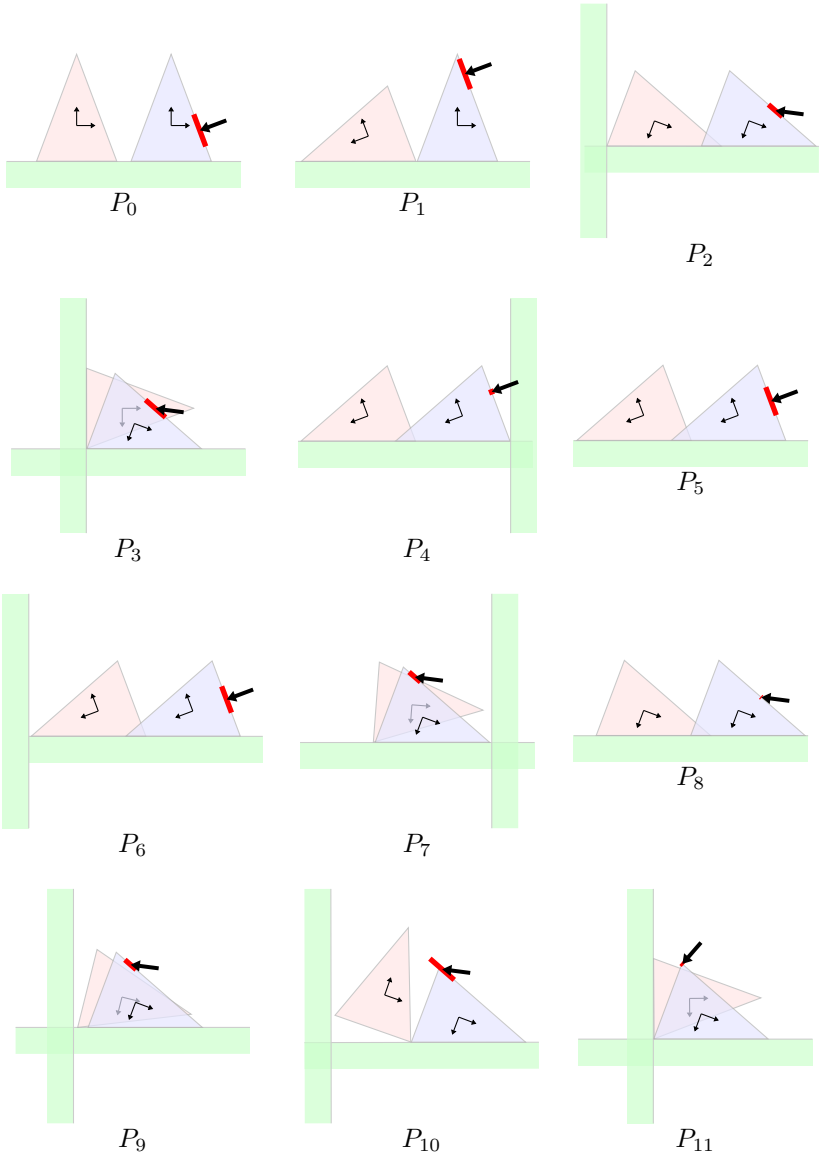


Figure B.11: Learned affordances for the isosceles triangle ground and walls trials. One standard deviation of the interaction point range on the face is drawn in red either side from the interaction point.

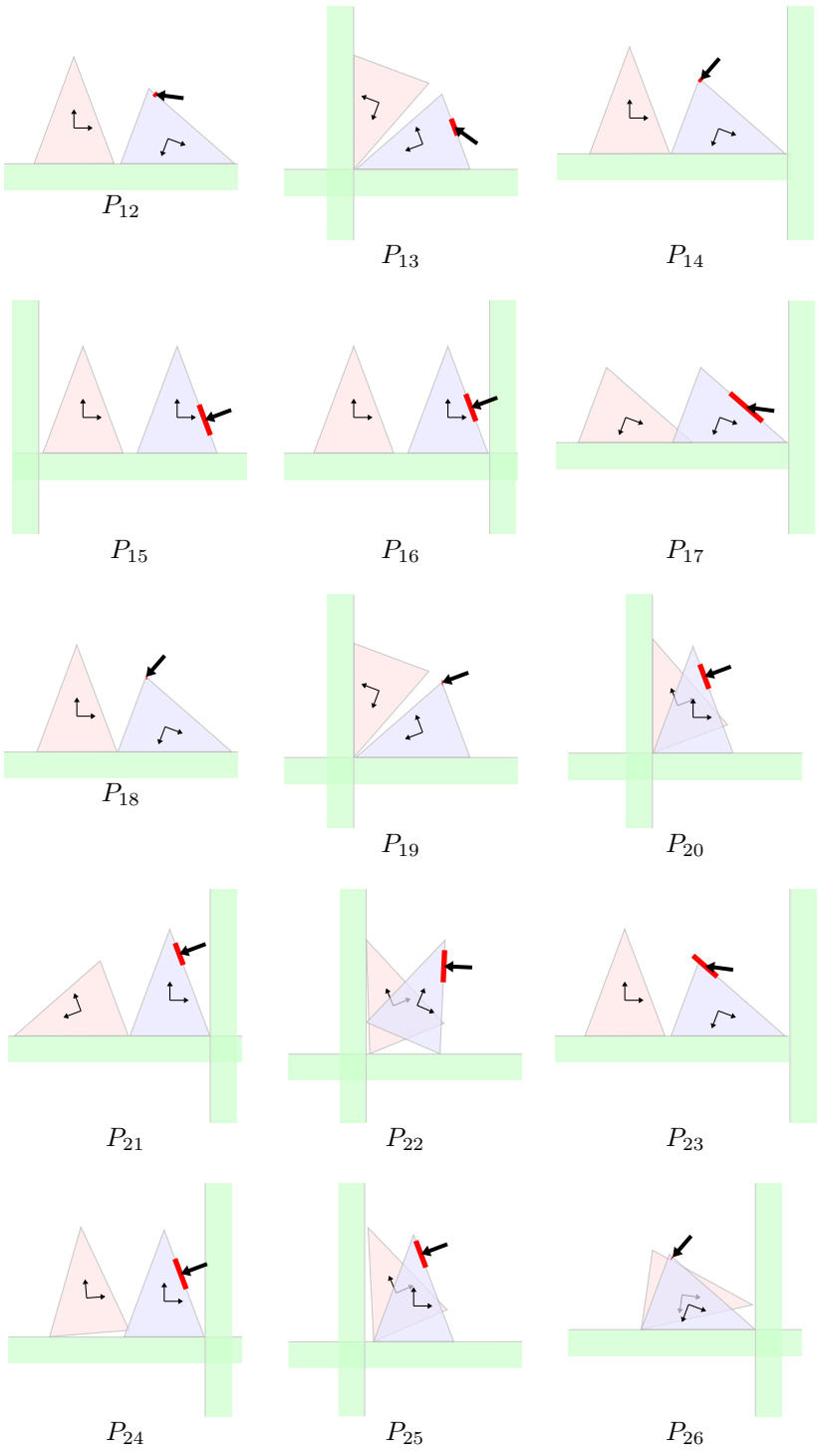


Figure B.11: (cont.) Learned affordances for the isosceles triangle ground and walls trials.

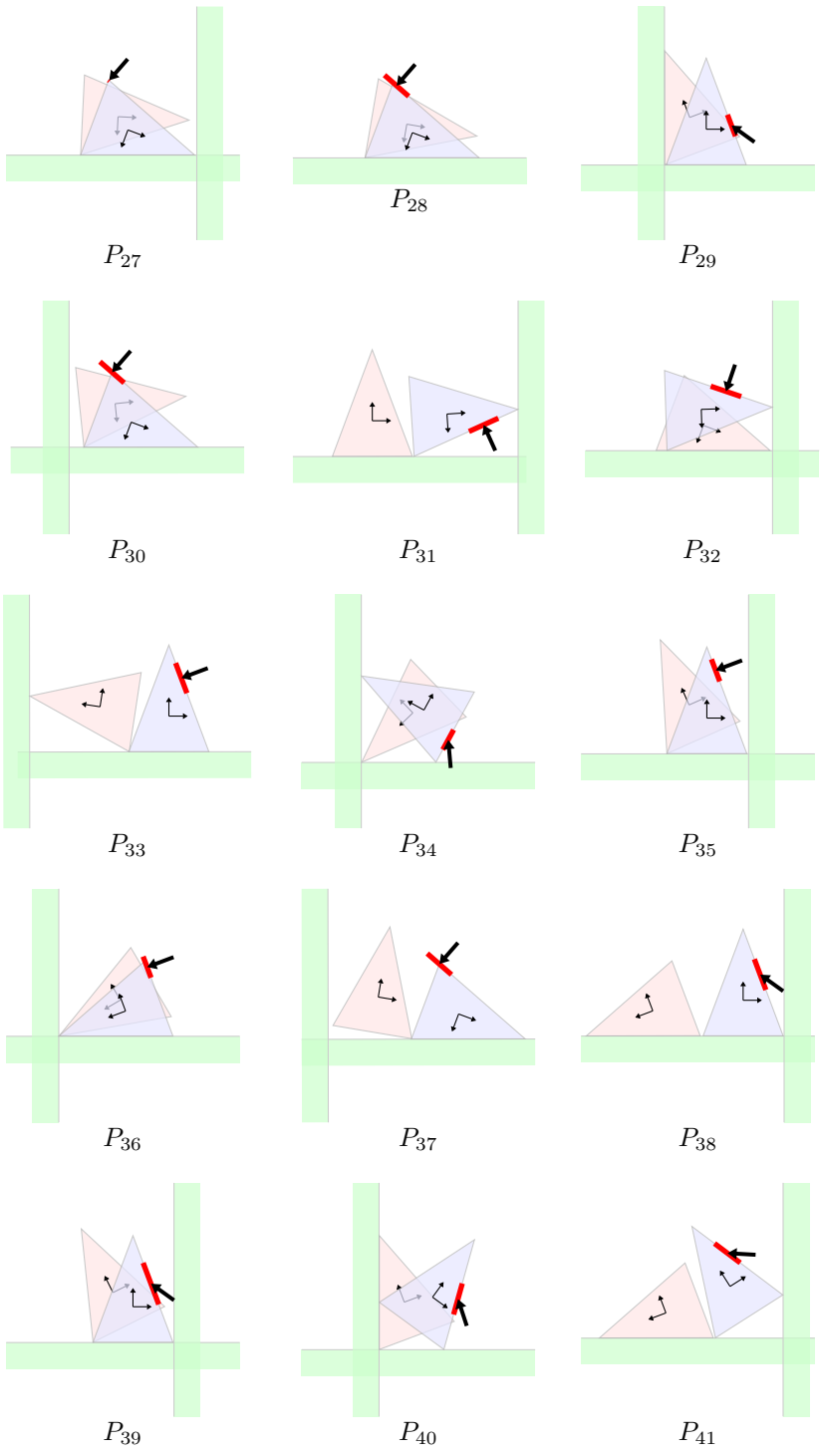


Figure B.11: (cont.) Learned affordances for the isosceles triangle ground and walls trials.

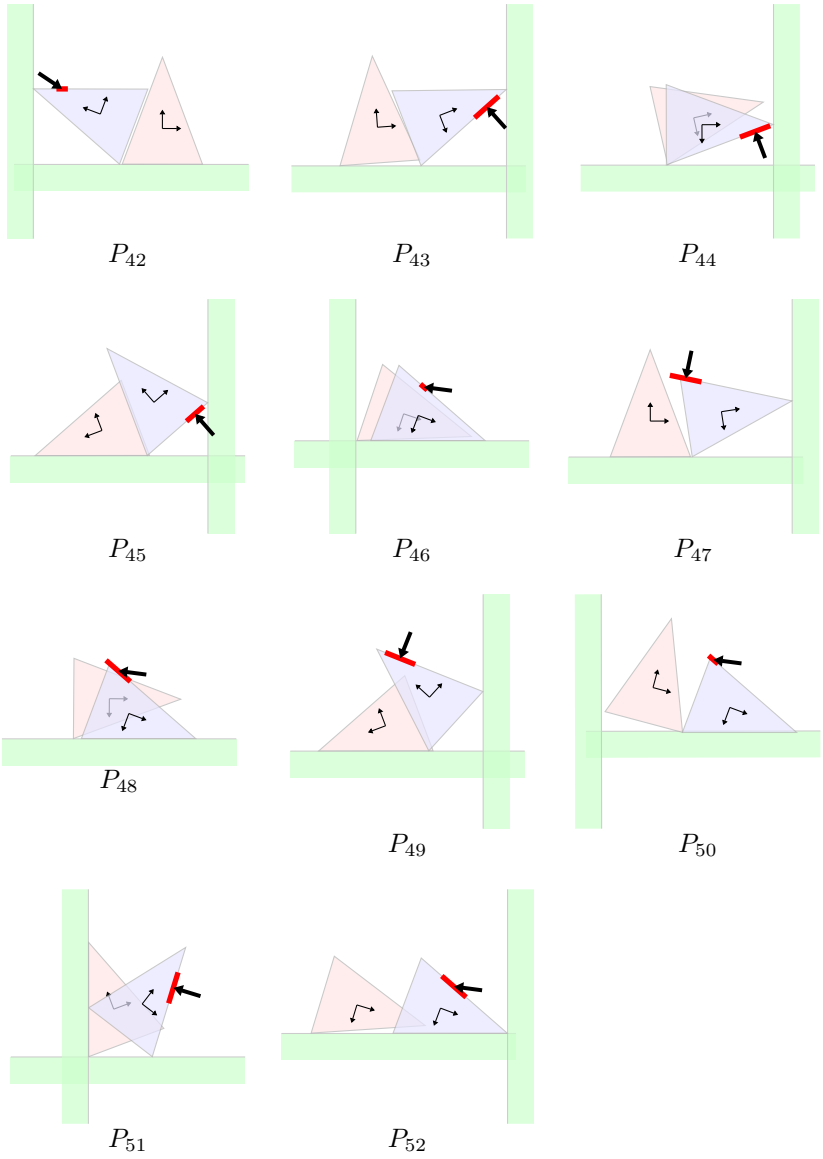


Figure B.11: (cont.) Learned affordances for the isosceles triangle ground and walls trials.

To generate the interest graphs such as Figure B.18, affordances are matched across multiple sets of trials. Matching can sometimes be problematic as when an affordance is performed and updated, changing its average, it can become a closer match to a different affordance. The affordance itself does not change, only the label identifying it. Hence, the interest graphs can have missing sections, such as P_{20} and P_{40} from Figure B.18. This simply means that the affordance better matched a slightly different one from another set of trials during the current set of trials.

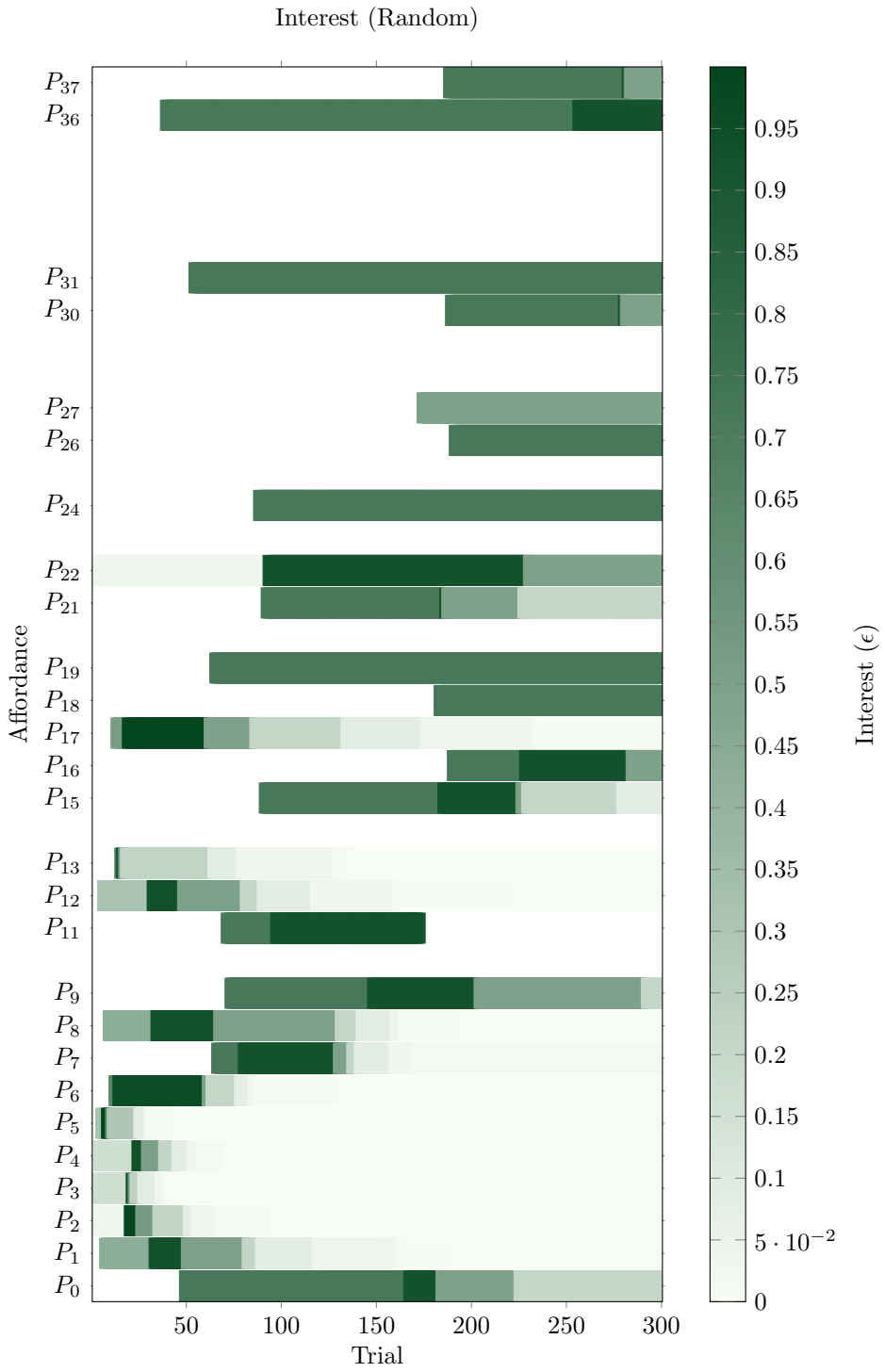


Figure B.12: The interest for each affordance in a single set run using random exploration of an isosceles triangle with ground and wall constraints.

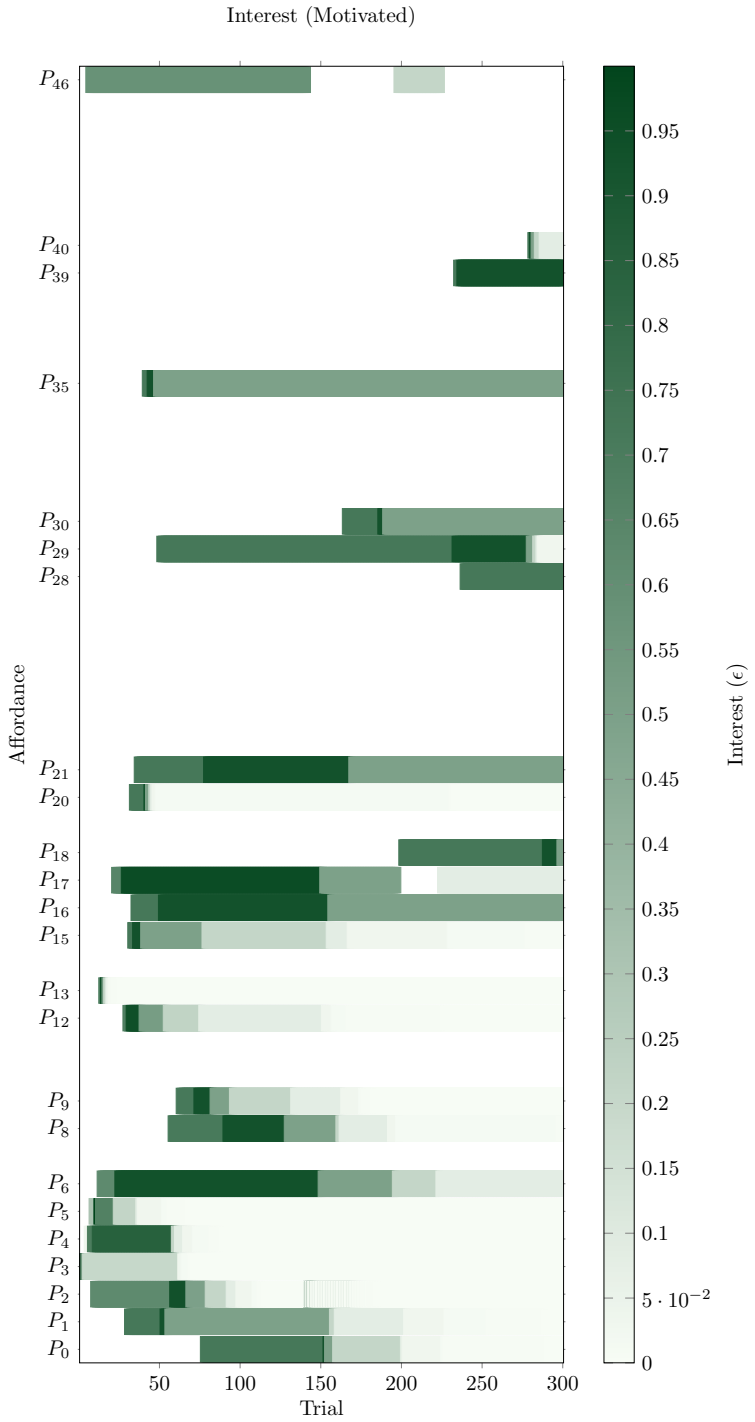


Figure B.13: The interest for each affordance in a single set run using motivated exploration of an isosceles triangle with ground and wall constraints.

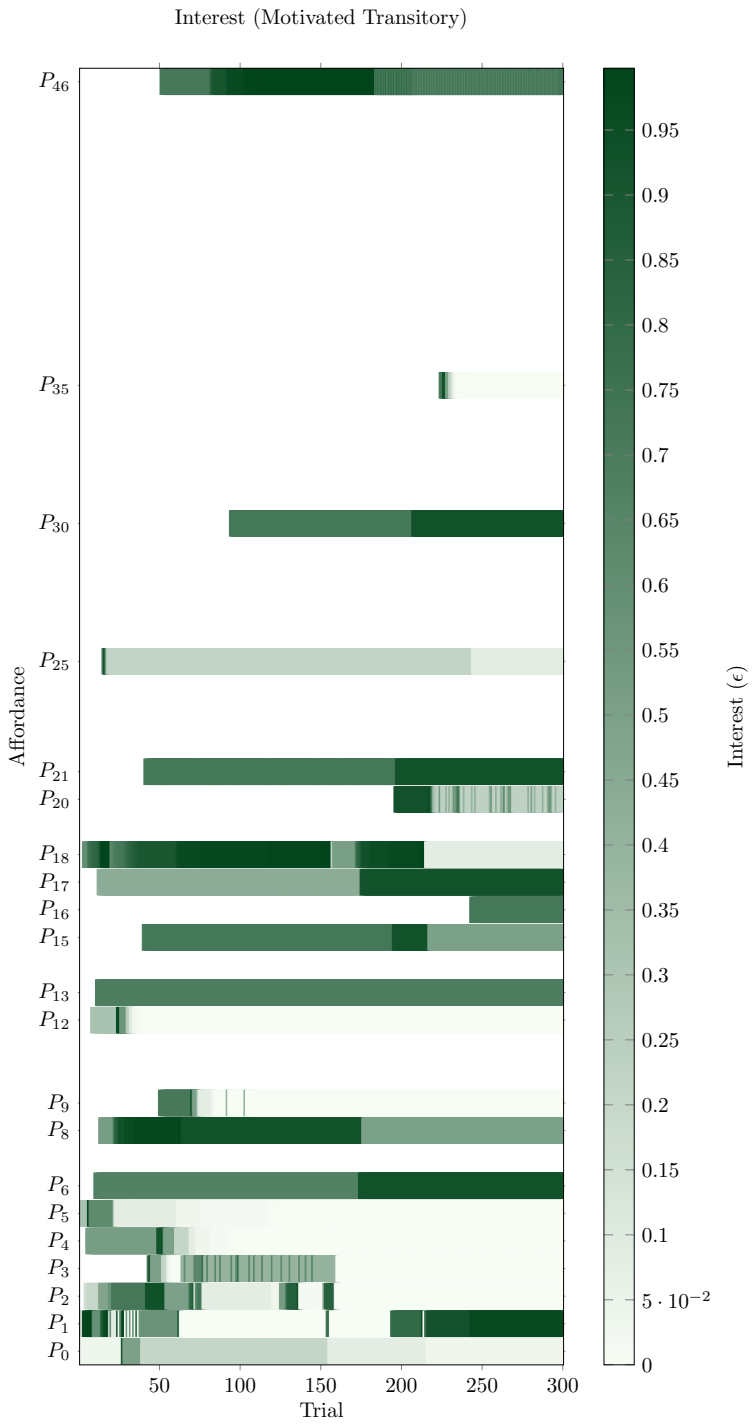


Figure B.14: The interest for each affordance in a single set run using motivated transitory exploration of an isosceles triangle with ground and wall constraints.

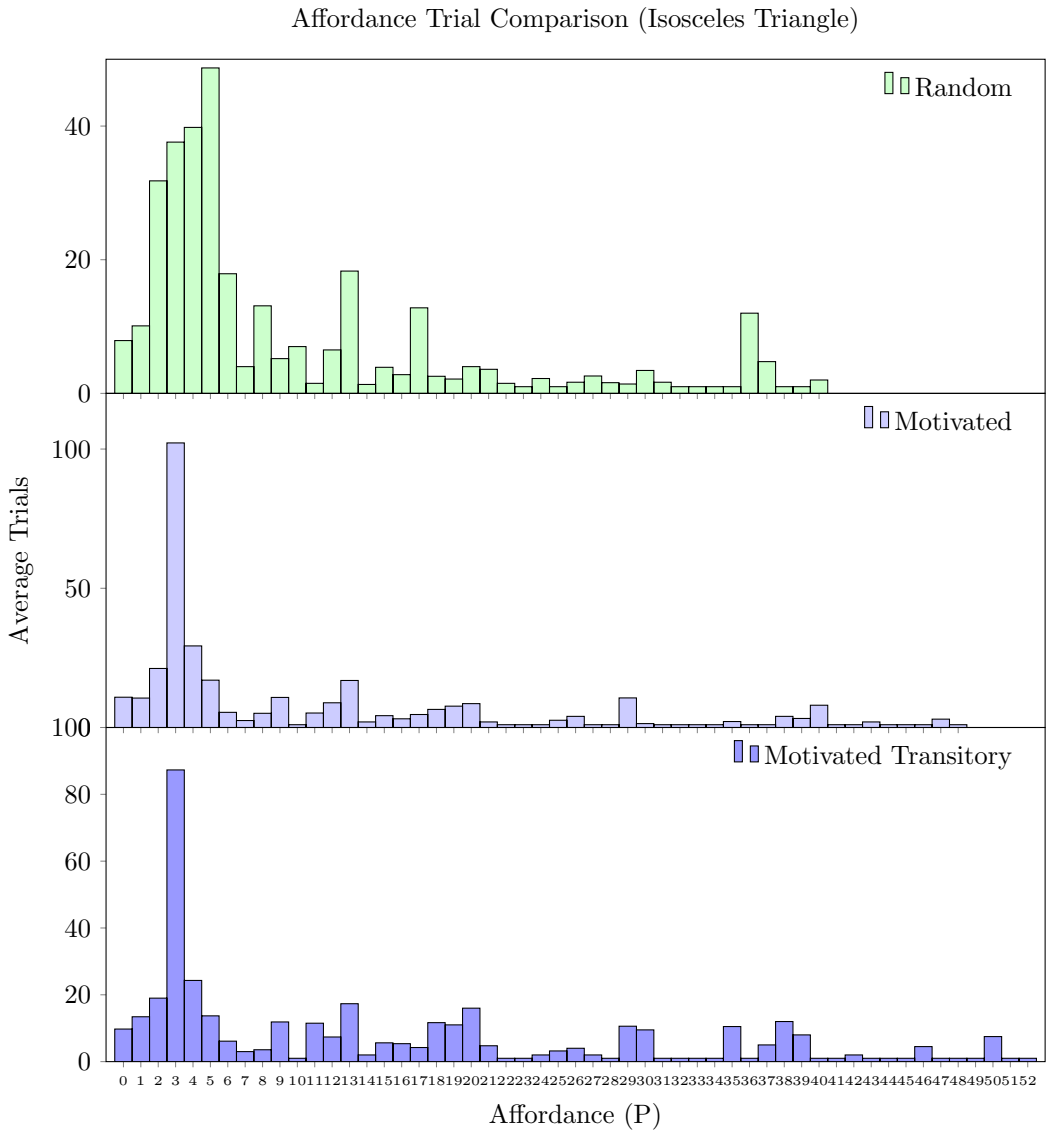


Figure B.15: Comparing the average number of trials for each affordance in each exploration mode of an isosceles triangle with ground and wall constraints.

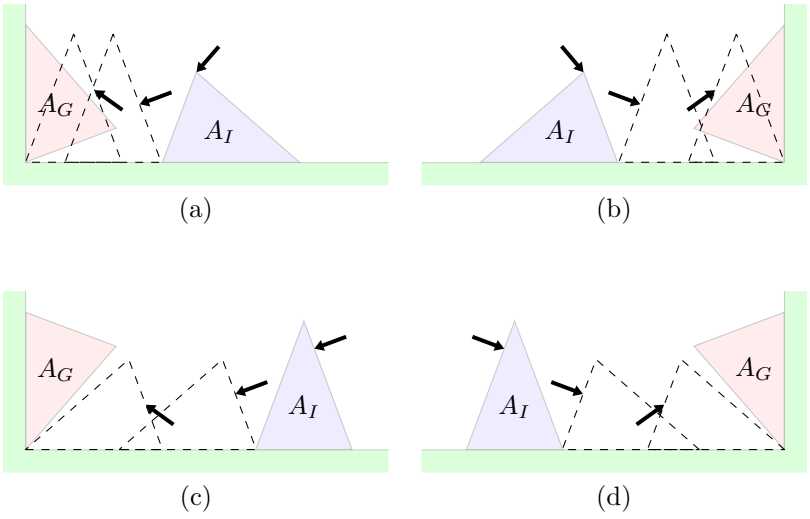


Figure B.16: Various goals and generated plans to complete them using learned affordances of an isosceles triangle with ground and wall constraints. A_I is the initial position and A_G is the goal position. The dotted frames indicated intermediary positions in the plan where a different affordance is performed.

B.2 Rectangle Affordance Results

This section contains results for the rectangle simulations not covered in Section 6.2.5.

B.2.1 Ground and Walls Environment

See Section 6.2.5 for comparison of trials and generated plans.

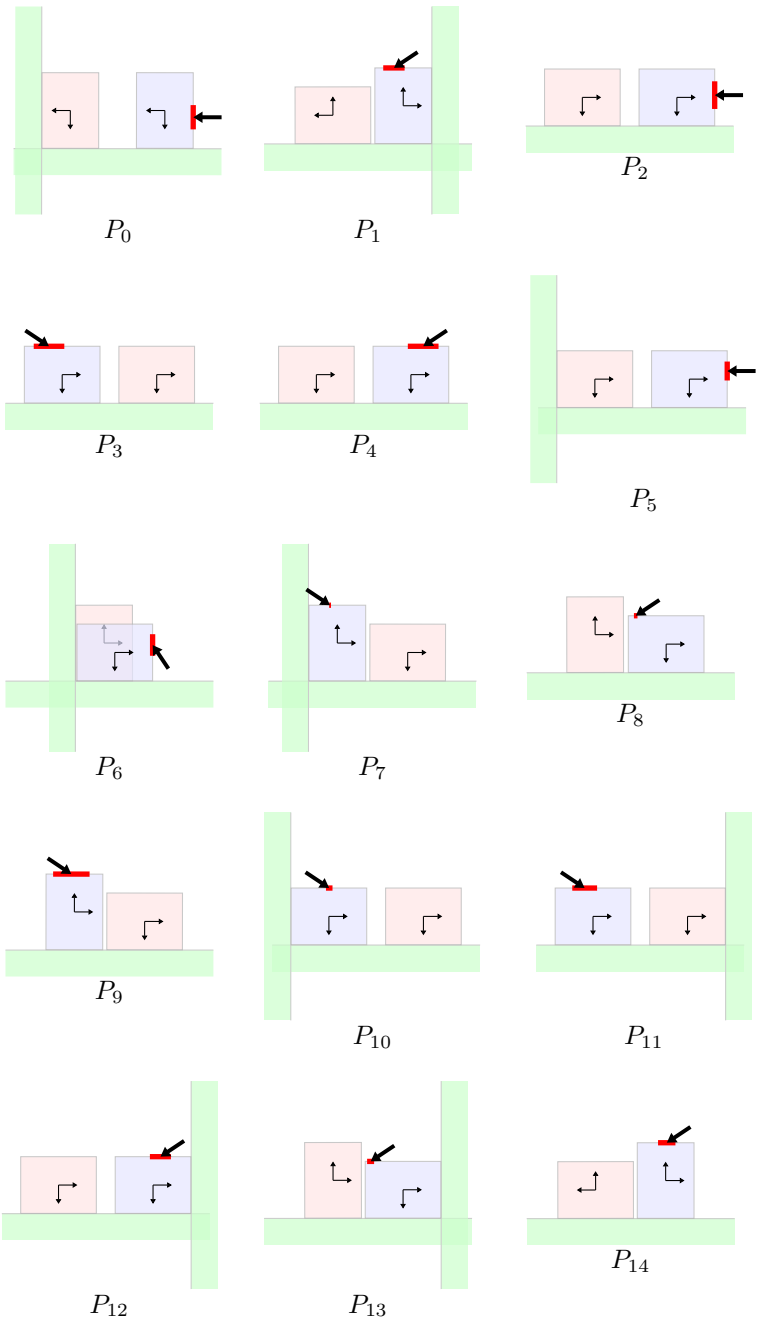


Figure B.17: Learned affordances for the rectangle ground and walls trials. One standard deviation of the interaction point range on the face is drawn in red either side from the interaction point.

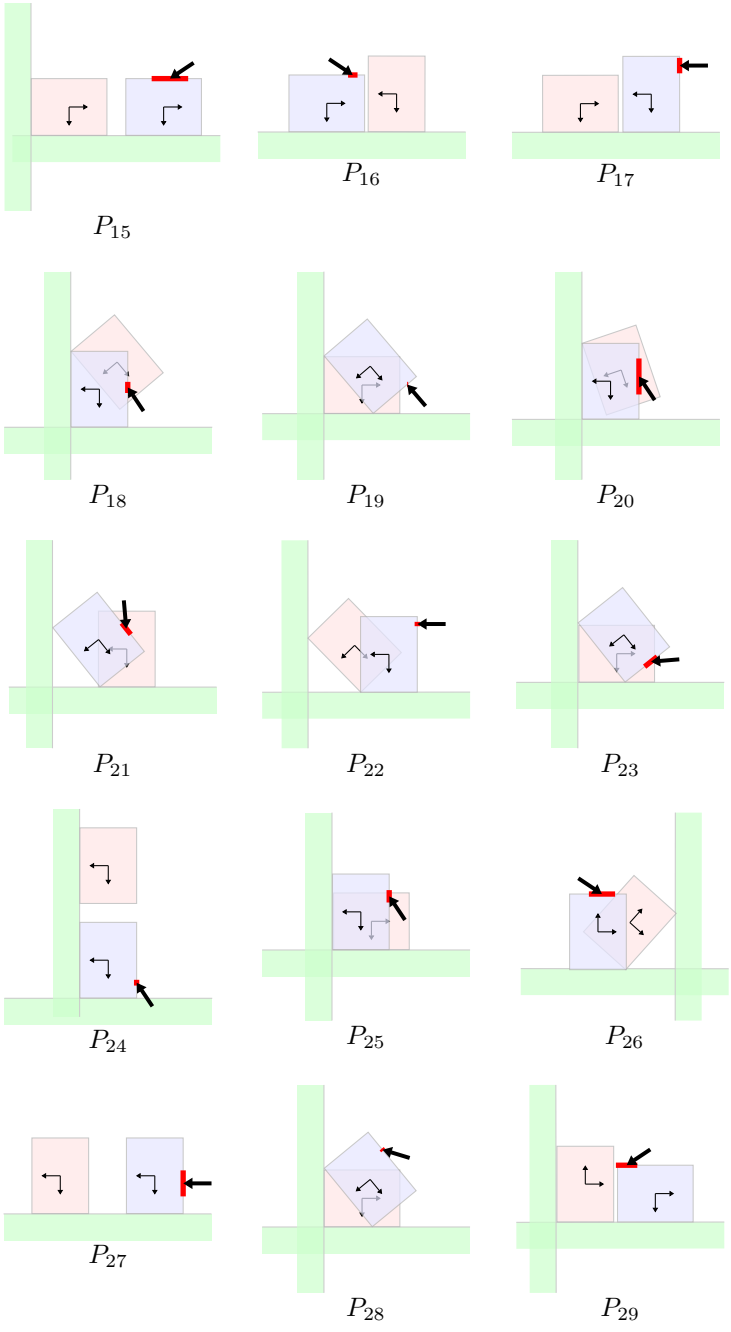


Figure B.17: (cont.) Learned affordances for the rectangle ground and walls trials.

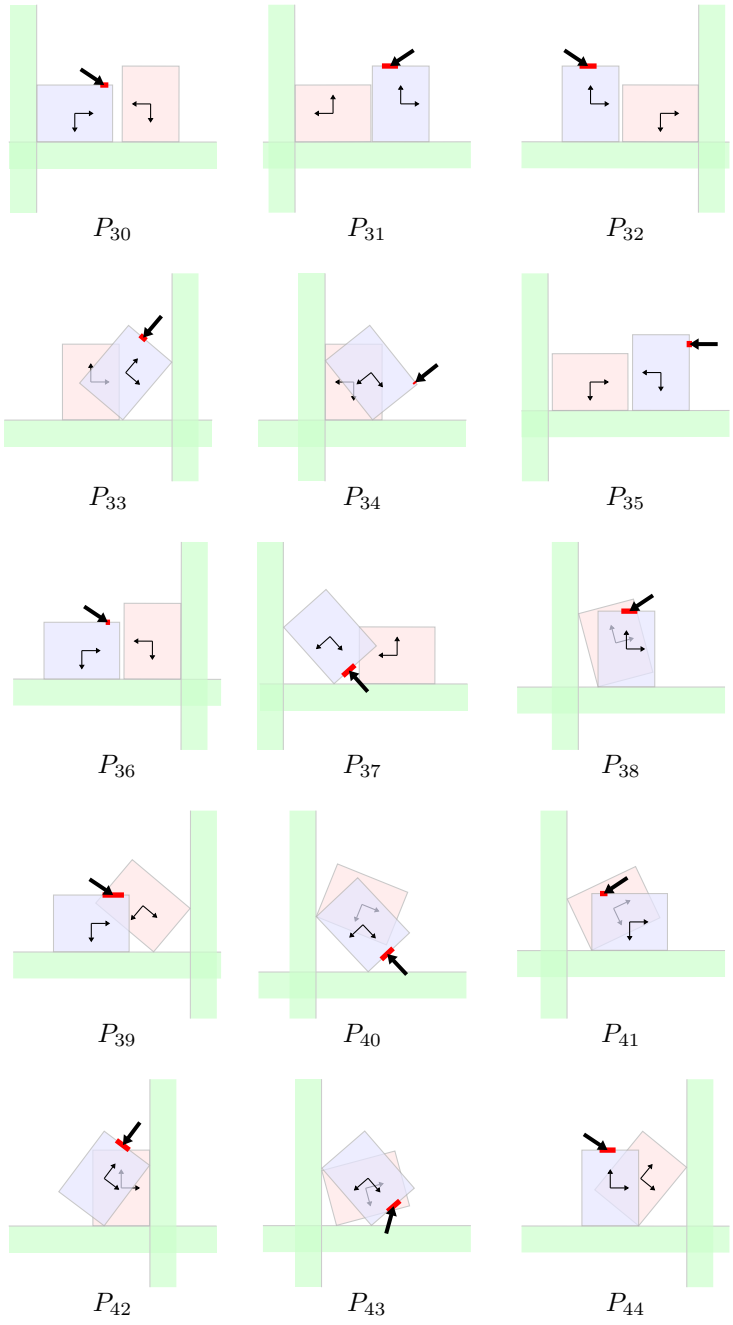


Figure B.17: (cont.) Learned affordances for the rectangle ground and walls trials.

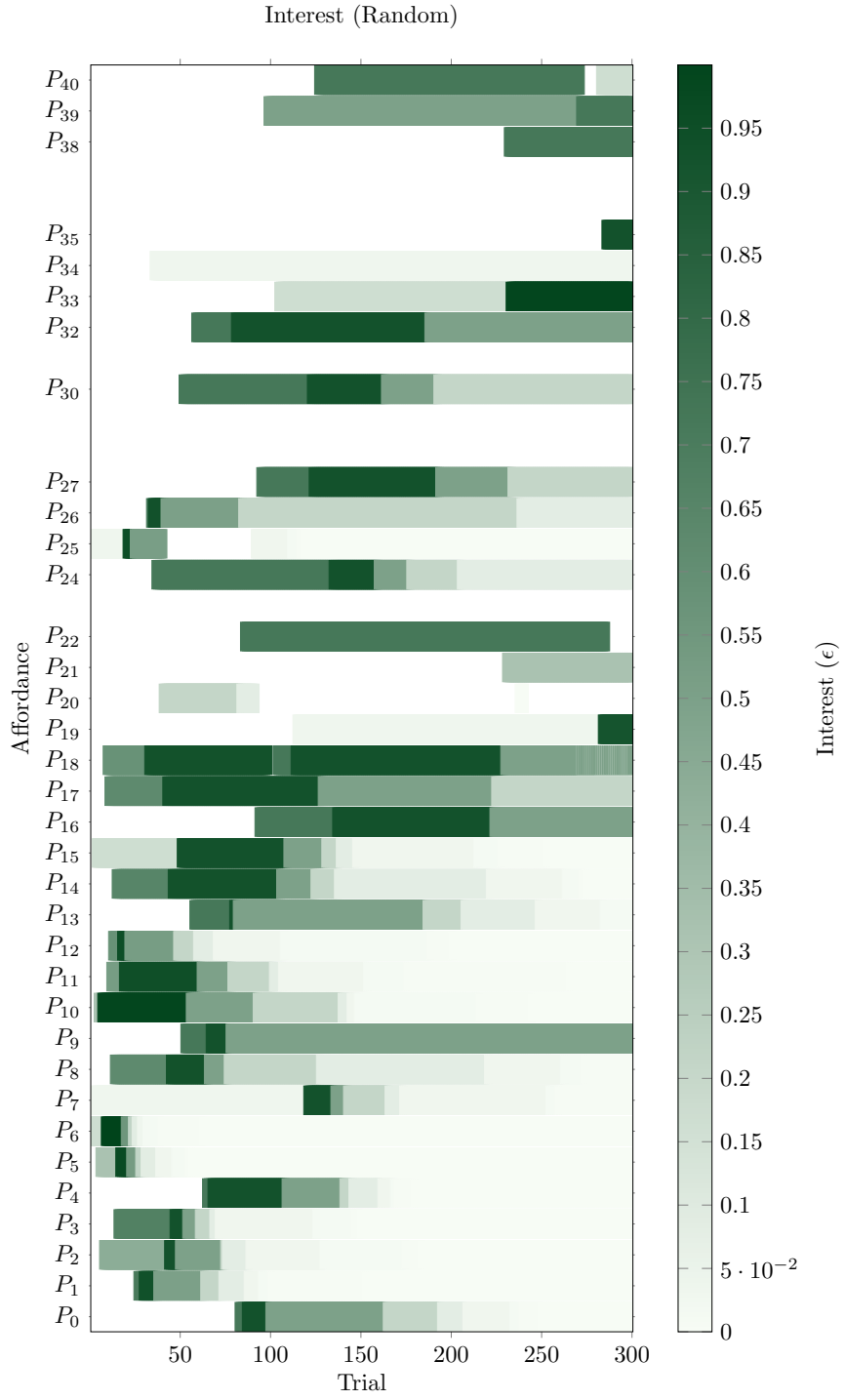


Figure B.18: The interest for each affordance in a single set run using random exploration of a rectangle with ground and wall constraints.

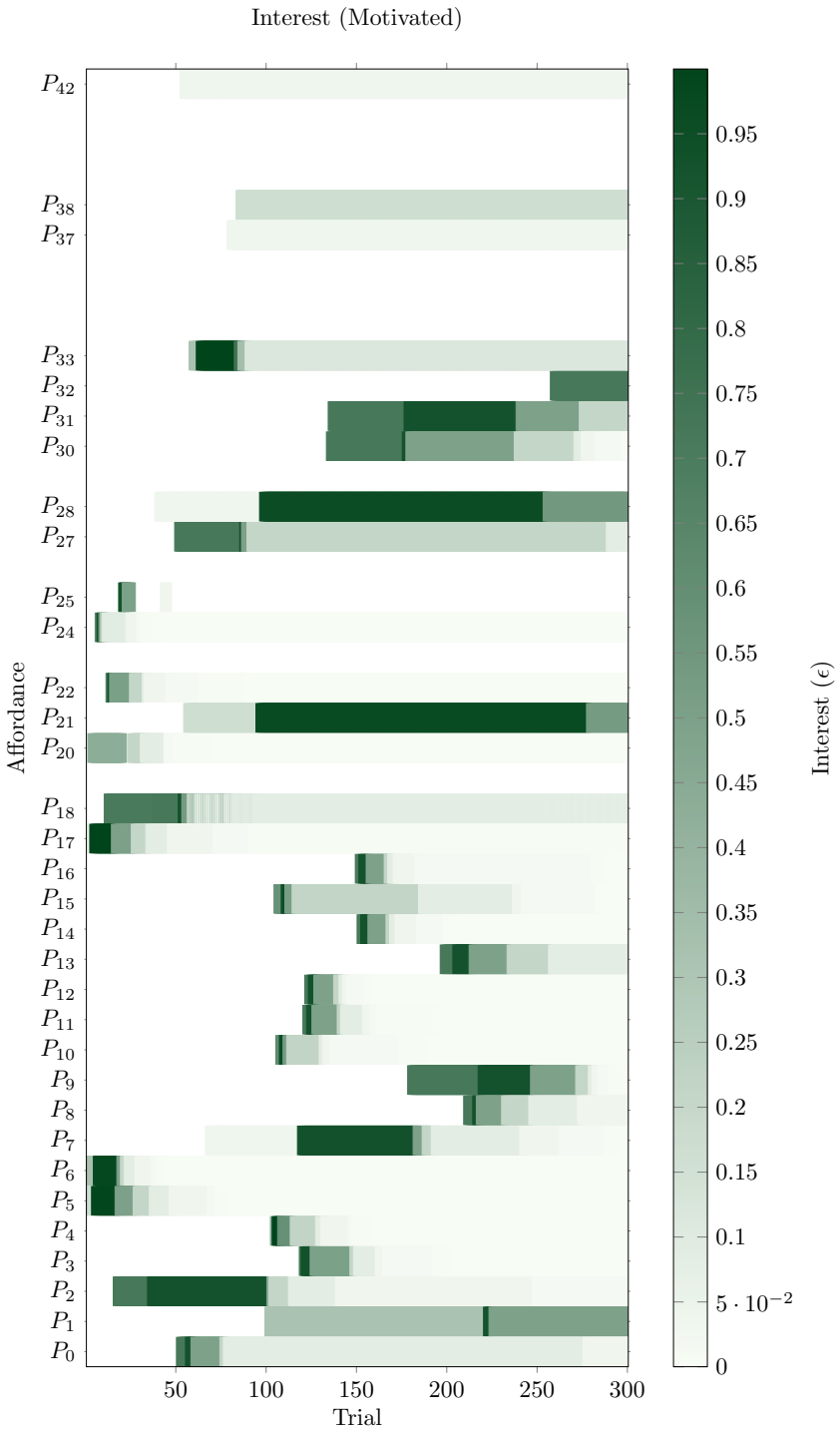


Figure B.19: The interest for each affordance in a single set run using motivated exploration of a rectangle with ground and wall constraints.

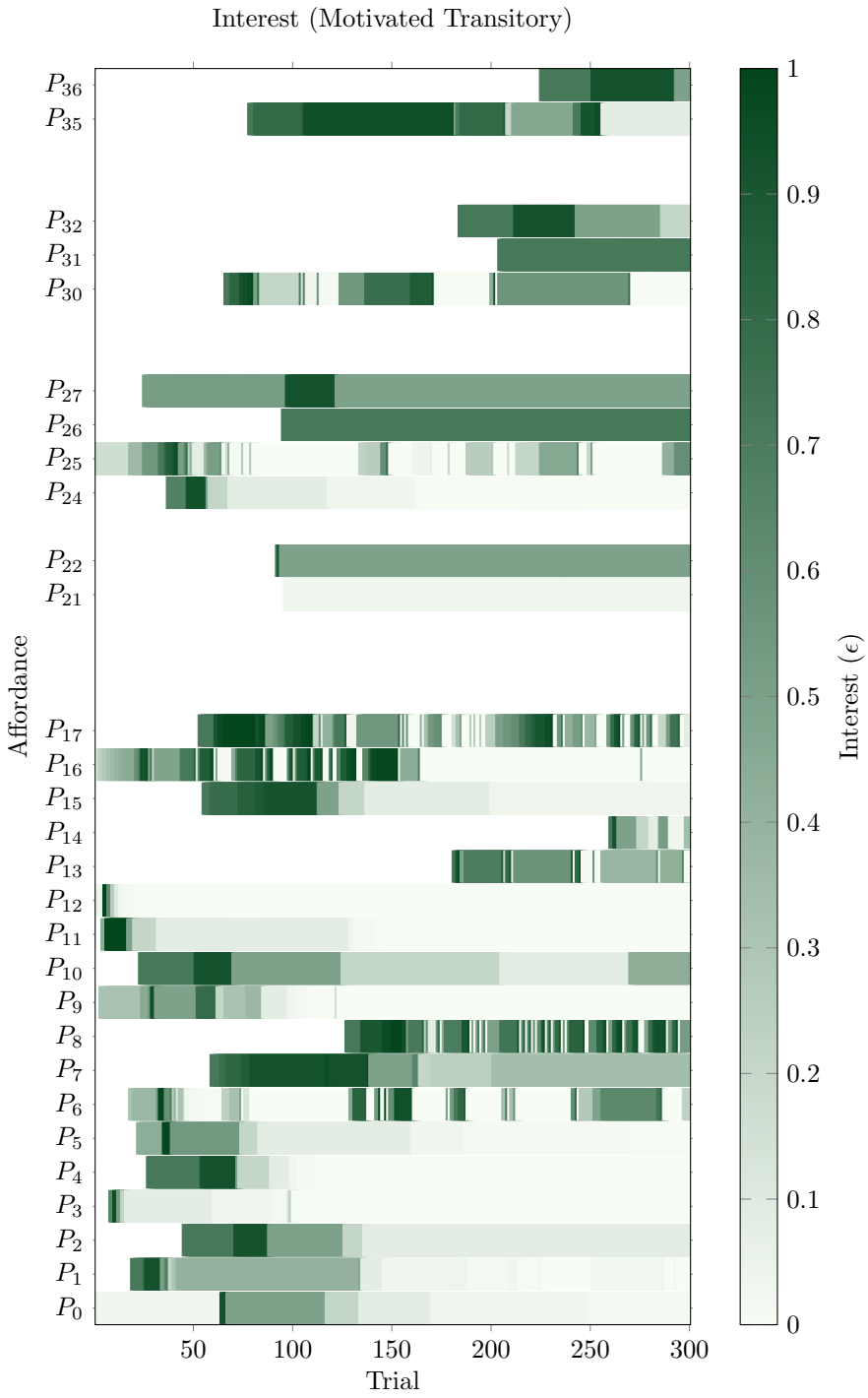


Figure B.20: The interest for each affordance in a single set run using motivated transitory exploration of a rectangle with ground and wall constraints.

B.3 Affordance Tables

The following tables show the data which forms an affordance. The tables show; the transformation descriptor position, $\hat{T}_x$ and $\hat{T}_y$ and the rotation $\hat{T}_\alpha$, the transformation magnitude T^d, the face f the affordance is linked to (symmetrical faces are referenced instead of actual faces), the initial orientation I_α, the force orientation F, the mean interaction point X_μ, the variance of the observed interaction points X_σ, the average number of trials $Trials$, the orientation of gravity relative to the face G, the initial orientation of any constraints in contact C^I, finally the final orientation of any constraints in contact C^F.

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C^I	C^F
P_0	0.351123	0.936329	-1.57556E-008	0.18915	2	110.556	90	0.34365	0.0140143	16.8	159.444	-110.556	-110.556
P_1	0.0167701	0.0684268	0.997515	0.674499	2	110.556	90	0.794798	0.00930445	16.3	159.444	-110.556	138.888
P_2	0.351124	0.936329	0	0.18915	1	110.556	90	0.535917	0.0547152	25.1	159.444	-110.556	-110.556
P_3	0.753843	0.657054	0	0.267741	0	138.924	146.31	0.480917	0.0372637	25.2	131.076	-138.924	-138.924
P_4	0.0645089	0.0452574	0.99689	0.674921	0	138.888	146.31	0.924379	0.00243963	12.5	131.112	-138.888	110.556
P_5	0.057816	0.039409	0.997549	0.674476	0	138.888	90	0.9864	0.00016465	4.1	131.112	-138.888	110.556

Table B.1: Isosceles triangle ground environment simulation affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C^I	C^F
P_0	-4.48039e-008	1	-7.75279e-009	0.311596	1	90	90	0.384298	0.0203177	6.8	-180	-90	-90
P_1	0.0678951	0.00907042	0.997651	0.548642	0	-179.939	146.31	0.43705	0.0593247	6.2	89.9393	179.939	90
P_2	5.20509e-007	1	0	0.39646	0	90	90	0.609417	0.0605791	18.6	180	-90	-90
P_3	-1	-0.000396	0	0.202684	1	179.977	33.6901	0.62425	0.0505983	19.4	90.0227	-179.977	-179.977
P_4	1	2.03852e-006	0	0.202811	1	180	146.31	0.36305	0.0512712	18.2	90.0001	-180	-180
P_5	0.0673936	-0.00910757	0.997685	0.548995	1	180	146.31	0.921082	0.00233358	9.22222	90	-180	89.9998
P_6	-0.0827588	-0.00909702	-0.996528	0.549631	1	180	33.6901	0.127869	0.00296804	10.3	90	-180	-90
P_7	-0.0677807	0.00917165	-0.997658	0.54926	0	-179.959	33.6901	0.378735	0.0433463	6.8	89.9587	179.959	-90
P_8	-0.00910337	0.068078	0.997638	0.548982	1	90.0061	90	0.857258	0.00597249	6	179.994	-90.0061	180

Table B.2: Rectangle ground environment simulation affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C_1^I	C_1^F	C_2^I	C_2^F
P_0	0.351633	0.936138	-0.000597405	0.339185	0	110.589	90	0.287781	0.0135945	9.75	159.411	-110.589	-110.556	-	-
P_1	0.016741	0.068349	0.997521	0.674495	0	110.556	90	0.815747	0.00697565	13.4286	159.444	-110.556	138.888	-	-
P_2	0.752904	0.658104	-0.00597246	0.00773235	0	138.888	146.31	0.46903	0.0114402	19	131.112	-138.888	-138.88	-	131.12
P_3	0.080581	-0.014358	0.996645	0.125918	0	138.888	146.31	0.531663	0.0440696	87.3	131.112	-138.888	-159.509	131.112	110.491
P_4	0.351177	0.936309	-0.000133147	0.00561941	1	110.556	90	0.653742	0.0227477	24.3	159.444	-110.556	-110.556	-20.5564	-
P_5	0.351125	0.936328	-1.91416E-006	0.195441	1	110.556	90	0.52145	0.0401623	13.7	159.444	-110.556	-110.556	-	-
P_6	0.351159	0.936316	-0.000086998	0.0100337	1	110.556	90	0.486834	0.0301473	6.125	159.444	-110.556	-110.556	-	159.444
P_7	0.08662	-0.011977	0.996169	0.09914	0	138.888	146.31	0.8675	0.00844861	3	131.112	-138.888	-155.116	-48.8876	-
P_8	0.753426	0.657533	0.000003	0.277573	0	138.888	146.31	0.509259	0.0188156	3.55556	131.112	-138.888	-138.888	-	-
P_9	0.178032	0.0641271	0.981933	0.0408001	0	138.888	146.31	0.834198	0.014678	11.875	131.112	-138.888	-145.471	-	124.529
P_{10}	0.0670005	0.0304999	0.997287	0.55579	0	138.888	146.31	0.97	0.02	1	131.112	-138.888	130.034	-	40.0342
P_{11}	0.0781765	-0.0161505	0.996809	0.128114	0	138.888	90	0.995833	0.000123611	11.5	131.112	-138.888	-159.872	131.112	110.128
P_{12}	0.061248	0.042408	0.997221	0.674697	0	138.888	146.31	0.921466	0.00468457	7.375	131.112	-138.888	110.556	-	-
P_{13}	0.0963974	0.046885	0.994238	0.299245	1	110.556	146.31	0.561611	0.0125762	17.3333	159.444	-110.556	-159.444	159.444	110.556
P_{14}	0.058245	0.039781	0.997509	0.674503	0	138.888	90	0.985	0.000225	2	131.112	-138.888	110.556	-48.8879	-
P_{15}	0.351148	0.93632	-5.45534E-005	0.00561956	0	110.556	90	0.30974	0.0120428	5.625	159.444	-110.556	-110.556	-	159.444
P_{16}	0.35116	0.936316	-0.000073	0.00562	0	110.556	90	0.425909	0.00913928	5.375	159.444	-110.556	-110.556	-20.5562	-
P_{17}	0.753422	0.657537	6.20963E-006	0.307924	0	138.888	146.31	0.469146	0.0178231	4.22222	131.112	-138.888	-138.888	-48.8876	-
P_{18}	0.0578167	0.0394097	0.997549	0.674476	0	138.888	90	0.99754	0.00672553	11.6667	131.112	-138.888	110.556	-	-
P_{19}	0.0963354	0.0465351	0.994261	0.299238	1	110.556	90	0.997273	7.43802E-005	11	159.444	-110.556	-159.444	159.444	110.556
P_{20}	0.0696733	0.0436221	0.996616	0.125525	0	110.556	90	0.716567	0.0138681	16	159.444	-110.556	138.888	159.444	-131.112
P_{21}	0.0168175	0.0684967	0.99751	0.674301	0	110.589	90	0.7675	0.0111365	4.75	159.411	-110.589	138.888	-20.5891	-
P_{22}	-0.002607	0.04574	0.99895	0.276474	0	87.4259	90	0.77	0.02	1	-177.426	-87.4259	137.193	-177.426	-132.807
P_{23}	0.060926	0.042127	0.997253	0.674677	0	138.888	146.31	0.93	0.02	1	131.112	-138.888	110.556	-48.8877	-
P_{24}	0.321498	0.759296	0.565781	0.00694065	0	110.556	90	0.59	0.02	2	159.444	-110.556	-111.201	-20.5559	-
P_{25}	0.067996	0.043478	0.996738	0.142687	0	110.556	90	0.819686	0.0119638	3.2	159.444	-110.556	-133.925	-	136.075
P_{26}	0.078433	-0.02071	0.996704	0.075912	0	138.888	90	0.9625	0.00001875	4	131.112	-138.888	-151.32	-48.8878	-61.3203
P_{27}	0.0791097	-0.0171255	0.996719	0.108431	0	138.888	90	0.995	0.000025	2	131.112	-138.888	-156.646	-48.8876	-
P_{28}	0.0781582	-0.021738	0.996704	0.0666272	0	138.888	90	0.96	0.02	1	131.112	-138.888	-149.8	-	-
P_{29}	0.071221	0.047999	0.996305	0.125565	0	110.556	146.31	0.368371	0.0240542	10.6	159.444	-110.556	138.888	159.444	-131.112
P_{30}	0.0798068	-0.011062	0.996749	0.161523	0	138.888	90	0.999167	0.0100153	9.5	131.112	-138.888	-165.343	-	104.657
P_{31}	-0.071396	0.024872	0.997138	0.526277	0	24.3274	90	0.67	0.02	1	-114.327	-24.3275	-110.556	65.6725	-
P_{32}	-0.028354	0.05664	-0.997992	0.136162	0	161.217	90	0.43	0.02	1	108.783	-161.217	-138.888	-71.2167	-48.8879
P_{33}	0.035237	0.06545	0.997233	0.496101	0	110.556	90	0.69	0.02	1	159.444	-110.556	168.152	-	78.1517
P_{34}	-0.009357	0.010056	0.999906	0.441486	1	61.3555	146.31	0.32	0.02	1	-151.355	-61.3555	-133.892	-151.355	136.108
P_{35}	0.067534	0.043751	0.996757	0.145733	0	110.556	90	0.781389	0.010578	10.5	159.444	-110.556	-134.425	-20.5559	-
P_{36}	0.107012	0.009072	0.994216	0.060053	1	110.556	90	0.92	0.02	1	159.444	-110.556	-120.367	159.444	149.633
P_{37}	0.061825	0.03452	0.99749	0.615837	0	138.919	90	1	0.02	5	131.081	-138.919	120.142	-	30.1424
P_{38}	0.0171732	0.0694985	0.997434	0.674553	0	110.556	146.31	0.575	0.0208417	12	159.444	-110.556	138.888	-20.5561	-
P_{39}	0.066619	0.045082	0.99676	0.161653	0	110.556	146.31	0.550091	0.0371136	8	159.444	-110.556	-137.032	-20.5559	-
P_{40}	-0.019045	0.039809	0.999026	0.345159	0	74.4198	146.31	0.46	0.02	1	-164.42	-74.4198	138.92	-164.42	-131.08
P_{41}	0.039312	0.066566	0.997007	0.476737	0	143.011	146.31	0.61	0.02	1	126.989	-143.011	138.888	-53.0106	-
P_{42}	-0.077629	0.017908	-0.996821	0.423231	0	179.879	33.6901	0.75	0.0025	2	90.121	-179.879	-110.556	90.121	-
P_{43}	-0.064701	0.038853	0.997148	0.444549	0	41.7858	90	0.77	0.02	1	-131.786	-41.7858	-114.624	48.2142	-
P_{44}	-0.036623	0.064203	0.997265	0.076873	0	20.6834	90	0.83	0.02	1	-110.683	-20.6834	-33.2805	69.3166	-

Table B.3: Isosceles triangle ground and walls environment simulation affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C_1^I	C_1^F	C_2^I	C_2^F
P_{45}	-0.072117	0.016261	0.997264	0.423494	1	41.1594	90	0.79	0.02	1	-131.159	-41.1594	-110.556	48.8406	-
P_{46}	0.440039	0.303704	0.845062	0.0145643	0	138.888	146.31	0.711288	0.0081623	4.5	131.112	-138.888	-140.91	-	129.09
P_{47}	0.072139	0.025363	0.997072	0.496182	0	168.152	90	0.95	0.02	1	101.848	-168.152	110.556	-78.1517	-
P_{48}	0.105004	0.00719383	0.994446	0.123164	0	138.888	146.31	0.9	0.02	1	131.112	-138.888	-159.013	-	-
P_{49}	0.041611	0.059114	0.997384	0.383876	0	158.2	90	0.78	0.02	1	111.8	-158.2	138.888	-68.1999	-
P_{50}	0.0644214	0.0322803	0.997401	0.58264	0	138.888	146.31	0.967308	0.0115355	7.5	131.112	-138.888	125.624	-	35.6235
P_{51}	-0.0198961	0.038424	0.999063	0.354093	0	72.9832	90	0.63	0.02	1	-162.983	-72.9832	138.888	-162.983	-131.112
P_{52}	0.68848	0.556799	0.464727	0.00801789	0	138.854	146.31	0.62	0.02	1	131.146	-138.854	-139.467	-48.8543	-

Table B.3: (cont.) Isosceles triangle ground and walls environment simulation affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C_1^I	C_1^F	C_2^I	C_2^F
P_0	-0.00108329	0.999999	0.00127313	0.296915	1	89.9379	90	0.411501	0.0187255	9.7	-179.938	-89.9379	-90	-	180
P_1	0.0677581	0.0091771	0.99766	0.549007	0	-180	146.31	0.664234	0.0395617	9.2	89.9999	180	90	-90.0001	-
P_2	0.000012	1	0.000195	0.438062	0	90	90	0.539826	0.0249156	8	180	-90	-90.0141	-	-
P_3	-1	0.000001	0	0.202811	1	-180	33.6901	0.674139	0.0360959	15.9	90	180	180	-	-
P_4	1	3.57746E-008	0	0.202811	1	180	146.31	0.337617	0.0395706	11.1	90	-180	-180	-	-
P_5	-0.000426725	0.999998	0.00187475	0.0794648	0	89.9756	90	0.643127	0.0336126	12.1	-179.976	-89.9756	-90	-	180
P_6	0.00912312	0.0103135	0.999905	0.547694	0	90.0133	146.31	0.631635	0.0313114	18.3	179.987	-90.0133	180	179.987	90
P_7	-0.0678189	0.00910761	-0.997656	0.549011	0	-180	33.6901	0.627467	0.00878979	3	89.9998	180	-90	89.9998	-
P_8	0.0684357	-0.00910674	0.997614	0.549034	1	180	146.31	0.902542	0.00179989	28	90.0001	-180	89.9999	-	-
P_9	-0.0677792	0.00905072	-0.997659	0.548437	0	179.906	33.6901	0.554501	0.0405982	14.9	90.0936	-179.906	-90	-	-
P_{10}	-1	-3.51316E-005	-0.000623443	0.370705	1	-180	33.6901	0.495474	0.0222075	8.5	90	180	-179.962	90	-
P_{11}	-1	-8.88831E-006	0	0.0449925	1	-180	33.6901	0.610144	0.0563391	15.7	90	180	180	-	-90
P_{12}	1	-0.000002	0	0.20281	1	-180	146.31	0.40277	0.0223188	15.6	89.9999	180	180	-90.0001	-
P_{13}	0.0674596	-0.00910675	0.99768	0.548996	1	-180	146.31	0.929298	0.00125545	16.1429	90	180	90	-90	-
P_{14}	0.0674705	0.00910895	0.99768	0.549002	0	179.999	146.31	0.480829	0.0483625	12	90.001	-179.999	90	-	-
P_{15}	0.999998	0.000184675	0.00199529	0.0323323	1	179.989	146.31	0.417002	0.0307465	6.5	90.0106	-179.989	180	-	90
P_{16}	-0.0672798	-0.00911178	-0.997693	0.549058	1	-180	33.6901	0.155985	0.00153762	19.7	90	180	-89.9888	-	-
P_{17}	-0.00907439	0.0682867	0.997624	0.548702	1	90.0532	90	0.877642	0.00583046	26.4	179.947	-90.0532	-180	-	-
P_{18}	0.0586665	-0.0384565	0.997537	0.244185	1	90	146.31	0.522886	0.00263866	8.11111	-180	-90	139.975	-180	-
P_{19}	-0.0325093	-0.0156365	0.999349	0.301521	0	40.4873	90	0.785556	0.00104074	2.66667	-130.487	-40.4873	-90	-130.487	180
P_{20}	0.051814	-0.049348	0.997437	0.114019	1	90	146.31	0.562556	0.010606	10.6667	-180	-89.9999	-108.687	-180	161.313
P_{21}	-0.0561032	-0.0363313	-0.997764	0.235332	1	128.582	33.6901	0.383844	0.0115555	3.42857	141.418	-128.582	-90	141.418	-
P_{22}	0.0177202	0.0670909	0.99759	0.270293	1	89.9998	90	0.901875	0.00983785	2	-180	-89.9998	-134.306	-	135.694
P_{23}	-0.0291845	-0.0174348	0.999422	0.314298	0	38.3853	33.6901	0.58	0.02	1	-128.385	-38.3853	-89.9999	-128.385	-180
P_{24}	1	-0.000001	0	0.043474	1	90	146.31	0.202514	0.00369458	5.5	180	-90.0001	180	180	-
P_{25}	-0.00912786	-0.0102058	0.999906	0.547774	1	90	146.31	0.705575	0.0274283	11.7143	180	-90	-180	180	90
P_{26}	-0.066117	-0.01922	-0.997627	0.255248	0	-180	33.6901	0.429028	0.0402804	7.66667	89.9999	180	-138.158	-	-48.158
P_{27}	0.000000151	1	-2.42067E-007	0.343436	1	90	90	0.399419	0.0110445	4.6	180	-90	-90	-	-
P_{28}	-0.0161157	0.0313791	0.999378	0.30819	1	129.391	146.31	0.701667	0.00918056	2.25	140.609	-129.391	180	140.609	90
P_{29}	0.0675391	-0.00915804	0.997675	0.548466	1	-180	146.31	0.88	0.0151167	1.5	90	180	90.0875	-	0.087471
P_{30}	-0.0730504	-0.00910315	-0.997287	0.549214	1	-180	33.6901	0.109534	0.00560937	7.16667	90	180	-89.9998	90	-
P_{31}	0.065087	0.009242	0.997837	0.549445	0	179.912	146.31	0.69	0.02	1	90.0878	-179.912	90.0001	-	5.72958E-005
P_{32}	-0.0673725	0.00910838	-0.997686	0.548997	0	-179.999	33.6901	0.543333	0.0132356	3	89.9994	179.999	-89.9999	-	0.000119532
P_{33}	0.044148	0.063859	0.996982	0.246032	0	139.678	90	0.663889	0.00251759	4.66667	130.322	-139.678	-179.983	-49.6776	-
P_{34}	0.00791362	0.0439495	-0.999002	0.23504	1	128.582	90	0.0366667	0.000188889	6	141.418	-128.582	-89.9999	141.418	-180
P_{35}	-0.00920383	0.0681418	0.997633	0.5494	1	89.9381	90	0.876706	0.00733237	5.5	-179.938	-89.9381	180	-	90
P_{36}	-0.0679859	-0.00911772	-0.997645	0.549188	1	-180	33.6901	0.155625	0.00437613	3.5	90	180	-89.9718	-	0.0281978
P_{37}	0.024778	-0.03865	-0.998946	0.80254	0	41.7398	90	0.35	0.02	1	-131.74	-41.7398	89.9919	-131.74	-
P_{38}	0.0622755	-0.0351628	0.997439	0.0905504	0	179.997	146.31	0.448333	0.0167389	1.5	90.0026	-179.997	165.162	-	75.1617
P_{39}	-0.111294	-0.0381047	-0.993057	0.245393	1	180	33.6901	0.21	0.02	1	90.0001	-180	-139.958	-	-49.9577
P_{40}	0.046399	0.054203	0.997451	0.150949	0	43.2439	90	0.46	0.02	1	-133.244	-43.2439	-157.984	-133.244	-
P_{41}	0.054789	-0.046151	0.997431	0.155027	1	-180	146.31	0.8425	0.00241875	4	89.9997	180	154.592	-	64.5919
P_{42}	-0.04545	-0.006307	0.998947	0.224762	0	143.107	90	0.59	0.02	1	126.893	-143.107	-180	-53.1068	-90
P_{43}	-0.011335	-0.008139	0.999903	0.389529	0	40.9306	146.31	0.52	0.02	1	-130.931	-40.9306	-104.93	-130.931	165.07
P_{44}	-0.0670412	-0.0207275	-0.997535	0.239837	0	-179.965	33.6901	0.546833	0.0405865	9.6	89.9647	179.965	-140.723	-	-50.7231
P_{45}	-0.554196	-0.0453612	-0.831149	0.0544279	1	180	33.6901	0.22	0.02	1	90	-180	-172.567	-	-82.5667

Table B.4: Rectangle ground and walls environment simulation affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C_1^I	C_1^F	C_2^I	C_2^F
P_0	0.0603559	0.0386174	0.99743	0.0830259	1	107.281	90	0.685	0.0332917	6	162.719	-107.281	-128.864	-17.2814	-38.8641
P_1	0.059959	0.0324418	0.997674	0.0824415	1	108.323	33.6901	0.501667	0.0405806	6	161.677	-108.323	-129.871	-18.3227	-39.8712
P_2	0.323641	0.94309	-0.0764023	0.036837	1	108.452	90	0.866	0.066584	5	161.548	-108.452	-108.291	-	-
P_3	0.31983	0.947349	-0.0154455	0.0827188	1	108.553	90	0.356667	0.00752222	6	161.447	-108.553	-106.298	-	-16.298
P_4	0.0981169	0.157905	0.982568	0.0321773	1	108.347	90	0.993333	8.88889E-005	3	161.653	-108.347	-115.832	-	-
P_5	0.208904	0.514921	0.831394	0.0979778	1	108.188	90	0.763333	0.0146889	3	161.812	-108.188	-129.138	-	-39.1383
P_6	0.0149909	0.075294	0.997049	0.457878	1	108.274	90	0.99	0.0001	2	161.726	-108.274	142.709	-	-
P_7	0.294017	0.954677	0.0463247	0.0417449	0	106.385	90	0.48	0.0530571	7	163.615	-106.385	-105.498	-	-15.4979
P_8	0.0869803	0.0309442	0.995729	0.210921	0	106.222	33.6901	0.55	0.0187	8	163.778	-106.222	-67.6113	-16.2215	-157.611
P_9	0.696976	0.499784	0.514237	0.0715909	2	141.494	90	0.95	0.02	1	128.506	-141.494	-150.043	-	-60.0428
P_{10}	0.724508	0.463306	0.510329	0.0112033	2	141.029	90	0.87	0.02	1	128.971	-141.029	-142.305	-	-
P_{11}	0.796325	0.604865	-0.00199207	0.0575601	2	142.537	33.6901	0.86	0.02	1	127.463	-142.537	-142.596	-	-
P_{12}	0.114242	0.211442	0.970691	0.0855264	1	106.779	90	0.943333	0.00535556	3	163.221	-106.779	-129.302	-	-39.3021
P_{13}	0.241449	0.961107	-0.134075	0.00700351	0	106.307	90	0.973333	0.000422222	3	163.693	-106.307	-107.769	-	-
P_{14}	0.152989	0.0161694	0.988096	0.0398526	0	107.178	33.6901	0.57	0.02	1	162.822	-107.178	-115.679	-17.178	-25.6792

Table B.5: Isosceles triangle robotic system affordances

$Affordance$	$\hat{T}_x$	$\hat{T}_y$	$\hat{T}_\alpha$	T^d	f	I_α	F	X_μ	X_{σ^2}	$Trials$	G	C_1^I	C_1^F	C_2^I	C_2^F
P_0	-0.00571615	-0.00524066	0.99997	0.544131	0	89.1184	33.6901	0.766667	0.0327889	12	-179.118	-89.1184	-179.833	0.881572	-89.833
P_1	-0.0313846	0.831734	0.554286	0.0101089	0	90.2469	90	0.61	0.0338	3	179.753	-90.2469	-90.2531	-	-0.253059
P_2	0.0370853	-0.0235787	0.999034	0.0749627	1	90.7658	33.6901	0.620714	0.0204066	14	179.234	-90.7658	-11.6087	-0.765795	-101.609
P_3	0.818411	0.0144862	0.57445	0.0368265	1	90.3187	33.6901	0.47	0.02	1	179.681	-90.3187	-0.141133	-0.318694	-
P_4	0.152583	0.934204	0.322461	0.00581313	1	92.0467	90	0.94	0.02	1	177.953	-92.0467	-91.9299	-	-
P_5	0.0180904	0.361862	-0.932056	0.0193655	1	90.0179	90	0.636	0.013264	5	179.982	-90.0179	-91.4775	-	-1.47755
P_6	0.00688357	0.00441339	0.999967	0.531193	1	90.0075	33.6901	0.61	0.02	1	179.992	-90.0075	-178.657	-0.00751905	-88.6574
P_7	0.016476	-0.0279196	0.999474	0.262682	0	89.7019	33.6901	0.98	0.02	1	-179.702	-89.7019	-136.665	0.29813	-46.6651
P_8	-0.0468698	0.989678	-0.135431	0.00743498	0	90.0232	90	0.5625	0.0320187	4	179.977	-90.0232	-89.5384	-	0.461605
P_9	0.0221407	-0.0106531	0.999698	0.363461	1	90.1352	33.6901	0.86	0.005	3	179.865	-90.1352	-143.967	-0.135209	-53.9666
P_{10}	0.0463622	0.997363	0.0558359	0.063881	1	92.3464	90	0.658333	0.0305139	6	177.654	-92.3464	-90.3156	-	-0.315611
P_{11}	0.791215	0.0239385	-0.61107	0.0305311	0	89.9288	33.6901	0.4	0.0036	2	-179.929	-89.9288	0.257885	0.0711923	-
P_{12}	0.0304243	0.877482	0.478643	0.00698882	0	91.9844	90	0.57	0.02	1	178.016	-91.9844	-93.2853	-	-
P_{13}	0.111095	-0.0470252	0.992697	0.117095	1	89.5571	33.6901	0.46	0.000266667	3	-179.557	-89.5571	-24.4357	0.442884	-
P_{14}	-0.948826	-0.00447798	0.315769	0.0243753	1	179.73	146.31	0.4775	0.00106875	4	90.2704	-179.73	-177.35	-	-

Table B.6: Rectangle robotic system affordances

B.4 Plan Tables

This section contains plan data for each move in the plan. The tables show; the position and orientation is given by x, y, α respectively, the face f, force orientation F and position X where interaction is desired, the uncertainty of each move ξ, finally the depth $depth$ in the plan tree the move was generated at.

Plan	*State*	*P*	x	y	α	f	F	X	ξ	*depth*
	A_G	-	0.0516333	0.0201542	-0.0161804	-	-	-	-	-
1	A_I	P_5	0.132616	0.0151539	-90.0112	1	146.31	0.919167	0.124955	1
	A_1	P_0	0.0956175	0.0201613	-0.0109112	3	90	0.38	0.176934	2
	A_G	-	6.92959E-009	0.02015	-5.94584E-008	-	-	-	-	-
2	A_I	P_2	-0.10235	0.01515	90	0	90	0.3875	3.06676E-005	1
	A_1	P_6	-0.0454105	0.0151501	90	3	33.6901	0.137778	0.161494	5
	A_G	-	-0.0335339	0.01515	90	-	-	-	-	-
3	A_I	P_8	0.0573523	0.020159	-0.034439	1	90	0.15625	0.00610607	1
	A_1	P_2	0.0947229	0.0151392	-90.0283	0	90	0.6125	3.06676E-005	2
	A_G	-	-0.0375601	0.01515	90	-	-	-	-	-
4	A_I	P_8	-0.141283	0.02015	180	1	90	0.84375	0.00610607	1
	A_1	P_2	-0.178657	0.0151526	269.994	2	90	0.3875	3.06676E-005	2
	A_G	-	-4.79418E-010	0.02015	1.25238E-006	-	-	-	-	-
5	A_I	P_5	0.000139143	0.01515	-90	1	146.31	0.919167	0.124955	1
	A_1	P_0	-0.0368598	0.0201502	0.000249332	1	90	0.62	0.176934	2
	A_G	-	0.217975	0.01515	-90	-	-	-	-	-
6	A_I	P_8	0.218251	0.02015	3.1202E-006	1	90	0.15625	0.00610607	1
	A_1	P_2	0.255624	0.0151526	-89.9939	0	90	0.6125	3.06676E-005	2

Table B.7: Rectangle simulation system ground environment generated plans

Plan	*State*	*P*	*x*	*y*	*α*	*f*	*F*	*X*	*ξ*	*depth*
	A_G	-	-0.000366693	0.0190105	-0.0116841	-	-	-	-	-
1	A_I	P_3	0.0867155	0.0133917	-110.556	0	146.31	0.478333	3.06676E-005	1
	A_1	P_5	0.0468858	0.0134171	-110.556	0	90	0.986	0.887423	5
	A_G	-	-0.0168707	0.0190062	0.000001512	-	-	-	-	-
2	A_I	P_5	-0.105629	0.0133917	110.556	2	90	0.014	0.887423	1
	A_1	P_0	-0.0587711	0.0190062	-3.42388E-005	0	90	0.656	0.000394889	2
	A_G	-	0.0871076	0.0133917	-110.556	-	-	-	-	-
3	A_I	P_1	-0.0171979	0.0190062	1.65824E-006	0	90	0.185714	0.00810494	1
	A_1	P_2	0.0299891	0.0133916	-110.556	1	90	0.4525	3.06676E-005	2
	A_G	-	-0.141013	0.0133917	110.556	-	-	-	-	-
4	A_I	P_1	-0.0278871	0.0190062	1.27151E-006	2	90	0.814286	0.00810494	1
	A_1	P_2	-0.0750741	0.0133916	110.556	1	90	0.5475	3.06676E-005	2
	A_G	-	-0.0213766	0.0190062	2.46517E-006	-	-	-	-	-
5	A_I	P_5	-0.017523	0.0133917	110.556	2	90	0.014	0.887423	1
	A_1	P_0	0.0293347	0.0190062	-3.42388E-005	2	90	0.344	0.000394889	2
	A_G	-	6.23822E-005	0.0190062	1.98243E-006	-	-	-	-	-
6	A_I	P_5	-0.00174835	0.0133917	-110.556	0	90	0.986	0.887423	1
	A_1	P_0	-0.0486061	0.0190062	4.10689E-005	0	90	0.656	0.000394889	2
	A_G	-	-0.045796	0.0133917	110.556	-	-	-	-	-
7	A_I	P_1	-0.0495225	0.0190189	-0.0344853	2	90	0.814286	0.00810494	1
	A_1	P_3	-0.0967129	0.0134328	110.522	2	33.6901	0.521667	3.06676E-005	2
	A_G	-	0.0149947	0.0133917	-110.556	-	-	-	-	-
8	A_I	P_1	0.0160452	0.0190062	1.29667E-006	0	90	0.185714	0.00810494	1
	A_1	P_3	0.0632321	0.0133916	-110.556	0	146.31	0.478333	3.06676E-005	2

Table B.8: Isosceles triangle simulation robotic system ground environment generated plans

Plan	*State*	*P*	*x*	*y*	α	*f*	*F*	*X*	ξ	*depth*
1	A_G	-	-0.28485	0.0201536	5.25204e-005	-	-	-	-	-
	A_I	P_8	-0.216843	0.01515	-90	1	146.31	0.916667	0.334735	1
	A_1	P_0	-0.252645	0.0201732	0.0903339	3	90	0.406	0.540627	2
2	A_G	-	0.28485	0.0201536	6.03854E-005	-	-	-	-	-
	A_I	P_3	0.192011	0.0151651	90.0427	3	33.6901	0.64	0.274813	1
	A_1	P_{36}	0.24895	0.0152075	90.0427	3	33.6901	0.13	0.542688	5
3	A_G	-	-0.202824	0.0151531	-89.9912	-	-	-	-	-
	A_I	P_{25}	-0.28485	0.02015	1.87821e-006	3	146.31	0.723	0.326098	1
	A_1	P_{10}	-0.279314	0.0151499	89.9999	3	33.6901	0.563333	0.286282	2
	A_2	P_3	-0.205739	0.0151458	89.9999	3	33.6901	0.64	0.274813	3
4	A_G	-	0.194231	0.01515	90	-	-	-	-	-
	A_I	P_1	0.28485	0.02015	2.79206e-006	0	146.31	0.655882	0.440488	1
	A_1	P_4	0.24765	0.0151118	90	3	146.31	0.36625	0.238815	2
5	A_G	-	-0.221894	0.0151503	-90	-	-	-	-	-
	A_I	P_{19}	-0.27537	0.0250836	-138.572	0	90	0.74	0.592701	1
	A_1	P_{10}	-0.279866	0.0151848	-90.1921	1	33.6901	0.563333	0.286282	2
	A_2	P_3	-0.252799	0.0150925	-90.1921	1	33.6901	0.64	0.274813	3
6	A_G	-	0.226048	0.01515	-90	-	-	-	-	-
	A_I	P_{21}	0.275508	0.0251354	-219.514	3	146.31	0.6475	0.109312	1
	A_1	P_{47}	0.260512	0.0203897	-180.931	1	146.31	0.713529	0.36529	2
	A_2	P_3	0.223234	0.0159505	-90.8818	1	33.6901	0.64	0.274813	3
7	A_G	-	-0.275807	0.0251763	37.1448	-	-	-	-	-
	A_I	P_4	-0.179396	0.01515	-90	1	146.31	0.36625	0.238815	1
	A_1	P_8	-0.226378	0.0151502	-90	1	146.31	0.916667	0.334735	4
	A_2	P_{22}	-0.26218	0.0201734	0.0903339	3	90	0.852	0.476767	5
8	A_G	-	0.275294	0.0250373	-42.3123	-	-	-	-	-
	A_I	P_{16}	0.191978	0.01515	-90	1	33.6901	0.654167	0.201495	1
	A_1	P_{15}	0.219046	0.01515	-90	1	33.6901	0.150556	0.705291	2
	A_1	P_{25}	0.256846	0.0201687	-179.928	3	90	0.128571	0.616624	3

Table B.9: Rectangle simulation system walls and ground environment generated plans

Plan	*State*	*P*	*x*	*y*	α	*f*	*F*	*X*	ξ	*depth*
1	A_G	-	-0.286607	0.0253968	20.5584	-	-	-	-	-
	A_I	P_{18}	-0.214546	0.0133941	-110.56	0	90	0.998462	0.832985	1
	A_1	P_{15}	-0.261404	0.019012	-0.00396142	2	90	0.451667	0.449311	2
	A_2	P_{29}	-0.278786	0.0190257	-0.00396142	2	146.31	0.5625	0.724942	3
2	A_G	-	0.286609	0.025292	-20.556	-	-	-	-	-
	A_I	P_{18}	0.201309	0.0133917	110.556	2	90	0.00153846	0.832985	1
	A_1	P_{15}	0.248167	0.0190062	-3.42388E-005	0	90	0.548333	0.449311	2
	A_2	P_{29}	0.278787	0.019006	-3.42388E-005	0	33.6901	0.4375	0.724942	3
3	A_G	-	-0.286609	0.0354794	159.444	-	-	-	-	-
	A_I	P_1	-0.177311	0.0190062	1.85678E-006	2	90	0.785897	0.667379	1
	A_1	P_6	-0.223419	0.0133922	110.557	1	90	0.616667	0.640932	2
	A_2	P_{13}	-0.264688	0.0133994	110.553	1	146.31	0.499167	0.168615	3
4	A_G	-	0.286609	0.0355413	-159.444	-	-	-	-	-
	A_I	P_1	0.181052	0.0190062	2.05229E-006	0	90	0.214103	0.667379	1
	A_1	P_6	0.227161	0.0133922	-110.557	1	90	0.383333	0.640932	2
	A_2	P_{13}	0.264688	0.0133994	-110.553	1	33.6901	0.500833	0.168615	3

Table B.10: Isosceles triangle simulation robotic system walls and ground environment generated plans

Plan	*State*	P	x	y	α	f	F	X	ξ	*depth*
1	A_G	-	0.164	-0.110384	-89.0254	-	-	-	-	-
	A_I	P_{10}	0.109333	-0.113538	0.784713	1	90	0.341667	0.678571	1
	A_1	P_6	0.159655	-0.113538	0.784713	1	146.31	0.39	1.0698	2
2	A_G	-	0.160495	-0.113188	-181.417	-	-	-	-	-
	A_I	P_8	0.105829	-0.110735	-89.2181	2	90	0.4375	0.55857	1
	A_1	P_0	0.163123	-0.110735	-89.2181	2	146.31	0.233333	0.384245	2

Table B.11: Rectangle robotic system generated plans

Plan	*State*	P	x	y	α	f	F	X	ξ	*depth*
1	A_G	-	0.127819	-0.118905	-18.9896	-	-	-	-	-
	A_I	P_4	0.0693837	-0.113801	87.5354	2	90	0.01	0.896008	1
	A_1	P_5	0.104252	-0.118256	-21.1878	0	90	0.0266667	0.116826	2
2	A_G	-	0.151013	-0.11816	-18.9973	-	-	-	-	-
	A_I	P_4	0.0749213	-0.113552	88.218	2	90	0.01	0.896008	1
	A_1	P_6	0.109841	-0.117591	-19.8567	0	90	0.52	0.614697	2
3	A_G	-	0.16592	-0.110294	73.0309	-	-	-	-	-
	A_I	P_3	0.119386	-0.113844	89.4458	2	90	0.643333	0.15315	1
	A_1	P_0	0.16197	-0.113844	89.4458	2	90	0.315	0.178193	2
4	A_G	-	0.164334	-0.10197	-71.473	-	-	-	-	-
	A_I	P_4	0.0977753	-0.113838	89.2866	2	90	0.01	0.896008	1
	A_1	P_6	0.132667	-0.11811	-20.097	0	90	0.52	0.614697	2
	A_2	P_7	0.150865	-0.11811	-20.097	0	146.31	0.45	0.55944	3

Table B.12: Isosceles triangle robotic system generated plans

Bibliography

[1] T. H. Speeter, "Primitive based control of the utah/mit dextrous hand," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, vol. 1, 1991, pp. 866–877. 2, 35

[2] J. Kober and J. Peters, "Learning motor primitives for robotics," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 2112–2118. 2, 20

[3] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 763–768. 2, 73

[4] J. Peters and S. Schaal, "Reinforcement learning for parameterized motor primitives," in *Neural Networks, 2006. IJCNN '06. International Joint Conference on*, 2006, pp. 73–80. 2

[5] L. Lonini, L. Dipietro, L. Zollo, E. Guglielmelli, and H. I. Krebs, "An internal model for acquisition and retention of motor learning during arm reaching," *Neural computation*, vol. 21, no. 7, 2009. 2

[6] K. Muelling, J. Kober, and J. Peters, "Learning table tennis with a mixture of motor primitives," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, 2010, pp. 411–416. 2, 20

[7] J. Kober, B. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 834–839. 2, 20

[8] O. Fuentes and R. C. Nelson, "Learning dextrous manipulation skills using the evolution strategy," in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 1, 1997, pp. 501–506. 2, 35, 73

[9] D. J. Cappelleri, J. Fink, B. Mukundakrishnan, V. Kumar, and J. C. Trinkle, "Designing open-loop plans for planar micro-manipulation," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 637–642. 2

[10] M. Kopicki, J. Wyatt, and R. Stolkin, "Prediction learning in robotic pushing manipulation," in *Advanced Robotics, 2009. ICAR 2009. International Conference on*, 2009, pp. 1–6. 2, 73

[11] M. Lopes, F. S. Melo, and L. Montesano, "Affordance-based imitation learning in robots," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 1015–1021. 3, 73

[12] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory–motor coordination to imitation," *Robotics, IEEE Transactions on*, vol. 24, no. 1, pp. 15–26, 2008. 3, 21, 28, 35

[13] D. Kraft, R. Detry, N. Pugeault, E. BaÅ§eski, J. Piater, and N. KrÃijger, "Learning objects and grasp affordances through autonomous exploration," in *Computer Vision Systems*. Springer, 2009, pp. 235–244. 3

[14] D. E. Berlyne, "Novelty and curiosity as determinants of exploratory behaviour," *British Journal of Psychology. General Section*, vol. 41, no. 1-2, pp. 68–80, 1950. 3, 79

[15] ——, *Conflict, arousal, and curiosity*. New York: McGraw-Hill, 1960. 3, 79

[16] P.-Y. Oudeyer and F. Kaplan, "How can we define intrinsic motivation?" in *proceedings of the 8th international conference on epigenetic robotics: modeling cognitive development in robotic systems*, 2008. 3

[17] K. E. Merrick, "A comparative study of value systems for self-motivated exploration and learning by robots," *Autonomous Mental Development, IEEE Transactions on*, vol. 2, no. 2, pp. 119–131, 2010. 3, 79, 80

[18] S. Marsland, U. Nehmzow, and J. Shapiro, "On-line novelty detection for autonomous mobile robots," *Robotics and autonomous Systems*, vol. 51, no. 2, pp. 191–206, 2005. 3

[19] K. Merrick, "Evaluating intrinsically motivated robots using affordances and point-cloud matrices," in *In Proceedings of the Ninth International Conference on Epigenetic Robotics*, 2009, pp. 105–112. 3

[20] A. Baranes and P.-Y. Oudeyer, "Intrinsically motivated goal exploration for active motor learning in robots: A case study," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 1766–1773. 3

[21] Q. Xinyu and Y. Minghai, "Visual novelty based internally motivated q-learning for mobile robot scene learning and recognition," in *Image and Signal Processing (CISP), 2011 4th International Congress on*, vol. 3, 2011, pp. 1461–1466. 3

[22] P. Raif and J. A. Starzyk, "Motivated learning in autonomous systems," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, 2011, pp. 603–610. 3, 76

[23] S. Hart and R. Grupen, "Intrinsically motivated affordance discovery and modeling," in *Intrinsically Motivated Learning in Natural and Artificial Systems*. Springer, 2013, pp. 279–300. 3, 20

[24] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986. 3

[25] M. A. Peshkin and A. C. Sanderson, "The motion of a pushed, sliding workpiece," *Robotics and Automation, IEEE Journal of*, vol. 4, no. 6, pp. 569–598, 1988. 3, 31, 96

[26] M. Salganicoff, G. Metta, A. Oddera, and G. Sandini, *A vision-based learning method for pushing manipulation*. University of Pennsylvania, Department of Computer and Information Science, 1993. 3, 96

[27] B. Ridge, D. SkoËĞcaj, and A. Leonardis, "Towards learning basic object affordances from object properties," in *International Conference on Cognitive Systems*, Karlsruhe, Germany, 2008. 3

[28] C. Zito, R. Stolkin, M. Kopicki, and J. L. Wyatt, "Two-level rrt planning for robotic push manipulation," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 678–685. 3

[29] R. A. Russell and F. J. Paoloni, "A robot sensor for measuring thermal properties of gripped objects," *Instrumentation and Measurement, IEEE Transactions on*, vol. 1001, no. 3, pp. 458–460, 1985. 3

[30] Y. Hasegawa, H. Ioka, T. Fukuda, and K. Kanada, "Dexterous manipulation from pinching to power grasping - strategy selection according to object dimensions and grasping position," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, 2003, pp. 785–790. 3

[31] K. Masahiro, U. Jun, M. Yoshio, and O. Tsukasa, "Recognition of in-hand manipulation along with rolling contact using orbital motion of contact points on object surface," in *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, 2006, pp. 167–172. 3

[32] H. Dobashi, A. Noda, Y. Yokokohji, H. Nagano, T. Nagatani, and H. Okuda, "Derivation of optimal robust grasping strategy under initial object pose errors," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 2096–2102. 3

[33] R. Jakel, S. R. Schmidt-Rohr, M. Losch, and R. Dillmann, "Representation and constrained planning of manipulation strategies in the context of programming by demonstration," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 162–169. 3

[34] L. Qiang, M. Meier, R. Haschke, H. Ritter, and B. Bolder, "Object dexterous manipulation in hand based on finite state machine," in *Mechatronics and Automation (ICMA), 2012 International Conference on*, 2012, pp. 1185–1190. 3

[35] A. L. Ciancio, L. Zollo, G. Baldassarre, D. Caligiore, and E. Guglielmelli, "The role of thumb opposition in cyclic manipulation: A study with two different robotic hands," in *Biomedical Robotics and Biomechatronics (BioRob), 2012 4th IEEE RAS & EMBS International Conference on*, 2012, pp. 1092–1097. 3, 20, 96

[36] T. H. Vose, P. Umbanhowar, and K. M. Lynch, "Sliding manipulation of rigid bodies on a controlled 6-dof plate," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 819–838, 2012. [Online]. Available: http://ijr.sagepub.com/content/31/7/819.abstract 3, 96

[37] H. Chang-Soon and K. Sasaki, "Control program of two-fingered dexterous manipulation with primitive motions," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3, 2003, pp. 2902–2907. 3

[38] R. D. Howe and M. R. Cutkosky, "Integrating tactile sensing with control for dextrous manipulation," in *Intelligent Motion Control, 1990. Proceedings of the IEEE International Workshop on*, vol. 1, 1990, pp. 369–374. 3, 96

[39] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt, "Learning relational affordance models for robots in multi-object manipulation tasks," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 4373–4378. 3

[40] C. C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments," *IEEE Robotics and Automation Magazine*, vol. 14, no. 1, p. 20, 2007. 11

[41] J. Kenney, T. Buckley, and O. Brock, "Interactive segmentation for manipulation in unstructured environments," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 1377–1382. 11

[42] A. Saxena, L. Wong, M. Quigley, and A. Y. Ng, "A vision-based system for grasping novel objects in cluttered environments," in *Robotics Research*. Springer, 2011, pp. 337–348. 11

[43] D. Katz, A. Venkatraman, M. Kazemi, J. A. Bagnell, and A. Stentz, "Perceiving, learning, and exploiting object affordances for autonomous pile manipulation," in *Proceedings of Robotics: Science and Systems*, Berlin, Germany, 2013. 11

[44] R. R. Murphy, "Introduction to ai robotics," in *Introduction to AI Robotics*, 1st ed. Massachusetts Institute of Technology, 2000. 16, 36

[45] R. Gaschler, D. Katz, M. Grund, O. Brock, and P. A. Frensch, *Intelligent object exploration*. Citeseer, 2012. 20

[46] R. T. Lauer, P. H. Peckham, and K. L. Kilgore, "Eeg-based control of a hand grasp neuroprosthesis," *NeuroReport*, vol. 10, no. 8, pp. 1767–1771, 1999. [Online]. Available: http://journals.lww.com/neuroreport/Fulltext/1999/06030/EEG_based_control_of_a_hand_grasp_neuroprosthesis.26.aspx 20

[47] P. Sikka, "The role of tactile sensing in robot manipulation," Ph.D. dissertation, University of Alberta, 1994. 20

[48] P. Sikka and B. J. McCarragher, "Monitoring contact using clustering and discriminant functions," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2. IEEE, 1996, pp. 1351–1356. 20

[49] J. J. Gibson, *The ecological approach to visual perception*. Routledge, 1986. 21

[50] B. Moldovan, P. Moreno, and M. van Otterlo, "On the use of probabilistic relational affordance models for sequential manipulation tasks in robotics," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1290–1295. 21

[51] E. Ugur and E. Sahin, "Traversability: A case study for learning and perceiving affordances in robots," *Adaptive Behavior*, 2010. 21

[52] B. Ridge, D. Skocaj, and A. Leonardis, "A system for learning basic object affordances using a self-organizing map," in *Proc. ICCS*, 2008. 28

[53] M. T. Mason, "Manipulator grasping and pushing operations," Ph.D. dissertation, Massachusetts Institute of Technology, 1982. 28

[54] K. M. Lynch, "The mechanics of fine manipulation by pushing," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE, 1992, pp. 2269–2276. 28

[55] D. E. Angelaki and K. E. Cullen, "Vestibular system: the many facets of a multimodal sense," *Annu. Rev. Neurosci.*, vol. 31, pp. 125–150, 2008. 46

[56] W. H. Li and L. Kleeman, "Real time object tracking using reflectional symmetry and motion," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 2798–2803. 61

[57] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008. 61

[58] W. H. Li and L. Kleeman, "Interactive learning of visually symmetric objects," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 4751–4756. 61

[59] F. Attneave, "Some informational aspects of visual perception," *Psychological review*, vol. 61, no. 3, p. 183, 1954. 61

[60] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in neurorobotics*, vol. 1, p. 6, 2007. 73

[61] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine learning*, vol. 8, no. 3-4, pp. 293–321, 1992. 76

[62] N. Chentanez, A. G. Barto, and S. P. Singh, "Intrinsically motivated reinforcement learning," in *Advances in neural information processing systems*, 2004, pp. 1281–1288. 76

[63] Y. Kobayashi, H. Fujii, and S. Hosoe, "Reinforcement learning for manipulation using constraint between object and robot," in *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, vol. 1, 2005, pp. 871–876 Vol. 1. 76

[64] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 2, pp. 265–286, 2007. 76

[65] K. Merrick and M. L. Maher, "Motivated learning from interesting events: Adaptive, multitask learning agents for complex environments," *Adaptive Behavior*, vol. 17, no. 1, pp. 7–27, 2009. [Online]. Available: http://adb.sagepub.com/content/17/1/7.abstract 76, 78

[66] X. Huang and J. Weng, "Inherent value systems for autonomous mental development," *International Journal of Humanoid Robotics*, vol. 4, no. 02, pp. 407–433, 2007. 76, 79, 82

[67] A. Baranes and P.-Y. Oudeyer, "Robust intrinsically motivated exploration and active learning," in *Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on*. IEEE, 2009, pp. 1–6. 76

[68] J. Sungmoon, L. Minho, H. Arie, and J. Tani, "Developmental learning of integrating visual attention shifts and bimanual object grasping and manipulation tasks," in *Development and Learning (ICDL), 2010 IEEE 9th International Conference on*, 2010, pp. 165–170. 76

[69] U. Schiefele, "Interest, learning, and motivation," *Educational psychologist*, vol. 26, no. 3-4, pp. 299–323, 1991. 76

[70] P. J. Silvia, "What is interesting? exploring the appraisal structure of interest," *Emotion*, vol. 5, no. 1, p. 89, 2005. 79

[71] T. B. Kashdan and P. J. Silvia, "Curiosity and interest: The benefits of thriving on novelty and challenge," *Oxford handbook of positive psychology*, vol. 2, pp. 367–374, 2009. 79

[72] T. B. Kashdan, P. Rose, and F. D. Fincham, "Curiosity and exploration: Facilitating positive subjective experiences and personal growth opportunities," *Journal of personality assessment*, vol. 82, no. 3, pp. 291–305, 2004. 79

[73] P. J. Silvia, "InterestâĂŤthe curious emotion," *Current Directions in Psychological Science*, vol. 17, no. 1, pp. 57–60, 2008. 79

[74] G. Loewenstein, "The psychology of curiosity: A review and reinterpretation," *Psychological bulletin*, vol. 116, no. 1, p. 75, 1994. 79

[75] J. Schmidhuber, "Formal theory of creativity, fun, and intrinsic motivation (1990âĂŞ2010)," *Autonomous Mental Development, IEEE Transactions on*, vol. 2, no. 3, pp. 230–247, 2010. 79

[76] R. B. Zajonc, "Attitudinal effects of mere exposure," *Journal of personality and social psychology*, vol. 9, no. 2p2, p. 1, 1968. 79

[77] R. N. Hughes, "Intrinsic exploration in animals: motives and measurement," *Behavioural Processes*, vol. 41, no. 3, pp. 213–226, 1997. 79

[78] D. Wood-Gush and K. Vestergaard, "The seeking of novelty and its relation to play," *Animal Behaviour*, vol. 42, no. 4, pp. 599–606, 1991. 79

[79] S. Sreenivasan, "Quantitative analysis of the evolution of novelty in cinema through crowdsourced keywords," *Sci. Rep.*, vol. 3, 2013. [Online]. Available: http://dx.doi.org/10.1038/srep02758 79

[80] L. da Fontoura Costa and R. M. Cesar Jr, *Shape analysis and classification: theory and practice*. CRC press, 2010. 96

[81] J. Flusser, "On the independence of rotation moment invariants," *Pattern recognition*, vol. 33, no. 9, pp. 1405–1410, 2000. 96

[82] M. A. Erdmann and M. T. Mason, "An exploration of sensorless manipulation," *Robotics and Automation, IEEE Journal of*, vol. 4, no. 4, pp. 369–379, 1988. 96

[83] R. A. Russell, *Robot tactile sensing*. Prentice-Hall, Inc., 1990. 96

[84] H. Osumi, N. Ishii, K. Takahashi, K. Umeda, and G. Kinoshita, "Optimal grasping for a parallel two-fingered hand with compliant tactile sensors," in *Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on*, vol. 2, 1999, pp. 799–804. 96

[85] P. Sikka and H. Zhang, "Tactile-assisted robotic object manipulation," *Fourth International Symposium on Robotics and Manufacturing*, pp. 83–88, 1992. 96

[86] P. Sikka, H. Zhang, and S. Sutphen, "Tactile servo: Control of touch-driven robot motion," in *Experimental Robotics III*. Springer, 1994, pp. 219–233. 96

[87] V. Emeli, C. C. Kemp, and M. Stilman, "Push planning for object placement in clutter using the pr-2," in *PR2 Workshop: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, 2011, pp. 25–30. 96

[88] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman, "Push planning for object placement on cluttered table surfaces," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 4627–4632. 96

[89] C. Kukjin and K. D. Wise, "A high-performance silicon tactile imager based on a capacitive cell," *Electron Devices, IEEE Transactions on*, vol. 32, no. 7, pp. 1196–1201, 1985. 98

[90] R. A. Russell, "Compliant-skin tactile sensor," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4. IEEE, 1987, pp. 1645–1648. 98

[91] E. Catto, "Box2d physics engine," *World Wide Web electronic publication*, 2009. 110

[92] M. Hu, "Visual pattern recognition by moment invariants," *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 179–187, 1962. 132

[93] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973. 133