A DEVELOPMENTAL FRAMEWORK FOR LEARNING AFFORDANCES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EMRE UĞUR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

DECEMBER 2010

Approval of the thesis:

A DEVELOPMENTAL FRAMEWORK FOR LEARNING AFFORDANCES

submitted by **EMRE UĞUR** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Asst. Prof. Dr. Erol Şahin _____
Supervisor, **Computer Engineering Dept., METU**

Assoc. Prof. Dr. Erhan Öztop _____
Co-supervisor, **School of Engineering Science, Osaka University**

Examining Committee Members:

Prof. Dr. Göktürk Üçoluk
Computer Engineering, Middle East Technical Univesity _____

Asst. Prof Dr. Erol Şahin
Computer Engineering, Middle East Technical Univesity _____

Prof. Dr. Billur Barshan
Electrical and Electronics Engineering, Bilkent University _____

Prof. Dr. İsmail HakkıToroslu
Computer Engineering, Middle East Technical University _____

Asst. Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering, Middle East Technical University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: EMRE UĞUR

Signature :

ABSTRACT

A DEVELOPMENTAL FRAMEWORK FOR LEARNING AFFORDANCES

Uğur, Emre

Ph.D., Department of Computer Engineering

Supervisor : Asst. Prof. Dr. Erol Şahin

Co-Supervisor : Assoc. Prof. Dr. Erhan Öztop

December 2010, 192 pages

We propose a developmental framework that enables the robot to learn affordances through interaction with the environment in an unsupervised way and to use these affordances at different levels of robot control, ranging from reactive response to planning. Inspired from Developmental Psychology, the robot's discovery of action possibilities is realized in two sequential phases. In the first phase, the robot that initially possesses a limited number of basic actions and reflexes discovers new behavior primitives by exercising these actions and by monitoring the changes created in its initially crude perception system. In the second phase, the robot explores a more complicated environment by executing the discovered behavior primitives and using more advanced perception to learn further action possibilities. For this purpose, first, the robot discovers commonalities in action-effect experiences by finding effect categories, and then builds predictors for each behavior to map object features and behavior parameters into effect categories. After learning affordances through self-interaction and self-observation, the robot can make plans to achieve desired goals, emulate end states of demonstrated actions, monitor the plan execution and take corrective actions using the perceptual structures employed or discovered during learning.

Mobile and manipulator robots were used to realize the proposed framework. Similar to

infants, these robots were able to form behavior repertoires, learn affordances, and gain prediction capabilities. The learned affordances were shown to be relative to the robots, provide perceptual economy and encode general relations. Additionally, the affordance-based planning ability was verified in various tasks such as table cleaning and object transportation.

Keywords: affordances, developmental robotics, sensory-motor learning, cognitive robotics, robot perception

ÖZ

SAĞLARLIK ÖĞRENİMİ İÇİN GELİŞİMSEL BİR ÇERÇEVE

Uğur, Emre
Doktora, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Yrd. Doç. Dr. Erol Şahin
Ortak Tez Yöneticisi : Doç. Dr. Erhan Öztop
Aralık 2010, 192 sayfa

Robotun ortam ile etkileşimi yolu ile sağlarlıkların (ing. affordances) gözetimsiz öğrenilmesini ve reaktif tepkiden planlamaya robot denetiminin farklı düzeylerinde sağlarlıkların kullanımını sağlayan gelişimsel bir çerveçe önerilmiştir. Gelişimsel Psikoloji'den ilham alarak robotun hareket keşifleri iki ana ardışık aşamada gerçekleştirilmiştir. İlk aşamada, sınırlı sayıda temel aksiyon ve reflekse sahip olan robot, bu aksiyonları egzersiz ederek ve başlangıç ilkel algı sistemindeki degişiklikleri gözlemleyerek yeni davranış primitifleri keşfetmektedir. İkinci aşamada, daha ileri hareket olanaklarını öğrenmek için, daha gelişmiş bir algı sistemi kullanarak ve bir önceki adımda keşfettiği davranışları uygulayarak daha karmaşık ortamları araştırmaktadır. Bu amaçla, öncelikle robot, etki kategorileri bularak, aksiyon-etki deneyimlerinde oluşan benzerlikleri keşfetmektedir. Daha sonra cisim özellikleri ve davranış parametrelerini etki kategorilerine eşlemek amacı ile her davranış için öngörücüler kurmaktadır. Robot, sağlarlıkları etkişim ve gözlem yoluyla öğrendikten sonra, öğrenme sırasında keşfedilen algısal yapıları kullanarak, istenen hedefleri gerçekleştirmek için planlar yapabilmekte, başkası tarafından gösterilen aksiyonlar ile elde edilen son durumları taklit edebilmekte ve planın yürütülmesini takip edip düzeltici hareketler yapabilmektedir.

Önerilen çerçeveyi gerçekleştirmek için gezer ve manipülatör robotlar kullanılmıştır. Bebek-

lere benzer şekilde, bu robotlar, hareket repertuarları geliştirebilmiş, sağlarlıkları öğrenebilmiş ve tahmin becerisi kazanabilmiştir. Öğrenilen sağlarlıkların robotlara göreli olduğu, algıda ekonomi sağladığı ve genel ilişkileri kodladığı gösterilmiştir. Ayrıca, sağlarlık tabanlı plan yapabilme kabiliyeti masa temizleme ve cisim taşıma gibi çeşitli görevlerde doğrulanmıştır.

Anahtar Kelimeler: Sağlarlık, gelişimsel robotik, sensör motor ögrenmesi, bilişsel robotik, robot algısı

to my parents

ACKNOWLEDGMENTS

In the first place I would like to express my sincere gratitude to my advisor, Erol Şahin for his supervision, advice, and guidance. He has always been an extraordinary advisor and mentor for the last 8 years. He gave me the chance to participate in several state-of-the-art projects, work in a superb environment, enrich my growth as a student and researcher.

I am also full of gratitude to my thesis monitoring and defense committee members Göktürk Üçoluk, Billur Barshan, Hakkı Toroslu, and İlkay Ulusoy for their guidance and support.

I am grateful to the members of the KOVAN lab for creating an inspiring and pleasant atmosphere. Special thanks to Fatih Gökçe for his support in mechanical and electrical problems and Hande Çelikkanat for her support in programming with PC clusters. Both Hande and Fatih helped me a lot in practical problems in Turkey when I was in Japan. Ali Emre Turgut has been both a teacher and friend for me. Last but not least, it was great pleasure to discuss about affordances with Maya Çakmak and Mehmet Remzi Doğar.

I would like to show my gratitude to Mitsuo Kawato for giving me the opportunity to work at such a stimulating research environment as an intern researcher. The time I had at ATR and in Japan was a very enriching period of my life. I would like to salute the staff at ATR for all their help. I am deeply thankful to Yu Shimizu for her friendship and always being up for a cup of coffee at ATR's veranda. I am greatly indebted to Mari Fukaumi for her kind support in administrative issues. Thanks to Brian Moore for always being there with his warm smile. I am thankful to Nao Nakano for his technical support. Finally, I would like to thank to Mario Gamez and Ludovic Righetti for being good friends.

I would like to acknowledge Barbara Webb from Edinburgh University, Aleš Ude from Jožef Stefan Institute, Frank Guedrin from University of Aberdeen, and Minoru Asada from Osaka University for their constructive comments on affordances and developmental approach.

I would like to acknowledge the financial support of TÜBİTAK (The Scientific and Technical Research Council of Turkey) through the National Scholarship Programme for Ph.D.

Students. This work was partially funded by the European Commission under ROSSI (FP7-21625) and MACS (FP6-004381) projects. This thesis was also supported by Advanced Telecommunications Research Institute International and National Institute of Information and Communications Technology, Japan. I would like to also acknowledge the computational resources provided by the High Performance Computing Center, Department of Computer Engineering, Middle East Technical University.

My friends İzlem Gözükeleş, Emre Akbaş, Candaş Kılıç, Aykut Erdem, Erkut Erdem, Pilar Suguimoto and Yu Shimizu were always 'online' and by my side. Thank you!

I owe deep gratitude to Erhan Öztop, my boss, co-advisor, mentor, and friend who has always been the first person to help whatever the matter is. It has been great pleasure having the chance to work with him, discussing about science and research, going to sea and mountains, listening to Zülfü while talking about life and many other things. I would also like to thank to Mai-san for always being kind to me.

Finally, I am deeply and forever indebted to my family for their love, support and encouragement throughout my entire life. This work would not be possible without their help. (Anne, Baba ve Ahmet: Hersey için çok teşekkürler!)

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

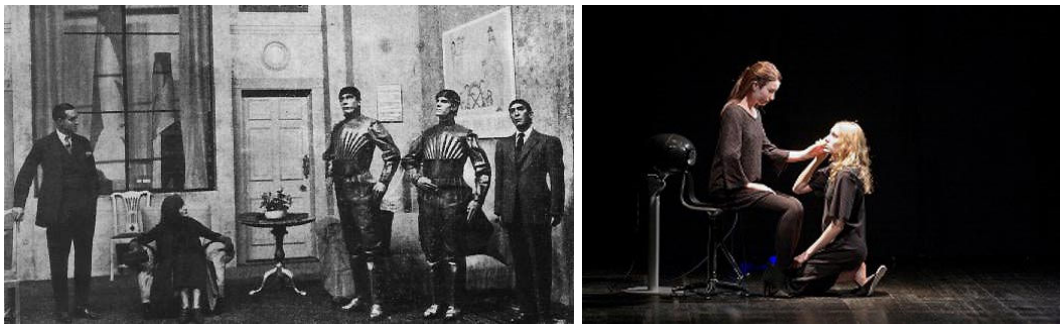
"Failure of existing rules is the prelude to a search for new ones. "

— Thomas Kuhl, 1962

It has been almost a century, since the *robot*[1] word first appeared in Karel Čapek's science fiction play [16], where human-like biological robot characters were played by human actors as intelligent workers that can serve in any job. This year (in 2010), we witnessed a real robot (built by Prof. H. Ishiguro from Osaka University, Japan) took stage and acted as an android giving company to a (real) woman suffering from a fatal illness [62].

If an intelligent robot "is a mechanical creature which can function autonomously" [104, p. 3] or " is a machine that senses, thinks, and acts" [9, p. 2] as defined in robotics textbooks, then Prof. Ishiguro's android (which is controlled by a human operator) can hardly be considered an intelligent robot. On the other hand, Grey Walter's three-wheeled robots (1950) with prototube eyes and vacuum tube amplifiers can be considered as intelligent since they could exhibit emergent behaviors while searching for its battery charging locations [2]. As another example from old days, despite its slow behavior execution and its susceptibility to dynamic and uncertain world, Stanford Research Institute's Shakey robot (1970), with bump detectors, range finders and radio antennas, can be considered more intelligent since it had planning abilities [104]. Coming to today, the state-of-the-art robots which are designed for specific tasks exhibit impressive abilities. Mobile vehicles can drive 13,000 km. from Italy to China in traffic [61] autonomously, humanoid robots can climb stairs (Asimo), run at 7 km/hr speed (Toyota), and shake hands with presidents of the world states. However, simply put, no robot today can exhibit perceptual, motor or intellectual capabilities of a 3 years-old child.

[1] Robot is derived from *robota* which means self-labor or forced-labor in Czech language.



(a) Rossum's Universal Robots (1921) [16] (b) Real android (left) on stage (2010) [62]

Figure 1.1: From human actors who play robots to an android starring alongside a human actress.

Over the past several decades, researchers have taken different approaches to create intelligent machines. The first approach was to create complicated knowledge-bases for world representation and use logic-based methods to make inferences. However, these robots lacked fast response in dynamical and uncertain environments, since the sensing and acting were connected through a slow planning module. As an alternative approach, simple robots were programmed with tight sensor-actuation couplings so that they were able to give fast reactions and survive in the dynamic world, however the tasks that could be undertaken by these robots were simple. Next, low-level reactive control is combined with high-level reasoning systems to obtain the advantage of both. However bridging these two layers of control was not-straightforward due to the differences in design constraints and the representational gap between these levels.

Roboticists also utilized ideas from social sciences and nature to overcome the difficulty in intelligent robot development. In [109] robots evolved through generations based on Darwinian principle of survival of the fittest, in [12] simple robots exhibit emergent and collective intelligent behavior inspiring from social insects, in [112] they grasped objects based on models derived from monkey and human neuroimaging studies.

1.1 Robotics and Developmental Psychology

As Weng et al. put "although notable, none of these (the methods used so far) is powerful enough to lead to machines having the complex, diverse, and highly integrated capabilities of an adult brain, such as vision, speech, and language" [155]. As an attempt to find and

alternative and 'better' way to develop intelligent robots, in the beginning of 2000's a new field was established at the intersection between developmental psychology, cognitive science, developmental neuroscience and robotics. The aim of this approach, presented as *Cognitive Developmental Robotics* by Asada et al. [4], *Autonomous Mental Development* by Weng et al. [155], *Epigenetic Robotics* by Zlatev and Balkenius [162], and *Developmental Robotics* by Lungarella et al.[92], was to create "truly intelligent machines" [155]. *Developmental Robotics* in short, was born as an alternative to previous robot-learning approaches that were task and designer dependent. The new approach argued that a robotic developmental pathway similar to humans or animals with high order cognitive skills is the right way for obtaining intelligent robots. Although there exists differences in to which extent biological development is to be followed, common to all, developmental robotic approach has the following main characteristics:

- First of all, the learning agent must be embodied in a physical body and situated in the environment it physically interacts. Developmental psychology argues that the brain, body and environment are tightly coupled and the development of cognition is only possible within (and is being directed by) this coupling. In other words, physical embodiment enables cognitive development through interactions with the environment. The interactions can take place with the object in robot's world in an unsupervised way, or it can be scaffolded by other agents (or parents) in the environment, corresponding to sensorimotor and social development.

- The development must be incremental, i.e. the perceptual, motor and cognitive development must follow a pathway going from simple to complex. More complex skills must be discovered using and based on the simpler skills that were learned before. Furthermore, similar to infant development [119, 36] the incremental development can occur in qualitatively different stages.

- The development must be task-independent and should be led by the environment (including robot's own body and changes in its morphology if possible). For example, instead of specifically learning how to grasp objects, the robot that interacts with the environment can develop many manipulation skills including grasping. Parental scaffolding is important and necessary in development of higher-level cognitive functions, however self-exploration of the environment through interaction is important in devel-

opment of basic sensorimotor system.

- The sensory-motor system of the robot must be limited in initial phases of the development. The robot's perception system is bombarded with large amount of and complex data received from its sensors such as vision and tactile. Similar to biological systems who has limited sensory and motor capabilities initially (such as limited visual acuity and range in newborn infants), the complexity of the sensor data and motor commands should be limited initially and limitations should be relaxed incrementally while the robot develops necessary perceptual and cognitive structures that can process that data.

- The robot is 'born' with low-level sensorimotor representation. However, the increased cognitive abilities of the robot would require higher-level perceptual and motor representations. Thus, the robot needs to learn high-level concepts in its perceptual and motor aparata, i.e. should develop perceptual and behavioral categories.

1.2 Robotics and Ecological Psychology

The core assumption of developmental robotics, *embodiment* and *situatedness* has been studied in Ecological Psychology for decades. Along this line of research, one of the most influential Ecological Psychologist J.J.Gibson, coined the term affordances, that (according to him) "refers to both the environment and the animal in a way that no existing term does (and) implies the complementarity of the animal and environment." [55, p. 127].

According to J.J.Gibson, the action possibilities (affordances) provided by the environment can be directly perceived by humans without any intermediate high-level object recognition step. That is, humans do not need to recognize the action-free meanings of the objects and make complex inferences over these meanings in order to act on them. For example we do not identify the objects with their action-free labels such as chairs, couches or stones when we need to throw them or sit on them. Instead, we look for a specific combination of the object properties taken with reference to us and our action capabilities in order to detect their 'throwability' or 'sittability' affordances.

Although it is not the classical engineering approach of 'identify and then act', this strategy appears to be the one employed by our brains. It is known that the cerebral cortex processes visual information in at least two channels, the so called dorsal and ventral pathways. The

ventral pathway appears to be responsible for object identification, whereas the dorsal pathway is mainly involved in perception for action [34, 56, 57, 146]. These data suggest that an agent does not necessarily need to possess object recognition capability to learn about its environment, and use this knowledge for making plans.

Robotics has long suffered from the problem of processing vast amount of information available in perception, representation and decision making levels. Affordance-based view provides the agent a simplified means of perception and representation of the environment. Furthermore, this perception and representation is grounded in agent's actions and body, and the environment it acts. Thus, utilizing affordances theory in robotics has great potential and indeed it has been recently explored in many robotic studies [3, 26, 27, 31, 93, 46, 133, 132, 30, 99, 139, 43, 48, 129, 60, 37, 65, 101, 117, 159, 100, 121].

Affordances can also be used to deal with the problem of creating reactive robots with high-level reasoning systems. As mentioned in the beginning of this chapter, there exists a representational gap between the continuous sensory-motor experiences of a robot and the symbolic planning operators of Artificial Intelligence. The mapping of the symbols used in these operators onto the sensory-motor readings of the robot's continuous world is typically referred as part of the symbol grounding problem [64] and has been studied since the days of STRIPS [44]. These studies [83] typically assume that the relations that bind the pre-coded symbols (such as pre-conditions and effects of an operator) are given, and aim to learn the mapping from these symbols to the continuous sensory-motor readings of the robot. Recently, it has been argued that symbols "are not formed in isolation" and that "they are formed in relation to the experience of agents, through their perceptual/motor apparatuses, in their world and linked to their goals and actions" [135, p. 149]. Learning affordances through robot-environment interactions can enable the formation of high-level grounded concepts that are used in high-level reasoning systems and bridge the gap between two (or more) levels of control.

1.3 The Aim of the Thesis

This thesis aims to propose a developmental framework to enable the robot to learn affordances through interaction with the environment in an unsupervised way and use these affordances at different levels of robot control from reactive response to planning. In this respect,

our approach can be viewed at the intersection of Autonomous Robotics, Developmental Sciences and Ecological Psychology. Affordance learning is studied in two main domains of robotics, namely mobile robotics and manipulation systems. Our studies in mobile robotics are more focused on studying Gibsonian affordances since the experiments conducted to show the utility of affordance perception are more related to mobility related behaviors such as climbability, pass-through-ability, pass-under-ability, etc. On the other hand, we shifted our focus to (relations with) infant development during our affordance learning and behavior discovery studies, thanks to the similarities of biological and robot hand/arm systems and the corresponding skills. In the rest of this section, we will provide the ideas derived from infant development and affordances theory that are used in the development of the proposed system.

1.3.1 Ideas Adopted from Infant Development

In this section, we will enumerate the phases of infant development that guide us in establishing the progressive learning framework that this thesis follows:

- New-born babies have many innate reflexes such as pupil reflex to light, sucking reflex or palmar-grasp reflex. Palmar reflex in particular is "integrated into later intentional grasping" [120, p. 7] after repeated activation of the reflex and execution of grasp action. This reflex is not always stable and by 6 months of age, it disappears [124, p. 199].

- By 4 months of age, infants learn to perceive the reachability boundaries [124, p. 199] and they can successfully reach to the objects [14, p. 41].

- By 5 months of age, infants slow down their hand speeds when grasping objects, i.e. they learned adjusting hand reach speed by this age [124, p. 100].

- It takes 9 months for infants to reach for objects with correct hand-orientation and adjust their grip size based on objects' size before contact. These parameters of reach develops later then hand-speed parameter since "babies younger than 9-months lack a fully-developed map between visually perceived orientations and corresponding hand orientations" [124, p. 200].

- Between 7-9 months, babies explore the environment and objects using various behaviors including grasp, drop, and hit [4]. We think that, by this time, the infant has already

developed a set of behavior primitives from its most basic movement primitive, 'move arm'.

- Between 7-9 months, they learn the causality relations and object dynamics in response to their actions [4]. It is plausible to think that while interacting with the environment, babies monitor these consequences of their actions and relate the consequences to the visual properties of the objects they interact with. In other words, they learn object affordances in this phase.

- By 10-12 months, they can imitate actions and they can generate multi-step plans to accomplish goals (such as reaching toys) [157]. Since the symbolic representation develops only after 18 months [119], probably the sub-symbolic structures or concepts discovered within infant's sensorimotor representation during affordance learning are represented as symbols and used for imitation and multi-step planning.

- Psychologists believe that a mechanism called 'intrinsic motivation' exists in order to drive open-ended cognitive development of humans, and infants in particular [156]. Thanks to this mechanism, infants exhibit spontaneous exploration and curiosity during their joy of play and 'maximize' their learning extent and speed.

1.3.2 Ideas Adopted from Affordances Theory

In this section, we will enumerate the main attributes of affordances concept that guide us in establishing the affordance learning framework. The following attributes were first identified in [33] and re-interpreted for affordances in our study.

- *Affordances can be viewed from three perspectives; namely, agent, observers, and environment*. The agent's affordances correspond to a representation that *resides inside the agent* and the observer's affordances can be viewed as the representation when we (observers) analyze the execution of the agent based on it's affordance perception.

- *Affordances are acquired relations* This acquisition can correspond to evolution, learning and trial-and-error based design. In our case, the structures (that are used for learning affordances) are assumed to exist (for example acquired through evolution) and we will focus on learning of affordances. The learning of affordances was not particularly

a focus of J.J.Gibson, but studied by E.Gibson. We will discuss this in the next chapter, Section 2.2.

- *Affordances encode general relations* pertaining to the agent, environment interaction, such as balls: balls are rollable. Naturally, exceptions to these general relations, such as "the-red-ball-on-my-table is not rollable (since it is glued to the table)" do exist. However, unlike affordance relations, these specific relations possess little predictive help over other cases, such as whether the-blue-ball-on-my-table is rollable or not.

- *Affordances provide perceptual economy*. The concept of affordances is often used as support for minimality in perception to argue that one does not have to perceive all the qualities of their environment in order to accomplish a simple task such as wandering around.

- *Affordances are relative.* This argument, generally accepted within most contexts, is usually linked to the complementarity of the organism and the environment. According to this view, the existence of an affordance is neither defined by the environment nor by the organism alone but through their interaction. For instance, the *climbability* of a stair step is not only determined by the metric measure of the height, but also by one's leg length. In case of *observer's affordance*, the existence of *climbability* affordance depends on the ratio between height of the stair and agent's leg length. However, from *agent's perspective*, since it learned *climbability* affordance with a fixed leg length, the affordance (that resides in agent's mind) only depends on the height of the stairs.

- *Affordances provide support for multi-step prediction and planning*. Discovering affordances from low-level and continuous sensorimotor experience of the robot corresponds to formation a higher-level world representation, i.e. abstract perceptual states. In other words, the robot learns to represent the state in terms of discovered affordances. Furthermore, learning to perceive affordances corresponds to acquiring prediction ability over these perceptual states. An agent can make multi-step predictions and accomplish goals that are encoded in discovered state representations by generating plans with the acquired prediction ability.

1.4 The Organization of the Thesis

The realization and implementation of the affordance-learning framework follows simple to complex progression and this progression does not always correspond to infant's development time-line. The former chapters include limited versions of the affordances learning framework by postulating simplifying assumptions, and in later chapters these limitations are gradually relaxed. The robot perception, the behavior representation, and the details of the learning algorithms also differ among experiments. Thus, in the beginning of each chapter, a section named **Framework Implementation** gives details of the representational details and postulated assumptions. At the end of each chapter, **Discussion** section describes the assumptions to be relaxed in the next chapter.

Chapter 2 gives an overview of the theory of affordances and its use in different fields. The topics discussed in this chapter, namely the definition and formalization of the affordance concept, the previous affordance based robot control systems, and the affordance-related animal experiments, will serve as a basis in our affordance-based robot learning framework.

Chapter 3 describes how affordances are encoded in 'robot's mind' and provides the affordance learning framework in its most generic form. First, the affordance representation is described in a relational structure that encapsulates the robot behaviors, the initial perception of the world, and the change in perception due to the behavior execution. Then, the unsupervised affordance learning method that use robot's interaction experience with the environment is defined. At the end, how learned affordances can be utilized in goal-oriented robot control is discussed. Note that the realization of this framework was progressively developed through time and this development is reflected in chapter organizations. In other words, in each chapter different (but overlapping) parts of the framework is implemented.

Chapter 4 gives the details of the robot platforms used in this thesis. The differential drive mobile robot platform with laser range finder is used in Chapters 5-7. The manipulator robot platform that is composed of an anthropomorphic hand-arm robot system and infrared range camera is used in Chapters 8-10.

Chapter 5 studies the learning and perception of traversability affordances on a mobile robot equipped with range sensing ability. The environment is said to be traversable in a certain direction, if the robot (moving in that direction) is not enforced to stop as a result of contact

with an obstacle. Thus, if the robot can push an object (by rolling it away), that environment is said to be traversable even if the object is on robot's path, and a collision occurs. Through experiments inspired by Ecological Psychology, we show that the robot, by interacting with its environment, can learn to perceive the traversability affordances. We show that three of the main attributes that are commonly associated with affordances, that is, affordances being relative to the environment, providing perceptual economy, and providing general information, are simply consequences of learning from the interactions of the robot with the environment. Using the learned affordance detection ability, the real robot can successfully navigate in an office environment cluttered with objects that it has never interacted before.

In Chapter 5, the robot learns traversability affordances by random exploration and batch learning. However, we discussed that infants use 'intrinsic motivation' in exploration to optimize the speed and extent of their learning, and they learn in an open-ended manner. Thus, in **Chapter 6** we study a curiosity-based online learning algorithm that automatically chooses novel situations to interact based on previous experience and show that with curiosity-based learning method, the robot can learn traversability affordance using less exploration time.

In Chapters 5 and 6, the affordances are learned through supervision of the behavior designer who explicitly sets success criteria for each behavior execution. As we discussed that infants can explore the environment in a goal-free means without any supervision. **Chapter 7** studies unsupervised learning of affordances where the mobile robot interacts with the objects in its environment using a pre-coded repertoire of behaviors. It records its interactions in a triple that consist of the initial percept of the object, the behavior applied and its effect, defined as the difference between the initial and the final percept. The method allows the robot to learn object affordance relations which can be used to predict the change in the percept of the object when a certain behavior is applied. These relations can then be used to develop plans using forward chaining. Using this method, the real mobile robot with limited manipulation capabilities can make and execute multi-step step plans such as 'move-forward-left', 'move-forward-right', and 'lift', in order to lift a novel unreachable object.

In Chapters 8-10, we use manipulation robot platforms and study the affordances provided to them. **Chapter 8** shows that through self-interaction and self-observation, an anthropomorphic robot equipped with a range camera can learn object affordances and use this knowledge for planning. Similar to Chapter 7, the robot discovers commonalities in its action-effect

experiences by discovering effect categories using a novel hierarchical categorization algorithm. After learning, the robot can make plans to achieve desired goals, emulate end states of demonstrated actions, monitor the plan execution and take corrective actions using the perceptual structures employed or discovered during learning. This chapter can be viewed as extension of the same ideas in Chapter 7 to manipulation environment, but with a better effect category discovery mechanism, incorporation of emulation and monitoring mechanisms and a implemented closed-loop control architecture. At the end, if the robot observes an empty table as goal, then it can clear the table by pushing or lifting or dropping the objects. If the robot observes one object lifted in the air, it can bring other objects to the same position by pushing several times and lifting. If it observes two objects that are close to each other as goal and those objects are separated, it can generate plans to bring them closer. All these plans are made in robot's perceptual space and they are based on learned affordances and learned prediction ability.

Chapter 9 studies not only learning of the existence of affordances provided by objects, but also the behavioral parameters required to actualize them, and the prediction of effects generated on the objects in an unsupervised way. This chapter extends previous chapters by using parametric behaviors and including the behavior parameters into affordance learning and goal-oriented plan generation. Furthermore, for handling complex behaviors and complex objects (such as execution of precision grasp on a mug), the perceptual processing is improved by using a combination of local and global features. In short, object affordances for object manipulation are discovered together with behavior parameters based on the the monitored effects.

Upto Chapter 10, we assumed the existence of a behavior repertoire that was learned in a previous developmental phase. Thus, we manually designed supposedly learned behaviors inspiring from infant development literature so that the learned affordance prediction abilities based on those behaviors can be used in a goal-oriented way. In **Chapter 10**, we relax this last assumption and propose a method that enables the robot to discover behavior primitives from one basic action using limited tactile and visual perception. Additionally, we improve the effect category discovery method that was developed in Chapter 8 and propose a visual representation inspired from the affordance representation in the parietal cortex of macaque monkeys [114].

In this thesis, all robotic experiments are inspired from the ideas in Theory of Affordances or Developmental Psychology domains. Thus the results of these experiments are discussed in corresponding domains in **Chapter 11**. In the same chapter, the stance of this work among other robotic studies is also identified. In particular, we reviewed the related robotic studies using the terminology developed in this thesis and emphasize our contributions to the field of Autonomous Robotics.

CHAPTER 2

AFFORDANCES

This chapter gives an overview of the theory of affordances and its use in different fields. After summarizing J.J. Gibson's affordance concept and explaining E.J. Gibson's ideas on affordance learning in humans, we describe our affordance formalism that is used as a base in robot control in this thesis. Next, we discuss the robotic studies that utilized affordances for different purposes: detecting traversable paths for mobility, detecting object affordances for manipulation, and minimizing robot perception for detecting only action-relevant properties of the environment. At the end, we will review the human and animal experiments that show the existence of affordance detection systems. The definition and formalization of the affordance concept, the previous affordance based robot control systems, and the affordance-related experiments in Ecological Psychology will serve as a basis in our affordance-based robot learning framework.

2.1 Affordances: A computational View

The concept of affordances was introduced by J. J. Gibson to explain how inherent "values" and "meanings" of things in the environment can be directly perceived and how this information can be linked to the action possibilities offered to the organism by the environment [55].

> "The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill. The verb to afford is found in the dictionary, but the noun affordance is not. I have made it up. I mean by it something that refers to both the environment and the animal in a way that no existing

Figure 2.1: Affordances in real life. In the first photo, the monkey *holds out* the *stick* to *push* the banana. In the second, it *climbs* to the *stick* to *reach* the banana. In the third, it *stacks* the *boxes* and *climbs* over them to *reach* the target. Here, banana affords *pushability* and *reachability*, stick affords *holdability* and *climbability*, and box affords *stackability* and *climbability*. Monkeys can detect and act on these affordances. The photos were taken during Wolfgang Kohler's experiments [84].

> term does. It implies the complementarity of the animal and the environment."
>
> (J. J. Gibson, 1979/1986, p. 127)

In this sense, a stone affords throwing, a flat rigid surface affords walking, a mug affords grasping etc. The definition of the term often depends on the field it is used in; in fact Gibson himself gave differing definitions over the course of his publications [76]. In general, the question 'what does this mug afford for me?' can be equated with 'what type of actions can I apply on this mug?'. One clear fundamental notion of the affordance concept is that object recognition is not a necessary step for interacting with objects. Instead affordances are directly perceived by humans without creating object models with further 'mental calculation' of the otherwise meaningless perceptual data. That is, a specific combination of object properties with respect to the agent and its action capabilities are enough to detect the affordances of a given object (and act on it).

The concept, described through inspirational but also vague discussions by J.J. Gibson such as the one quoted above, turned out to be very influential and has attracted interest from a wide range of fields, ranging from Neuroscience and Human Computer Interaction to Autonomous

Robotics. In our earlier work [33], we summarized the context behind the conception of the term, speculated on its evolution within J.J. Gibson's studies, and reviewed the usage of affordances in different fields. We concluded that the confusion surrounding the concept had stemmed from that fact that J.J. Gibson's own ideas on the concept were not finalized during his lifetime and was left in an ambiguous state.

Placing the concept of affordance on a general computational ground is difficult due to its elusive and multi-facet nature. Recently, we [33] proposed a computational interpretation of the affordance concept that was shown to be effective for mobile robot control [139, 141, 140, 39, 144, 18]. The proposed formalism agrees with the Gibsonian view that affordances are relations within the agent-environment system, but it also extends this view by arguing that these relationships can also be represented in the agent (a.k.a. robot).

2.2 Learning of Affordances

J.J. Gibson was not particularly interested in development and "his concern was with perception" [136] only. As a result, he did not discuss the concept of affordances from a developmental point of view, and only mentioned that affordances are learned in childhood [55]. It is generally accepted that infants' exploration, through physical interaction with the environment, is very important in development of locomotion related perceptual and motor skills [1]. E.J. Gibson argued that learning is neither the construction of representations from smaller pieces, nor the association of a response to a stimulus. Instead, she claimed, learning is "discovering *distinctive* features and *invariant* properties of things and events" [51] or "discovering the information that specifies an affordance" [52]. Learning is not "enriching the input", but discovering the critical perceptual information in that input. She named this process of discovery *differentiation*, and defined it as a "narrowing down from a vast manifold of (perceptual) information to the minimal, optimal information that specifies the affordance of an event, object, or layout" [52].

2.3 Affordance Formalization

In [33], after reviewing a number of affordance formalization proposals, we proposed a new formalization of affordances, based partially on Chemero's formalization [20] and outlined

how affordances can be used at different levels of robot control, ranging from perception and learning to planning. One key feature of this framework is that the affordances are defined from the viewpoint of the acting agent.

Specifically, the formalism defined affordances as general relations that pertain to the robot-environment interaction and claimed that they can be represented as triples that consist of the initial percept of the object, the behavior applied and the effect produced. Formally, an affordance is an acquired relation that can be represented as a nested triplet

$$(\textit{effect}, (\textit{entity}, \textit{behavior}))$$

indicating that when the agent applies the *behavior* on the *entity* the *effect* is generated. Here, the term *entity* denotes the environmental relata of the affordance and represents the initial state of the environment (including the state of the agent) as perceived by the agent. Entity is a high level term that can encapsulate the perceptual representation of an agent at different complexity levels, ranging from raw sensory data to the features extracted from the environment. Although for some affordances the term object would perfectly encapsulate the environmental relata, for others, the relata may be too complex to be confined to an object - such as the layout among multiple objects. In the rest of the chapter, we will freely use object instead of *entity* for the sake of clarity. 'Behavior' represents the physical embodiment of the agent's interaction. It is an internal representation that defines a unit of action that can often take parameters for the initiation and online control. As in the entity definition, the level of complexity is not part of the definition; therefore a simple joint rotation, as well as a grasping action directed to an object can be considered as behaviors. Finally, the term *effect* denotes the change in perceptual state and environment that is generated by the agent's execution of the *behavior* on the *entity*. More specifically, a certain behavior applied to a certain entity should produce a certain *effect*. For instance, the *lift-ability* affordance implicitly assumes that, when the *lift* behavior is applied to a *can*, it produces the effect *lifted*[1], meaning that the can's position, as perceived by the agent, is elevated.

Based on these arguments, we argue that through its interactions with a can, a robot can acquire *relation instances* of the form:

$$(\textit{lifted}, (\textit{black-can}, \textit{lift-with-right-hand}))$$

[1] Note that 'lifted effect' is a label used to describe the sensory change here; in the agent's world this just corresponds to an internal representation which is not assigned any label.

meaning that there exists a potential to generate the effect *lifted* when *lift-with-right-hand* is applied to *black-can*. Note that the term *black-can* is a label for us humans indicating the perceptual representation of the black can by the interacting agent. Similarly, *lifted* and *lift-with-right-hand* are labels for the related perceptual and proprioceptive representations. For instance the representation of black can be a raw feature vector derived from all the sensors of the robot looking at the *black-can* before it attempts to apply its lift behavior.

Arguing that affordances should be relations with predictive abilities, rather than a set of unconnected relation instances, we proposed a process of generating *equivalence classes* that can be applied on this representation. For instance, a robot can achieve the effect *lifted*, by applying the *lift-with-right-hand* behavior on a *black-can*, or a *blue-can*. It can thus learn a relation:

$$(\textit{lifted}, (<\textit{*-can}>, \textit{lift-with-right-hand}))$$

where <*-*can*> denotes the derived invariants of the entity equivalence class.

The nesting inside the affordance triplet provided support for planning over learned affordances (as shown in the following Chapters), and can be removed within the context of this study for simplicity as:

$$(\textit{lifted}, <\textit{*-can}>, \textit{lift-with-right-hand})$$

2.4 Affordances in Robotics

2.4.1 Affordances and Mobile Robots

In Autonomous Robotics, the learning and perception of traversability[2] in mobile robots has recently started to attract interest. Although traversability can be considered a fundamental capability for mobile robots, it has long been limited to the problem of simple obstacle avoidance where the robot tries to avoid making any physical contact with the environment, and heads only to open spaces. In general, proximity sensors are employed to detect whether there is an object or not. When such approaches are used, the robot's response would be the same whether it encounters an impenetrable wall or a balloon that can just be pushed aside

[2] The verb traverse is defined as "to pass or move over, along, or through." Hence traversability refers to the affordance of being able to traverse. The learning and perception of traversability is a fundamental competence for both organisms and autonomous mobile robots because most of their actions depend on their mobility.

without any damage. A stair that is traversable for a hexapod robot may not be traversable for a wheeled one. Similarly a white vertical flat surface may be an impenetrable wall in one environment whereas in another environment a similar surface may be a door that can just be pushed to open. Therefore, a method that can automatically learn the traversability affordances from the robot's interactions with the environment would be valuable for robotics.

The interest in the learning of traversability has recently been fueled by the LAGR (Learning Applied to Ground Robots) program [73] whose aim was to support the development of algorithms that enable robots to navigate in off-road environments efficiently and robustly. In this program, the robots are required to learn the traversability characteristics of the environment and plan paths based on short-range and long-range traversability predictions. The traversability is not defined simply as obstacle avoidance, which prevents robot motion in grasslands and vegetated areas. The robots are expected to avoid from bushes and shrubs while driving over soft grasses of similar height.

Most teams competed in this program carried out training under the self-supervision of the robot's own signals, such as bumpers, inertial navigational system, wheel encoders and camera images [6, 128]. The robots learned and predicted short-range traversability of the environment mainly through range images obtained from laser range-finder or stereo vision; and long-range traversability through color camera features. Typically, the pixels of range image and color images were first projected onto local grid squares and then learning was performed in the constructed map environment. In this approach, regions in the Cartesian space of the robot were assigned as traversable and non-traversable, so that the robot avoided entering these regions. In [110] the robot planned paths directly over images obtained from color cameras, however their intermediate steps included high-level 3D processing such as ground plane detection, identification of points above the ground plane, and terrain slope computation. We think a projection onto a nominal ground plane should avoided since we believe that it is against the direct perception view. Further, the actions should be tightly coupled to perception of the affordances, i.e. move behaviors must be directly executed whenever they are afforded. In some of the cited studies, perception of obstacles and free-spaces were hand-coded and learning was done based on the obstacle-ground distribution of these objects [128]. In some other studies such as [6], features related to the physical affordances were carefully identified by hand and used directly, for example the ground plane and heights of the objects over this plane were explicitly computed. Contrary to those approaches, we ex-

pect automatic discovery of the relevant and invariant features through learning in a large, low-level and generic feature space. [80] discussed the traversability problem explicitly in relation to Gibsonian affordances, and claimed they "learn a direct mapping from observations to affordances", we believe that the use of maps in a global frame hardly classifies this study in that context.

Affordances were incorporated into the robot's world model in order to generate robust and simpler plans in [90]. The world was divided into pre-determined overlapping regions. Regions provided different affordances such as liftability and switch-triggerability, that corresponds to existence of objects and switches that can be liftable and triggerable respectively. The core idea of this work lies in the representation level of affordances. High-level region affordances were included in world model and planning was performed only based on this information. The robot explored the environment to gather this high-level information in initial stages, detected the particular objects or perceptual cues for corresponding affordances, however did not inform the world model about the details. After sufficient exploration of the environment, the plan was generated through use of region affordances. During plan execution, when robot's high level operators such as lift_in_region_1 was activated, robot tried to detect the *liftable* objects through its perceptual cue detectors and executed *lift* action.

2.4.2 Affordances and Manipulator Robots

In the context of manipulation based affordance learning, [46] studied the rollability affordances of the objects using vision, and claimed that manipulation can be used to ease and ground visual perception [45]. In [133, 132], Stoytchev et. al. studied the so-called 'binding affordances' and 'tool affordances', where learning binding affordances corresponds to discovering the behavior sequences that result in the robot arm binding to different kinds of objects, whereas learning tool affordances corresponds to discovering tool-behavior pairs that give the desired effects. Although these studies are important in the context of learning through exploration, in both studies, the objects were differentiated using their colors only, and no association between the visual features (that affect the affordances) of the objects and the corresponding affordances were established, giving no room for the generalization of the affordance knowledge for novel objects.

In [101], a general probabilistic model was proposed based on Bayesian networks to learn the

relationship between actions, objects, and effects through interaction with environment. The object properties that have no influence on other components of the system could be discovered by the network and filtered out during task execution, however the formed object classes were not based on the generated effects. Tool affordances for a robot were learned in [129] but the the object dealt with was kept fixed, so affordances of the objects were not learned. Fritz et al. [49] demonstrated a system that learned to predict the lift-ability affordance for different objects, where predictions were made based upon features of object regions extracted from camera images. In [60], the object affordances were learned through interaction for a task that requires categorization of container and non-container objects.

The concept of 'object-action complexes' (OACs), which argues that objects and actions are tightly linked, is also relevant to affordances. Along the lines of the concept of OACs, [86] used the assumption that combinations of certain visual features suggest certain grasp actions for 'things' in a scene, and named an 'object' as the set of visual features that move in the scene in accordance with the executing grasping action through a process called 'birth of objects'. This work was extended in [117] by learning effects of actions (such as filling, moving) from its preconditions and its effects. In [159], the concept of OACs was linked to the predictability of the environment and the body of the robot and how these can be used to improve the robots model of the world and itself.

2.4.3 Affordances and Robot Perception

Affordances provide perceptual economy as mentioned in Section 1.3.1 since the agent (either robot or animal) needs to perceive only the action-relevant properties of the environment to detect the affordances provided. Relevant regions/cues/features in the environment have been automatically discovered in many robotic tasks such as robot localization [161, 88], object tracking [77], and robot navigation [116, 15]. In [161] the features used for simultaneous localization and map building were filtered out to increase the speed of the process. The features that increase uncertainty in robot localization were filtered out using an entropy-based method. Similarly [88] selected the most discriminative features to recognize the location of the robot by measuring the information entropy that was calculated from posterior probabilities of location classes given the feature values. [77] selected a number of image points among many of the detected ones to track the moving objects from a mobile platform. In [116], the

robot was tele-operated first, discrete motor states were differentiated and the salient features that consistently co-occur in same motor states were discovered and later used autonomous navigation phase. In navigation of the robot, [15] selected and used the features that were persistent over the course of previous runs. These studies used feature selection as a means to discover features that will allow the recognition of 'visual landmarks' and did not use them to learn general relations about the environment. On the other hand, in [47] the action-relevant features were discovered through evolutionary algorithms and later used in robot navigation Specifically, a mobile robot was controlled by motor outputs of the network, which were activated by the weighted sum of perceptual inputs. The resolution, position, orientation as well as feature computation strategies were evolved for visually guided actions.

2.5 Affordance Experiments in Ecological Psychology

In Ecological Psychology, the learning and perception of traversability in organisms is probably one of the well-studied topics on affordances. Although it is not known precisely which visual cues are actually used in space perception [126], many organisms are known to use visual perception to detect whether the environment's spatial layout allows them to carry out their locomotor activities, such as crawling, walking or jumping.

Simple amphibians are known to perceive whether varying size barriers, apertures, overhead holes and pits afford locomotion or not. Toads, known to possess depth perception through stereopsis [24] tend to walk into shallow pits, and jump over deeper ones [89]. Leopard frogs, when challenged with a stimulus at their rear, tend to jump only through apertures that are larger than their own bodies [72]. The relation between the aperture and their body width is complex, since the dynamics of the interaction also depend on the orientation of the gap and frog's jumping direction. Thus, frog's choice of jumping through the gap has a 'realistic' relationship to its body width in absolute metrics only if the gap is placed directly in front of the frog. Still, the frog can correctly predict the 'pass-through-ability' affordance in different situations independent of absolute metrics as observed from outside. In another study [25], prey was presented to the toads behind fence barriers with gaps and the movement direction decisions of the toads were studied. It was observed that although the toads moved towards impassable fences and attempted to directly snap at the worms in some situations, they generally managed to choose the collision-free route by either passing through the gaps

or detouring around the barrier, depending on the placement of the worm. Further, it was shown the toads tend to select wider gaps [25] or larger over-head holes [125] and jump over smaller-size ones even the smaller size holes also afford jumping.

Visual space perception and sensitivity to differences in spatial layout are detected in humans at early ages. Newborns show visual sensitivity and attempt to interact with slanted objects [5]. Young infants withdraw their heads or lean forward when encountered obstacles and apertures before gaining their locomotion ability [160]. Older infants with crawling or walking ability are able to detect more complex affordances such as traversability of rigid/non-rigid surfaces and act accordingly [54]. They can use both haptic and visual information provided by the environment and perceive the traversability affordance implicitly taking into account their mode of locomotion. There also exist situations where haptic and visual cues are contradictory. For example in the so-called "visual cliff" experiments [53], crawling infants are placed on a glass surface part of which is placed on a table covered with a textured sheet, whereas the remaining part is kept on air (supported from only its sides). Thus although the glass is a rigid surface, the part on the table gives appearance of solidity, and the other part becomes a visual cliff. In such situations, crawling infants tend not to go over the apparently unsupported surface even if their mothers call them from the other side.

In experiments conducted with human adults, subjects were queried on the existence (or non-existence) of affordances. Warren's stair-climbing experiments [152] have generally been accepted as a seminal work on the analysis of affordances, constituting a baseline for later experiments which seek to understand affordance-based perception in humans. According to Warren, "to determine whether a given path affords locomotion, the behaviorally relevant properties of the environment must be analyzed in relation to relevant properties of the animal and its action system". Thus a specific set of values of the functionally related variables is identified for the activity of stair-climbing. Since the environment should be perceived in terms of *intrinsic* or *body-scaled* metrics, not in absolute or global dimensions, this specific set of values is expressed as π, a dimensionless ratio of animal property (leg-length) and environment property (stair-height). The particular value of these ratios that signaled the existence of an affordance were called the *critical points*. It was argued that critical points remain constant across humans with different body sizes and provide a natural basis for perceptual categories (e.g. categories of climb-able and not-climb-able). Moreover, these points reflect the underlying dynamics of the system and these categories can be correctly perceived by humans.

In one experiment, a number of tall and short human subjects were asked to judge whether different stair-ways looked climb-able or not, and the height of the stair where climb-ability disappears is recorded for each person. It was indeed found that the proportion of the recorded stair height to leg-length (a.k.a. *critical* π ratio) is constant regardless of the heights of the subjects. Furthermore, this predicted ratio is equal to the ratio calculated analytically taking into account the dynamics of a bi-pedal biomechanical system, which proves that people are able to correctly perceive traversability affordances provided by stairways.

Warren's studies were followed by studies that further explore the underlying mechanisms of traversability in different environments and that identify the visual channels and cues in human affordance perception. For example slanted surfaces were included into the environment in [81] and human subjects were shown to correctly predict the walk-on-ability affordance of slopes when they perceived them at a distance. The roles of optical and geographical slants which imply relative and absolute measures were discussed in the detection of these affordances. [153] studied which properties of the environment and human body are used in visual guidance for walking through apertures. The constant π ratio was defined as a proportion of the environment-related variable *aperture width* and action-related organism variable *shoulder width*. In the experiments, where subjects were asked to judge whether they can pass through apertures without rotating their shoulders, the predicted critical π ratio was found to be compatible with the real one, the one found by actually executing the actions. This critical ratio was also found to be constant among subjects with narrow and broad shoulders. It was further shown that static human subjects with monocular vision looking through a reduction screen (which limits the view) were as successful as moving subjects with binocular vision in the detection of pass-through-able apertures. Thus, stereo vision and optic flow were not necessarily involved in the process of traversability perception, however 'perceived eyeheight'[3] as an intrinsic measure is shown to be used. Traversability was also studied in environments with barriers [95], where human subjects were asked to judge the pass-under-ability of barriers at different heights. The predictions of the subjects were found to be valid as in previous experiments and compatible with the constant critical ratio π, defined as the proportion of subject-to-barrier height. Instead of passing-under, when the subjects were asked whether they can walk-over obstacles [28] or gaps [21] of different sizes, the traversability detection

[3] Eyeheight is known to be used to perceive the body-scaled geometrical dimensions such as size and distance of the objects [55]. In [154], eyeheight is defined as the height at which a person's eyes would pass through a wall while walking and looking straight in a natural and comfortable position.

was found to be successful as well.

In summary the experiments discussed above were generally used to show that organisms can perceive whether the surface layout affords traversability or not[4]. The properties of the organism and environment related to the action were identified and a dimensionless ratio between these properties was used to describe the dynamics of the affordances. Some studies further explored the nature of the perceptual cues used in detection of these affordances, however the question of how these cues are used precisely, still needs further elaboration.

2.6 Conclusion

In this chapter, we discussed affordances in the contexts of Ecological Psychology and Autonomous Robotics. We provided the formalism for using affordances in autonomous robot control. However, the details of this formalism in terms of affordance representation and learning is missing. Thus, in the next chapter, we will ground this formalism by providing a detailed description of how affordances will be represented by the robot. Additionally, we will describe the methods that enable the robots to discover the affordances provided by the environment, learn making predictions based on these affordances, and use this prediction ability in goal-oriented fashion in the next chapter.

[4] Humans can also perceive action possibilities not related to traversability such as sittability provided by surfaces[96] or graspability provided by objects[108]. For a more complete discussion of experiments not related to locomotion, please see [33].

CHAPTER 3

FORMAL DESCRIPTION OF AFFORDANCE LEARNING FRAMEWORK

In this chapter, the affordance learning framework is provided in its most generic form. The encoding of affordances is given in a relational structure that encapsulates the robot behaviors, the initial perception of the world, and the change in perception due to the behavior execution. Additionally, the multi-step unsupervised affordance learning method that use robot's interaction experience with the environment is described. At the end, how learned affordances can be utilized in goal-oriented robot control is discussed.

3.1 Affordance Encoding

As mentioned in the previous chapter, affordances are represented by (*effect*, *entity*, *behavior*) nested triplets. The actual encoding of these components varies based on robot's perceptual and actuation capabilities. In this thesis, the affordances framework is implemented in mobile and manipulation robot platforms, so especially the *behavior* component varies significantly among chapters. Furthermore, in some cases the environment is perceived as a whole and in other cases as a collection of detected objects. Thus, *entity* and *effect* can encode environment or object features depending on the learning targets.

3.1.1 Behavior Encoding: $b_i(\alpha)$

Behavior corresponds to an open-loop pre-defined action, represented by $b_i(\boldsymbol{\alpha})$, where i refers to the index of the behavior and $\boldsymbol{\alpha}$ represents the free parameter list of the behavior. There are

two types of behaviors:

Non-parametric Behaviors: Some behaviors are encoded as discrete actions that include no parameter. For example, *drive-forward* behavior which is implemented as driving the mobile robot forward for certain distance is such a behavior. In this case, size of parameter list ($|\boldsymbol{\alpha}|$) is zero and the behaviors are simply shown as b_i.

Parametric Behaviors: Some behaviors are modulated by a number of free parameters. For example, *power-grasp(α)* is a parametric behavior, where the robot hand approaches to the object from α direction and grasps it. In this case, the size of parameter list is one, and the behavior is represented as $b_i(\alpha)$.

Note that, object-oriented behaviors, such as *lift* or *grasp* use the object's position as an argument unlike behaviors such as *drive-forward*. Although object's position is also a parameter for those actions, it is not included into the parameter list ($\boldsymbol{\alpha}$) since it is not a free parameter, i.e. it is fixed given the object that is acted on.

3.1.2 Entity Encoding: $\boldsymbol{f}^{()}$

Entity corresponds to the initial perception of the robot before behavior execution. An *entity* can correspond to object features, environment features, robot's proprioception or any combination of this perceptual data. In other words, *entity* is encoded as a list of features that are computed by different perceptual processing channels. It is symbolized by $\boldsymbol{f}^{()}$, where $\boldsymbol{f}$ is the feature vector and the superscript $^{()}$ denotes that no behavior has been executed yet. In this thesis, $\boldsymbol{f}^{()}$ is computed either from the environment or a detected object, but not both. In other words, the robot can encode either object or environment properties in one entity. On the other hand, multi-object environment are perceived as a list of entities, and represented by $[\boldsymbol{f}^{()}_{o_1}, \boldsymbol{f}^{()}_{o_2} ... \boldsymbol{f}^{()}_{o_m}]$ where o_j is used as object identifier. Robot's proprioception and tactile sensor readings can also be included in entity representation in both cases.

3.1.3 Effect Encoding: $\boldsymbol{f}^{b_i}_{\textbf{effect}}$

Effect corresponds to the difference between final and initial perception of the robot and is

defined as the vectorial difference between final and initial features:

$$\boldsymbol{f}^{b_i}_{\text{effect}} = \boldsymbol{f}^{(b_i)} - \boldsymbol{f}^{()}$$

where $\boldsymbol{f}^{(b_i)}$ represents the feature vector of the entity perceived after b_i behavior is executed.

3.2 Exploration

In all experiments, the robot goes into an exploration phase in the simulator to gather experience that is later used in affordance learning. The exploration phase, consists of episodes, where the robot interacts with the objects, and monitors the changes. In the beginning of episode k, the robot first computes the feature vector $\boldsymbol{f}^{()}$ for the environment to be acted upon. Then the robot executes behavior b_i with parameters $\boldsymbol{\alpha}$ and computes the effect feature vector $\boldsymbol{f}^{b_i}_{\text{effect}}$. The robot executes all of its behaviors with different parameters in random situations and records its experience. The data from an interaction is recorded in the form of $< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i(\boldsymbol{\alpha}) >$ tuples, i.e. (*effect*, *entity*, *behavior*) instances (Algorithm 5).

3.3 Learning Affordances

In this section, we will discuss how gathered experience during exploration phase is utilized to learn the affordances of the objects. The data collected as tuples during the exploration phase are stored in a repository

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i(\boldsymbol{\alpha}) >\}$$

and is used by the robot to learn the affordances of objects. The learning process consists of three steps: the unsupervised discovery of effect categories, the discovery of relevant features for affordance prediction, and the training of classifiers to predict the effect categories from object features. The learning process is applied separately for each behavior as detailed below.

3.3.1 Discovering Effect Categories

In the first step, similar effects are grouped together to get a more general description of the effects that the behavior repertoire of the robot can create. In this thesis, we used the following methods to find effect categories:

- In Chapters 5 and 6, the effect categories are pre-defined and pre-coded as *success* and *failure* for each behavior.

- In Chapter 7, the robot self-discovers fixed number of effect categories using a standard clustering algorithm.

- In Chapters 8-10, the robot self-discovers variable number of effect categories using a novel hierarchical categorization method. In the lower level, channel-specific effect categories are found by clustering in the space of each feature channel, discovering separate categories for visibility, position, shape, etc. In the upper level, the channel-specific effect categories are combined to obtain all-channel effect categories using the Cartesian product operation. The proposed hierarchical clustering method is superior to simple one-level clustering method, since the results of one-level clustering is sensitive to the relative weighting of the effect features in different channels that are encoded in different units.

Each feature vector in the set of $\{\boldsymbol{f}^{b_i}_{\text{effect}}\}$ is assigned to one of the effect categories ($E^{b_i}_{id}$) during clustering process. Then, for each category a prototype effect vector ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$) is computed as the average of the category members. In order to represent the experience of the robot in a more compact way, the continuous effect vectors are replaced by effect category id's and their prototypes; and the repository is transformed into the following form:

$$\{E^{b_i}_{id}, \boldsymbol{f}^{()}, b_i\}, \{< E^{b_i}_{id}, \boldsymbol{f}^{b_i}_{\text{prototype},id} >\}$$

Here, the first list corresponds to the set of affordance relation instances where effects are generalized and the second one corresponds to the list of <effect-category-id, prototype vector> pairs.

3.3.2 Learning Relevant Features

The robot's perceptual system is bombarded with large amount of data received from its sensors. As discussed in the Introduction Chapter, it is sufficient to perceive only action-relevant properties of the environment to perceive the affordances. The robot benefits from this characteristics of affordance perception by finding the relevant features of each behavior and using only these features during affordance prediction and execution.

In Chapter 5, relevant features are selected based on a distance metric without considering how the selection would affect the performance in the later classification phase. In Chapters 8 and 9 on the other hand, feature relevance is measured based on feature's performance of the classifier used in the next phase, and this approach gives near-optimal results. The other chapters don't utilize relevant feature selection mechanism.

3.3.3 Learning Effect Category Prediction

In this step, classifiers are trained to predict the effect category for a given feature vector and a behavior parameter list by learning the $(\boldsymbol{f}^{()}_{\text{relevant}}, \boldsymbol{\alpha}) \rightarrow E^{b_i}_{id}$ mapping. Effectively, this establishes a forward model, $Predictor^{b_i}(\boldsymbol{f}^{()}_{\text{relevant}}, \boldsymbol{\alpha})$ that returns $E^{b_i}_{id}$ for each behavior.

At the end of these two learning steps, affordance relations are encoded as:

$$\{Predictor^{b_i}()\}, \{< E^{b_i}_{id}, \boldsymbol{f}^{b_i}_{\text{prototype},id} >\}$$

or

$$\{\{Predictor()\}, \{< E_{id}, \boldsymbol{f}_{\text{prototype},id} >\}\}^{b_i}$$

allowing the robot to 'know' the effect of a behavior in terms of the effect category and its prototype.

3.4 Use of Affordances in Task Execution

The predictors allow the robot to predict the *effect category* that is expected to be generated on an *entity* by a *behavior* that is controlled with a particular *parameter*:

$$E^{\text{predicted}}_{b_i,id} = Predictor^{b_i}(\boldsymbol{f}^{()}_{\text{relevant}}, \alpha) \tag{3.1}$$

The predicted percept of the entity can be found as:

$$\boldsymbol{f'}^{(b_i(\alpha))} = FM^{b_i}(\boldsymbol{f}^{()}, \alpha) = \boldsymbol{f}^{()} + \boldsymbol{f}^{b_i}_{\text{prototype},id^{\text{predicted}}} \tag{3.2}$$

Effectively, this corresponds to a forward model ($FM()$) that returns the next perceptual state of the entity. By successively applying this model, the robot can predict the perceptual state of the entity for any number of sequentially executed behaviors.

Different control systems are utilized to use the learned affordance prediction capability in a goal-oriented way for various tasks in the following chapters.

- The goals can be set to obtain certain effect categories such as *traversed* or *lifted*. In Chapters 5 and 6, the effect categories are defined as *success* and *fail* for traversability in different directions. During execution, the robot finds the set of behaviors (movement directions) that are predicted to result in *successful* traversability by predicting the effect categories using Equation 3.1. Then it chooses one of these behaviors either based on a priority mechanism or in order to minimize the risk of collision.

- The goals can be set to achieve desired percept (entities). For example, in order to achieve a goal where the object gets close to the mobile robot, entities' desired distance feature is set accordingly. As another example, if the goal is to bring the objects to a fixed position, entities' desired position features are set accordingly. During execution, if the current entity does not satisfy the desired constraints, i.e. current entities' corresponding feature is not close to the desired feature value, the robot needs to find the sequence of behaviors that are predicted to transform the current values to the desired ones. For this purpose, the robot makes multi-step predictions using Equation 3.2, starting from its initial entity feature vector and finds a sequence of behaviors which are predicted to transform current entity to the desired one. In Chapter 7 the desired entity features (the goals) are manually set by the programmer. In Chapter 8, in observation stage, the goal is shown to the robot and the robot encodes the goal in terms of desired entity features itself.

3.5 Conclusion

In this chapter, the details of affordance encoding is described in a relational structure that encapsulates *behaviors*, *entities* and *effects*. Furthermore, the multi-step affordance learning method, where the effect categories are discovered in the first step, relevant features are found in the second step, and the mapping from entities to effect categories are learned in the final step, is given.

The realization and implementation of this framework was progressively developed through time. The following chapters, where affordances framework is realized with different robot

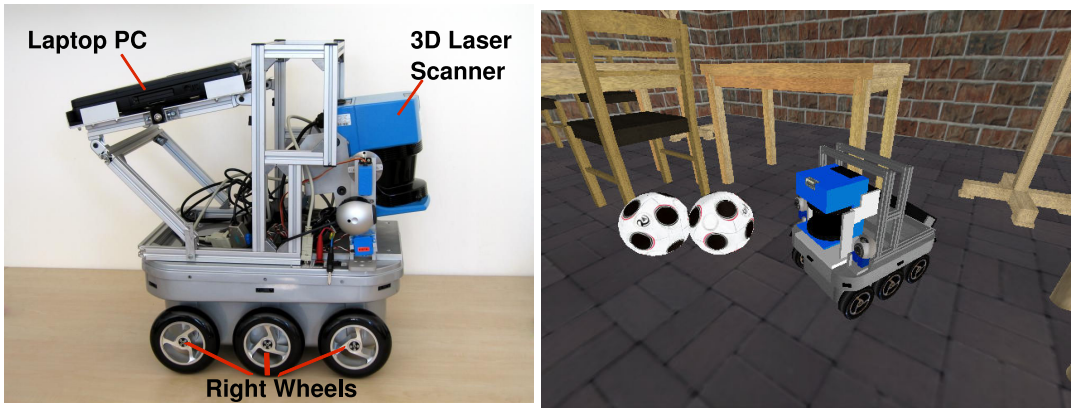
experiments, implement different (sometimes overlapping) parts of this framework. While the former chapters include limited versions of this framework, later chapters tend to be more inclusive. These limitations are characterized by postulating simplifying assumptions in affordance representation and learning mechanisms. On the other hand, the robot perception, the behavior representation, and the details of the learning algorithms also differ among experiments. Thus, in the beginning of each chapter, a section named **Framework Implementation** gives details of the postulated assumptions that will progressively relaxed.

CHAPTER 4

ROBOT PLATFORMS

In this thesis, two different robot platforms are used to learn and test the rich set of affordances. The first platform is composed of a mobile robot (KURT3D) with a simple manipulator and a 3D laser range finder. KURT3D is mainly used to study traversability affordances in Chapters 5-7. The second platform is a 23 degree of freedom (DOF) anthropomorphic hand-arm robot system with a 3D infrared range sensor. This platform is used to discover action primitives and to study manipulation affordances such as graspability, rollability or reachability in Chapters 8-10.

4.1 Mobile Robot Platform



(a) The robot platform (b) The simulator

Figure 4.1: The KURT3D mobile robot platform and its simulator.

A medium-sized ($45cm \times 33cm \times 47cm$) differential drive mobile robot (Kurt3D), equipped with a 3D range finder, and its physics-based simulator, is used as the first experimental

Figure 4.2: A sample range image from mobile robot.

platform (Figure 4.1). The 3D range finder is based on SICK LMS 200 2D laser scanner, rotated vertically with an RC-servo motor [70]. The 3D laser scanner has a horizontal range of 180°, and is able to sweep a vertical range of ±82.8° in 45 seconds to produce a 720×720 range image. A sample range image is shown in Figure 4.2.

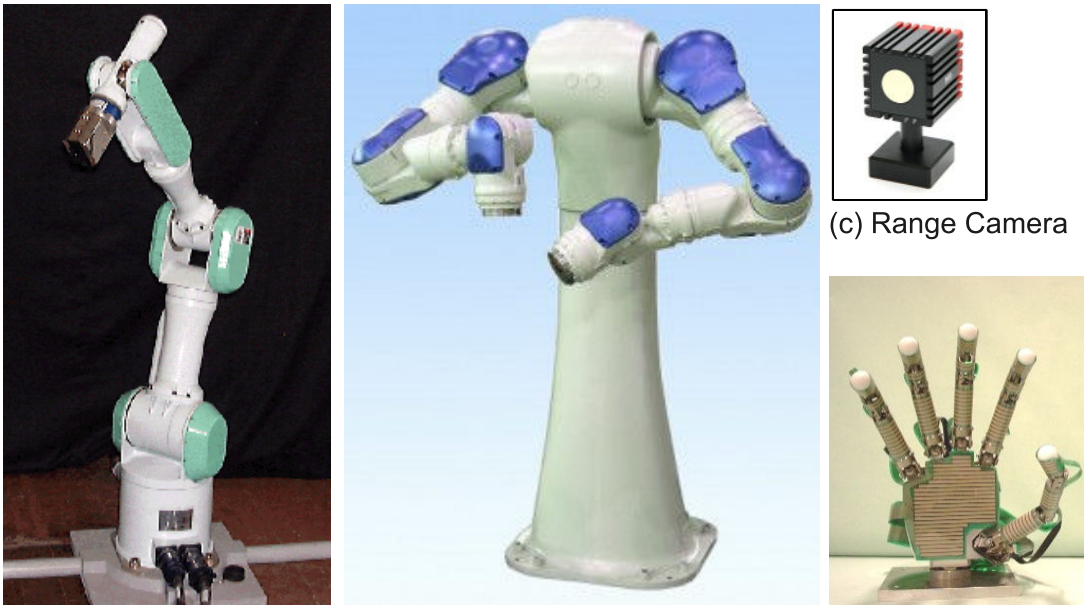
A crane arm is mounted on top of the robot with 3 degrees of freedom and an electromagnetic gripper at the end of the arm is used to manipulate magnetizable objects. The arm can rotate around itself, move the gripper back-and-forth in a range of 55*cm*, and lift its magnet up and down.

The laser scanner, Kurt3D and robot's environment is simulated in MACSim [147], a physics-based simulator that is built using ODE (Open Dynamics Engine) [130], an open-source physics engine. The sensor and actuator models are calibrated against their real counterparts. Fig. 4.1(b) shows a scene from the simulator.

4.2 Manipulator Robot Platform

An anthropomorphic robotic system, equipped with a range camera, and its physics-based simulator is used as the second experimental platform. This system uses 7 DOF robot arms, either PA-10[1] robot that is placed on the ground or Motoman[2] robot that is placed on a vertical bar similar to human arm as shown in Figure 4.3. A five fingered 16 DOF robot hand[3] is

[1] Mitsubishi Heavy Industries.
[2] Yaskawa Electric Corporation [29]
[3] Gifu Hand III, Dainichi Co. Ltd., Japan [91]



(a) PA10 arm (b) Motoman arm (c) Range Camera (d) Gifu hand

Figure 4.3: The actuator and sensors for manipulator platform. Only the right Motoman arm is used in this thesis. The distributed tactile sensor on Gifu hand are visible as horizontal bars inside palm and on fingers.

mounted on the arms to enable manipulation. The maximum length of PA-10, Motoman and Gifu hand is 134 cm., 123 cm., and 23 cm., respectively. There are tactile sensors distributed on the surface of the fingers and palm with a total number 624 measurement points [78]. For environment perception, an infrared range camera [4], with 176x144 pixel array, 0.23° angular resolution and 1 cm distance accuracy is used. Along with the range image, the camera also provides grayscale image of the scene and a confidence value for each pixel (Figure 4.4).

The simulator (Figure 4.5), developed using the Open Dynamics Engine (ODE) library, is used during the exploration phase. The parameters of the simulator, such as friction, mass of the objects, forces on robot hand and arm are adjusted to make the interactions realistic. The tactile sensor is simulated by placing one binary touch sensor to the palm and each finger link, obtaining $3 \times 5 + 1 = 16$ total touch values. The range camera is simulated by sending a 176×144 ray array from camera center with 0.23° angular intervals. For each ray, the first contact with any surface is retrieved using ODE functions, distance between the contact point and ray origin point is used as range value, and a Gaussian noise with $\mu = 0, \sigma^2 = 0.2$ is added to account for camera noise. The range camera's accuracy is best between 1-2 meters.

[4] SwissRanger SR-4000 [71]

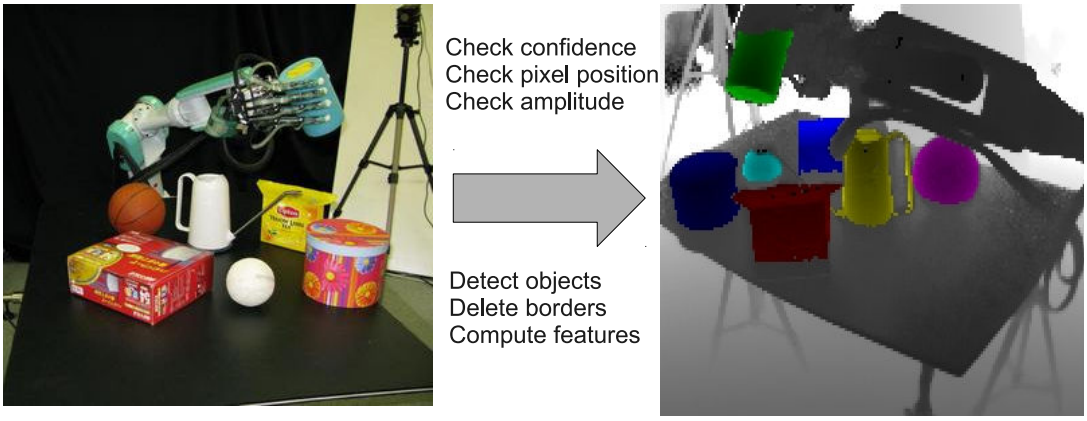


Figure 4.4: 23 DOF hand-arm robotic platform and the range image. In (a), the hand-arm system, infrared range camera (on the top-right) and the objects that are used in this study are shown. In (b), the range image obtained from the range camera and the detected objects are shown where range is encoded in grayscale and in color for the environment and objects, respectively.

In order to avoid any collision with the robot arm, the range camera is placed 1 meter away to the left of the robot as shown in Figure 4.4. The noise inherent to infrared range camera that is caused by surface characteristics and color of the objects is not modelled. Instead, the objects with surfaces that give consistent and accurate readings are used in the real world experiments. Additionally, as the object borders tend to give noisy readings, they are discarded.

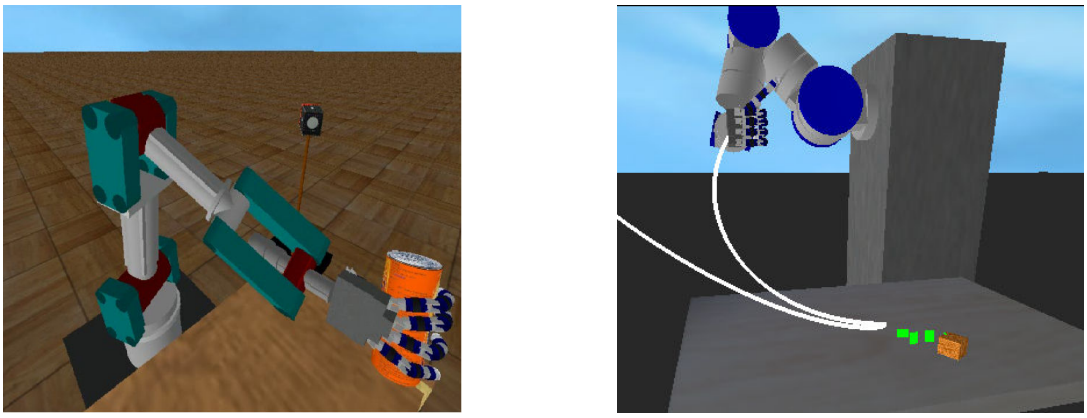


Figure 4.5: Simulation environment of manipulator robots.

CHAPTER 5

TRAVERSABILITY: A CASE STUDY FOR LEARNING AFFORDANCES IN MOBILE ROBOTS

In this chapter, we describe the implementation of one part of the affordance formalism [33] for learning and perception of traversability affordances on a mobile robot equipped with range sensing ability. Through systematic experiments, that are inspired by those used in Ecological Psychology, we show that the robot, by going through an exploration phase, can learn to perceive the traversability affordances in its environment, build a "sense of its of body size" and achieve perceptual economy.

5.1 Introduction

The environment is said to be traversable in a certain direction, if the robot (moving in that direction) is not enforced to stop as a result of contact with an obstacle. Thus, if the robot can push an object (by rolling it away), that environment is said to be traversable even if the object is on robot's path, and a collision occurs.

In this chapter, through experiments inspired by Ecological Psychology, we will show that the robot, by interacting with its environment, can learn to perceive the traversability affordances. We will consider three of the main attributes[1] that are commonly associated with affordances in robotics; namely,

- *Affordances are relative*. This argument, generally accepted within most contexts, is usually linked to the complementarity of the organism and the environment. According

[1] Although these arguments are certainly inspired from J.J. Gibson's own writings, we refrain from attributing them to him, in order to avoid the discussion of what he actually meant (or did not mean) in his writings.

to this view, the existence of an affordance is neither defined by the environment nor by the organism alone but through their interaction. For instance, the climb-ability of a stair step is not only determined by the metric measure of the step height, but also by one's leg-length.

- *Affordances provide perceptual economy*. The concept of affordances is often used as support for minimality in perception to argue that one do not have to perceive all the qualities of their environment in order to accomplish a simple task such as wandering around. In this sense, one would directly perceive the traversability of a path without recognizing the objects on its path and making additional "mental inferences" over them.

- *Affordances provide general information*. The discussion on affordances are mostly based on the general relations that pertain to the interaction of the organism with its environment such as sit-ability, climb-ability, and cross-ability. It is usually assumed that the use of affordances enables one to deduce whether a designer's chair that he sees for the first time would support sittability, or whether a coconut shell can be used to carry water in the place of a cup.

5.2 Framework Implementation

- **Behavior:** *Behaviors* correspond to discrete pre-defined actions without any parameter. Thus, they will be represented by b_i.

- **Entity:** The *entity* is computed from the whole perceived environment, without any object detection process. Thus, no object identifier is included in its notation. The entities are perceived only prior to behavior execution and they are not perceived after the execution. Thus, $^{()}$ superscript which includes the list of executed behaviors is dropped from $\boldsymbol{f}^{()}$ notation since the environment is not represented after any behavior execution. As a result, the entity feature vector is denoted by $\boldsymbol{f}$.

- **Effect category:** The *effect categories* are pre-defined (*success/traversable* and *fail/not-traversable*), and the means to compute these categories are provided by behavior designer. So, there is no effect category discovery step. On the other hand, the behavior designer does not provide effect category prototypes, thus the system can only learn

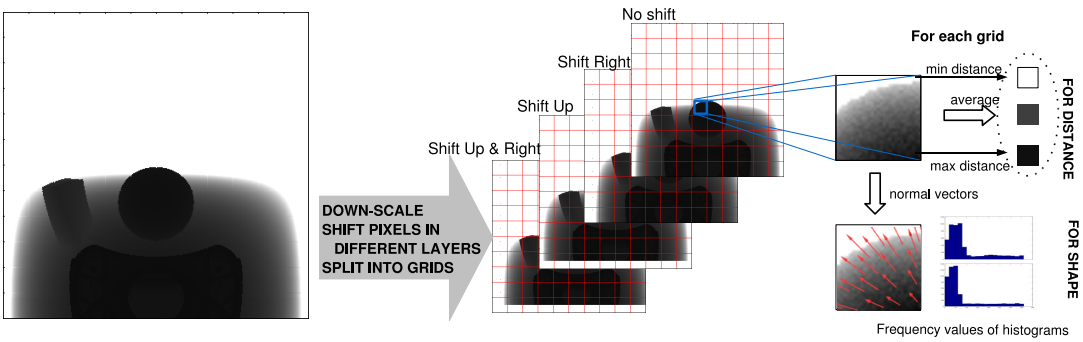


Figure 5.1: Perceptual processing of the range image.

whether a behavior will succeed or not. Since the *effect categories* are defined as *success* and *fail*, we will not use the effect category notation $E_{id}^{b_i}$. Instead, r^i is used to denote the success of behavior b_i.

- **Effect:** Since the robot does not perceive and represent the (*final*) entity after behavior execution, the change in entity feature vector, i.e. the *effect*, cannot be not computed. Thus, *effect* is neither represented nor used in this chapter.

- **Affordance relation instance:** The affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< r^i, \boldsymbol{f}, b_i >\}$$

5.3 Experimental Setup

5.3.1 Perception

The robot perceives the world through its 3D range scanner by scanning the environment to produce a range image of resolution 720×720. As sketched in Figure 5.1, the image is first down-scaled to 360×360 pixels in order to reduce the noise, and split into uniform size squares in a grid. The grid squares are shifted in order to have a representation that provides overlaps. Finally, low-level generic features are extracted for each grid square where 3 features are related to distance characteristics of the grid square and 36 features to shape. The features of the different grid squares are then collected and stored in a large one-dimensional feature vector $\boldsymbol{f}_k$ that represents the perception of the robot before k^{th} interaction.

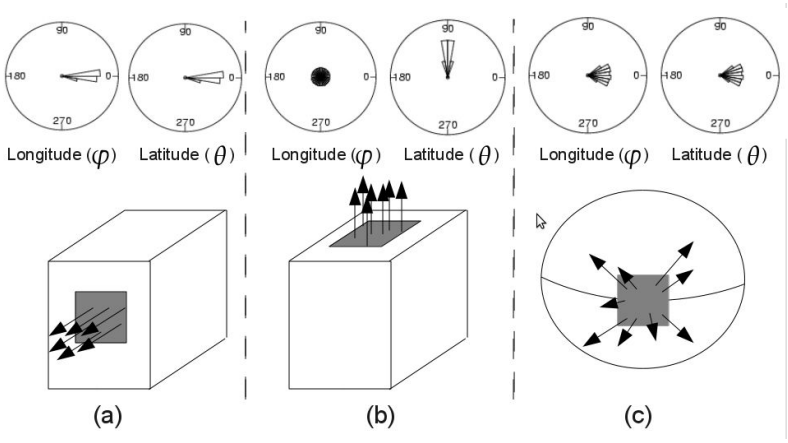


Figure 5.2: Sample angular histograms. **(a)** Vertical, **(b)** horizontal and **(c)** spherical surface patches and their corresponding angular histograms of normal vectors in latitude (θ) and longitude (φ). In (b), the orthogonal projection of the normal vectors onto the horizontal plane should create zero size vectors in ideal conditions and the angles in longitude in this situation should be undefined. However, the noise in sensing creates small random perturbations in these normal vectors which in turn results in randomly distributed angular histograms in longitude.

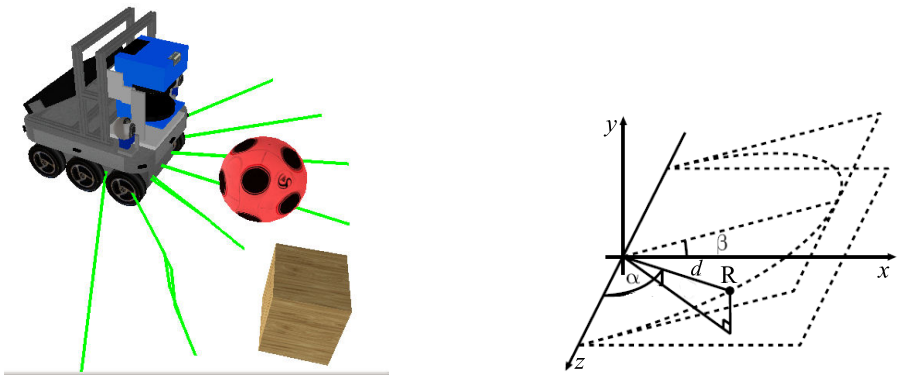
The distance-related features of each grid square are defined as the minimum, maximum, and mean range values of that grid square. In order to derive the shape-related features, the position of each pixel in the range image (see Figure 5.3(b)) relative to the laser scanner is computed using:

$$\mathbf{p}_{r,c} = \begin{bmatrix} d_{r,c} sin(\alpha_{r,c}) cos(\beta_{r,c}) \\ d_{r,c} sin(\alpha_{r,c}) sin(\beta_{r,c}) \\ d_{r,c} cos(\alpha_{r,c}) \end{bmatrix}$$

where d is the distance measured, r and c are the row and column indexes of the corresponding point, respectively. After finding the positions, the normal vector of the local surface around each point is computed by using the positions of the two neighbors in the range image:

$$\mathbf{N}_{r,c} = (\mathbf{p}_{r-n,c} - \mathbf{p}_{r,c}) \times (\mathbf{p}_{r,c-n} - \mathbf{p}_{r,c})$$

where n corresponds to the neighbor pixel distance and is set to 5. In spherical coordinates, the unit length 3D normal vector is represented by two angles, polar (θ) and azimuthal (φ) angles that encode information along latitude and longitude, respectively. The polar angle (θ) corresponds to the angle between y-axis and the normal vector, whereas φ is the angle between z-axis and the normal vector's orthogonal projection on x-z plane. After polar and azimuthal angles are computed for each pixel, angular histograms are computed in both dimensions for each grid square and are sliced into 18 intervals of 20° each. At the end, frequency values of



(a) (b)

Figure 5.3: Robot behaviors and 3D perception. **Left:** The simulated robot and the trajectories recorded during the execution of its 7 different move behaviors. Note that in some cases, the robot's movement is obstructed by the box. **Right:** The coordinate system of 3D scanning. The two planes correspond to the different 2-D slices sensed in 2D scanning.The laser beam reflected from R is transmitted at α degrees in the scanning plane of the scanner which has a pitch angle of β.

angular histograms are used as 36 shape related features. This representation encodes the distribution of the local surface normal vectors of the corresponding grid square. Figure 5.2 demonstrates distribution of normal vectors, and angular histograms corresponding to the particular grid squares in three different situations.

5.3.2 Behaviors

The robot is provided with seven pre-coded non-parametric behaviors to move in different directions. The execution of a behavior, b_j where $0 \leq j \leq 6$, consists of first rotating the robot in place for a certain angle (one of $0°, \pm 20°, \pm 40°, \pm 60°$), and then driving it forward for $70cm$, as shown in Figure. 5.3(a). The robot can measure its actual displacement and change in its orientation through its wheel encoders and use this information to detect whether the behavior succeeded or not. If the change in orientation is within $[-5°, +5°]$ and the displacement of the robot is within $[65cm, 75cm]$, the behavior is judged to be successful.

5.3.3 Interactions

The interaction of the robot with its environment consists of episodes. In episode k, the robot first computes the feature vector $\boldsymbol{f}_k$ for the environment to be acted upon. Then the robot

executes behavior b_j and records the result (r_k^j) as success or fail. During the *exploration phase*, which takes place in MACSim for obvious safety reasons, the robot executes all of its behaviors within a given environment and records its experience in the form of affordance relation instances as $< r_k^j, \boldsymbol{f}_k, b_j >$ triplets (Algorithm 1).

Algorithm 1 Exploration phase

for each trial k (from 1 to m) **do**
  Put the robot in a randomly constructed environment.
  Make a 3D scan
  Compute feature vector, $\boldsymbol{f}_k$
  for each behavior b_j **do**
    Perform b_j
    Find result of behavior, r_k^j.
    Put $< r_k^j, \boldsymbol{f}_k, b_j >$ into repository.
    Reset robot and object positions.
  end for
end for

5.4 Learning Affordances

For a given behavior, the robot discovers the relevant features of the environment for traversability (or non-traversability) and learns to map these relevant features to its affordances.

Within the context of this study, learning is conducted as a batch process that takes place after the exploration. The learning phase consists of two steps as explained in Algorithm 2 and is carried out separately for each behavior. In the first step of learning, the ReliefF method [82, 85] is used to automatically pick out the relevant features for the perception of traversability. This method estimates the relevancy of each feature, based on its impact on result of behavior execution (traversable/non-traversable). Specifically, the relevancy of a feature is increased, if the feature takes similar values for the situations that have same execution results, and it has different values for situations that have different results. Once the relevancy values for the features are calculated (see Algorithm 3), a threshold can be used to mark a subset of the features as relevant.

Algorithm 2 Learning phase

1: **for** each behavior b_j **do**
2: Fetch samples $< r_k^j, \boldsymbol{f}_k >$ from repository for behavior b_j.
3: Find a set of relevant features $\mathcal{F}^j$ using Algorithm 3
4: Train the SVM model, $Predictor^{b_j}()$, with relevant features.
5: Store $\mathcal{F}^j$ and $Predictor^{b_j}()$ for perception of affordances in *execution mode*.
6: **end for**

Algorithm 3 Computation of feature weights for behavior b_j

n_f: number of features

$\boldsymbol{w}_d$: weight of d^{th} feature

m: number of iterations, experimentally set to 1000

$\boldsymbol{f}_l$: the feature vector computed in l^{th} situation (interaction)

$\boldsymbol{f}_l[d]$: the normalized value of d^{th} feature computed in j^{th} situation (interaction)

1: $\boldsymbol{w}_d \leftarrow 0$, where $1 \leq d \leq n_f$, n_f is number of features (initialize weights)
2: **for** $i = 0$ to m **do**
3: Select a random feature vector $\boldsymbol{f}_l$ from $\{< r_k^j, \boldsymbol{f}_k, b_j >\}$
4: Compute distance of $\boldsymbol{f}_l$ to all other samples in $\{\boldsymbol{f}_k\}$
5: Find 10 feature vectors closest to $\boldsymbol{f}_l$ with execution results r_l^j, i.e. find the most similar situations to the l^{th} situation with the same result. Put them into set of nearest hits, $\mathcal{H}$. ($\mathcal{H} = \{\boldsymbol{f}_{1'}, ..\boldsymbol{f}_{10'}\}$)
6: Find 10 nearest feature vectors with execution results different from r_l^j, and put them into set of nearest misses, $\mathcal{M}$. ($\mathcal{M} = \{\boldsymbol{f}_{1''}, ..\boldsymbol{f}_{10''}\}$)
7: **for** $d = 0$ to n_f **do**
8: $\boldsymbol{w}_d \leftarrow \boldsymbol{w}_d - \frac{1}{m \cdot 10} \sum_{j=1}^{10} \mid \boldsymbol{f}_l[d] - \boldsymbol{f}_{j'}[d] \mid + \frac{1}{m \cdot 10} \sum_{j=1}^{10} \mid \boldsymbol{f}_l[d] - \boldsymbol{f}_{j''}[d] \mid$
9: **end for**
10: **end for**

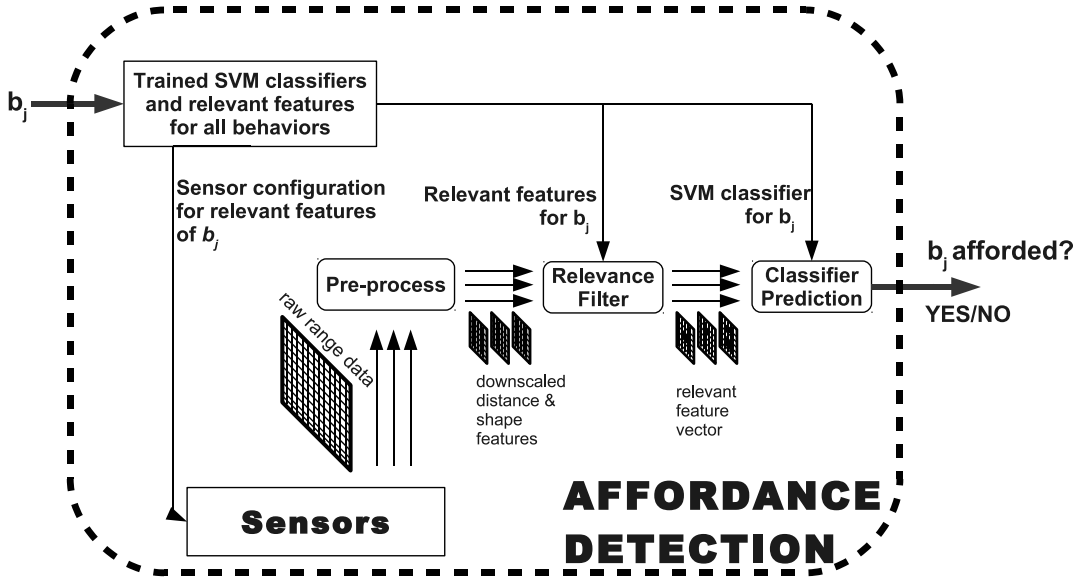


Figure 5.4: The affordance prediction module. This module receives the behavior id (or direction) as input and predicts the behavior's affordance based on the percept of the environment. Sensors are configured in order to compute only the relevant features. The previously trained SVM classifier for that particular behavior predicts the result of interaction.

During the second step of learning, SVMs (Support Vector Machines)[2] [148] are used to classify the relevant features into traversable or non-traversable target categories. The SVMs are chosen for their robustness against noisy inputs and their scalability in dealing with large datasets and input spaces (between 1000-5000 interaction samples and 100-500 features in our case). In SVMs, the optimal hyper-plane that separates traversable and non-traversable situations in feature space is learned based on the most informative points, also called the support vectors, in the training set. Although the literature has proposed several kernels to categorize linearly non-separable samples, we used a linear kernel (with a single tolerance parameter c) since more complex kernels did not increase the performance in our case.

5.5 Predicting Affordances

The set of relevant features and the SVM classifiers trained for each behavior can be used to predict the existence (or nonexistence) of the traversability affordance in a given environment using an affordance prediction system illustrated in Figure 5.4.

[2] The LibSVM software that is used in this study, is available at [19]

5.6 Experiments

We are interested in how a mobile robot, equipped with 3D range sensing ability, can perceive, learn and use the traversability affordance required to wander in an environment filled with different types of objects that change the traversability of the environment depending upon their shape, size, and relative position and orientation with respect to the robot. Towards this end, we used the following geometric objects and structures during the exploration phase:

- rectangular boxes ( ) that are non-traversable,
- spherical objects ( ) that are traversable since they could roll in all directions,
- cylindrical objects in upright position ( ) that are non-traversable,
- cylindrical objects lying on the ground ( ), that may or may not be traversable,
- ramps, that may or may not be traversable,
- gaps, that may or may not be traversable.

At this point, please keep in mind that the description provided above is rather crude for a number of reasons. First, the traversability of the robot is determined as a result of its physical interaction with the objects (which is implicitly implemented using the ODE physics-engine library). Second, the robot does not have the concept of an object, and our discussion at the level of objects is only to ease our discussion. Third, in our experiments, multiple objects can be present and the traversability is a complex function of not only the objects but also their layouts. Fourth, the size, relative placement and orientation of the objects vary during the experiments and hence their traversability.

5.6.1 Parameter optimization

Both the perceptual representation of the robot and the learning phase contains a number of parameters that needs to be optimized to obtain the best affordance prediction performance. Regarding the perceptual representation, the effect of grid size as well as the effect of overlapped (versus non-overlapped) grid representation needs to be decided. Regarding the learning phase, the relevancy threshold and the tolerance parameter to be used during the training

of the SVMs needs to be optimized. Towards this end, we carried out 5000 interactions during which the robot faced up to 12 objects that were placed at random locations and with different orientations. The objects are chosen to be boxes, cylinders (upright and lying), and spheres of random sizes.

As a result of the optimization process, described in the Appendix, we decided to use a representation with 5×5 grid with 4 overlapping layers for perception. Using these parameters, the feature vector $\boldsymbol{f}_k$ consists of $4 \times (5 \times 5) \times 39 = 3900$ features. The parameters of the learning phase were optimized for each behavior separately. Specifically, $100 - 400$ of the features were chosen to be relevant and the tolerance parameter was chosen as $250 - 500$. This setting provides a prediction accuracy of approximately 87% in environments randomly generated as described above.

A number of issues needs to be discussed to understand why the prediction accuracy is not higher: the traversability of an environment is a complex function of both the individual objects as well as the layout of the objects in the environment. First, the interaction between the robot and the objects can be complex. For instance, even small differences in the point of contact between the robot's body and a lying cylinder can affect the outcome of the interaction. Second, due to line-of-sight some objects may be invisible. For instance, a lying cylinder can become non-traversable due to a box behind it. Third, the grid square representation may lump patches from different objects, such as a patch from sphere and a patch from a box, producing confusing instances for the classifier.

In order to analyze the performance of the proposed method, we conducted experiments using both the simulated and the physical robot in different settings. These settings are inspired by the experimental settings used in Ecological Psychology with the aim of providing a more direct link to the studies carried out there. Specifically, we carried out experiments to evaluate:

- whether the learned *affordances are relative* to the robot,
- whether the learning of *affordances provided perceptual economy* to the robot, and
- whether the learned *affordances generalized well in novel environments*, that were not interacted during training.

In all the experiments reported below, unless otherwise stated, we carried out 5000 interac-

tions using the parameters obtained through the optimization process. During the evaluation of prediction accuracies, the training set was split into 5, and 5-fold cross-validation was performed.

5.6.2 Are Learned Affordances Relative?

The first set of experiments aimed to analyze whether the learned traversability affordances were related to the physical characteristics of the robot, such as its body dimensions and the capabilities of the robot.

5.6.2.1 Body Size

During the exploration phase, the robot is faced with a random number of boxes that hang (fixed) in space within the cubic $1m^3$ volume in front of the robot. The dimension [5cm – 20cm], position and orientation of the boxes as well as their number $[0-12]$ are kept random. The optimization process yielded approximately 160 relevant features for each behavior and the best affordance predictor was found to have a success rate of approximately 90%. In this experiment, the ratio of non-traversable environments in the exploration phase varied between $[39.64\%, 43.70\%]$ for different behaviors.

In order to analyze what the robot has actually learned, we conducted two experiments. In the first experiment, a box was placed on the ground and moved along the longitudinal and lateral axes within a $81cm^2$ area with 3cm gaps. For each position, the robot predicted the existence of traversability for the go-forward behavior. Figure 5.5(a) marks the positions where the robot predicted non-traversability. In the second experiment, the box was placed directly in front of the robot with no lateral deviation and moved along the longitudinal and vertical axes with 3cm gaps as illustrated in Figure 5.5(b). The results clearly show that the robot had acquired a "sense of its body" and was taking its body size into account in predicting traversability.

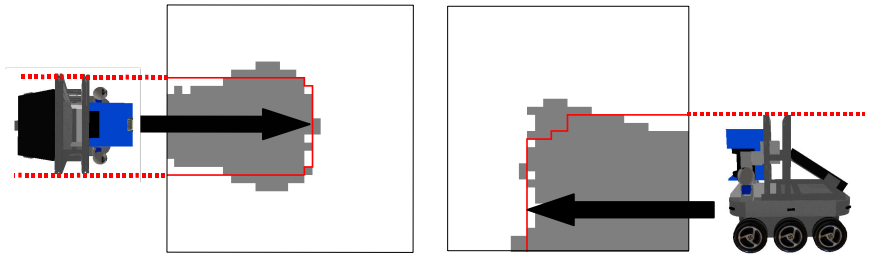


Figure 5.5: Collision boundary prediction by affordance perception. The robot's movement is shown with arrows. In (a) an object (illustrated by a small gray square) is shifted along longitudinal and lateral axes, and in (b) the object is shifted along longitudinal and vertical axes. Each gray square corresponds to a different setup that does not afford traversability. The objects that locate other than gray areas afford traversability. The lines that lie at the end of the movement arrow and extend from the robot's body correspond to real collision boundaries for the go-forward behavior. The height, width and depth of these lines correspond to critical points for drive-under-ability, pass-through-ability, and go-forward-ability, respectively.

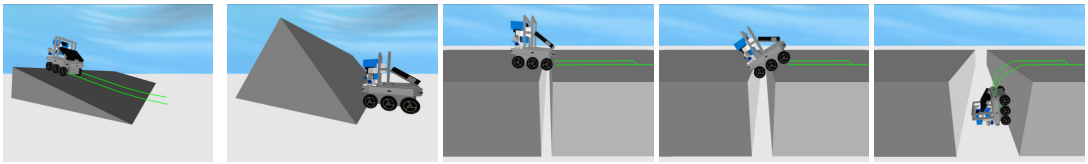


Figure 5.6: *Climb-able* and *not-climb-able* ramps; *cross-able* and *not-cross-able* gaps.

5.6.2.2 Ramps

During the exploration phase, the robot faced ramps placed in the vicinity at random position, orientation, width, height and slopes. As illustrated in Figure 5.6, the ramps afford traversability based on their slopes and relative orientations with respect to the robot. The robot interacted with four different move behaviors in each situation. During exploration, the ratio of non-traversable environments in the test set varied between $[77.10\%, 25.40\%]$ for different behaviors. The most relevant 160 features were selected to obtain approximately 95% prediction success.

In order to understand what the robot has actually learned, we compared the *critical slope* values of the ramp beyond which it becomes non-climb-able (corresponding to actual) or is perceived as non-climb-able (corresponding to predicted). Specifically, the ramp is placed in front of the robot in seven different orientations and is incrementally elevated. The predicted

Table 5.1: The predicted and actual *critical angles* of the ramps for climb-ability.

Ramp Orient.	Predicted/Actual *Critical Angles* for Behaviors		
	↑	↗	↗
-45°	12° / 11.5°	33.5° / 28°	- / -
-30°	10° / 8.5°	15.5° / 13.5°	30.5° / 26.5°
-15°	8° / 7.5°	9.5° / 9.5°	32° / 33.5°
0°	9° / 8°	7.5° / 8°	14° / 11°
15°	8° / 7.5°	8.5° / 8°	8.5° / 8.5°
30°	9° / 8.5°	8.5° / 8°	8° / 7.5°
45°	12° / 11.5°	9° / 8.5°	9° / 8.5°

and actual critical angles are shown in Table 5.1. A couple of observations can be made based on these results. First, as expected, the actual critical angles change with the relative orientation of the ramp with respect to the robot as well as to the type of the behavior. Second, the predicted critical angles are very close to the actual values, indicating that the learning was successful. Note that these critical values are a function of many things, such as the friction between the robot's wheels and the floor, the weight of the robot and the power of its motors, and is difficult, if not impossible, to model.

5.6.2.3 Gaps

During the exploration phase, the robot faces gaps on the ground that are randomly placed within a distance of [10 cm - 100 cm] in front of the robot. As illustrated in Figure 5.6, the gaps afford traversability based on their width and relative orientation with respect to the robot. The ratio of non-traversable environments in exploration phase varied between $[51.40\%, 26.25\%]$ for different behaviors. As in the previous experiment, after a 5-fold cross validation training was completed, the most relevant 160 features were used to achieve an average of 95% of prediction success.

We systematically analyzed the change in *critical width* for cross-ability affordance, and compared the actual and predicted values in Table 5.2. The results show that both the actual and predicted *critical width*s change with the relative orientation of the gap. This result is compatible with the dynamics of a differential drive robot since it becomes difficult to cross a gap when the gap orientation is perpendicular to robot motion.

Table 5.2: The predicted and actual *critical widths* of gaps for cross-ability.

Gap Orient.	Predicted/Actual *Critical Widths* for Behaviors		
	↑	↗	↗
-45°	30cm / 30cm	33cm / 33cm	36cm / 36cm
-30°	27cm / 30cm	27cm / 27cm	24cm / 24cm
-15°	21cm / 21cm	15cm / 18cm	15cm / 15cm
0°	12cm / 12cm	9cm / 18cm	15cm / 21cm
15°	24cm / 24cm	24cm / 24cm	21cm / 24cm
30°	30cm / 30cm	24cm / 27cm	21cm / 24cm
45°	30cm / 30cm	21cm / 27cm	27cm / 33cm

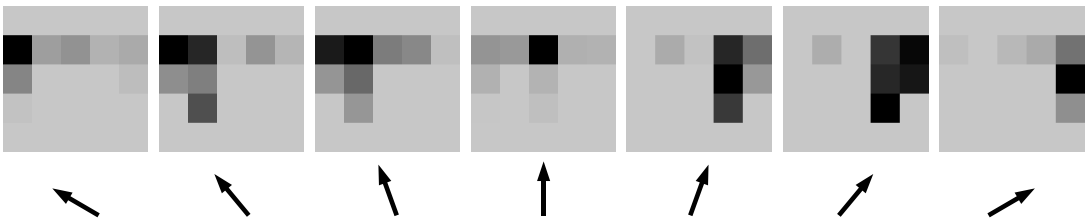


Figure 5.7: Grids relevant for different move behaviors.

The experimental results reported in this section have shown that the learned traversability affordances are relative with respect to the physical embodiment and capabilities of the robot. We argue that these experiments can be considered similar to Warren and Whang's studies [153] on the go-through-ability of apertures (relating one's shoulder width to aperture widths), Marcilly and Luyat's study [95] on pass-under-ability of barriers, Kinsella-Shaw et al.'s study [81] on the walkability of slanted surfaces, and Jiang and Mark's study [75] on the cross-ability of gaps.

5.6.3 Do Learned Affordances Provide Perceptual Economy?

We analyzed the number of relevant features chosen during the experiment described in Section 5.6.1. With the optimized threshold, 100 – 400 features among 3900 were selected to be relevant to perceive the traversability affordance for each different behavior. In other words, at most, 10% of the whole feature set was found to be relevant to determine whether a behavior is afforded or not.

5.6.3.1 Spatial Distribution of Relevant Features

Figure 5.7 shows the grid squares that include relevant features for each of the behaviors. In the plots, the darkness of a grid square is proportional to its relevancy. Couple of observations can be made at this point. First, the bottom of the range image corresponding to the robot's body, and the top of the image which lies above the robot's height were discovered to be irrelevant for all the behaviors. Second, the relevant grid squares tend to be aligned with the direction of the movement. Only the grid squares at the center of the range image are discovered to be relevant for the go-forward behavior, whereas the relevant grid squares for behaviors that turn left are grouped on the left part of the range image. The locations of the relevant grid squares and direction of movement are not consistent in some cases due to the existence of very close large non-traversable objects which cover the range image in the training phase. The non-symmetrical distribution of the relevant grid squares for symmetrical behaviors is probably due to the use of a relatively small training set with respect to the size of the feature vector.

A closer inspection of the relevancy grid squares also reveals that vertical shape of the objects are more important than the horizontal shape for the traversability affordance. Although and  have horizontally different shapes, they have the same traversability affordance. On the other hand, the vertical shape distinguishes the traversabilities of ,  , and .

5.6.3.2 Distribution of Feature Categories

We grouped the features into *i*) distance related ones, *ii*) shape related ones in lateral axis, and *iii*) shape related ones in longitudinal axis in order to analyze their relevancy. When the most relevant 20 features are considered, 65%, 30%, and 5% of them correspond to distance, lateral shape, and longitudinal shape related groups, respectively. Hence, the vertical shape of the objects is more important than their horizontal shape for perceiving traversability. Although and have horizontally different shapes, they have the same traversability affordance. On the other hand, the vertical shape distinguishes the traversabilities of , , and . Note however that if one considers all the most relevant 320 features, the number of features of shape relate groups becomes much larger than the distance related group since the shape related groups include more features (18 features each) compared to distance related group (3

features).

5.6.4 Do Learned Affordances Generalize?

The experiments reported above have used similar testing environments to the ones used during training. Although the randomness in the size, placement and orientation of the objects as well as the randomness between the layout of the objects in the environment indicate that the affordances learned by the robot can successfully generalize, one still wonders how well the learned affordances will generalize to objects with which the robot has never interacted during the exploration phase.

5.6.4.1 Novel Objects

In this section, the generalization capability of the system when encountered with novel structured objects is analyzed. In previous experiments, training was performed with all types of objects included. In order to assess the generalization performance, the robot should encounter with objects it had never seen during training phase. Since such a training should be done in the lack of some object types, the training setup is constrained to include only a subset of object types. Testing, on the other hand, is performed with all types of objects, so that the affordance prediction for novel situations can be evaluated. In both training and testing, only one object is placed in front of the robot, and the go-forward behavior is executed.

Table 5.3 shows the results obtained from 15 different experiments. The left-most two columns of the table show case number and the set of objects in the environment where the corresponding classifier is trained. The second row shows the object types used for testing and the prediction accuracy obtained. The following observations can be made:

- When the training set includes only traversable objects (case 4), the classifier predicts traversability in all cases. When only non-traversable objects are included (cases 2, 3), the traversability of the environment is mainly determined by the relative position of the object. For instance in cases 2 and 3, spheres are predicted to be traversable with 30% accuracy.
- In case 1, the robot is trained with only , yet it is able to predict the affordances

Table 5.3: Generalization of learned affordances. The left-most two columns show case number and the set of objects in the environment where the corresponding classifier is trained. The second row shows which object types are included into the test sample set, where each set contains only one object type. For each of the given training set, and test object, the accuracy of the learned classifier's predictions are given in the rest of the table.

Case	Training obj. types	Prediction Accuracy (%)			
1		96	95	86	100
2		66	97	94	31
3		75	98	99	30
4		55	30	34	100
5		96	98	91	88
6		97	98	96	80
7		95	83	78	100
8		70	97	97	30
9		93	93	86	100
10		95	93	91	100
11		95	98	93	81
12		97	92	87	100
13		95	98	94	100
14		95	96	90	100
15		95	97	92	100

of all other object types that it did not interact with before. This is due to the fact that a can be traversable or non-traversable, depending on its relative orientation with respect to the robot. In this case, the classifier correctly predicts that and are non-traversable (95% and 86% success), and that were traversable (100% success).

- We believe that the main reason behind the successful prediction of traversability for novel objects in cases 5, 6, 7, 11, 12, 13, and 15 is due to the inclusion of in training. The accuracy for is increased in some cases because other objects included into the training have similarities with . For example has similar shape distribution on vertical axes, and thus a similar affordance to .

- In case 9, since the training set contains samples for both success and fail, the affordances of novel objects (and) are also correctly predicted.

- Case 15 includes all types of objects and the corresponding classifier gives best results when compared to the others.

As a result, we can say that our method successfully predicts the affordances of the completely

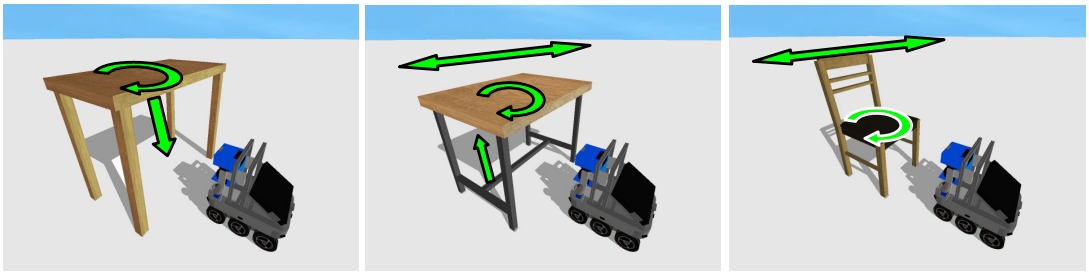


Figure 5.8: Left: The wooden table. Middle: The iron table. Right: The chair.

novel object classes that were never met before.

5.6.4.2 Complex Real-World Objects

After the robot was trained with only simple objects, as reported in Section 5.6.1, we evaluated the affordance prediction ability of the robot on the following complex real-world objects, over 1000 interactions:

- **A wooden table** with a leg width of 5.5*cm* is placed in front of the robot in random orientations. When the table is placed almost orthogonal to robot's movement direction, the robot can drive through both wide and narrow legs of the the wooden table. The height and orientation of the table is varied so that in some situations the robot can drive below the table and in some situations it cannot. The prediction success rate obtained is 85.5% in a test set in which 64.9% of the tables are non-traversable .

- **An iron table** with a leg width of 3.5*cm* is placed in front of the robot. The robot cannot drive under the table since the bars that connect the legs prevent the robot's locomotion. In this case, the table is not only just rotated in place but is also shifted laterally with respect to the robot, and the height of the connecting bars were also changed to enable locomotion. Because of the connecting bars, the ratio of non-traversable situations in the test set is higher then previous one (74.2%) . The robot is observed to correctly predict traversability with 81.7% success. The prediction accuracy is low when compared to wooden table case since the legs and bars that connect the legs are smaller in size. Note that objects included during the training phase have dimensions of at least of 5*cm*.

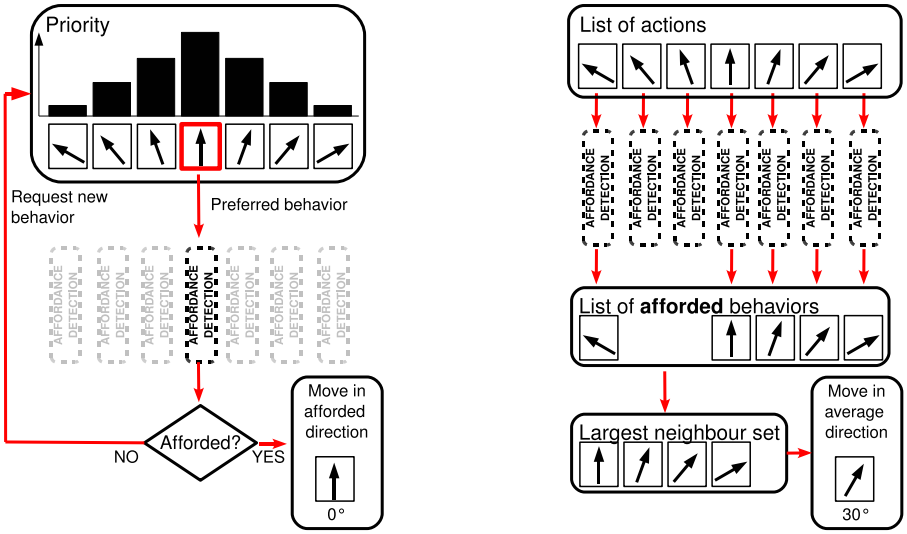
- **A chair** with a seat, a backrest part with supporting rails, and legs of 4*cm* in width is placed in front of the robot. The structure of the chair is kept fixed in the experiments, but its position and orientation with respect to the robot was varied. The robot predicts the traversability with 94.7% accuracy in the test set with 68.4% ratio of non-traversable cases . This accuracy rate is very close to the rates presented in Section 5.6.2.1 Figure 5.5 (a), where boxes are shifted laterally in front of the robot. Although the width of the legs were small, the robot correctly predicts the non-traversability of the chair successfully, due to the seat. Note that seat of the chair is never elevated so it is always predicted as non-traversable.

5.6.5 Full Demonstration

The results reported so far have been obtained in only relatively simple and episodic experiments. In order to use the learned affordance predictors in navigating a robot in a relatively unstructured environment, we have proposed two execution architectures for *aggressive navigation* and *cautious navigation*, as sketched in Figure 5.9.

The aggressive navigation architecture tries to minimize the turns the robot has to make during navigation by prioritizing the move behaviors, as shown in Figure 5.9(a). It prefers to go-forward as much as possible and make minimal turns. The architecture uses the affordance prediction modules shown in Figure 5.4 to detect the affordances. Specifically, the architecture queries the existence of affordances for each behavior in the order of priority and executes the first one that is supported by the environment.

We used the classifiers trained through interactions with simple objects, as reported in Section 5.6.1, in a virtual room cluttered with office objects of different sizes and types. The trajectory of the robot, controlled using the aggressive navigation architecture, is shown in Figure 5.10. We would like to comment on the decisions made by the robot at the positions marked with numbers on the figure. Situation (1) is predicted to be traversable for the go-forward behavior since the table is high enough to drive-under, and the width between the legs is wide enough to pass through. Situations (4) and (5) are not traversable for the go-forward since the coffee table is not sufficiently high and the aperture between the legs of the shelf is narrow. The robot is able to pass-through the legs of different tables in situation (2) and correctly predicts the traversability of the garbage bins in situation (3). The robot makes



(a) Aggressive navigation

(b) Cautious navigation

Figure 5.9: The mobile robot control systems in execution mode. In **aggressive navigation**, the high-priority behavior is immediately executed if it is afforded. In **cautious navigation**, all afforded behaviors are considered and the average direction of neighbor afforded behaviors is used.

an incorrect prediction in the last step (6), i.e. it predicts that the aperture width between the leg of the table and extension of the hanger affords traversability. Note that the robot does not have a preference for driving towards open spaces and can prefer driving towards spherical and cylindrical objects equally, wherever they afford traversability.

The cautious navigation architecture, sketched in Figure 5.9(b), takes a more conservative approach in order to minimize the risk of collisions. In this architecture, the robot is also driven using the learned affordance detection system. However different from aggressive navigation, the robot moves only if more than one neighbor behavior is afforded. The largest set of afforded behaviors that are neighbors to each other is identified based on the SVM classifier, the average direction of the corresponding behaviors is computed, and the robot moves in that average direction. As a result, new behaviors would automatically be added to the repertoire where the robot turns at angles of a factor of 10° and move forwards, not only a factor of 20° as originally designed.

We used the classifiers trained with interactions obtained from simple objects, as reported in Section 5.6.1, on the physical KURT3D robot platform using the cautious navigation architecture. As shown in Figure 5.11, Kurt3D is placed in an office environment with objects that have different physical characteristics and affordances. In this experiment, the robot decides

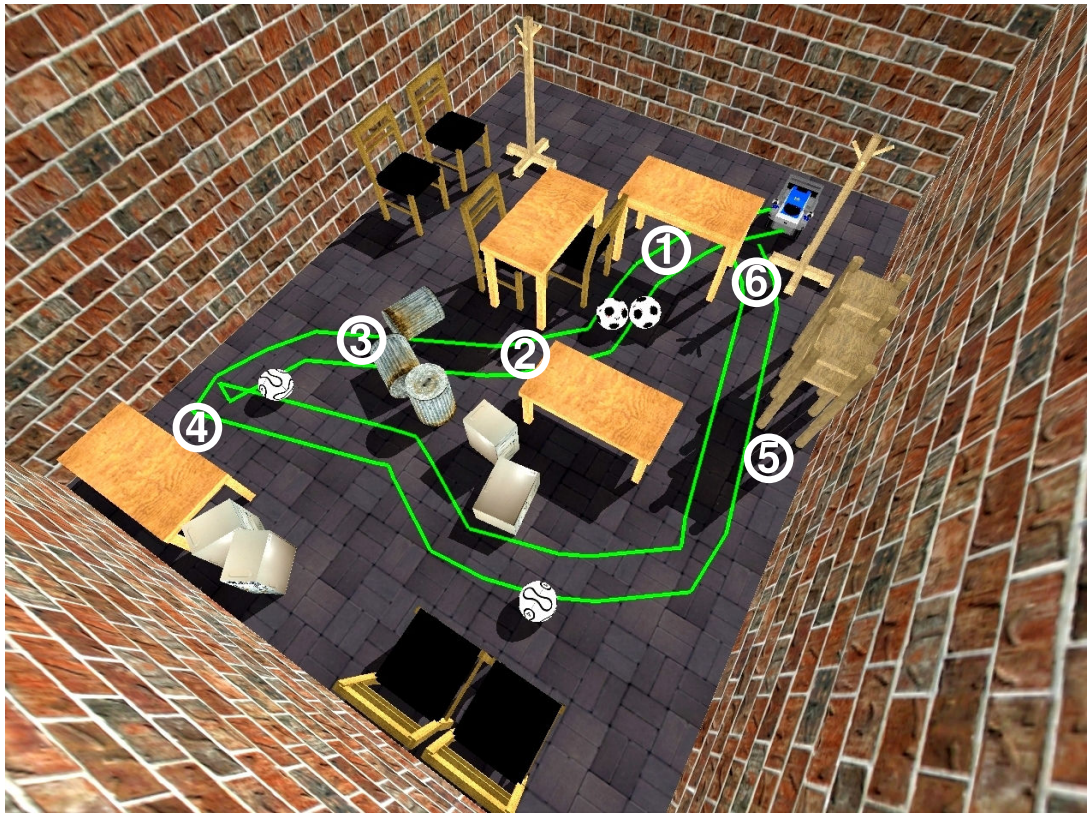


Figure 5.10: The course of the simulated robot in *aggressive* navigation mode.

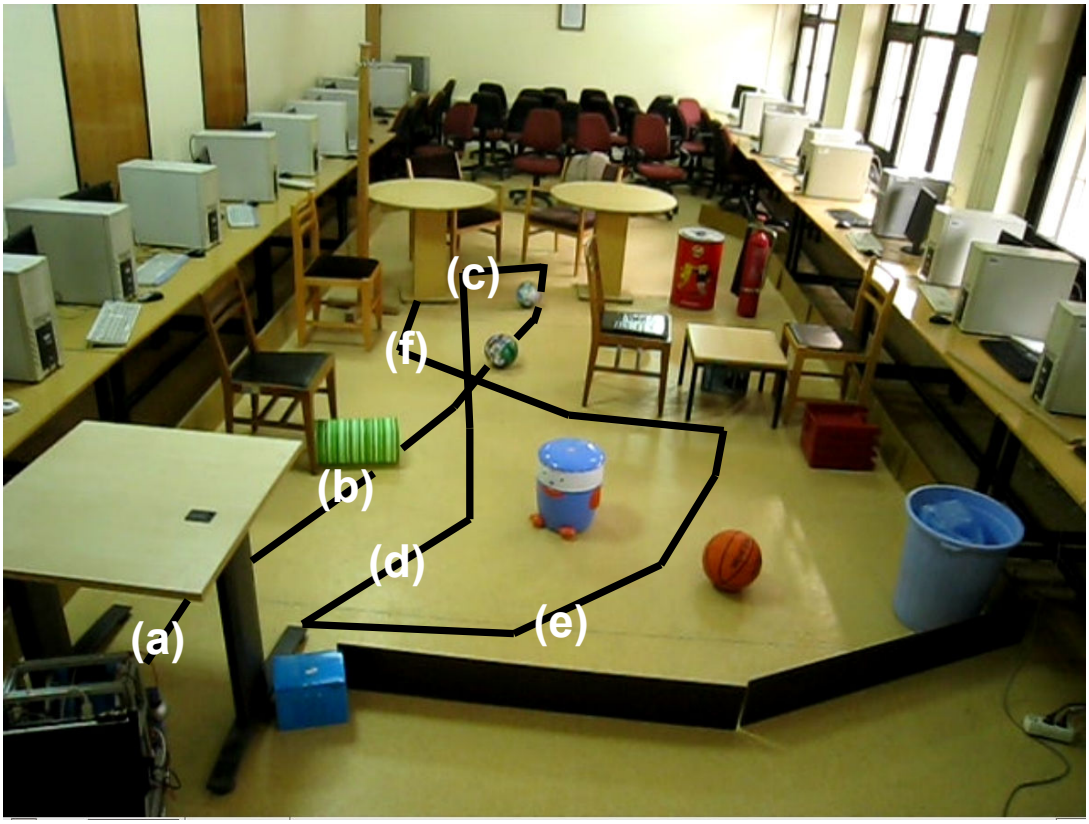
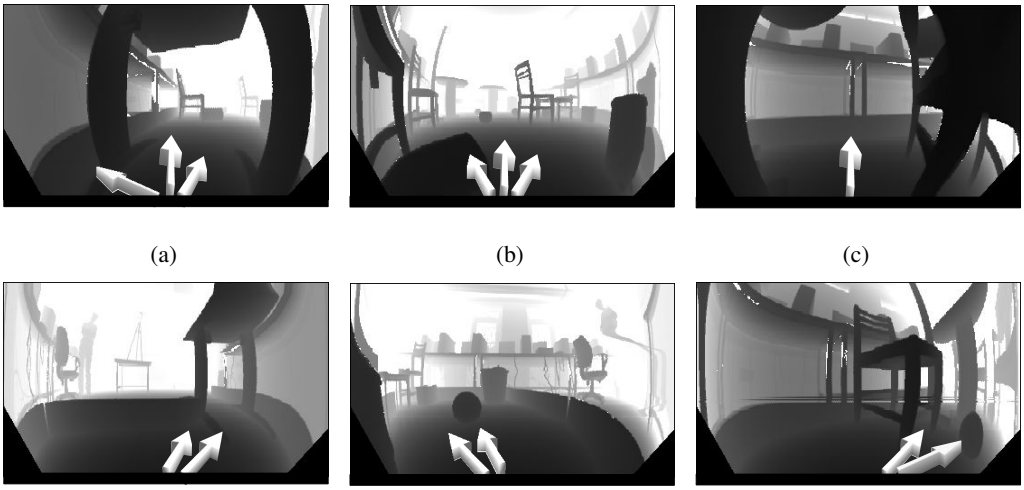


Figure 5.11: The approximate course of the robot resulting from the execution of the controller described in Figure 5.9(b) in a real room. The controller tries to drive the robot as cautious as possible by averaging the direction of the neighbor afforded behaviors. The plastic balls and the cylindrical light shade are predicted go-over-able, the tables are predicted go-under-able, the apertures between table legs and tables' bases are predicted pass-through-able. The robot perceives the extension of the table on the left as traversable and collides with it. The basket ball on the right is also incorrectly predicted as not-traversable for some behaviors, and the robot avoided from it.

the traversability of the environment 31 times through its course. Some situations critical to our discussions are identified on the figure and the corresponding range images used in feature computation are shown in Figure 5.12 together with the afforded behaviors. Figure 5.12(a) shows that the robot correctly learned its own body dimensions, i.e. the *go-under-ability* of the table and *pass-through-ability* through its legs are correctly perceived. Only three of the seven behaviors are afforded because a collision would occur for other behaviors. The largest set of afforded neighbor behaviors in this case is $\{\uparrow, \nearrow\}$, so the robot first rotates 10° around itself and then goes forward.

Figures 5.12(b) and 5.12(e) show the range images of the situations that the robot encounters with *roll-able* objects. In these cases, the robot correctly predicts the affordances of the behav-



(a) (b) (c)

(d) (e) (f)

Figure 5.12: Real range images used for traversability prediction.

iors, which drive the robot towards these objects. In (b), the robot goes over the cylindrical object whose orientation with respect to the robot makes it convenient to push, and rolls it aside. Moreover, the robot correctly predicts that the complex chair on the left and the upright cylindrical trash bin on the right do not afford traversability. The robot also deals with the confusing situation where the traversable cylinder and the non-traversable chair locate in the same direction; it decides that the behaviors and are not afforded. In (e) the robot does not go towards the ball since the other afforded behavior moves the robot further to the left than expected.

In Figure 5.12(c) only go-forward ( ) behavior is afforded, however the robot does not drive forward since at least two neighbouring behaviors should be afforded. Indeed the aperture width between the table base on the left and the chair on the right is not large enough for the robot to pass-through as predicted. This example situation shows how the robot is protected from a collision by being *cautious*. However in Figures 5.12(d) and 5.12(f), the robot cannot avoid collision even in this mode. In (d), the extension of the leg of the table is very small, in fact smaller than any object encountered during training interactions. So the robot incorrectly predicts that behaviors, which drive the robot towards that direction, namely and , are afforded. In (f), the robot also makes an incorrect prediction on affordance of behavior. Although, the average direction of the set { , } is taken as the movement direction, the robot cannot avoid a major collision with the chair on the right.

5.7 Feature Relevance

In this chapter, we used two off-the-shelf methods from Machine Learning research; namely the ReliefF method to extract relevant features for traversability, and the SVMs as classifiers for predicting traversability. These specific methods were chosen over other alternatives due to their scalability, robustness and computational complexity. Specifically, both methods scale well with the dimension of the inputs, with the size of the training dataset, and perform robustly when faced with noisy data. Moreover, after training, the SVMs store only a small number of parameters and have low computational complexity during execution. However, it should be noted that one can consider the use of other methods instead if properties such as scalability and low computational complexity are not required.

5.7.1 Feature Selection

The methods that select relevant features can be roughly categorized in two groups; namely *wrappers* and *filters* [11]. The filter methods select features based on metrics such as distance and correlation without considering how the selection would affect the performance in classifier phase. The wrapper methods on the other hand measure the relevance of features based on the performance of classifier used in the second phase and can produce near-optimal results [35]. For instance, among wrapper methods the Schemata Search algorithm [102] starts from an empty relevant feature subset and incrementally adds a new feature to the subset that increases the classification performance most. The method iterates as long as the performance increase remains positive. On the other hand, among filter methods the ReliefF method computes the relevancy of each feature using the correlation of the features with the categories independent of the classifier to be used in the second phase.

Although wrapper methods give better results compared to filter methods in general, the classification phase tends to have high computational complexity in most applications and does not scale well with the dimension of the inputs and the size of the training dataset. Filter methods have lower computational complexity since they do not utilize the classification phase in computing the relevancy of the features. However, they fail to detect redundancies in the feature sets and produce less than optimum sets. For instance, if relevant feature is duplicated in the representation, both copies of the feature will be included in the set, despite the fact that

the inclusion of the second copy does not improve the performance of classification.

Table 5.4 lists the 20 most relevant features discovered by the ReliefF method for predicting the traversability of the go-forward behavior. The feature list can roughly be categorized into three sets: (1) features measure the minimum or the mean of the distance values coming from the central grid squares, and (2) the latitude and (3) longitude features coming from almost similar grid squares measuring the normal vector histograms between certain degrees ($[0°, 20°]$for latitude, and $[60°, 80°]$for longitude). In order to analyze the redundancy of these features, we used the *sequentialfs* (sequential feature selection) method provided in the MATLAB package [97], within the wrapper category. The *sequentialfs* method generates near-optimal relevant feature sets in a way similar to the one used in Schemata Search[102]. Starting from an empty relevance feature set, it selects one feature and adds it to the feature set of previous iteration. At each iteration, a candidate feature set for each not-yet-selected feature is formed by adding the corresponding feature to the previous feature set. Then, the candidate feature sets are evaluated through 10-fold cross-validations on SVM classifiers that use these candidate feature sets. The best performing candidate set is then transfered to the next iteration.

Table 5.5 ranks the most important 20 features found after the 20 iterations[3] of *sequentialfs* method. The table also includes the rank of each feature as evaluated by the ReliefF method for comparison purposes. As expected, the most relevant feature discovered by the *sequentialfs* method ranks also high on the ReliefF ranking[4]. It can be seen that due to the incremental nature of the *sequentialfs* method, there is little correspondence between the rankings of the two methods. However, given that *sequentialfs* method produces a more optimal set of features, we can now go back to the set of relevant features listed in Table 5.4 to support our claim that there exist a lot of redundancy among the information carried out by the features that are ranked high by the ReliefF method. For instance, only 5 distance related features appear in Table 5.5, as opposed to 13 in the Table 5.4 indicating the redundancy in the relevant feature set.

Towards this end, we analyzed the performance of the classifiers that are trained with the most

[3] The processing took approximately two days of computation on a PC with Dual Core 1.86 GHz CPU and 1 GB RAM and did not allow us to proceed further.

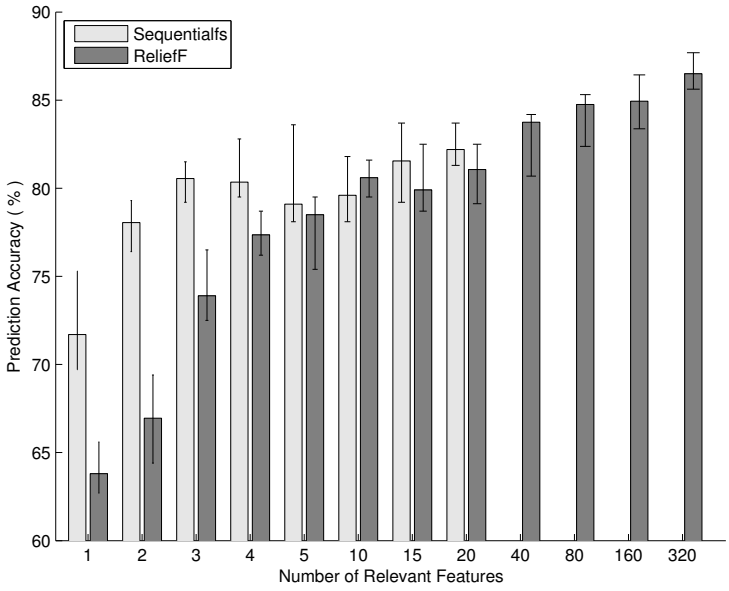
[4] In order to understand why the most relevant feature of the *sequentialfs* method did not rank at the top of ReliefF ranking, one needs to realize the fact that the ReliefF method counts the number of correct classifications, whereas the SVM's used by the *sequentialfs* method optimizes the least square error over the training data set, and that these two may not necessarily match.

Table 5.4: The 20 most relevant features discovered by the ReliefF method for the go-forward behavior.

Rank	Type	Feature	Grid position
1	distance	min	
2	latitude(θ)	$[0°, 20°]$	
3	distance	min	
4	latitude(θ)	$[0°, 20°]$	
5	latitude(θ)	$[60°, 80°]$	
6	distance	min	
7	distance	mean	
8	distance	min	
9	latitude(φ)	$[0°, 20°]$	
10	distance	mean	
11	distance	min	
12	distance	min	
13	distance	min	
14	latitude(φ)	$[60°, 80°]$	
15	distance	mean	
16	distance	mean	
17	longitude(φ)	$[0°, 20°]$	
18	distance	min	
19	distance	min	
20	latitude(θ)	$[60°, 80°]$	

Table 5.5: The 20 most relevant features discovered by the *sequentialfs* method for the go-forward behavior.

sequentialfs rank	Type	Feature	Grid position	ReliefF rank
1	distance	min		3
2	latitude(θ)	$[40°, 60°]$		113
3	latitude(θ)	$[20°, 40°]$		70
4	latitude(θ)	$[-40°, -20°]$		428
5	latitude(θ)	$[20°, 40°]$		243
6	longitude(φ)	$[40°, 60°]$		423
7	longitude(φ)	$[40°, 60°]$		290
8	distance	min		438
9	longitude(φ)	$[0°, 20°]$		302
10	latitude(θ)	$[60°, 80°]$		14
11	longitude(φ)	$[-80°, -60°]$		256
12	latitude(θ)	$[60°, 80°]$		291
13	latitude(θ)	$[-20°, 0°]$		280
14	longitude(φ)	$[40°, 60°]$		585
15	distance	min		167
16	longitude(φ)	$[0°, 20°]$		293
17	distance	mean		232
18	latitude(θ)	$[20°, 40°]$		196
19	longitude(φ)	$[0°, 20°]$		131
20	distance	min		46



(a)

Figure 5.13: Relevancy results in training like environments. The prediction accuracies obtained with the most relevant n features in environments used during training.

important n features obtained by the *sequentialfs* and the ReliefF method in environments with the same characteristics of the training environment where varying number of different types of objects are randomly placed in the frontal area of the robot.

Figure 5.13 plots the performance of the classifiers that are trained with the most important n features obtained by the *sequentialfs* ($n \leq 20$) and the ReliefF method ($n \leq 320$) for predicting the traversability of the go-forward behavior. The evaluation was made in environments with the same characteristics of the training environment where varying number of different types of objects are randomly placed in the frontal area of the robot. The bars show best, median, and worst prediction accuracies that are obtained over 10 different test sets. Two observations can be made based on these results. First, the prediction performance increases with the number of relevant features. Second, for a given number of relevant features, the classifiers perform better with the feature set obtained from *sequentialfs* than the one obtained from ReliefF.

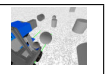
5.7.2 Traversability Problem

The difficulty inherent in learning traversability of the robot, as studied in this chapter, begs further analysis in order to ensure that the problem is not reduced to a trivial one through the choice of the particular feature representation. Figure 5.13 shows that classifiers can achieve prediction performance of approximately 64% (72%) using the most relevant feature (minimum distance from one of the central grid squares) discovered by the ReliefF (*sequentialfs*) method. The inclusion of the next three features raises the performance to 78% (80% for *sequentialfs*), and the performance gradually reaches 87% with the use of 320 features.

In order to analyze in detail how the inclusion of additional features contributed to the prediction performance, we used seven exemplary setups as shown in Table 5.6. It can be seen that through the use of the most relevant feature only, classifiers merely link the existence/non-existence of an object in the frontal area to traversability and do not take into account their rollability. The inclusion of the second most relevant feature detected by the *sequentialfs* method allows the detection of traversability in lying cylinders that are properly aligned but fails the detection of traversability in spheres. In a similar manner, the inclusion of the second most relevant feature detected by ReliefF method allows the classifier to detect the traversability of spheres but fails on lying cylinders. As can be seen, the classifiers trained by the most three (four) relevant features detected by the *sequentialfs* (ReliefF) method are able to detect the traversability affordances in the first six setup that included single objects. However, when the scene is cluttered with multiple objects, such as the seventh setup, where the robot faces both a close-by sphere and a further-away box in its frontal view, then the prediction becomes difficult, and requires the use of 320 features.

A closer inspection of Figure 5.13 shows that as the number of features used by the classifier increases from 20 to 320, the performance merely increases from 82% to 87%. Hence, one may question whether the amount of performance gain is sufficient to justify the extra cost. At this point, we would like to point out that the prediction performances reported in the above experiments are determined by the distribution of the test environments, and may not be very representative of the real-world performance to be expected from the robot. Towards this end, we conducted an experiment to evaluate the performance of the classifiers by hanging boxes (which are non-traversable) around the *critical points* for traversability. Specifically, in a similar setup shown in Figure 5.5, we systematically placed boxes (non-traversable) inside

Table 5.6: The traversability prediction results in eight exemplary setups. *Seqfs(n)* and *ReliefF(n)* denote classifiers trained with the first *n* relevant features discovered by the *sequentialfs* and ReliefF methods. Setups: *1*: no object, *2*: a box, *3*: a rotated box, *4*: sphere, *5*: upward cylinder, *6*: lying cylinder that is traversable, *7*: lying cylinder that is not traversable, *8*: a mixed environment where the robot sees a close-by sphere and a box that is slightly further away.

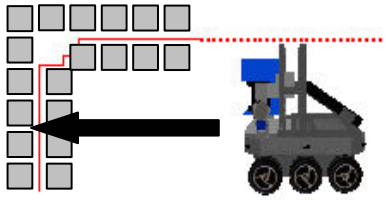
								
	1	2	3	4	5	6	7	8
Seqfs (1)	√	√	√	X	√	X	√	X
Seqfs (2)	√	√	√	X	√	√	√	X
Seqfs (3)	√	√	√	√	√	√	√	X
Seqfs (4)	√	√	√	√	√	√	√	X
Seqfs (20)	√	√	√	√	√	√	√	X
ReliefF (1)	√	√	√	X	√	X	√	X
ReliefF (2)	√	√	√	√	√	X	√	X
ReliefF (3)	√	√	√	X	√	X	√	X
ReliefF (4)	√	√	√	√	√	√	√	X
ReliefF (320)	√	√	√	√	√	√	√	√

or outside of the collision boundaries within a 10*cm* band as shown in Figure 5.14(a) and plotted the prediction performance in Figure 5.14(b). The results show that as the number of relevant features increase from 20 to 320, the performance increases from 65% to 85%, a significant gain in borderline situations.

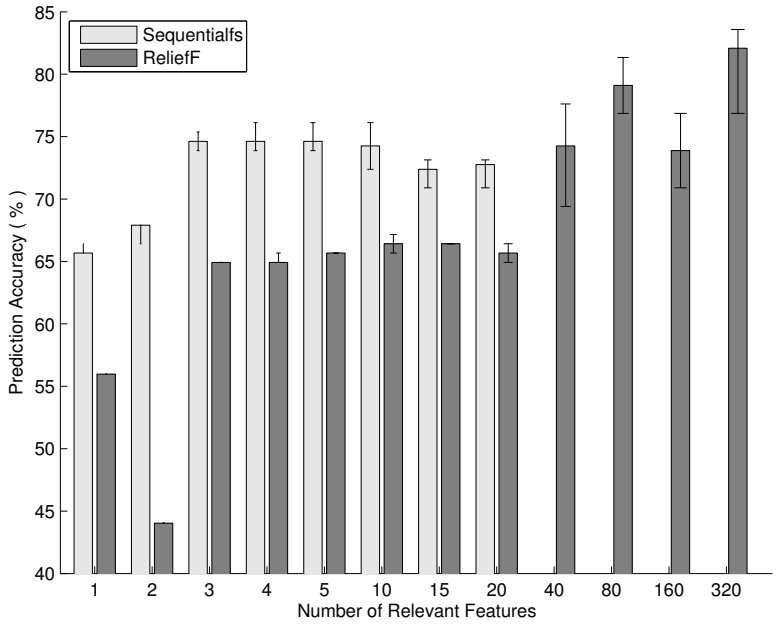
Finally, we would like to point out that the sufficiency of using a linear kernel in SVM classifiers does not necessarily imply the simplicity of the problem, since many learning problems that require complex high-dimensional kernels at low-dimensional feature spaces are transformed into simpler problems that can be linearly separable through the use of high-dimensional features.

5.8 Conclusion

In this chapter, we studied the learning and perception of traversability affordance in organisms and robots with the hope of appealing to readers from both Ecological Psychology and Autonomous Robotics. Hence the contributions of this chapter are two-fold: first, from a robotics point of view, it presents a method for the learning and perception of traversability on mobile robots using range images. Specifically, we proposed a set of features for representing



(a)



(b)

Figure 5.14: Boundary experiment setup and the prediction accuracies. (a) The robot is asked to predict the traversability of environments which include a hanging box that is placed either on one of the given gray squares or a position in between these squares. Note that the boxes have same dimensions as the squares. (b) The prediction accuracies were obtained with the most relevant n features in the boundary experiment. Note that, different classifiers that are trained with same features can make same predictions if the number of features is small.

the shape and distance information on range images that are shown to provide a good degree of generalization, and a scalable method towards learning affordance relations (specifically, learning entity equivalence classes in our formalism). The learning method uses off-the-shelf machine learning methods that are highly scalable with the input dimension. The proposed method shows that one can start with a large feature vector that contains all types of feature detectors that one can propose, and have it reduced down to only a fraction after training. In this sense, the robot can minimize the load on its perceptual processing after learning to achieve perceptual economy. A systematic analysis of the method and its performance under different parameter settings, and in both simulated and physical environments, showed that despite the simplicity of perceptual representation, the method can provide a good degree of generalization, as demonstrated in Section 5.6.5 where upon training with only simple object types in a simulated environment, the robot can navigate successfully among complex objects in the real-world.

Second, from an Ecological Psychology point of view, this chapter shows that the formalization proposed in our earlier work [33], can indeed be used to make the robots learn the affordances in its environment. The proposed formalism, which we admitted to extend the Gibsonian view on affordances, had received criticism from Chemero [22] who claimed that affordances are relations that exist within the agent-environment system and that they cannot be represented in a robot. Through systematic experiments, that are inspired by the ones used in Ecological Psychology, we show that the robot, by going through an exploration phase, can learn to perceive the traversability affordances in its environment, and build a "sense of its of body size" and achieve perceptual economy. By conducting experiments that show that our robot can learn generalizable relations about its interaction with the world that are related to its physical size and capabilities while achieving perceptual economy, we claim to support our view.

The study presented in this chapter has a number of limitations that can mainly be attributed to the use of our 3D range sensing equipment, which takes almost 40 seconds to produce a range image. First, the speed of sensing limits the reactivity of the robot, and doesn't leave much room for the robot to immediately perceive and react to changes in the environment. Second, the slowness also makes it prohibitive to obtain large quantities of data to be used for learning, which was tackled through the use of a physical simulator. However, the physical simulators bring in their own constraints, such as the difficulty of access to 3D models of real-world

objects which then limits the type of interactions that can be explored in simulation. These limitations can be addressed through using stereo vision systems that operate using standard cameras or through the use of 3D cameras that can provide range images at large frame rates.

Finally, we would like to point out that although the use of range images makes it easier to link and generalize the perceptual features with the physical affordances of the environment, the proposed methodology does not pose any limitation on the type sensing device. As a matter of fact, the use of regular camera images may indeed be used to discover/develop image features that are relevant to affordances may produce interesting insights to computer vision similar to the ones shown in [45].

5.9 Discussion

In this chapter, the robot is required to make large number of interactions (5000) in order to learn traversability affordance of each behavior. During robot's exploration, the environment it interacts is randomly constructed, in other words the robot 'chooses' to explore any random environment independent of the experience it gained. Furthermore, the learning is performed in a batch manner: The robot interacts with the world many times, all experience is accumulated, and only after then the learning is performed.

In the Introduction Chapter, we argued that 'intrinsic motivation' mechanism is utilized during infant exploration to optimize speed and extent of learning. Furthermore, learning should be considered as an open-ended online process. The random exploration strategy followed in this chapter together with batch learning process hardly satisfies these criteria. Therefore, in the next chapter, we will study a curiosity-based online learning algorithm that automatically chooses novel situations to interact based on previous experience.

CHAPTER 6

CURIOSITY-DRIVEN LEARNING OF TRAVERSABILITY AFFORDANCE

6.1 Introduction

In the previous chapter, the robot made 5000 interactions to learn the traversability affordances. However, learning is a costly process in robotics. Ideally, the robot should physically interact with its environment exploring its environment and testing its behavioral abilities in different situations. However, even for simple tasks, such as avoiding objects, a large number of interactions, some of which may result in physical damage to the robot, need to be carried out to drive the learning process. Hence, the learning process is not only time-consuming and costly in terms of the physical wearing out of the robot, but is also risky, since some of the interactions may result in physical damage to the robot. Therefore, it is essential that the interactions of the robot during the learning phase be minimized with minimal or no degradation of learning.

The problem of selection of the best training data to increase the performance and speed of learning has been studied in the field of Machine Learning (Active Learning) and particularly in Developmental Robotics. In these studies, as stated in [111], generally two modules are used: the *learner* and the *meta-learner*. In these systems, the *learner* is responsible from the learning process, whereas *the meta-learner* is responsible from selection of the next sample, which would increase the speed of the learning process.

In this chapter, we study the learning of traversability affordance on a mobile robot and investigate how the number of interactions required can be minimized with minimal degradation on the learning process. Instead of using a *meta-learner* we utilized a curiosity-based scheme

on the *learner* itself to increase the speed of the affordance learning and minimize the number of interactions with minimal degradation in learning process. Specifically, we propose a two step learning process which consists of bootstrapping and curiosity-based learning phases. In the bootstrapping phase, a small set of initial interaction data are used to find the relevant perceptual features for the affordance, and a Support Vector Machine (SVM) classifier is trained. In the curiosity-driven learning phase, a curiosity band around the decision hyper-plane of the SVM is used to decide whether a given interaction opportunity is worth exploring or not. Specifically, if the output of the SVM for a given percept lies within curiosity band, indicating that the classifier is not so certain about the hypothesized effect of the interaction, the robot goes ahead with the interaction, and skips if not. Our studies within a physics-based robot simulator show that the robot can achieve better learning with the proposed curiosity-driven learning method for a fixed number of interactions. The results also show that, for optimum performance, there exists a minimum number of initial interactions to be used for bootstrapping. Finally, the trained classifier with the proposed learning method is also successfully tested on the real robot.

6.2 Framework Implementation

- **Behavior:** *Behaviors* correspond to discrete pre-defined actions without any parameter.

- **Entity:** The *entity* is computed from the whole perceived environment, without any object detection process. Thus, no object identifier is included in its notation. The entities are perceived only prior to behavior execution and they are not perceived after the execution.

- **Effect category:** The *effect categories* are pre-defined and fixed, and the means to compute these categories are provided by behavior designer. So, there is no effect category discovery step. On the other hand, the behavior designer does not provide effect category prototypes, thus the system can only learn whether a behavior will succeed or not. The *effect categories* are defined as *traversable/success* and *non-traversable/fail*.

- **Effect:** Since the robot does not perceive and represent the (*final*) entity after behavior execution, the change in entity feature vector, i.e. the *effect*, cannot be not computed. Thus, *effect* is neither represented nor used as in previous chapter.

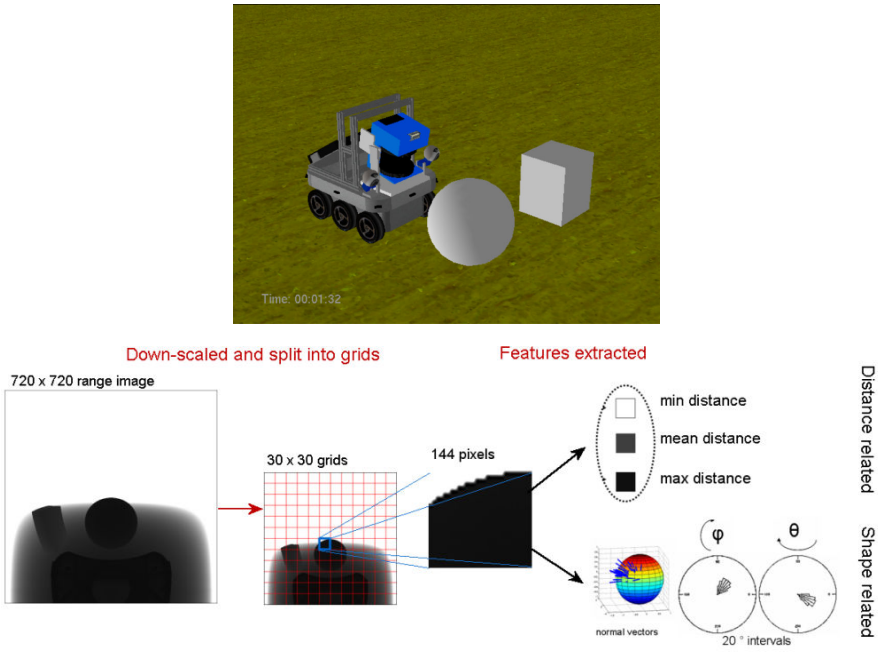


Figure 6.1: The simulated robot in MACSim is shown on top. At the bottom, The range image obtained in this situation and the operations applied to this image are shown. The 360×360 pixel range image is divided into $30 \times 30 = 900$ grids of 12×12 pixels, and the angular histogram is divided into 18 intervals, so that total number of features computed over a downscaled range image is $900 \times (3 + 2 \times 18) = 35100$ where 3 corresponds to the three distance values (minimum, maximum, and mean) and the multiplication by 2 corresponds to the two angle channels.

6.3 Experimental Setup

6.3.1 Perception

The perceptual processing of the robot is similar as in previous Chapter. The range image is downscaled, split into grids, and distance and shape related features are computed for each grid. Different from previous chapter, the grids are not shifted and the resolution of perceptual representation is not optimized. In other words, in this chapter one grid layer is used and the number of grids is kept fixed as shown in Figure 6.1.

6.3.2 Behaviors

Same as previous chapter, the robot is provided with seven simple hand-coded actions, which result in movement in seven different directions. One of the actions makes the robot go

forward, while the others first rotate the robot around its own vertical axis for a certain period and then drive it forward. Along with each action, the expected displacement of the robot is provided as its success criteria.

6.3.3 Interactions

In the learning phase the robot learns a mapping between environmental situations and the results of its actions by physically interacting with the environment. In each interaction episode, the robot is placed at a random position and orientation in a training room which includes a number of randomly placed objects.

6.4 Learning Affordances

After the robot perceives its environment using the 3D range scanner and computes a feature vector, the *learner* that is trained up-to that point determines whether the current situation is an interesting one or not, based on the computed feature vector. If the learner is certain about the effect to be produced, the robot will choose not to interact with the environment to test its hypothesis and will be "beamed" to a different position in the room. However, if *the learner* is not certain about the result of executing a particular action in that situation, the robot will execute the action and observe the result of that action using a pre-defined success metric (displacement vector). Then, *the learner* is updated using the feature vector and the result of the action.

The learning process consists of two phases:

6.4.0.1 Bootstrap Phase

In this phase, a small set of training samples ($n_{bootstrap}$) are obtained by interacting with the environment without any novelty check. Since time and space requirements of learning from samples with 35100 features would be huge, the learning is done using only a subset of these features. This subset includes the features which are relevant for a particular action, and affordance learning for that action is performed using only that subset.

ReliefF algorithm[82], which estimates the relevance of each feature based on its impact on the target category (traversable/non-traversable) of the samples, is used for feature selection. After computing the relevances using ReliefF, the most relevant n features are chosen. Although ReliefF does not work optimally with such a small sample set and high number of features, by setting n to a relatively large number, most of the relevant features would be included in the obtained subset. We set n to 250 in our experiments.

The bootstrap period is also required to initiate the training. Thus, the set of training samples, obtained in this phase are used to train a classifier in a batch manner. The details of the classifier, which learns a relation that maps the (initially perceived) relevant features to predict the success/fail result of applying that action, will be given below.

6.4.0.2 Curiosity-driven Learning Phase

Different from the approaches mentioned in Section 6.1, we will use *the learner* both to select the next sample and to learn from experience. A training sample in our domain is obtained through perceiving the environment, physically interacting with it, and storing the perceptual data together with the result of the robot's interaction (afforded/not-afforded). Thus, if *the learner* decides that a candidate sample is not interesting enough, it will not be included in training. In this case, there is also no need to execute the action since only the perceptual data are used by *the learner* to determine whether that sample is interesting or not. As a result, we use an online-learner, which determines the novelty of the perceptual data, and executes the action only if the perceptual data are interesting enough for that action.

Support Vector Machines (SVMs) are used to learn the mapping between perceptual data and affordance classes (traversable/non-traversable). In SVMs, the optimal hyper-plane that separates two classes is found, based on the most informative samples (the support vectors) in the training set. The new test sample's class is predicted based on its relative location with respect to this hyper-plane in the feature space. We made an assumption that SVMs are more certain in their class prediction of a new sample, if that sample is further away from the hyper-plane, and less certain if sample is closer to the hyper-plane. Thus, when the robot is in an environment, where it is almost certain about the affordances provided, it will bypass this environment without executing any action, and look for more novel situations. On the contrary, when the robot encounters a new situation, if the feature vector computed in

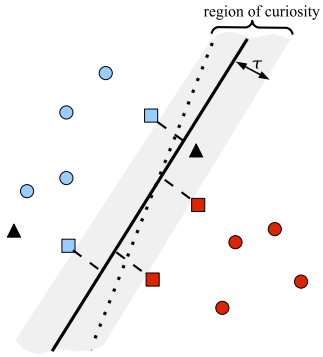


Figure 6.2: Use of the trained SVM's hyper-plane to find interesting situations. The mechanism which selects interesting samples for training is demonstrated. The continuous line demonstrates the separating plane that is constructed so far, the square shaped samples demonstrates support vectors, and the circular shaped ones show the samples used in previous training steps, but not serve as support vectors. The triangular shaped samples are the candidates whose classes (traversable/non-traversable) are not known. Current SVM is more certain about the class of the sample on the left, so this candidate will not be included in the training set. However, the candidate on the right is very close to the hyper-plane and SVM is not certain about its class, thus it will be included in training. A probable modification in the hyper-plane is shown with dashed line after SVM is updated with this candidate sample.

that situation is close to the hyper-plane, SVM will conclude that this situation is interesting enough to be included in training. In this case, the robot executes the action, and SVM is updated using the feature vector and the result of that action. Thus, the novelty of the candidate is determined based on its distance to the hyper-plane that is constructed so far. If the distance is smaller than a fixed threshold τ then the sample is considered as an interesting one, if it is bigger than τ, it is skipped. Fig. 6.2 provides a simple and clear demonstration of the idea.

Although, SVMs are used as batch learning systems in general, some online implementations, where the samples are fed to the learning system in an incremental manner, are able to produce similar results. We used the LASVM software [13] for online updating of the SVM and making predictions on the candidate and test samples. A linear kernel (with tolerance parameter set as 1) was used since more complex kernels did not increase the performance in our case.

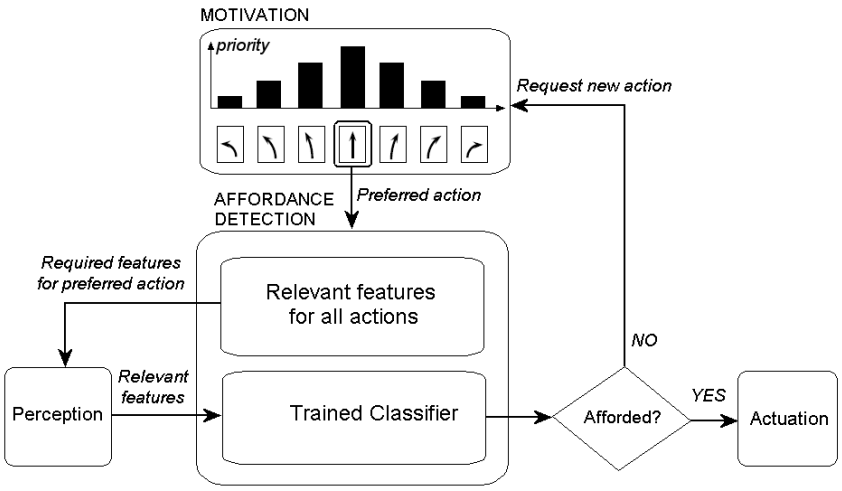


Figure 6.3: Use of the trained affordance classifiers in behavior selection.

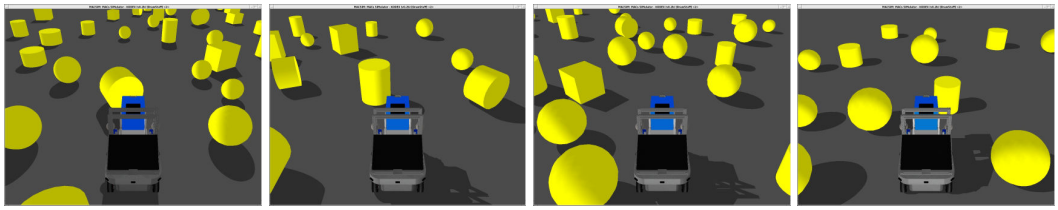
6.4.1 Control

The robot is driven using a simple control system (Fig. 6.3), which utilizes learned relevant feature perception and affordance classification schemes explained in the previous sections. Whenever a new action is requested, the motivation based control system sets a new *preferred* action with highest priority, among a set of actions with fixed priorities. The features which are relevant to the *preferred action* are then requested from perception, and these features are supplied to the trained classifier (SVM) to predict whether this action is afforded or not. If the immediate environment does not afford this action, a lower priority action is requested from the motivation module. Otherwise, it is executed (robot moves in a certain direction for a certain duration), and a new action is requested upon the completion of the action.

6.5 Experiments

The learning is conducted in an online-fashion, where first $n_{bootstrap}$ samples are collected for feature selection and initiating the classifier. Then the learning continues in a curiosity-driven way by selecting most interesting situations based on the distance threshold τ . As a result, two parameters, $n_{bootstrap}$ and τ determine the speed and performance of learning.

The learning is performed in MACSim, where the robot is placed in a 3×3 m^2 square room, which includes 100 randomly scattered objects with dimensions in the range [20cm – 40cm].



(a) (b) (c) (d)

Figure 6.4: Example situations from curiosity-based learning phase. Curiosity-based learner found the two left-most situations interesting, executed go forward action and updated the the classifier based on the result of its actions. However the two right-most situations are found to be uninteresting and were not included in training. (a) Corresponds to a situation where boundaries of the cylinder's surface is similar to the sphere's from the robot's point of view, and the learner is required to be fine-tuned. (b) Corresponds to a situation where the object locates in the boundaries of the go-forward action. (c) The space in front of the robot is clear. (d) This situation seems to be similar to (b), however the (smaller) object in (d) is closer than the object in (b).

For each action, an online-SVM is trained using 3000 different samples, which are obtained by making 3000 different interactions in this room. During this phase, only the interesting samples are used in training the SVM (Fig. 6.4).

After training, the robot is transferred into another virtual room with similar characteristics and 2000 test samples are collected in the second room. These 2000 samples are used to evaluate and compare the performances of the controllers trained with differing values of $n_{bootstrap}$ and τ. In the next section we examine the effect of these two parameters on the speed and performance of the learning system, based on the system's prediction accuracy over the 2000 testing samples.

6.5.1 Effect of Bootstrap Period

The number of bootstrap samples, $n_{bootstrap}$ affects the quality of the feature selection process and the classifier's performance. If $n_{bootstrap}$ is large, the relevant features are more accurately selected, and more samples will be included in initial training without any curiosity check. In these experiments, in order to examine the effect of bootstrap period, the prediction accuracies of the classifier are computed for $n_{bootstrap}$ values of 10, 25, 50, and 100 on the testing set. In the box and whisker plot (Fig. 6.5), the prediction accuracy of the classifier on the

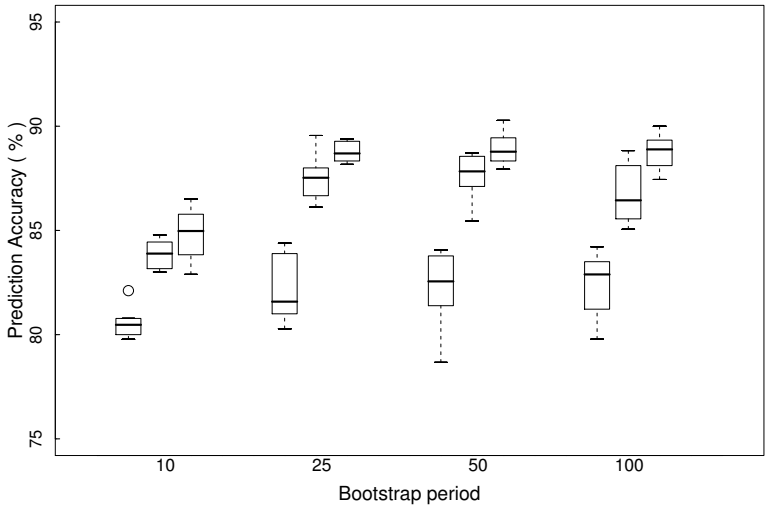


Figure 6.5: The effect of the bootstrap period on prediction accuracy. The bootstrap period required to select the relevant features and train an initial learner is adjusted, and the speed/performance plot is demonstrated. Successive three boxes correspond to values at 100^{th}, 250^{th}, and 400^{th} interactions. Curiosity parameter τ is fixed to 0.5.

test set is plotted against the bootstrap parameter, where each box represents the accuracy distribution of 10 different classifiers obtained from different orderings of the training samples. In this plot, for each value of the $n_{bootstrap}$, three successive boxes are drawn, corresponding to the prediction accuracy values at the 100^{th}, 250^{th}, and 400^{th} interactions. When $n_{bootstrap}$ is selected as 10, the performance of the classifier remains below %90 since 10 samples are insufficient for selecting the relevant features and bootstrapping an initial classifier with the ability to select interesting samples. The values greater than 25 does not further increase the performance, thus, 25 initial samples are found to be sufficient to bootstrap the learning process.

6.5.2 Effect of the Curiosity Parameter

The curiosity parameter τ determines the width of the band around the decision hyper-plane of the SVM. As the τ gets larger, more samples will be selected as interesting. The effect of τ is examined by training different classifiers with different τ values (eg. 0.05, 0.10, 0.50, 1.00, and no curiosity). In the box and whisker plot (Fig. 6.6), the prediction accuracy of the classifier on the test set is plotted against τ, where each box represents the accuracy distribution of 10 different classifiers corresponding to different orderings of the training samples. Similar to the previous figure, for each value of the τ, three successive boxes are drawn, corresponding

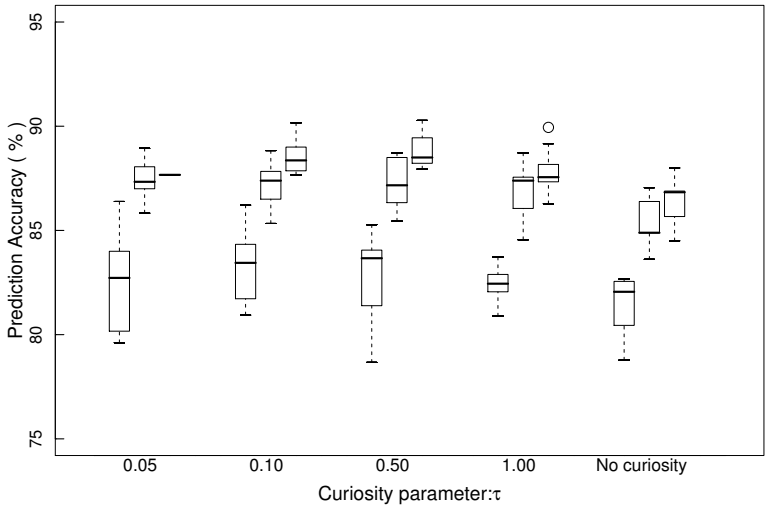


Figure 6.6: The effect of curiosity threshold on the prediction accuracy. The change in the prediction accuracy of the affordance perception during the learning phase. The thresholds which determine the curiosity level of the robot are compared. Successive three boxes correspond to values at 100^{th}, 250^{th}, and 400^{th} interactions. $n_{bootstrap}$ is fixed to 50.

to the prediction accuracy values at the 100^{th}, 250^{th}, and 400^{th} interactions. As shown, curiosity parameters that are too small keeps the system away from interacting with interesting situations. On the contrary, curiosity parameters that are too large slows down learning by including uninteresting samples in training. As a result, we selected $\tau = 0.50$ as the curiosity parameter to be used in the next section.

6.5.3 Using Traversability Affordance

In order to demonstrate the overall behavior of the robot, and its ability in perceiving the traversability affordance in the environment, it is placed in a room cluttered with objects of various shapes and size (Fig. 6.7). The controller used in this experiment was trained with $\tau = 0.5$ and $n_{bootstrap} = 50$. Here, the robot is additionally controlled by the motivation system which favors driving forward. Whenever the move-forward action is not afforded, a lower priority action is executed if it is afforded. As shown in the Fig. 6.7, the robot successfully wanders in the room. Note that the robot does not only drive towards the open-spaces, but if a higher priority action requires it, it chooses to drive over spherical and cylindrical objects in appropriate orientations, since they afford traversability. It also successfully avoids boxes and upright cylindrical objects by not driving towards them.

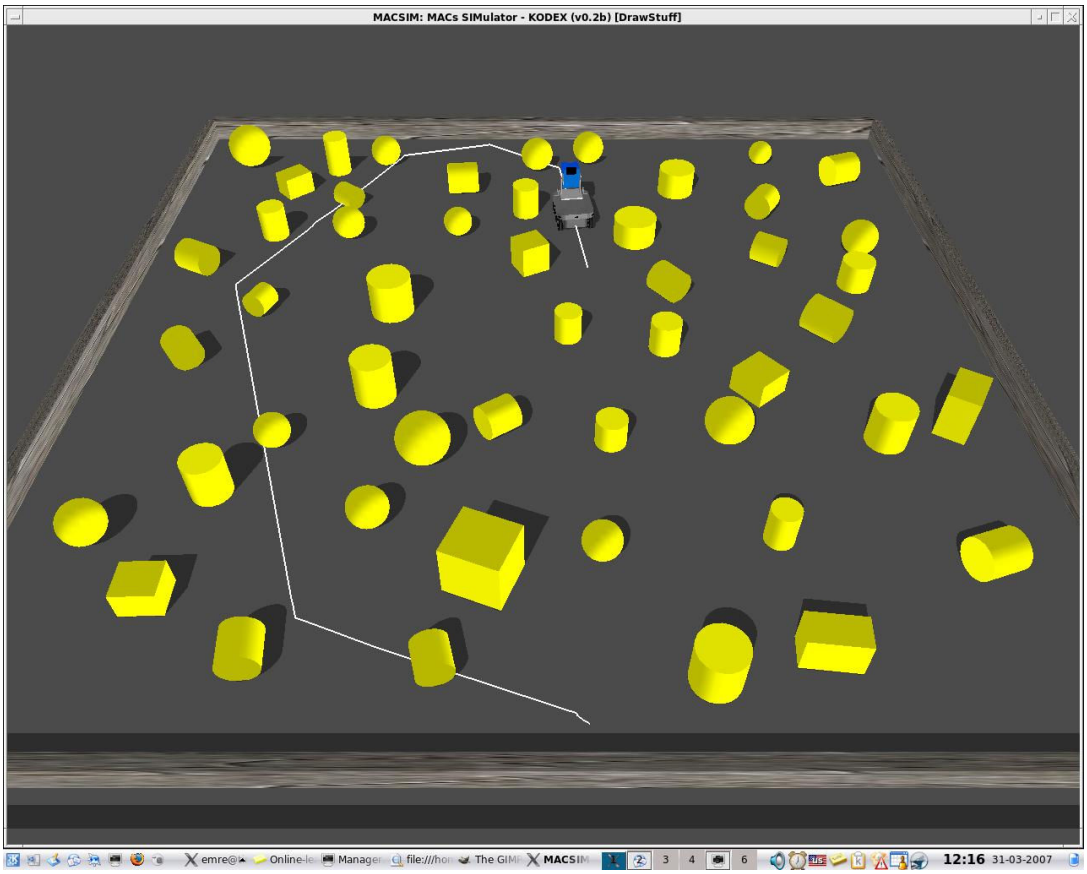


Figure 6.7: The course of the simulated robot trained with curiosity-driven scheme.

Figure 6.8: The course of the real robot trained with curiosity-driven scheme. The initial position of the robot is shown in the left-most figure. The robot first goes forward, then turns left since trash-bin does not afford traversability. Third snapshot shows the robot driving over the spherical object. The path of the robot is shown in the last figure.

The controller used in the simulator is also transferred to the real Kurt3D robot. Various objects, including simple geometrical ones, and office environment object like trash bins and boxes are then placed on the way of Kurt3D to test the controller. As shown in Figure 6.8, the robot is able to correctly perceive the affordances of the box, cylindrical, and spherical objects, and act without colliding with non-traversable objects and driving over traversable ones.

6.6 Conclusion

In this chapter, we studied the learning of traversability affordance on a mobile robot and investigated how the number of interactions required can be minimized with minimal degradation on the learning process. Specifically, we proposed a two step learning process which consists of bootstrapping and curiosity-based learning phases. In the bootstrapping phase, a small set of initial interaction data were used to find the relevant perceptual features for the affordance, and a Support Vector Machine (SVM) classifier was trained. In the curiosity-driven learning phase, a curiosity band around the decision hyper-plane of the SVM was used to decide whether a given interaction opportunity is worth exploring or not.

The effects of two parameters of our learning system, τ and $n_{bootstrap}$, which serve as the curiosity threshold and number of bootstrap samples respectively, are examined in systematic experiments. Selecting τ small keeps the system away from interacting with interesting situations, and selecting it large slows down learning since uninteresting samples are used in training. As for $n_{bootstrap}$, while small values degrade the performance of the system, large values does not improve the performance after a certain threshold.

The affordance perception system, trained using optimized parameters, was tested in a room cluttered with objects of varying shapes. In this environment the robot was able to predict the traversability affordances of the objects, and wander around the room. The trained controller was also transferred to the real robot, which was also successful in predicting the traversability affordance of real world objects.

6.7 Discussion

In this chapter (and previous one), the robot was able to learn how to detect traversability affordance. In both chapters, the behavior designer provided the means of assessing whether robot's movement was successful or not. Thus, the robot learned affordances in a supervised way for a specific task: being able to traverse a certain distance.

In the Introduction Chapter, we discussed that 7-9 months-old infants can explore the environment in a goal-free means without any supervision. During this exploration, they monitor the consequences of their actions, and relate the consequences to the visual properties of the objects and environment. Furthermore, we argued that the development must be task-independent and must be led by the surrounding environment. Instead of externally providing the *success* or *fail* result labels for actions, robots or infants should be able to discover what type of effects can be generated in the environment. In that way, the robot may find not only two categories but more during exploration, such as 'no-contact and traversed', 'contact and traversed', and 'contact and not-traversed'.

Following this argument, the limitation of supervised learning will be removed in the next chapters enabling the robot to learn affordances in an unsupervised way. Furthermore, during this learning it will discover abstract concepts and sub-symbolic structures that can be used in multi-step plan generation.

CHAPTER 7

MULTI-STEP PREDICTION AND PLANNING IN A MOBILE ROBOT

7.1 Introduction

In this chapter, we propose a method that allows a robot to learn the symbolic relations that pertain to its interactions with the world and show that they can be used in planning. Specifically, the mobile robot interacts with the objects in its environment using a pre-coded repertoire of behaviors and records its interactions in a triple that consists of the initial percept of the object, the behavior applied and its effect that is defined as the difference between the initial and the final percept. The method allows the robot to learn object affordance relations which can be used to predict the change in the percept of the object when a certain behavior is applied. These relations can then be used to develop plans using forward chaining. The method is implemented and evaluated on a mobile robot system with limited object manipulation capabilities. We have shown that the robot is able to learn the physical affordances of objects from range images and use them to build symbols and relations that can be used in making multi-step predictions about the affordances of objects and achieve complex goals.

7.2 Framework Implementation

- **Behavior:** *Behaviors* correspond to discrete pre-defined actions without any parameter. Thus, they will be represented by b_i as in previous chapters.

- **Entity:** The *entity* corresponds to the feature vector computed for one object and is used interchangeably with the term *object* in this chapter. The robot learns object affordances

by interacting with one object at a time. On the other hand, this learning enables the robot to make predictions over multiple objects. Thus an object identifier is included in representing *entity/object* (when required) as follows: $\boldsymbol{f}^{()}_{o_j}$. Here, $\boldsymbol{f}$ corresponds to the feature vector, o_j is the object identifier and () includes the list of the behaviors executed so far.

- **Effect category:** Different from the previous chapters, the effect categories are not decided by the behavior designer. Instead, the robot discovers a fixed number of *effect categories* ($E^{b_i}_{id}$) for each different behavior b_i during interactions. Further, each effect category has a representative effect prototype vector ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$), which is also found by the robot.

- **Effect:** Different from previous chapters, the robot perceives and represents the change in perception of the entities during its behavior executions. The *effect* feature vector ($\boldsymbol{f}^{b_i}_{\text{effect}}$) represents this change and is used to learn affordances and make predictions.

- **Affordance relation instance:** The affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i >\}$$

7.3 Experimental Setup

7.3.1 Perception

The robot perceives the world through its 3D range scanner. However, different from previous chapters, the robot is assumed to have object detection capability to manipulate the objects. So, instead of computing a feature vector for whole environment, it detects each object and computes a feature vector for each object. The feature computation is performed as follows:

First, the range image is down-scaled to 360×360 for noise reduction and is subtracted from the *background image* that was obtained from an empty environment. The resulting image is segmented and the popped-up regions are assumed to be objects. For each object, o_j, a feature vector, $\boldsymbol{f}^{()}_{o_j}$ is computed (see Fig. 7.1). The perception of the robot before any behavior execution is denoted as $[\boldsymbol{f}^{()}_{o_1}, \boldsymbol{f}^{()}_{o_2} ... \boldsymbol{f}^{()}_{o_m}]$, where m is the number of objects segmented. $\boldsymbol{f}^{()}_{o_1}$ is a

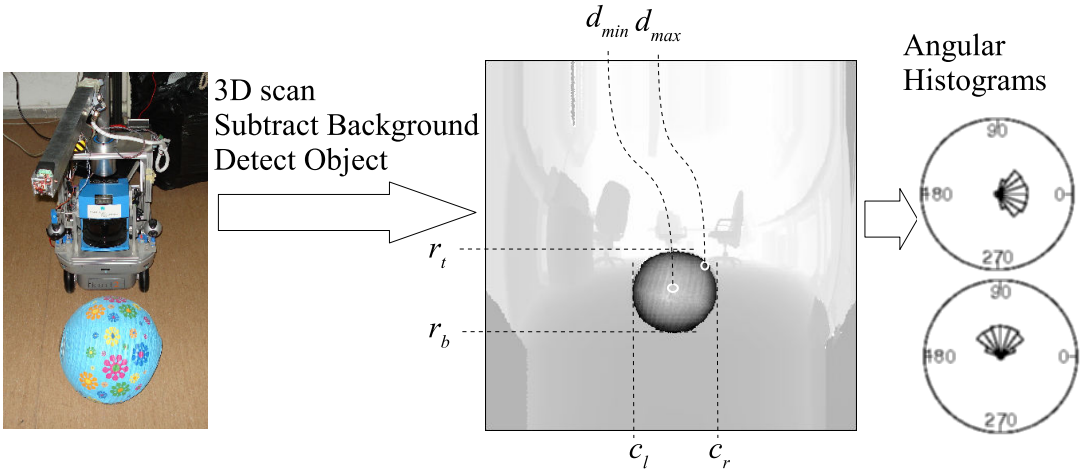


Figure 7.1: Mobile robot and object perception. On the left, the robot and a spherical object shown. On the right, the range image obtained from the 3D scan is given. The subtracted background and other objects are blurred. Distance, relative position and shape related features are shown.

vector of size 44 and is represented as follows:

$$\boldsymbol{f}^{()}_{o_1} = [d_{min}, d_{avg}, d_{max}, a, r_t, r_b, c_l, c_r, \varphi_1 \ldots \varphi_{18}, \theta_1 \ldots \theta_{18}]$$

where $d_{min}, d_{avg}, d_{max}$ denotes the minimum ,average and maximum range values, a is the area measured in pixels, r_t, r_b, c_l, c_r are the indexes of the top and bottom rows, and the left and right columns of the bounding box, and φ_i and θ_j represent the frequency histogram of normal vector angles in latitude and longitude as detailed in Section 5.3.1.

For each object, the effect created by a behavior is computed as the difference between the final and initial features:

$$\boldsymbol{f}^{b_i}_{\text{effect}} = \boldsymbol{f}^{(b_i)} - \boldsymbol{f}^{()}$$

where $\boldsymbol{f}^{b_i}_{\text{effect}}$, $\boldsymbol{f}^{(b_i)}$, and $\boldsymbol{f}^{()}$ represents the effect, final and initial feature vectors, and b_i represents the behavior executed.

7.3.2 Behaviors

The robot is equipped with five **move behaviors** and one **lift behavior**. The move behaviors (*move-forward*, $move_{\pm 30^\circ}$, and $move_{\pm 60^\circ}$) rotate the robot as specified by the type of the behavior and drives it forward for $40cms$. The robot is also endowed with a open-loop lift

behavior, which can be triggered by a detected object region in the range image to lift the object whose relative position can be computed from the range image.

7.3.3 Interactions

The robot interacts with three types of objects; namely boxes, cylinders and spheres, at different size and orientations. During the execution of its move behaviors, the robot may experience collisions with objects and face with different consequences. For instance, when the robot collides with boxes or upright cylinders, it would come to a stop as a result of the physical interaction. However, when the robot collides with a sphere, the sphere would roll away not blocking the robot's movement. The robot may or may not get blocked when it collides with lying cylinders depending on the relative orientation of the cylinder. The lift behavior would succeed in lifting an object, if the object is within the arm length of the crane and has a flat top (assuming that all objects are magnetizable). In this sense, all boxes and cylinders are liftable, whereas spheres and lying cylinders are not since a one-point-contact of the electromagnetic gripper is not sufficient to produce the necessary force for lifting.

7.4 Learning Affordances

The robot first discovers what type of effects it can generate in the environment, i.e. computes the effect categories by grouping similar effects together. Then, it learns the forward model to predict the effect categories given object features and behaviors. The data collected during interaction phase is in the following form:

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i >\}$$

Effect categories are found for each different behavior separately and by using the set of effect vectors $\{\boldsymbol{f}^{b_i}_{\text{effect}}\}$. Effect feature space is clustered using k-means clustering method for each behavior. Each cluster is assigned to an effect category id ($E^{b_i}_{id}$) and cluster centers are used as corresponding category's representative prototype ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$).

Effect category prediction is performed by learning the mapping between $\boldsymbol{f}^{()} \rightarrow E^{b_i}_{id}$ mapping, for each behavior separately. For this purpose, one classifier ($Predictor^{b_i}()$) is trained

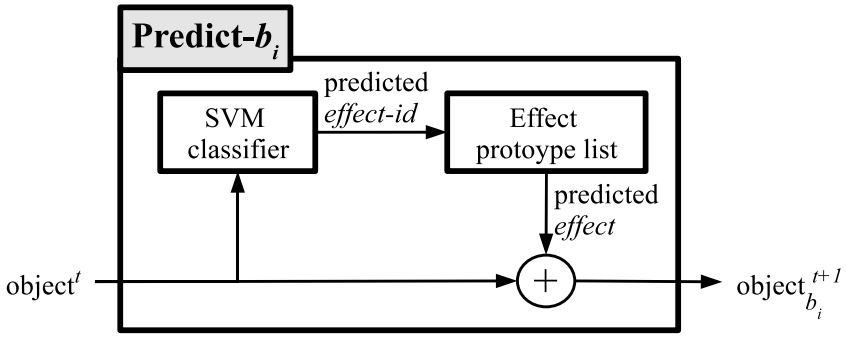


Figure 7.2: The `Predict-` operator. Is is trained to predict the next state of an object based on the predicted effect of applying behavior b_i.

for each behavior b_i where the initial object features are used as inputs and corresponding effect categories are used as target categories. Support Vector Machine (SVM) classifiers with Radial Basis Function (RBF) kernels are used because they are robust in the face of noisy input and able to deal non-linear mapping in large datasets and input spaces.

The trained SVM classifiers allow the robot to predict the type of *effect* a behavior is expected to generate when applied on a given object $\boldsymbol{f}^{()}_{o_j}$ using:

$$E^{b_i}_{id} = Predictor^{b_i}(\boldsymbol{f}^{()}_{o_j})$$

The predicted percept of the object after the application of the behavior can then be computed as (see Fig. 7.2):

$$\boldsymbol{f}^{(b_i)}_{o_j} = \boldsymbol{f}^{()}_{o_j} + \boldsymbol{f}^{b_i}_{\text{prototype},id}$$

7.5 Planning Using Learned Affordance Relations

The learned affordance relations can be used as operators for planning.

- **States:** A state is represented as the set of objects perceived or expected to be perceived after execution a number of behaviors:

  $$S^{(b_1 \ldots b_l)} = [\boldsymbol{f}^{(b_1 \ldots b_l)}_{o_1} \ldots \boldsymbol{f}^{(b_1 \ldots b_l)}_{o_m}]$$

  where o_m corresponds to the m$^{\text{th}}$ perceived object, and $\boldsymbol{f}^{(b_1 \ldots b_l)}_{o_m}$ is the percept of object m after execution of the behavior sequence $\{b_1 \ldots b_l\}$.

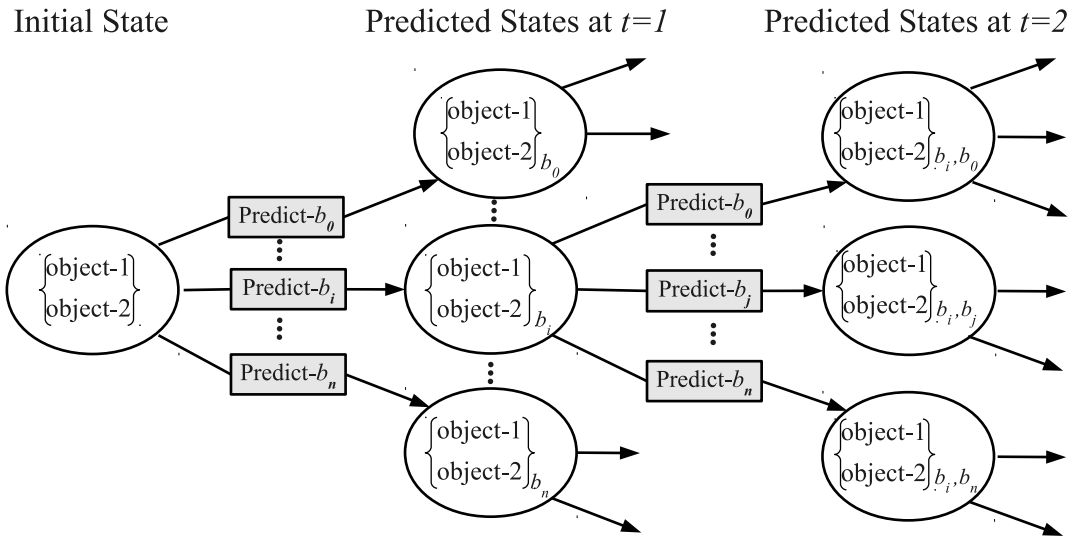


Figure 7.3: The breadth-first construction of the plan tree. States include one or more objects whose next states are predicted based on the operators in Fig. 7.2.

- **Actions:** The pre-coded behaviors; namely the five move behaviors and the lift behavior, constitute the actions. Different from standard techniques, the actions do not have any pre-conditions and their description does not include pre-defined state transition rules. All actions are applicable in all states, where the next state depends on the learned effect prediction operators summarized in Fig. 7.2.

$$S^{(b_1 \ldots b_l)} \xrightarrow{b_k} S^{(b_1 \ldots b_l, b_k)}$$

- **Goals:** A goal is specified as a partial state, in terms of values of some object features within states. The user can define a goal based on feature values of any object, of a particular object or the combination of both. For example, the state that includes an object feature vector with $d_{min} < 0.1m$ will satisfy the goal of *approach any object*. As another example, the goal of *pick-up a particular object* is satisfied in a state, where the bottom-most row feature value of the corresponding object is large ($r_b > 180$) in the range image.

- **Plan generation:** Forward chaining is used to generate totally ordered plans starting from the initial state (See Fig. 7.3). This process can be viewed as the breadth-first construction of a plan tree where the branching factor is the number of behaviors. The next states are computed using the prediction operator in Fig. 7.2. If the state in any

time step satisfies the goal, the sequence of the behaviors which lead the initial state to the goal is accepted as a potential plan.

7.6 Experiments

The learning experiments are conducted in a physics based simulator where the robot is verified against the real robot in [139]. where Gaussian noise is used in sensor and actuator modeling. One random object o (among) is placed in $[-90°, +90°]$ of robot's frontal area, in a random orientation and size $[20cm - 40cm]$. The robot makes 3D scans before and after executing one of its behaviors (b) to compute the object ($\boldsymbol{p}_o$) and effect (($\boldsymbol{\xi}_o^b$) feature vectors. For the lifting behavior 1000 interactions are simulated, whereas for the move behaviors 3000 interactions are simulated. The resulting set of relation instances **I** are then used in training.

7.6.1 The Learning of Lift Behavior

The set of effects ($\{< \boldsymbol{f}_{\text{effect}}^{lift}\}$) are split into two clusters using k-means. After clustering phase completed, each object in the training set is assigned to an *effect-id*, based on the class to which the created effect category E_{id}^{lift} belongs to (Equation3.1). Fig. 7.4 shows the effect classes of these entities together with the shape and position information for 2-cluster case. The objects assigned to class + are the ones with flat top and close proximity to the robot. On the other hand, close objects with curved tops (spheres and lying cylinders) and all distant objects are assigned to a separate class ('$\bullet$'). Hence, we can conclude that the robot learned to distinguish successful and unsuccessful lift actions.

After assigning each object to an effect-class, in order to learn the mapping between initial percept of the objects and the corresponding effects an SVM classifier is trained. The parameters of SVM training with RBF kernel are optimized in a grid search. $c = 0.03$ and $\gamma = 32000$ are set as optimum cost of SVM and width of Gaussian respectively. After training is completed, prediction accuracy of SVM model is tested on a distinct set of relation instances, where the result of lift behavior is also included as ground truth. Fig. 7.5-left plots the prediction accuracy of liftability with respect to the size of the training sets. As results indicate, the SVM classifiers trained with more than 700 samples have performance over 90%. Additional

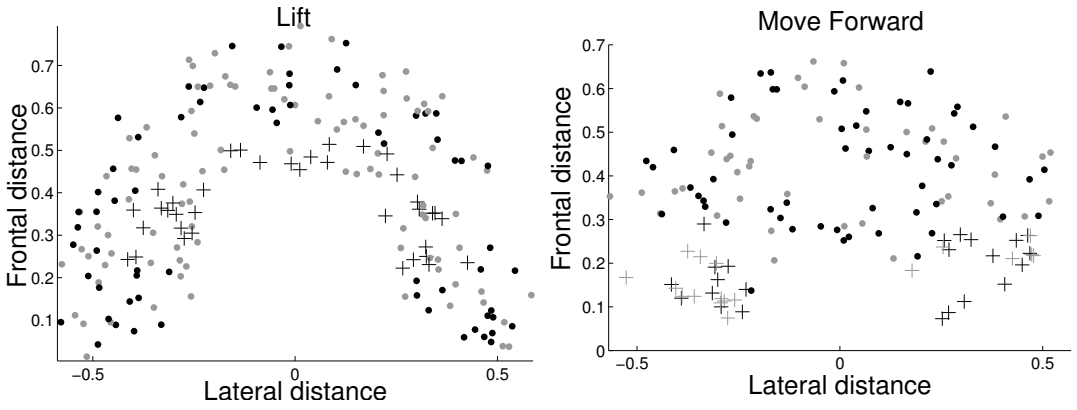


Figure 7.4: The effect categories obtained for *lift* and *move-forward* behaviors. Each environment in interaction phase includes only one object, and each marker corresponds to the placement of the object in a different environment. Dark markers represent boxes and upright cylinders; light markers represent spheres and lying cylinders. '.' (dot) and + (plus) illustrates how these objects are clustered at the end.

training do not increase the performance.

7.6.2 The Learning of Move Behaviors

The same type of learning is also applied to the data obtained from the five move behaviors. $c = 512$ and $\gamma = 0.125$ are set as optimum cost of SVM and width of Gaussian respectively. However, for simplicity, we will only discuss the results obtained from the move-forward behavior. Fig. 7.4 shows the effect categories for different objects together with the shape and position of those objects for 2 clusters. Independent of their shapes, objects located within $0.3m$ are within class +, and all distant objects are within class $\bullet$. Additionally, some $\bullet$ objects in front of the robot are closer than + objects. These observations show that the clustering process makes a distinction on effects based on whether the object disappears from the view of the robot or not.

Using the same training data, we varied the number of clusters being used to cluster the effect data, and measured the prediction accuracy . As the number of clusters increase, the clustering process also incorporates the differences in shapes of the objects. The accuracy of prediction of effect classes based on the objects are also examined and found to be over 90% for number of clusters up-to 10.

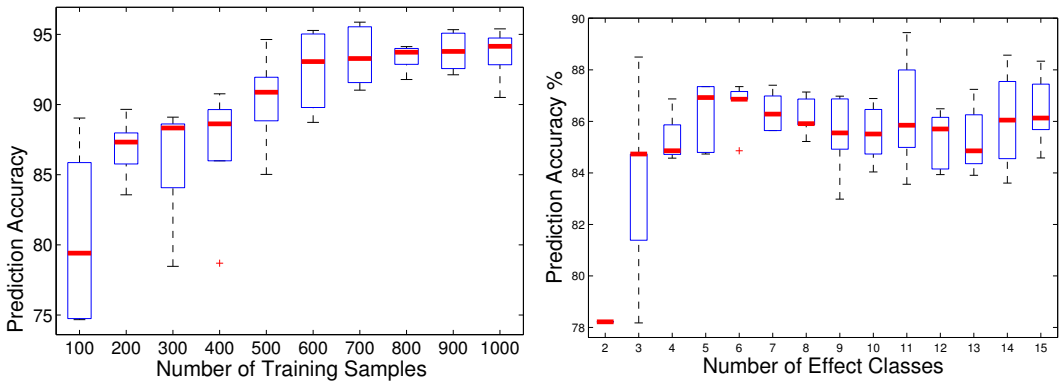


Figure 7.5: The effect of the training sample count and the effect class count on prediction accuracy. . **Left:** Performances of SVM classifiers trained with different number of samples in predicting lift effect classes (liftability). **Right:** The accuracy of liftability prediction in two-step planning. The boxes shows the distribution of prediction accuracy obtained in testing. The box is bounded by lower and upper quartile values, the line in the box refers to the median, and the whiskers show the extent of the data.

7.6.3 Two-step Planning

In this set of experiments, we evaluated the prediction accuracy of the robot to perceive the liftability of an object that are randomly placed within the $1m$ range of the robot. The robot applied the `Predict-`*move-forward* and `Predict-`*lift* operators (Fig. 7.2) to the initial percept of the object and using the final predicted percept of the object determine whether it's liftable or not. The predicted effect is then compared with the actual effect obtained by executing *move-forward* and *lift* behaviors.

Fig. 7.5-right plots the accuracy of liftability prediction for such two-step plans with respect to the number of effect clusters being used in the training of the *move-forward* behavior. The training set contained 3000 relation instances. Two points can be made. First, the average prediction accuracy of two-step plans for liftability (around 85%) is lower than the average prediction accuracy obtained from '1-step' plans. This is an expected result since as the objects get further away, the resolution of their perception degrades reducing the accuracy. Second, the number of effect clusters to be used in the training of the *move-forward* behavior should be greater than 2 to achieve an good prediction accuracy. This is probably due to the fact that the use of only 2 prototypes does not provide the necessary resolution to the *move-forward* behavior that can be propagated for planning.

Table 7.1: The prediction performance for one to four step plans.

				
1 step plan	0/0	103/103 100.0%	115/115 100.0%	0/4
2 step plan	0/0	538/605 88.92%	598/717 83.40%	0/134
3 step plan	0/61	727/889 81.77%	1072/1198 89.48%	0/251
4 step plan	0/153	339/559 60.64%	475/697 68.14%	0/111

7.6.4 Multi-step Planning

In this set of experiments, a randomly selected object is placed at a random position within $1m$ range from the robot. Based on the initial percept of the object, 1-step, 2-step, 3-step, and 4-step plans are generated for liftability and it's correctness is checked in simulation through the execution of the behavior sequence. Table 7.1 reports the results obtained from 3000 such interactions. Note that, the 1-step plans mean immediate liftability and correspond to objects that are close to the robot. Similarly, it can be assumed that objects that are liftable with longer plans are generally further away. Two observations can be made regarding the results. First, the number of potential plans first increase with the number of steps, and then decrease. This is probably due to the fact that forward chaining expands the set of potential states that can be reached and hence the possibilities. The decrease in the number of plans at 4-step plans is probably due to the fact that the objects are further away, and that the degradation in their resolution reduces the chances of planning to satisfy the goal of liftability. Second, as expected, the prediction accuracy of the plans goes down with the length of the plans as expected. Two reasons can be speculated for this, the loss of precision through the use of prototypes and the object being placed further away.

7.6.5 Case Study: Bringing an Object on Top of Another

In all the experiments reported so far, a single object is presented to the robot during evaluation, as has been done during the training phase. In this experiment, we put the robot into an environment containing multiple objects and specify the goal as the conjunction of two predicates.

In this experiment, the robot is asked to lift an object and go towards a button (pre-defined object). The robot is free to select the object to lift. The goal is defined over desired future entities based on the predicted outcomes following the execution of planned action sequence. The goal for lift is to obtain an outcome for any object, where the bottom part of the predicted outcome range image should be high, i.e. $r'_b > 180$ where r'_b is the 'bottom-row' feature of final feature vector. The goal for approach is defined as obtaining close proximity to a pre-defined object. The mean distance of the predicted final feature vector of that object, after the plan is executed should be small, i.e. $d_{mean} < 0.1m$. Thus, the overall goal is to obtain an outcome of any object which satisfies lift goal and an outcome of the pre-defined object which satisfies approach goal.

A plan for an environment that includes 5 objects is presented in Fig. 7.6, where the robot is required to lift any object and approach to a defined object (shown as a button). As shown, the generated plan is composed of three steps: $< move_{30}; lift(o_3); move_{-60} >$. We can make three observations. First, the robot is able to predict the liftability of object 3, before approaching. Moreover, the two cylindrical objects, one of which is laying on its side, and the other with a non-flat top, are correctly predicted to be non-liftable. Third, note that although object 1 is a cylinder with a round top, which is a novel object that was not used in training, the robot is able to predict its (not-)liftability.

7.6.6 Case Study: Novel Objects in Real World

The plan generation is also tested in real world for liftability. The learned affordance relations and effect prediction methods are directly transferred to real robot and its plan generation ability is tested. The environment contained six objects: A desktop world globe with a base, a box shaped power supply, an irregular hexagonal shaped metal piece, a triangular prism-shaped desk calendar, a can lying on top of another upright can (not visible), and an upside-down small pot as seen in Fig. 7.6. The segmented range image and the shortest plans to lift each of them (if there exists) is shown on the right hand side of the figure. The following plans are made. The pan is within the reach of the crane arm, and hence it can be directly lifted. The robot is required to move in order to pick-up the power supply or metallic piece. No plan is generated to lift the globe with base and the triangular prism since they did not have flat tops. However, it should be noted that the robot made an incorrect plan to lift the

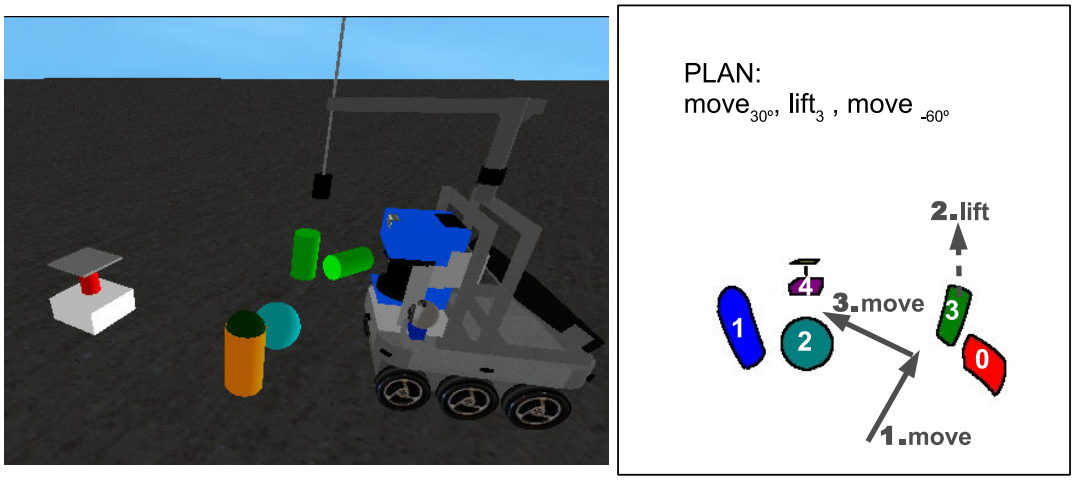


Figure 7.6: The generated plans for the 'bring an object on top of the button' task. On the left, 5 objects (including the button) are placed in the environment. On the right, the detected objects and the detected parts of the gripper arm in the range image are shown together with the generated plan. The object numbers in the range image are assigned automatically in the perception process, where button is numbered as 4. The robot cannot lift any object except the cylindrical shaped standing object on the right. However in order to lift it, the robot should approach to it by executing $move_{30^\circ}$ action. After the $lift_3$ step, $move_{-60^\circ}$ behavior is predicted to drive the robot towards the goal object.

lying cylinder on its left. This is probably due to the fact that the robot predicted the effect of move_{-60° wrongly to achieve liftability[1].

7.7 Conclusion

In this chapter, we studied a method that allows a robot to learn symbolic relations that pertain to its interactions with the world and showed that they can be used in planning. We have shown that a mobile robot can learn the physical affordances of objects from range images and use them to build symbols and relations that can be used in making multi-step predictions about the affordances of objects and achieve complex goals.

[1] Due to a mechanical breakdown in our crane, we were not able to test these plans on the real robot. Although this is unfortunate, we do not believe that it undermines the validity of the results presented due to two reasons. First, the learning of initial percept as well as the effects being produced, takes place on the range images, and that we used range images produced in real-world to test the learned relations. Second, by their very definition (as described in the last paragraph of the first page), the behaviors are assumed to be implemented in a closed-loop manner. Any failure (or success) in their execution would be due to their particular implementation and does not provide any implications for the learning method proposed in this chapter.

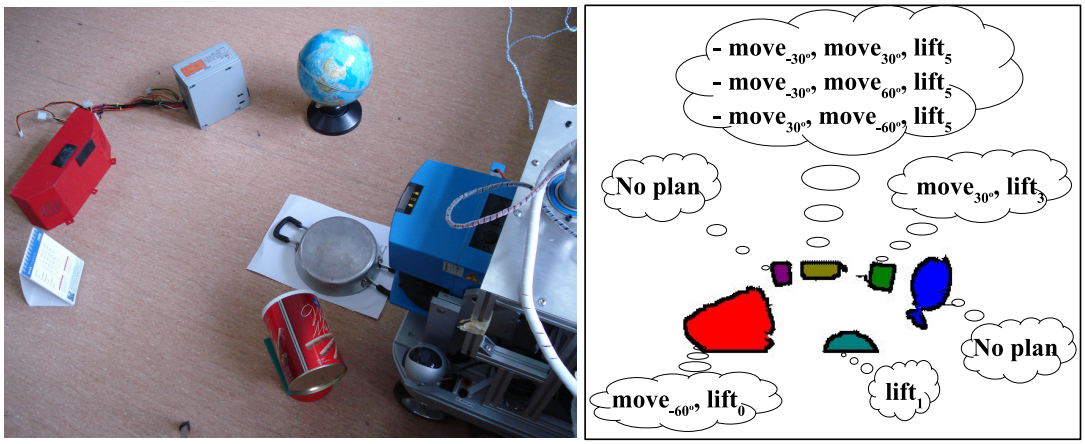


Figure 7.7: The generated plans for the 'lift any object' task with real robot. On the left, photograph of the environment where six real world objects are placed in front of the robot. On the right, the object regions detected in the range image are shown together with different plans generated for lifting. The bottom parts of the nearest two objects are not perceived and not seen in the range image since the laser beams are blocked by the robot base.

7.8 Discussion

In spite of the non-trivial learning and planning ability that our system exhibits, it has room for improvement.

The method that is used to discover effect categories is not robust. First, it uses a clustering algorithm where the number of clusters is fixed. Thus, the number of effect categories should be decided in advance. Second, the clustering in effect feature space is sensitive to the relative weighting of the effect features such as distances, pixel counts, histogram frequencies that are encoded in different units. In the next chapter, a channel-based hierarchical clustering algorithm will be proposed to obtain robust effect categories.

One other issue is that, blind execution of the generated plans is not realistic in dynamical environments. The predictions on object features created during plan generation can be used to monitor the plan execution and to check whether the change in the state is as the one that was predicted in the plan, to decide whether execution was successful or not. This problem will also be studied in the next chapter.

Last, we started studying manipulation affordances in this chapter in a very simple sense: If the object has a flat top surface and if it is in reach distance of the robot crane arm, then it's liftable. However, the manipulation in real life is much more complicated. First of all, when

a more complex manipulator, such as human's hand is considered, the graspability of the object is determined by the combination of many different properties of the object and hand. Furthermore, there are various actions that can be executed on objects such as pushing from different directions, grasping, lifting, dropping, shaking, etc. These and other issues related to affordance learning in manipulation domain will be studied in the next three chapters.

CHAPTER 8

GOAL EMULATION AND PLANNING IN A MANIPULATOR ROBOT

8.1 Introduction

For a growing infant, a major problem is to make sense of the continually incoming sensorimotor data by learning what changes she can generate in the environment. Only after this problem is overcome, the infant starts making plans and executes them for achieving goals, for example pulling the table cloth to reach a toy that is otherwise unreachable. It is plausible to think that earlier planning takes place in the perceptual domain of the infant, which is later augmented by symbolic planning capability as the infant forms symbolic representations through her interaction with the environment.

In this chapter, we consider the former phase of this developmental progression in a robotics context, where the anthropomorphic hand-arm robot learns its visuomotor capabilities by interacting with its environment. We are content that by adopting such a developmental approach, adaptive and human-like robotic systems can be synthesized. In the last decade, with similar views in mind, various developmental stages have been studied, modeled and transfered to robots. These stages correspond to acquisition of skills at different levels and ages, ranging from emergence of motor patterns before birth [87] and development of pattern generators for crawling [123] to language learning [68] (see [4] for a comprehensive review).

In the postnatal age of 7-10 months, the infant explores the environment actively. By observing the effects of her hitting, grasping and dropping actions on objects, she can learn the dynamics of the objects [4]. The infant in this stage has already acquired a number of manipulation behaviors and is able to detect different properties of objects such as shape,

position, color, etc. Using her motor skills, the infant interacts with the environment and observes the changes she creates via her perceptual system, accumulating knowledge about the relationships between objects, actions and the effects. This process effectively corresponds to the learning of the affordances [55] provided by the environment. The learning in this stage is largely performed in a goal-free fashion through self-exploration and self-observation [122, 17, 41, 151]. After approximately 9 months of age, the infant starts using the learned object-action-effect relations in a goal-directed way, anticipating a desirable change in the environment and behaving accordingly [119, 138, 158]. This skill ranges from recalling action-effect mappings to making simple plans that may involve multiple steps [157]. Goal-emulation, a form of imitation characterized by the replication of the observed end effect [151], starts after this period, and infants become skilled at imitating unseen movements after 12 months of age [40]. According to [42], infants learn to use anticipation for goal-directed actions in two phases. In the first phase, they execute random actions in the environment, self-monitor the changes, and learn the action-effect associations in a bi-directional way. Later, in the second phase, they start to control their actions by predicting the effects they can create.

In a similar vein, in the first phase, our manipulator robot experiences a goal-free self-exploration and self-monitoring phase where it discovers the affordances provided by the environment and learns how to use these affordances to predict the next perceptual state after the execution of a given behavior. In the second phase, the robot emulates goals presented in its sensory space by generating multi step plans based on the learned affordance and prediction capabilities.

8.2 Framework Implementation

- **Behavior:** *Behaviors* correspond to discrete pre-defined actions without any parameter. Thus, they will be represented by b_i as in previous chapters.

- **Entity:** The *entity* corresponds to the feature vector computed for one object and is used interchangeably with the term *object* in this chapter. The robot learns object affordances by interacting with one object at a time. On the other hand, this learning enables the robot to make predictions over multiple objects. Thus an object identifier is included in representing *entity/object* (when required) as follows: $\boldsymbol{f}^{()}_{o_j}$. Here subscript $\boldsymbol{f}$ corresponds to the feature vector, o_j is the object identifier (j^{th} detected object) and () includes the

list of the behaviors executed so far.

- **Effect category:** The robot discovers a variable number of *effect categories* ($E_{id}^{b_i}$) for each behavior b_i during it's interactions. A novel hierarchical unsupervised categorization method is proposed for this purpose. Further, each effect category has a representative effect prototype vector ($\boldsymbol{f}_{\text{prototype},id}^{b_i}$), which is also found by the robot.

- **Effect:** The robot perceives and represents the change in perception of the entities during its behavior executions. The *effect* feature vector ($\boldsymbol{f}_{\text{effect}}^{b_i}$) represents this change and is used to learn affordances and make predictions.

- **Affordance relation instance:** The affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< \boldsymbol{f}_{\text{effect}}^{b_i}, \boldsymbol{f}^{()}, b_i >\}$$

8.3 Experimental Setup

The anthropomorphic manipulator which is composed of PA-10 robot arm and Gifu robot hand is used with infrared range camera.

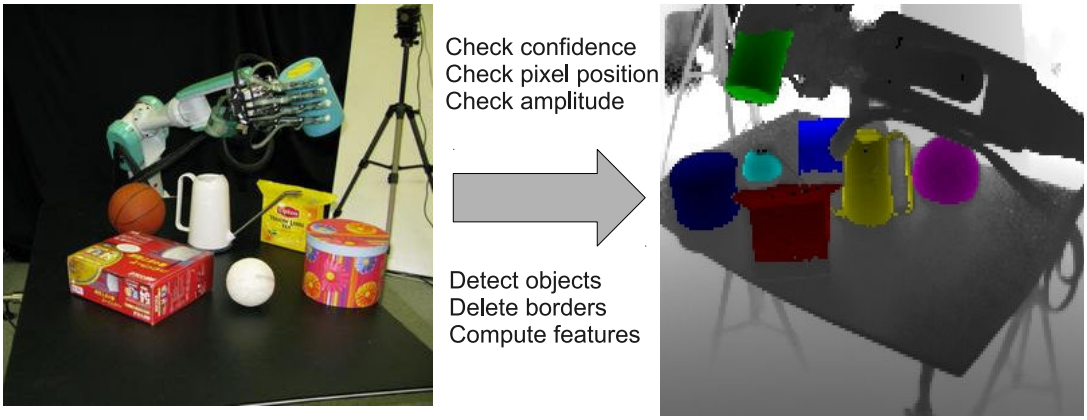
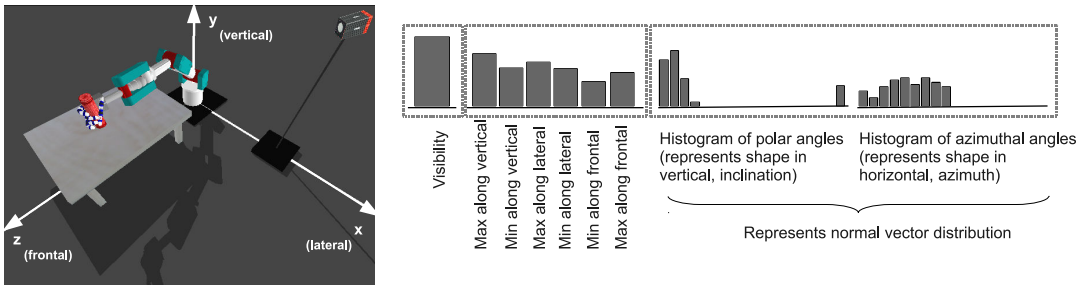


Figure 8.1: 23 DOF hand-arm robotic platform and the range image. In (a), the hand-arm system, infrared range camera (on the top-right) and the objects that are used in this study are shown. In (b), the range image obtained from the range camera and the detected objects are shown where range is encoded in grayscale and in color for the environment and objects, respectively.



a) Snapshot from the simulator

b) Object feature vector

Figure 8.2: (a) The robot arm grasps and lifts a cylindrical object in the physics based simulator. The coordinate system is also illustrated. (b) The 43-dimension feature vector computed for the object in the robot's hand in Figure 8.1 is given. It is composed of 1 visibility, 6 position and 36 shape related features whose values correspond to the height of the bars in normalized form.

8.3.1 Perception

Object Detection: The first step of pre-processing is to filter out the pixels whose confidence values are below an empirically selected threshold value. The robot's workspace consists of a black table, so region of interest is defined as the volume over the table, and black pixels are filtered out as the range readings from black surfaces are noisy. As a result, the remaining pixels of the range image are takes as belonging to one or more objects. These objects are segmented by the Connected Component Labeling algorithm [63] which differentiates object regions that are spatially separated by a preset threshold value (2 cm in the current implementation). In order to reduce the effect of camera noise, the pixels at the boundary of the object are removed, and median and Gaussian filters with 5x5 window sizes are applied. The detected objects on the range image of a sample setup is shown in Figure 8.1 (b). Finally, a feature vector for each object is computed using the 3D positions obtained from depth values of the corresponding object pixels as detailed in the next paragraph.

Object feature vector computation: The perception of the robot at time t is denoted as $[\boldsymbol{f}_{o_0}^{t,()}, \boldsymbol{f}_{o_1}^{t,()}..]$ [1] where $\boldsymbol{f}$ is a feature vector of size 43, and the superscript $^{()}$ denotes that no behavior has been executed on the object yet. Three channels of information are gathered and encoded in a feature vector for each object o_j (Figure 8.2 (b)). The first channel consists of

[1] Note that t and o_j are sometimes omitted in the rest of this chapter in order to ensure easy readability of the notation.

Algorithm 4 Object Detection

isConfident(p): true if confidence[p] ≥ confidence-threshold

isOnTable(p): true if position[p] is on table

isBright(p): true if amplitude[p] ≥ amplitude-threshold

setObjectPart(p): pixel p is assigned as object part

```
for each pixel p (from 0 to 174 × 144) do
    if (isConfident(p)) and (isOnTable(p)) and (isBright(p)) then
        setObjectPart(p)
    end if
end for
Find distinct objects with Connected Component Labeling
Remove pixels on object boundaries
Apply Median and Gaussian filters to object pixels
```

object visibility feature which encodes the knowledge regarding the existence of the object. The second channel corresponds to the distance perception of object's borders. Here, the points with minimum and maximum values along longitudinal, lateral and vertical axes are used as 6 position related features. The third channel encodes the shape related features, where the distribution of the local surface normal vectors are used. Specifically histograms of normal vector angles along the latitude and longitude are computed and used as follows.

The normal vector of the local surface around each point is calculated using the positions of the two neighbors in the range image:

$$\mathbf{N}_{r,c} = (\mathbf{p}_{r-n,c} - \mathbf{p}_{r,c}) \times (\mathbf{p}_{r,c-n} - \mathbf{p}_{r,c})$$

where $\mathbf{p}$ represents 3D position, n corresponds to the neighbor pixel distance and is here set to 5. In spherical coordinates, the unit length 3D normal vector is represented by two angles, polar (θ) and azimuthal (φ) angles that encode information along latitude and longitude, respectively. The polar angle (θ) corresponds to the angle between x-z plane and the normal vector, whereas φ is the angle between z-axis and the normal vector's orthogonal projection on x-z plane. After polar and azimuthal angles are computed for each pixel, two histograms are computed in θ and φ using a 20° bin size. Finally, the angular histograms represent the 36 shape related features.

Effect feature vector computation: For each object, the effect created by a behavior is

defined as the difference between its final and initial features:

$$\boldsymbol{f}^{(b_i)}_{\text{effect},o_j} = \boldsymbol{f}^{(b_i)}_{o_j} - \boldsymbol{f}^{()}_{o_j}$$

where $\boldsymbol{f}^{(b_i)}_{o_j}$ represents the final feature vector computed for object o_j after the execution of behavior b_i.

8.3.2 Behaviors

The robot interacts with the objects using three *push* behaviors and one *lift* behavior. The object position computed from the range camera is used as argument for the behaviors to enable the robot interact with objects placed in different positions. The hand is initially wide-open for all behaviors, is clenched into a fist during *push-forward* execution, and remains open for other *push* behaviors. For *push-forward*, *push-left*, and *push-right* behaviors the robot hand is brought to the rear, right and left side of the object, respectively. Then, the hand moves towards the object center, pushing the object in the appropriate direction. After behavior execution, the hand is placed to a 'home' position. In the *lift* behavior, the robot hand is placed at the back-right diagonal of the object first, then moved towards the object while the fingers are closed to grasp the object. After the fingers come to a halt, the hand is lifted vertically.

8.3.3 Interactions

The robot interacts with three types of objects: boxes, cylinders and spheres of different size and orientation. During the execution of *push* behaviors, the robot observes the consequences of its actions. For instance, when the robot pushes a box ( ) or an upright cylinder( ), the object is dragged during the execution of the behavior and stand still at the end of the action. However, when the robot pushes a sphere ( ), the object rolls away and falls down the table. The *lift* behavior would succeed in lifting an object, if the object is within the arm length of the robot and small enough to fit into the robot hand. However the consequences of the *lift* behavior execution is not limited to having lifted the objects and can be complex. For example, some spheres may roll out of the view after an attempt to grasp and lift, while large boxes will be pushed away but still remain in the view after the *lift* behavior execution.

Algorithm 5 Exploration phase

1: **for** each trial k (from 1 to m) **do**
2: Reset robot joint angles
3: Put a random object in random position, size, and orientation
4: **while** isObjectVisible() **and** isObjectPositionChanged() **do**
5: Perceive the environment and compute initial feature vector $\boldsymbol{f}^{()}$
6: Execute a random behavior b_i where $0 \leq i \leq 3$
7: Perceive the environment and compute effect feature vector $\boldsymbol{f}^{b_i}_{\text{effect}}$
8: Put $< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i >$ into repository.
9: **end while**
10: **end for**

8.4 Learning Affordances

The data collected as tuples during the exploration phase are stored in a repository

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i >\}$$

and is used by the robot to learn the affordances of objects. The learning process consists of two steps: the unsupervised discovery of effect categories, and the training of classifiers to predict the effect categories from object features. The learning process is applied separately for each behavior as detailed below. Note that the relevant features will be discovered during experiments.

8.4.1 Effect Category Discovery

In the first step, the effect categories and their prototypes are discovered through a hierarchical clustering algorithm (Figure 8.3). In the lower level, channel-specific effect categories are found by clustering in the space of each channel, discovering separate categories for visibility, position and shape. In the upper level, the channel-specific effect categories are combined to obtain all-channel effect categories using the Cartesian product operation. In Figure 3, where a hypothetical example is depicted, the effect category $E_1 = V_1 P_1 S_1$ stands for $E_1 = V_1 \wedge P_1 \wedge S_1$ and contains the effect feature vector instances which are classified as V_1, P_1, and S_1 when only the corresponding feature-channel is considered, respectively.

Finally, the effect categories that occur rarely (indicated in the figure as shaded regions) are automatically discarded together with their members. The proposed hierarchical clustering method is superior to simple one-level clustering method, since the results of one-level clustering are sensitive to the relative weighting of the effect features that are encoded in different units (e.g. continuous position features vs. binary visibility feature). Additionally, the performance of the clustering process is optimized by running the clustering algorithm multiple times and selecting the best clusters based on their utility in the second step of learning.

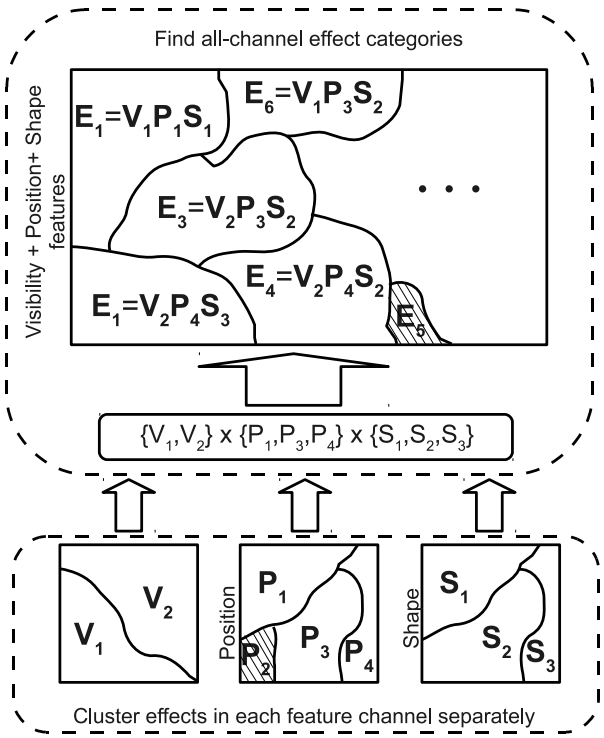


Figure 8.3: The proposed hierarchical clustering method to discover effect categories. Channel-specific and all-channel effect categories are shown on lower and upper levels, respectively.

After discovering the effect categories and assigning each feature vector in the set of $\{\boldsymbol{f}^{b_i}_{\text{effect}}\}$ to one of the effect categories ($E^{b_i}_{id}$), the prototype effect vectors ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$) are computed as the average of the category members. In order to represent the experience of the robot in a more compact way, the continuous effect vectors are replaced by effect category id's and their prototypes; and the repository is thus transformed into the following form:

$$\{E^{b_i}_{id}, \boldsymbol{f}^{()}, b_i\}, \{< E^{b_i}_{id}, \boldsymbol{f}^{b_i}_{\text{prototype},id} >\}$$

Here, the first list corresponds to the set of affordance relation instances where effects are gen-

eralized and the second one corresponds to the list of <effect-category-id, prototype vector> pairs.

8.4.2 Learning Effect Category Prediction

In the second step, classifiers are trained to predict the effect category for a given object feature vector and a behavior by learning the mapping $\boldsymbol{f}^{()} \rightarrow E_{id}^{b_i}$ mapping. Effectively, this establishes a forward model, $Predictor^{b_i}(\boldsymbol{f}^{()})$ that returns $E_{id}^{b_i}$ for each behavior.

At the end of these two learning steps, affordance relations are encoded as:

$$\{Predictor^{b_i}()\}, \{< E_{id}^{b_i}, \boldsymbol{f}_{\text{prototype},id}^{b_i} >\}$$

or

$$\{\{Predictor()\}, \{< E_{id}, \boldsymbol{f}_{\text{prototype},id} >\}\}^{b_i}$$

allowing the robot to 'know' the effect of a behavior in terms of the effect category and its prototype.

8.5 Learning Results

In the experiments, a table with 100×70 cm^2 surface area was placed in front of the robot with 40 cm distance, as shown in Figure 8.1. At the beginning of each exploration trial, one random object (, , or) of random size $[20cm - 40cm]$ was placed on the table at random orientation (see Algorithm 5). For all behaviors, 5000 interactions were simulated and the resulting set of relation instances were used in learning. The X-means algorithm [115] was used to find channel-specific effect categories[2], and Support Vector Machine (SVM) [148] classifiers were employed to learn effect category prediction.

In the rest of the section, the effect categories that were discovered using the proposed hierarchical clustering algorithm are interpreted, and the contributions of specific object features for affordance prediction, i.e. the features relevant to affordance perception, are assessed.

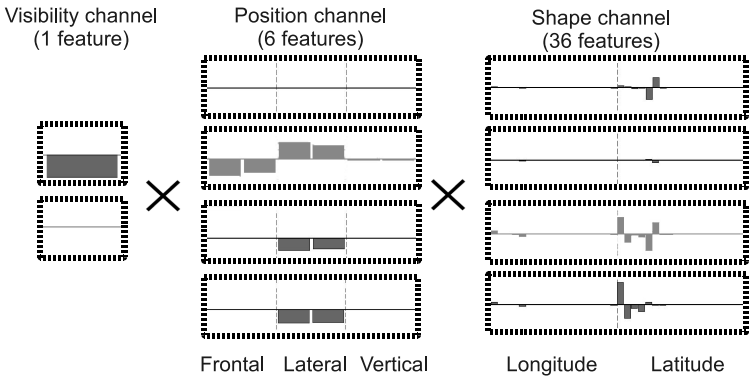


Figure 8.4: The effect categories discovered in different feature channels. Each dashed-box corresponds to a channel-specific effect category discovered for the *push-right* behavior. The category prototype vectors are represented by the bars in normalized form. For example, the position of the objects, which created the last effect category in position channel, reduced along lateral axis and did not change along other axes. The light colored prototypes are discarded since the number of members was below the threshold.

8.5.1 Discovered Effect Categories for Push Behaviors

The detailed results are given for only one representative behavior, *push-right* , as all the *push* behaviors produced similar effect categories. The channel specific effect categories discovered for the *push-right* behavior and their prototypes are shown in Figure 8.4. Two categories are discovered within the visibility channel. The first category corresponds to the disappearance of the object (indicated by a change of -1 on visibility feature) and the second category represents the effect where the object remains in the view (indicated by no change).

The changes in object position channel are represented by four distinct effect categories. The first category represents the case for no change in object position, and the third and fourth categories represent different magnitudes of object movement. The occurrence of the second effect category is very rare, i.e. the ratio of the members in this category to whole sample set is below a preset threshold (of 3%), hence is discarded. In the shape channel, four effect categories are discovered but one of them (third category) is discarded as its ratio was below the threshold.

The all-channel effect categories are computed by taking the Cartesian product of the channel-specific effect categories. The 2 categories in the visibility and 3 categories in both the position

[2] X-means implementation in Weka data mining software is used [94].

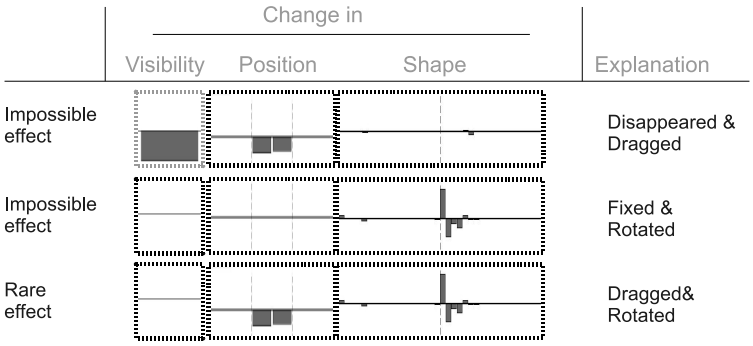


Figure 8.5: Impossible or rare effect categories that are formed through Cartesian product of channel-specific categories for push-right behavior. Some of the categories can be created due to inaccuracies in simulator and some of them do occur very rarely.

and shape channels generate $2 \times 3 \times 3 = 18$ all-channel categories. A pruning process is applied as in the lower level, to remove the impossible and rare effects based on the number of category members.

Figure 8.5 shows some of such categories that are obtained due to rare occurrence in robot's experience. The first illustrated category is physically impossible because the object disappears according to the visibility feature, and at the same time moves to a visible position based on the position feature. In the second category, object's position is not changed but it is rotated around. This is also impossible unless the object is attached to the table, which is not the case in our setup. The third category, where the object is pushed to the right and rotated, is possible but rare, as the objects are pushed from the center.

Next, we analyze the prototypes of remaining effect categories (Figure 8.6).

- The *unreachable effect* (Effect-2) corresponds to the prototype where no feature change is observed. The average distance of the objects that produced an *unreachable effect* is 124.4 cm indicating their *unreachability* given the kinematics of our robot.
- In the *disappear effect* (Effect-1), the visibility of objects drop from 1 to 0, indicating the objects falling off the table. This can happen when the objects are pushed and rolled out of the table. We found that most of the objects that fall under this category are spheres, since they are likely to roll away and fall from the table. However, boxes and upright cylinders placed on the edge of the table also fall under this category as they fall of the table when pushed. The *disappear effect* (Effect-1) was also created by the

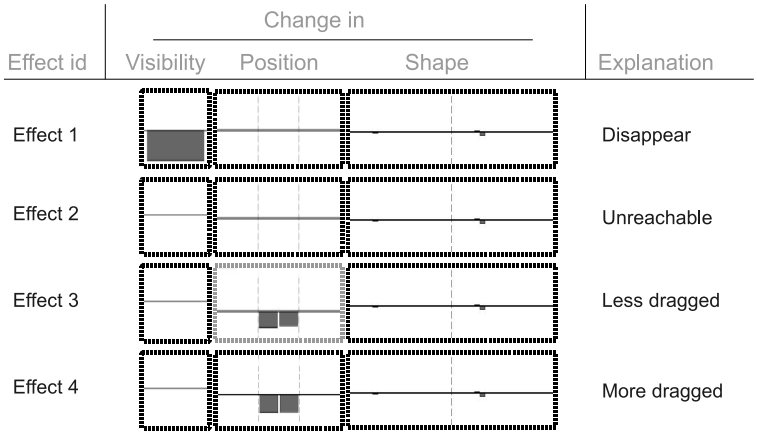


Figure 8.6: The effect category prototype vectors for push-right behavior. It can be seen that *push-right* has an effect on visibility and lateral position features but not in others.

objects which were elevated over the table. This happens when the robot had executed a successful *lift* behavior in the previous step. In such situations, a subsequent *push-right* behavior would open the hand causing the lifted object to drop and hence might make it disappear. Note that the disappearance of an object through dropping it (*lift* followed by *push-right*) was an unexpected emergent behavior.

- In the *less-dragged effect* (Effect-3) and the *more-dragged effect* (Effect-4), the lateral position of the objects are reduced (the objects are pushed right with respect to the robot) as a result of *push-right* behavior. These categories were created by only boxes and upright cylinders, and do not include any spheres since they always roll-away when pushed.

8.5.2 Discovered Effect Categories for Lift Behavior

Figure 8.7 shows the all-channel effect prototypes, discovered by the hierarchical clustering process for the *lift* behavior:

- The *unreachable effect* (Effect-2) corresponds to no significant change in the feature vector since it was created by (failed) interaction with unreachable objects, similar to Effect-2 for the *push-right* behavior.

- In the *disappear effect* (Effect-5), objects became invisible after execution of *lift* behav-

Figure 8.7: The effect category prototype vectors for lift behavior. It can be seen that *lift* behavior has an effect on vertical, frontal and lateral position features as well as visibility.

ior. This effect was created by (1) ungraspable large spherical objects that roll away after interaction, (2) ungraspable large objects that are pushed off from the left edge of the table, and (3) the objects that were already in robot's hand due to a previous *lift* behavior execution.

- In the *dragged effects* (Effect-1 & Effect-4), the vertical position of the object remains same, but its position on the table is changed indicating a drag over the table. This effect was created by large ungraspable objects that are not rollable. The objects that create *dragged effects* were pushed on the table for different amounts and in different directions since interactions with different object types and sizes result in different collision dynamics between the hand and object.

- In the *lifted effect* (Effect-3), the elevation of the objects (represented by first two columns) increase, corresponding to the cases where objects were successfully grasped and lifted.

The *lift* behavior was designed to grasp and lift the objects. Thus, from the designer's point of view, there can be two different outcomes resulting from the execution of the lift behavior: Either the object can be grasped and lifted successfully, or it can not be grasped, so can

not be lifted. However, when the effects that were obtained during the robot's exploration were clustered using the hierarchical clustering algorithm, five different effect categories were generated. These results show that the effect categories should not be limited to the definition of the behavior or the intention of the behavior designer, but should be discovered through interaction.

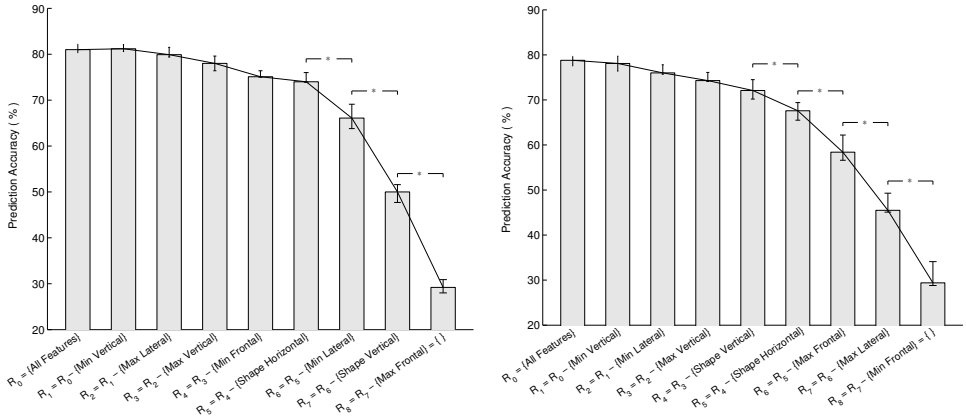
8.5.3 Effect Category Prediction Results

After the discovery of effect categories, the mapping from the initial object features to these categories is learned for each behavior b_i ($Predictor^{b_i}()$) by multi-class Support Vector Machines (SVMs). The Libsvm [19] software package was used with optimized parameters of the RBF kernel through cross-validated grid-search in parameter space. 4000 simulated interactions were used in training and a separate set of simulated 1000 interactions were used for testing. At the end, 95%, 84.3%, 82.2%, and 79.7% accuracy was obtained in predicting the correct effect categories for *push-forward*, *push-left*, *push-right*, and *lift* behaviors, respectively. The accuracy of *push-forward* is higher than other push behaviors since it has three effect categories (compared to four effect categories in other two *push* behaviors).

We analyzed the relevance of the features in affordance prediction for the *push-right* and *lift* behaviors using the Schemata Search [102] by computing the relevance of a feature based on its impact on the prediction accuracy. The Schemata Search is a greedy iterative method that starts with the whole feature set (R_0), and reduces it by removing the least relevant feature in each iteration. At each iteration (t), candidate subsets are formed by removing a different feature from R_{t-1} (remaining feature set of previous iteration), and they are evaluated by training SVM classifiers in 5-fold cross-validation. The subset with the highest mean prediction accuracy is chosen as R_t and transfered to the next iteration. The computation time is reduced by grouping the vertical (longitude) and horizontal (latitude) shape features, and treating them as single units.

Figure 8.8 shows the prediction accuracies of the feature sets produced by this method. In both plots, the first bar corresponds to the prediction accuracy with the full feature set (R_0) and the last bar corresponds to the accuracy without use of any features ($R_8 = \{\}$, base condition).

The effects of these features were further investigated by performing t-tests contrasting the



(a) Push-right behavior (b) Lift behavior

Figure 8.8: The prediction accuracies of the classifiers that are computed using different feature sets. The feature set is reduced by one feature in each iteration by deleting the most irrelevant one. The left-most and right-most bars in each plot show the results obtained using all and no features, respectively. Error bars on prediction accuracies indicate the best, median, and worst classifiers found by 5-fold cross-validation. Significant changes between adjacent feature subsets from t-test are shown (*: $p < 0.002$).

prediction accuracies of adjacent feature subsets. We found that the prediction accuracy changed significantly after removal of features from the subsets R_5 and R_4 for *push-right* and *lift* behaviors, respectively.

- For the the *push-right* behavior (Figure 8.8 (a)), the three most relevant features were *Min. Lateral, Shape Vertical*, and *Max. Frontal*. The *Shape Vertical* feature has direct relation to the rollability of objects, whereas the *Max. Frontal* and *Min. Lateral* features determine the object's position on the table and hence give information about whether the object is reachable or fallable from the edge. Note that, removing *Min. Frontal* feature from the training set did not have a significant effect on accuracy since existence of *Max. Frontal* in that set makes *Min. Frontal* redundant.

- For the *lift* behavior (Figure 8.8 (b)), *Min. Frontal* is among the most relevant four features together with *Max. Frontal*. This is unlike the case in *push-right*, where either of them would suffice for successful prediction. In the *lift* behavior, these together, define the size of the object, and so determine whether the object is graspable or not. The removal of *Shape Vertical* did not have significant effect on accuracy since the number of cases where the object rolled out of view was not high. However *Shape*

Horizontal feature was significant as it tells about the surface opposed by the fingers during grasping.

8.6 Stage 2: Use of Affordances in Task Execution

In this section, we present the methods that enable the use of learned affordances to accomplish tasks which require sequential planning. State space search algorithms are used for this purpose, where the the world state is represented in the perceptual space of the robot. Here, the world state corresponds to the list of feature vectors obtained from the objects in the environment. The initial world state can be represented as follows:

$$[\boldsymbol{f}^{()}_{o_0}, \boldsymbol{f}^{()}_{o_1}, .., \boldsymbol{f}^{()}_{o_m}]$$

where, () denotes the zero length behavior sequence executed on the objects, and m is the maximum number of objects. If the actual number of objects is less than m, the *visibility* features of non-existing objects are set to 0:

$$f^{()}_{o_j}[0] = 0, i \leq m$$

where 0 is the index of visibility feature.

State transition occurs when the robot executes one of its behaviors on an object. Only one object is assumed to be affected at a time during the execution of a single behavior, i.e. only the state of the corresponding object is changed during a state transition. For example, if the robot executes its 3$^{\text{rd}}$, 2$^{\text{nd}}$, 3$^{\text{rd}}$, and 1$^{\text{st}}$ behaviors on 1$^{\text{st}}$, 1$^{\text{st}}$, 2$^{\text{nd}}$, and 1$^{\text{st}}$ objects, respectively, where $m = 3$, the resulting state will be shown as:

$$[\boldsymbol{f}^{(b_3 \rightarrow b_2 \rightarrow b_1)}_{o_1}, \boldsymbol{f}^{(b_3)}_{o_2}, \boldsymbol{f}^{()}_{o_3}]$$

In the previous section, the robot acquires the ability to predict the next state ($\boldsymbol{f}^{(b_i)}$) based on the current state of the object $\boldsymbol{f}^{()}$ using SVM classifiers ($Predictor^{b_i}$) for each behavior (Figure 8.9). Based on this prediction scheme, the robot can estimate the total effect that a sequence of behaviors will create and use this to predict the final object state. Thus, any goal can be encoded in the perceptual state of the robot, and a search can be done through predicting effects of different behavior sequences to reach that goal state.

- **Goals:** The goals are represented by a set of constraints on the object features that are encoded in states. For example, the state that includes an object feature vector with $f_{o_2}^{(\dots)}[5] = [0.75m - 0.85m]$ will roughly satisfy the goal of *move the 2nd object to* $0.8m$ *distance along the frontal axis* where 5 corresponds to the index of the feature that encodes 'minimum distance along frontal axis'. As another example, the goal of *pick-up a particular object* is satisfied in a state, where $\boldsymbol{f}_{o_*}^{(\dots)}[2] = [0.35 - 0.45]$. Here, the 2nd feature corresponds to 'minimum position along vertical axis' and '$*$' corresponds to any object in robot's view, i.e. any object included in the world state description. If the task is to lift the 2nd object with $0.8m$ frontal distance to the robot, both features are required to be satisfied.

  Formally, the constraint set (goal) is composed of (object-index, feature-index, value and range) tuples $CS = \{(o_j, i_f, v, r)\}$. A state satisfies the goal if for all the constraints, the following inequality holds:

$$v - r \leq \boldsymbol{f}_{o_j}^{(\dots)}[i_f] \leq v + r$$

- **Goal Specification:** The straightforward means to set a goal is to manually decide what the constraints (features, objects, values, and ranges) are. In this way, one can encode any goal by manually setting the desired feature value ranges for any object or objects. However, this approach requires full knowledge of the representations of the states and the meaning of all the features. In case of any change in feature space, the goal setting procedure needs to be repeated. Furthermore, hand-tuned goal setting requires programmer intervention each time, making it time-consuming and inconvenient in a world with changing tasks and goals. A more convenient way is to demonstrate an action from which the robot can automatically extract the goal and encode it in its perceptual space. This second approach is used in next section, where the robot self-discovers the goals by observing the desired goal state of the object or objects, and then generates plans based on these.

- **Plan generation:** This refers to finding the behavior sequence required to transform the given state into the goal state. In this study, forward chaining is used to search the state space and find a sequence. Forward chaining uses a tree structure with nodes that hold the perceptual states and that correspond to (behavior-object) pairs. The execution of each behavior on each different object can transfer the state to a different state making

the branching factor of the search tree to be *number of behaviors* × *number of objects*. Starting from the initial state encoded in the root node, the next states for different behavior-object pairs are predicted for each state(Figure 8.9). Note that object features do not change if the behavior is not executed on them, thus only one prediction is performed and one feature vector is predicted in each transition.

In order to reduce the search time, the states with minimal distance to the goal state are expanded first. The distance between states is computed using the features that appear in the constraint set. When a state reached satisfies satisfies the goal constraints, the sequence of behavior-object pairs ($\{< b_i, o_j >\}$) that transfers the initial state to that state is returned as the plan.

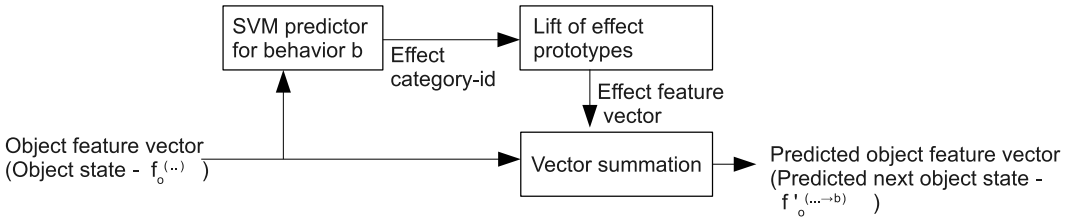


Figure 8.9: Next state prediction using the general affordance relations encoded in: $\{\{Predictor()\}, \{< E_{id}, \boldsymbol{f}_{\text{prototype},id} >\}\}^b$.

8.6.1 Control Architecture

In order to test the proposed method on the real robot platform, a control architecture that supports goal emulation through automatic goal specification was implemented. The robot, infrared range camera, and table were placed similar to the simulated interaction environment. A closed-loop robot control architecture, which can be viewed as a 3-layer hybrid architecture ([104, p. 257]), was used for this purpose (Figure 8.10). The *Perception Module* receives data from the range camera and computes the features of the objects, i.e. the state of the world, as described in Section 8.3.1. The *User Interface Module* is the means of communication with the robot: It shows the range image, the detected objects, and the features of the objects; gives a status report to the user about the plan being executed; and illustrates the search plan tree. Through the *User Interface Module* and the *Set-goal* command, the user can provide the goal environment to the robot, and he can initiate the process of goal-emulation

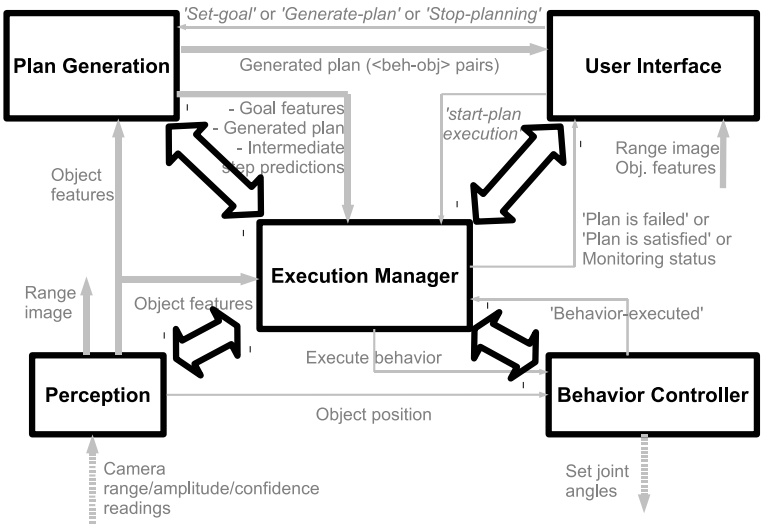


Figure 8.10: Robot control architecture.

in another environment by giving *Generate-plan* and *Start-plan-execution* commands. The predictions on object features that were calculated during the planning process and stored in the nodes of the search tree can be used to assess the difference between the predictions and actual perception of the environment. The *Plan Generation Module* stores the necessary knowledge $(\{\{Predictor()\}, \{< E_{id}, \boldsymbol{f}_{\text{prototype},id} >\}\}^{b_i})$ for making predictions in the perceptual space of the robot. It stores the goal state and starts the plan generation when the *Set-goal* and *Generate-plan* commands are received. Note that both goal state and initial state of planning are provided by the *Perception Module* .

The *Execution Manager Module* is responsible for the ordered execution of behaviors and monitoring of the plan execution. It receives the plan (behavior-object pairs) from the *Plan Generation Module* and when the *Start-plan-execution* command is sent through *User Interface Module*, the behaviors to be executed are sent one-by-one to the *Behavior Controller Module*. At each step, the *Execution Manager Module* checks whether the change in the state is as the one that was predicted in the plan, to decide whether the execution was successful or not. A mismatch detected during the execution of a behavior is reported to the *User Interface Module* causing the execution of the plan to stop. The *Behavior Controller Module* receives the behavior-id to be executed, generates a trajectory of the joint angles based on the specified behavior, object position and current joint angles, and sends it to the low-level controller of the robot arm and hand. This system is tested with several objects at varying positions in

different tasks as shown in the following sections, and used to assess the effectiveness of our approach for real world applications.

8.6.2 Goal Setting through Observation

We introduced *observation* and *imitation* phases to facilitate automatic goal setting. In the *observation* phase, the robot perceives the environment and encodes the goal based on the feature vectors obtained from the environment. In the *imitation* phase, the robot searches a sequence of behaviors that will transform the current state to the observed goal state. "What to imitate" is still an open question in developmental psychology and cognitive robotics [106, 107]. Here we followed a feature-channel-specific goal-emulation mechanism that prioritize some channels over others.

As mentioned earlier, the states are encoded in three different feature channels. We postulated a hierarchy of importance on these features for the agent. According to this, the visibility channel is the most important one since it determines whether an object exists or not. The position-channel represents the object's location (and relation to the robot and the other objects) in the world. Lastly, the shape channel gives information about the contour of the object. The robot first checks whether the object-visibility feature condition is satisfied or not. If not, it only focuses on satisfying the object-visibility condition. If it is already satisfied, then the robot makes a plan to obtain the observed position-related features. If both object-visibility and position-related features satisfy the goal constraints, the the shape-related features are chosen as the goal channel.

8.7 Stage 2: Results

8.7.1 One-Object Imitation

8.7.1.1 Clear the Table Task

The goal of this task was to keep the table clear, hence an empty table was shown to the robot in the *observation* phase. Since no object was perceived, the *object-visibility* feature was automatically set to 0. Later, during the *imitation* phase, different objects were placed on

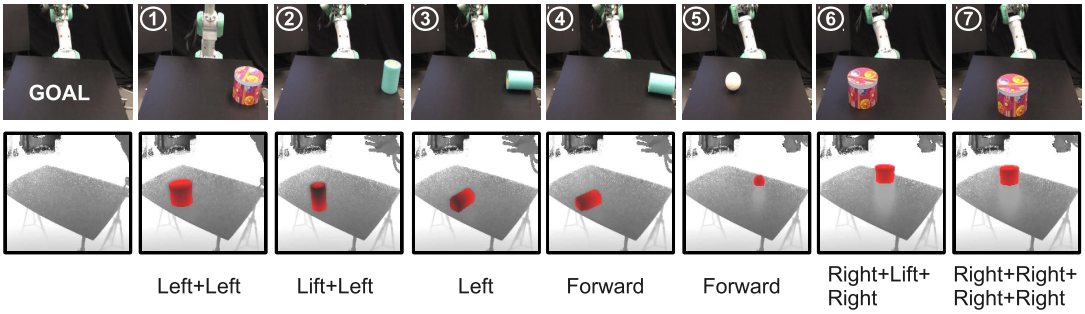


Figure 8.11: Clear the table task. In the *observation* phase, an empty table is shown to the robot and the robot sets the goal *object-visibility* feature to 0. Environment snapshots, range images and generated plans are given in top, middle and bottom rows, respectively.

the table and the robot generated and executed plans to reduce the *object-visibility* to 0.

The snapshots taken from this experiment are shown in Figure 8.11. In (1), the object was pushed and dropped from the left edge of the table using two *push-left* behaviors. In (2), a graspable object was placed at almost the same position and the robot generated a plan with *lift* and *push-left* behaviors. When these behaviors were executed, the robot lifted the object using *lift* behavior and then the object dropped from the hand in the beginning of *push-left* behavior. The object, that landed on the table was pushed from the edge of the table by the *push-left* behavior. In (3), the *push-left* behavior execution was predicted to drop the object from the table, however at the end of the *push-left* the object remained on the table. The plan monitoring module detected the failure and generated a new (correct) plan (4) to roll away the object in this slightly changed configuration. In (5), when a ball was placed on the table, the *push-forward* action was executed to roll it off the table. When a large non-rollable cylinder was placed in (6), a wrong plan was generated since the diameter of the large cylinder was on the decision boundary for liftability (grasp-ability). However, when the object's position was slightly changed, the system was able to make a new plan (7) with four subsequent *push-right* behaviors. This experiment verifies that through interaction the robot had learned the affordances related to physical characteristics and positions of the objects. Additionally, unsuccessful plan executions due to incorrect predictions could be corrected through the self-monitoring mechanism. Note that in order to save space, we did not include the experiments with the unreachable objects where no plan was generated, and box shaped objects that have similar movement characteristics with upright cylinders.

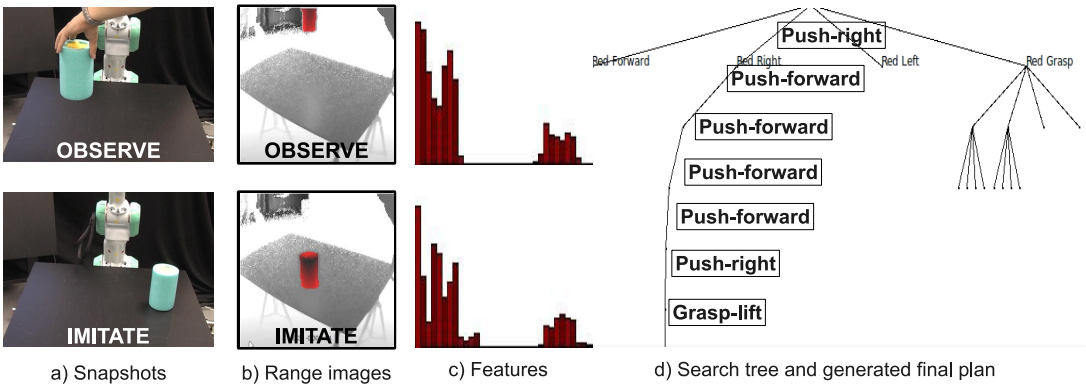


Figure 8.12: Move the object to a target position task. The first panel/column corresponds to the *observation* phase and the next panels correspond to the *imitation* phase steps. The details of the computed features (third panel) are described in the text.

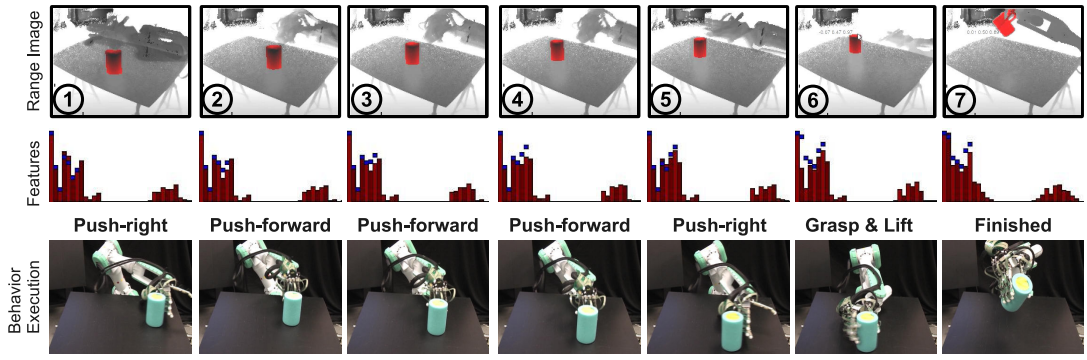


Figure 8.13: The execution steps of a 7-step plan that was generated to bring the object to the observed position in Figure 8.12 (a) top.

8.7.1.2 Move the Object to a Target Position Task

In the *observation* phase, an object lifted in the air was shown to the robot. The *observation* phase and the initial step of the *imitation* phase are shown in the upper and lower panels, respectively in Figure 8.12 (a-c). Visibility, distance and shape features were normalized and their magnitudes are shown by bars in compact form. Due to the priority-based automatic goal setting, the robot sets the goal based on position-related features and generates a plan which could transform the given object features to the observed ones. Figure 8.12 (d) shows the expanded nodes of the search tree, and the found plan.

The snapshots from the execution of the generated plan are shown in Figure 8.13. The top panel shows the initial range images before the execution of the corresponding behaviors. The figures in the middle panel show the feature values computed from the range image. The predictions made for each feature during planning for the visibility and position feature channels are indicated by small blue boxes. The lower panel illustrates the execution of each behavior. In the end, the 7-step plan was successfully executed bringing the object approximately to the goal configuration.

8.7.2 Two-Object Imitation

We can use the learned affordances to make predictions over multiple objects under the assumption that only one object is affected by each behavior execution. For this, not plan generation but the goal setting scheme needs to be modified for tasks involving multiple objects. In the case of two objects, the goal constraint set can be specified either absolutely or relatively. Inspired from goal-emulation in biology, our system sets the goals in accordance with the latter, where relation between objects is important. The robot computes the features of the objects in the *observation* phase, gets the vectorial difference between these features and encodes this difference as the desired goal to be achieved. Setting the goal in this way is also consistent with previous one-object imitation experiments if a second fixed object is assumed to exist (like the table or the robot's body).

The left-most panel of Figure 8.14 shows a goal configuration with two objects. The top and middle feature vectors in the second panel correspond to the robot's perception in this configuration and the bottom vector refers to the goal, computed as the difference between

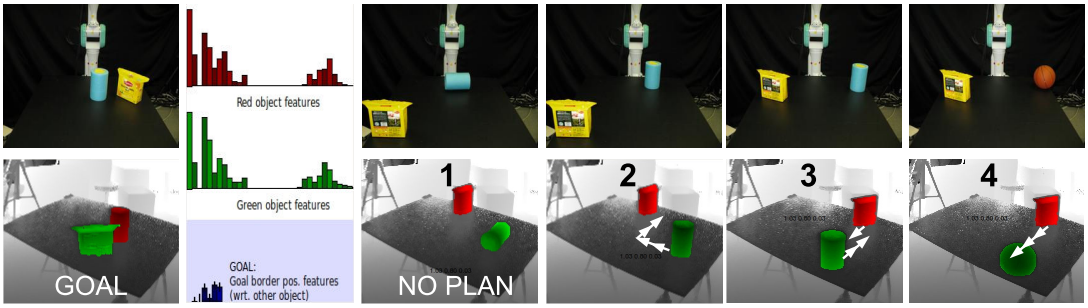


Figure 8.14: Two-object imitation. The left-most panel shows the placement of objects in the *observation* phase. The second panel shows the feature vectors of objects in *observation* phase, and the difference between these vectors encoded as the goal. Right-most 4 panels show the generated plans in different setups to achieve the goal.

position features of the two objects. The right-most four panels show different situations where the robot was expected to generate plans in order to achieve the goal. In situation (1), a lying cylindrical object was placed close to the robot and a box shaped object far away. In order to bring these objects closer, the robot needed to either pull the box towards the cylinder or push the cylinder towards the box. The system correctly predicted that the cylinder rolls away when pushed forward and the box can not be pulled back with the existing behaviors. Hence, no plan was generated. When the orientation of the cylinder was changed in (2), the robot predicted that the cylinder was no longer rollable, and it could be moved towards the box if pushed forward. As a result a 4-step plan was generated with 2 *push-forward* and 2 *push-right* behaviors on the cylinder. In (3), the box was placed closer to the robot, so instead of any *push-forward* behavior, the plan consisted of two *push-right* behaviors for the cylinder and one *push-left* behavior for the box. In (4), when the upright cylinder was replaced by a sphere, i.e. a rollable object, the generated plan only included behavior executions on the box.

8.8 Conclusion

In this chapter, we have shown that through self-interaction and self-obser-vation an anthropomorphic robot and a range camera system can learn the object affordances in an unsupervised way. The proposed learning system share crucial elements such as goal-free exploration and self-observation with infant development. After learning the robot can make plans to achieve desired goals and also emulate end state of demonstrated actions. The plans are based on affordance prediction capability and may involve multiple objects. Furthermore, the system

can monitor the plan execution and take corrective actions using the perceptual structures employed in learning.

In the first step of learning, the robot discovers commonalities in action-effect experiences by finding effect categories caused by its actions. For this purpose, the robot uses a novel hierarchical clustering algorithm that was developed for dealing with non-homogeneous feature spaces. This algorithm, first clusters effects into channel specific categories and then takes their Cartesian product to obtain all-channel effect categories. Predictors for each behavior are then trained to map object features into effect categories using non linear classifiers. Using the category prototypes, the robot can make predictions about the next perceptual state of the object acted upon enabling it to make multi-step plans for achieving goals represented as constraints defined over the object features.

The key aspect of our approach is that, the agent learns about its environment by discovering the effects it can generate through its actions, and forms forward models that enable it to predict the changes in the environment in terms of discrete effect categories as well as low level sensory changes. Predicting the 'change in state' rather then the 'next state' provides better generalization, and, at the same time, allows 'next state' prediction so that multiple steps into the future can be predicted facilitating multi-step planning. Finally, by representing the environment in relation to the effects, our agent 'understands' the world in regards to its own action capabilities, fully adhering to the action based perception view.

8.9 Discussion

In this chapter, the robot learned object affordances using a set of non-parametric behaviors. For example, instead of parameterizing a *push* behavior with approach direction, for simplicity we defined multiple behaviors (i.e. *push-left*, *push-right*, *push-forward*) for different instantiations of the same action. As another example, the robot can only *grasp* the objects from back since the approach direction is always same.

In the Introduction Chapter, we discussed that some action primitives are represented in separate areas in human brain. Although, *push* and *grasp* behaviors can be regarded as such action primitives with different control modules, it is highly improbable that push actions in different directions such as *push-left* and *push* -forward would be encoded in different areas and repre-

sented as different primitives. On the contrary, they must be represented by the same action primitive with a 'direction' parameter. In the Introduction Chapter, we further discussed that infants not only learn the visual properties of the objects but also adjust their reach direction while *grasping* objects. Thus, behaviors must be parameterized and these parameters must be learned during agent's exploration. The next chapter will study this problem, where all behaviors including *push*, *grasp*, and *move-hand* will be parameterized, and these parameters will be included into the problem of learning object affordances.

CHAPTER 9

GOING BEYOND THE PERCEPTION OF AFFORDANCES: LEARNING HOW TO ACTUALIZE THEM THROUGH BEHAVIORAL PARAMETERS

9.1 Introduction

As mentioned in the Introduction Chapter, infants between 7-10 months not only learn *what* type of affordances are offered by the object, but also discover *how* they can actualize them. For instance, they learn not only that a milk bottle is graspable, but also at which angle their hand should approach the bottle to successfully make the grasp. At this period, they demonstrate different modes of grasping such as power-grasp which relies on synergistic control of the hand as a whole, and precision-grasp that requires delicate distal finger control. It is not clear whether the two types of grasps develop from a single rudimentary grasping behavior or develop independently. However it is known that infants in that age do not have the complete adult level visuo-motor grasp execution ability [113], thus the control of grasp behavior develops with the perception of the affordance graspability.

This chapter extends previous chapters by (1) using parametric continuous behaviors instead of discrete ones, and by (2) addressing a more complex behavior, namely grasping. In the previous chapter, the robot learned how to predict the effect of its own actions with the assumption that the behaviors are discrete (i.e. had no free parameters); in this chapter we show how this assumption can be removed. The chapter also addresses learning to grasp in two modes: precision and power grasping. Grasp learning a is complex task [113]; here we adopt a minimalist representation for the grasp actions requiring two parameters, namely the target position and the approach direction. We choose to have the former to be determined by the

grasp (object) location uniquely. Therefore, it is not a free parameter of the grasping behavior once the object location is given. The approach angle, on the other hand, represents the freedom in grasping and used by our learning system to discover the grasp actions that are suitable for the given object.

9.2 Framework Implementation

- **Behavior:** *Behaviors* correspond to parametric pre-defined actions. A behavior is represented as $b_i(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ corresponds to the parameter list. In this chapter, each behavior has one parameter, so $b_i(\alpha)$ notation is used.

- **Entity:** The *entity* corresponds to the feature vector which includes object features and robot's tactile sensor readings. The robot learns affordances by interacting with one object at a time. Thus, entity will be represented as $\boldsymbol{f}^{()}$. where $\boldsymbol{f}$ corresponds to the feature vector and () includes the list of the behaviors executed so far.

- **Effect category:** The robot discovers a variable number of *effect categories* ($E^{b_i}_{id}$) for each behavior b_i during it's interactions. Further, each effect category has a representative effect prototype vector ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$), which is also found by the robot.

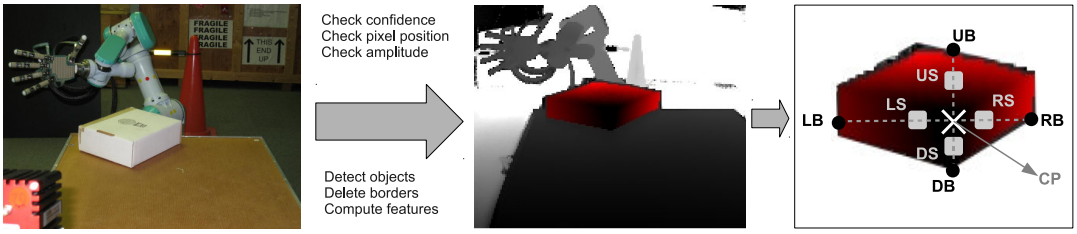
- **Effect:** The robot perceives and represents the change in perception of the entities during its behavior executions. The *effect* feature vector ($\boldsymbol{f}^{b_i}_{\text{effect}}$) represents this change and is used to learn affordances and make predictions.

- **Affordance relation instance:** The affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i(\alpha) >\}$$

9.3 Experimental Setup

The anthropomorphic manipulator which is composed of PA-10 robot arm and Gifu robot hand is used with infrared range camera (Figure 9.1).



a) Photograph of setup b) Range image c) Feature computation

Figure 9.1: In (a), the 23 DOF hand-arm robotic platform, infrared range camera (on the bottom-left) and one object that is used in this study are shown. In (b) the range image obtained from the range camera and the detected object are shown where range is encoded in grayscale and in color for the environment and the object, respectively. (c) The pixels and surface patches that are used in feature computation. The range image is scanned in four different directions starting from Closest Pixel (CP, shown by cross). Four neighbor rectangular surface patches and four border pixels are detected. U (up), D (down), L (left), and R (right) stand for four directions. Thus LS and LB means *left surface patches* and *left border*, respectively. Surface patches in different directions contain fixed number of (5x5=25) pixels at CP's neighborhood.

9.3.1 Perception

9.3.1.1 Object Detection

The first step of pre-processing is to filter out the pixels whose confidence values are below an empirically selected threshold value. The robot's workspace consists of a black table, so region of interest is defined as the volume over the table, and black pixels are filtered out as the range readings from black surfaces are noisy. As a result, the remaining pixels of the range image are belonging to one or more objects. These objects are segmented by the Connected Component Labeling algorithm [63] which differentiates object regions that are spatially separated by a preset threshold value (2 cm in the current implementation). In order to reduce the effect of camera noise, the pixels at the boundary of the object are removed, and median and Gaussian filters with 5×5 window sizes are applied. The detected objects on the range image of a sample setup is shown in Figure 9.1 (b). Finally, a feature vector for each object is computed using the positions obtained from depth values of the corresponding object pixels as detailed in the next paragraph.

9.3.1.2 Object Feature Vector Computation

The perceptual state of the robot is denoted as $[\boldsymbol{f}^{()}_{o_0}, \boldsymbol{f}^{()}_{o_1}..]$ where $\boldsymbol{f}$ is a feature vector of size 25, and the superscript $^{()}$ denotes that no behavior has been executed on the object yet. Six channels of information are gathered and encoded in a feature vector for the object.

Behavior execution on the objects are performed through interaction with objects' CP. Thus, the interaction results are affected by the properties of the CP and its local neighborhood. Thus, a number of pixels and surface patches, related to CP, are detected by scanning the range image in four different directions as shown in Figure 9.1 (c). Then, the following features are computed and included into the feature set:

- The position of CP (3 features).
- The distance of CP to each border pixel (4 features).
- The distance of CP to the center of each surface patch (4 features).
- The mean normal vector for each surface (4×3 = 12 features).
- The visibility of the object (1 binary feature).
- The touch sensor on the hand (1 binary feature).

In this feature vector, the first two channels represent the CP's global properties, in the environment and relative to the object. Third and fourth channels encode information about the CP's local properties. Last two channels correspond to information not related to the CP, i.e. refer to the knowledge regarding the existence of the object in the environment and in the hand, respectively.

9.3.1.3 Effect Feature Vector Computation

For each object, the effect created by a behavior is defined as the difference between its final and initial features:

$$\boldsymbol{f}^{(b_i)}_{\text{effect}} = \boldsymbol{f}^{(b_i)} - \boldsymbol{f}^{()}$$

where $\boldsymbol{f}^{(b_i)}$ represents the final feature vector after the execution of behavior b_i.

9.3.2 Behaviors

How the objects are affected from the execution of the same behavior, depends on the free parameters of these behaviors. For simplicity, each behavior is modulated with one parameter, α. The behaviors and their modulation strategy is as follows:

- **Open-hand(α)**: The robot rotates its wrist in α angle and opens its hand.
- **Move-hand(α)**: The robot moves its hand 10 cm in α *direction*.
- **Push-object(α)**: The robot pushes the object in for 10 cm approaching from α *direction*.
- **Power-grasp(α)**: The hand approaches wide-open from α *direction* to the CP of the object. When palm-touch sensor is activated or the hand reaches the desired position (CP), all the fingers are closed and the hand is lifted.
- **Precision-grasp(α)**: The hand approaches from α *direction* to the CP of the object. Different from power-grasp, only thumb and index fingers are used to make a precision grasp when the tip of these fingers reach CP. The hand is lifted after the fingers are closed.

During object manipulation the robot hand is moved only in horizontal plane above the table, thus *direction* parameter can also be represented by an angle.

9.3.3 Interactions

What type of interactions the robot can perform on the objects depend on the diversity of its behavior repertoire. In this chapter, five different behaviors, that are assumed to be learned in a previous developmental state, are used to manipulate the objects in the environment. These behaviors are triggered with different mechanisms based on the internal and external sensors. We postulate that manipulation behaviors are executed over object's CP to the robot. Thus, if an object is detected on the table, the position of the CP, computed from the range camera, is used to reach to and interact with the object by the behaviors triggered by external sensors. In case there is an object in robot's hand, the robot executes a different set of behaviors to manipulate the object.

Figure 9.2: The execution of power-grasp behavior and the final object range image. The arrow shows the corresponding approach direction (α).

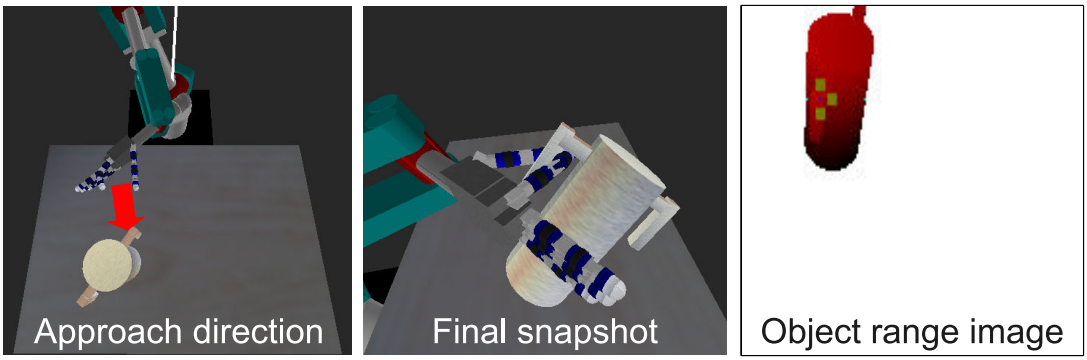


Figure 9.3: The execution of precision-grasp behavior and the final object range image. The arrow shows the corresponding approach direction (α).

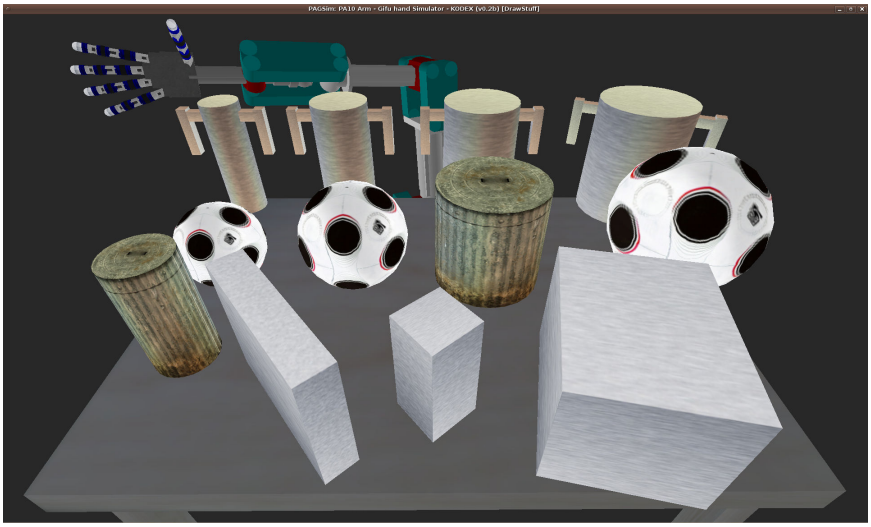


Figure 9.4: Sample objects that are used in learning. Note that the size and orientation of objects are randomly set.

9.3.4 Objects

The robot interacts with four types of objects; namely boxes, cylinders, spheres, and objects with handles, all in different size and orientations. As shown in Figure 9.4, only the handle dimension is kept fixed. During the execution of its behaviors with different parameters, the robot may experience interactions with objects and face with different consequences. For instance when the hand pushes boxes or upright cylinders in the middle of the table, the objects will remain on the table, but if it pushes spheres the objects will roll down the table. As another example, the same box can be grasped from one approach direction while cannot be grasped from other directions. Note that in order to avoid robot arm - camera collision, the camera is placed on the other side of the table. On the other hand, the robot interacts with closest point of the object and closest point is generally out of view of the camera. Thus, only symmetric objects, which provide mirrored but same information from robot and camera views, are used in experiments.

9.4 Learning of Affordance Relations

The exploration phase, conducted only in simulation, consists of episodes, where the robot interacts with the objects, and monitors the changes. The data from an interaction is recorded in the form of $< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i(\alpha) >$ tuples, i.e. (effect, (object,tactile), behavior) instances.

Here, α is the parameter of the behavior b_i used for interaction, $f^{()}$ and $f^{b_i}_{\text{effect}}$ denote the initial feature vector and the difference between final and initial feature vectors, respectively.

The learning process consists of two steps: the unsupervised discovery of effect categories, and the training of classifiers to predict the effect categories from object features. The learning process is applied separately for each behavior as detailed below.

Effect Category Discovery Effect categories are discovered by clustering the effects using the channel-based hierarchical clustering algorithm described in Chapter 8. In the lower level, channel-specific effect categories are found by clustering in the space of each channel, discovering separate categories for visibility, position and shape. In the upper level, the channel-specific effect categories are combined to obtain all-channel effect categories using the Cartesian product operation. In both levels, if the number of samples of any effect category is lower than a threshold, the corresponding effect category is discarded. After discovering the effect categories and assigning each feature vector in the set of $\{\boldsymbol{f}^{b_i}_{\text{effect}}\}$ to one of the effect categories ($E_{b_i,id}$), the prototype effect vectors ($\boldsymbol{f}^{b_i}_{\text{prototype},id}$) are computed as the average of the category members.

Learning effect category prediction In the second step, classifiers are trained to predict the effect category for a given object feature vector, a behavior id and behavior's parameter by learning the $(\boldsymbol{f}^{()}, \alpha) \rightarrow E_{b_i,id}$ mapping. Specifically, we used a Support Vector Machine (SVM) classifier with Radial Basis Function (RBF) kernel to learn this mapping for each behavior b_i, where $(\boldsymbol{f}^{()}, \alpha)$ is given as the input, and the corresponding $E_{b_i,id}$ as the target category.

9.5 Behavior Parameter Selection for Goal-Oriented Affordance Use

The trained SVM classifiers allow the robot to predict the *effect category* that is expected to be generated on an *object* by a *behavior* controlled with a particular *parameter*:

$$E^{\text{predicted}}_{b_i,id} = Predictor^{b_i}(\boldsymbol{f}^{()}, \alpha).$$

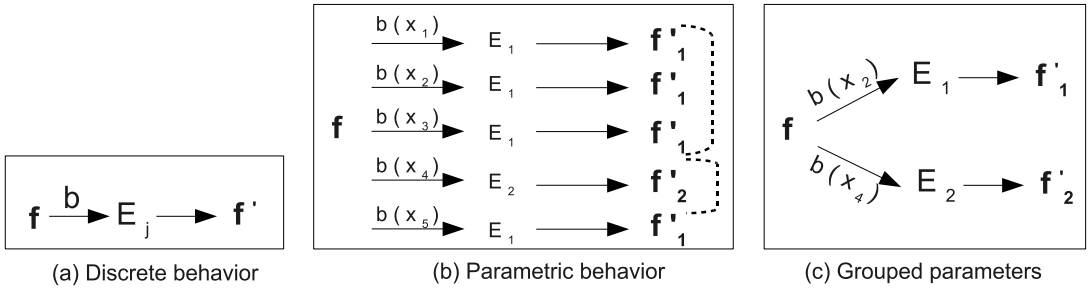


Figure 9.5: Behavior parameter selection to predict possible next object states. Given object features ($\boldsymbol{f}$) and behavior-id (b), the effect category (E_j) and the next state ($\boldsymbol{f'}$) can be predicted by using the corresponding *Predictor()* and prototype features. (a) and (b) shows next state prediction using discrete and parametric behaviors, respectively. A grouping and averaging mechanisms is used to choose the most reliable behavior parameters that transform the current object perceptual state to one of the possible states the corresponding behavior can transform.

The predicted percept of the object can be found as:

$$\boldsymbol{f'}^{(b_i(\alpha))} = FM^{b_i}(\boldsymbol{f}^{()}, \alpha) = \boldsymbol{f}^{()} + \boldsymbol{f}^{b_i}_{\text{prototype},id^{\text{predicted}}}$$

Effectively, this corresponds to a forward model (FM) that returns the next perceptual state of the object. By successively applying this model, the robot can predict the perceptual state of the object for any number of sequentially executed behaviors.

Predicting the next state of the object for any discrete behavior is straightforward since given initial object features, the SVM classifier will predict only one effect category and FM will give only one next perceptual state as shown in Figure 9.5.

On the other hand, one non-discrete behavior can create many different effects on the same object when controlled with different parameters. Next state predictions also depend on the behavior parameter since it is an input to *Predictor*(), thus different next state predictions can be obtained when whole parameter space of the behavior is considered as shown in Figure 9.5 (b). Still, the number of effect categories is fixed for each behavior and the possible next states are limited with this number. As a result, the problem can be transformed to 'finding the most reliable behavior parameter to reach a possible next state'. For this purpose, (1) a grid search is done in continuous parameter space; (2) behaviors which transform the current state to the same state are grouped together; (3) the largest group for each next different state is found; and (4) the mean parameter value in each group is selected as the best parameter

that transforms the current state to the corresponding next state. Figure 9.5 (c) illustrates this method in a simple example.

9.6 Experiments

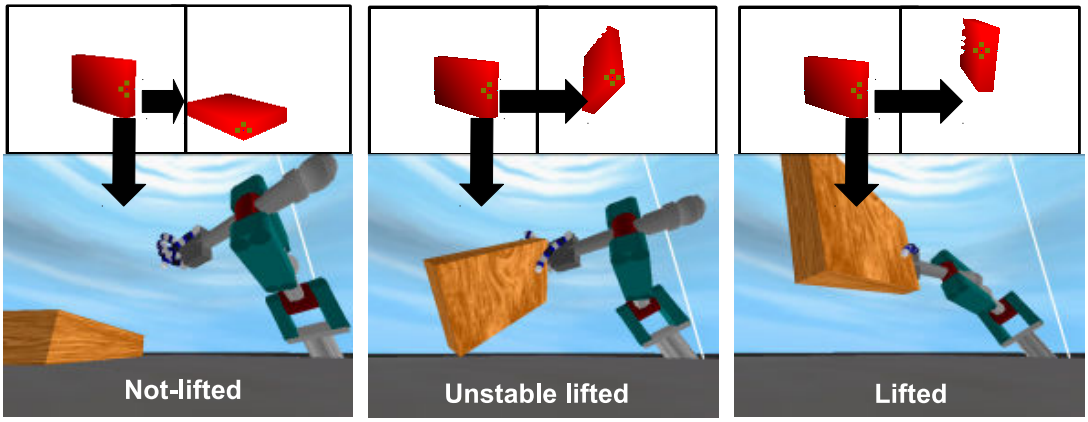
In the experiments, a table with 100×70 cm^2 surface area was placed with a distance of 40 cm in front of the robot, as shown in Figure 9.1. At the beginning of each exploration trial, one random object of random size [$8cm-40cm$] was placed on the table at random orientation. For all behaviors, 2000 interactions were simulated with random parameters and the resulting set of relation instances were used in learning. The X-means algorithm was used to find channel-specific effect categories[1], and Support Vector Machine (SVM) classifiers were employed to learn effect category prediction.

9.6.1 Discovered Effect Categories for Grasp Behaviors

For power-grasp behavior, two channel specific clusters are formed in each of the visibility, position and touch channel. In the upper level of the clustering, after cross-product operation, 4 clusters are found to represent whole effect space as shown in Table 9.1. Large objects could not be lifted and remained on the table resulting in *not-lifted effect*. Small objects could be lifted and as a result of this lifting, the height is increased and touch sensor is activated as shown in prototype of *lifted effect*. In some cases, the grasp is not stable, so the object slides from robot's hand during lifting but remains in contact with the hand, creating *unstable-lifted effect*. In this effect, the vertical position of the object is not increased (significantly), however the touch sensor remains activated. These cases are shown in Figure 9.6. The last effect, which is labeled as *disappeared* is created by spheres, that roll away during interaction with the hand.

For *precision-grasp* behavior, two channel specific clusters are formed in visibility and touch channels and 3 effect categories are obtained in the end of hierarchical clustering as shown in Table 9.2. Because the robot inserts one of its fingers through the aperture of the handle, the grasps are more stable once the object is hold. Thus an *unstable-lift* category was not formed as in *power-grasp*.

[1] X-means implementation in Weka data mining software is used [94].



(a) $\alpha = -35^\circ$ (b) $\alpha = -15^\circ$ (c) $\alpha = 25^\circ$

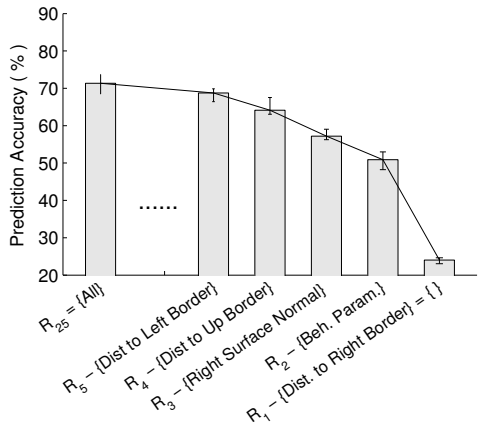
Figure 9.6: The interaction results for 3 different samples grasping cases are shown. Object angle is always kept as -45° but the approach angle α is changed.

Table 9.1: Effect category prototypes discovered for *power-grasp*. Only significant changes are given in the table. The comments are provided for the effect prototypes and are not used during experiments.

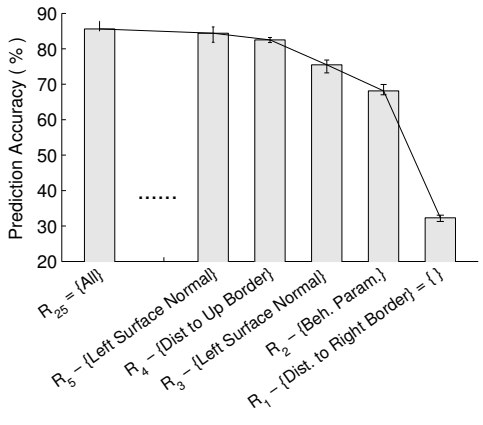
Effect id	**Visibility**	**Position (x,y,z)**	**Touch**	**Comment**
Effect 1	0	+3cm,+2cm,+2cm	0	Not-lifted
Effect 2	0	+3cm,**+13cm**,+3cm	**+1**	Lifted
Effect 3	0	+3cm,+2cm,+2cm	**+1**	Unstable lifted
Effect 4	**-1**	+3cm,+2cm,+2cm	0	Disappeared

Table 9.2: Effect category prototypes discovered for *precision-grasp*. Only significant changes are given in the table. The comments are provided for the effect prototypes.

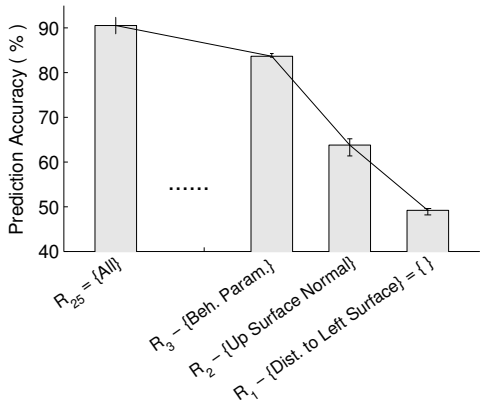
Effect id	**Visibility**	**Position (x,y,z)**	**Touch**	**Comment**
Effect 1	0	+6cm,-1cm,+4cm	0	Not-lifted
Effect 2	0	+5cm,**+10cm**,+2cm	**+1**	Lifted
Effect 3	**-1**	+6cm,-1cm,+4cm	0	Disappeared



(a) Power-grasp behavior (4 categories)



(b) Power-grasp behavior (3 categories)



(c) Precision-grasp behavior (3 categories)

Figure 9.7: The prediction accuracies of the classifiers that are computed using different feature sets. The feature set is increased by one feature in each iteration by adding the most successful one. The left-most and right-most bars in each plot show the results obtained using no and all features, respectively. Error bars on prediction accuracies indicate the best, median, and worst classifiers found by 10-fold cross-validation. The prediction accuracy did not drop significantly when the least irrelevant features were discarded from the training set. Thus their prediction accuracies are not shown, and they are represented with '.....'.

9.6.2 Effect Prediction in Power Grasp Behavior

After the discovery of effect categories, the mapping from the initial object features to these categories is learned for each behavior b_i ($Predictor^{b_i}()$) by multi-class Support Vector Machines (SVMs). The LibSVM [19] software package was used with optimized parameters of the RBF kernel through cross-validated grid-search in parameter space. 1000 simulated interactions were used in training and a separate set of simulated 1000 interactions were used for testing. At the end, 72% accuracy was obtained in predicting the correct effect categories for *power-grasp* behavior. The low accuracy is due to the difficulty in predicting *unstable-lifted* effect category since it corresponds to the critical point between success and failure in liftability. When this category is discarded from the sample set and a similar training procedure is applied, 85% accuracy is obtained in predicting the three categories.

We analyzed the relevance of the features in affordance prediction for the *power-grasp* and *precision-grasp*. For this purpose, we used Schemata Search [102] which computes the relevance of a feature based on its impact on the prediction accuracy. The Schemata Search is a greedy iterative method that starts with the full feature set (R_0), and shrinks it by removing the most least feature in each iteration. At each iteration (t), candidate subsets are formed by deleting a different feature from R_{t-1} (remaining feature set of previous iteration), and they are evaluated by training SVM classifiers in 10-fold cross-validation. The subset with the highest mean prediction accuracy is chosen as R_t and transfered to the next iteration.

Figure 9.7 (a) and (b) gives the prediction accuracy results with different feature sets, with and without *unstable-lifted* effect. When the feature relevance is examined, behavior parameter (α) is among the most relevant features as presented. The other relevant features represent CP's object-relative properties and CP's local surface angles. For example, *distance to right border* and *distance to left border* encodes the location of CP with respect to object and left/right surface normals represent the shape of the CP's local neighborhood.

We systematically analyzed the success in effect prediction by comparing real and predicted effect categories using a fixed size box which is graspable from one side and not graspable from the other side. It is rotated in 10° intervals and in each object orientation, *power-grasp* behaviors are executed with varying approach direction angles from $-70°$ to $40°$. Other approach direction angles are not considered since the object is not reachable with those angles.

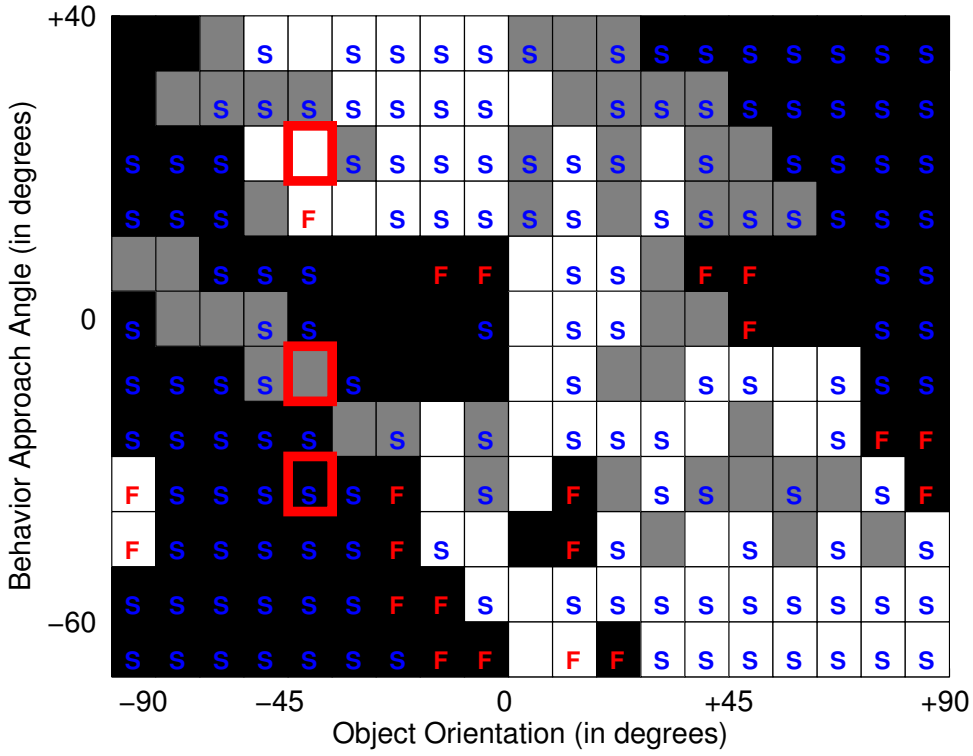


Figure 9.8: The comparison of real and predicted effect categories for different object orientations and power-grasp approach directions. The color of the regions corresponds to observed real effect categories; black: *Not-lifted effect*, white: *Lifted effect*, and gray: *Unstable lifted effect*. The 'S' and 'F' labels corresponds to prediction success and failure. If the prediction or real effect category is *unstable-lifted*, then the corresponding box is not labeled. The cases marked with bold red boxes are shown in Figure 9.6.

The real effect categories obtained during these interactions are illustrated with different colors in Figure 9.8. As shown in the figure, the relation between object-angle and behavior-approach-angle, which determines the liftability of the objects, is non-trivial as the robotic hand is not a simple gripper, but has high degree of freedom. There is hardly any symmetry or linear relation between these two components (e.g. while objects at 60° orientation are lifted by *power-grasp(*$-20°$*)*, objects at $-60°$ cannot be lifted by *power-grasp(*$20°$*)*. Furthermore, there are many 'gray' regions which correspond to *unstable-lifted effect* that are distributed between *lifted* and *dragged* regions. This is a consequence of the robot hand kinematics, which is different from a gripper. Our method was able to predict many effect categories correctly, however failed to predict some that reside in critical border.

9.6.3 Effect Categories and Learning Results for Other Behaviors

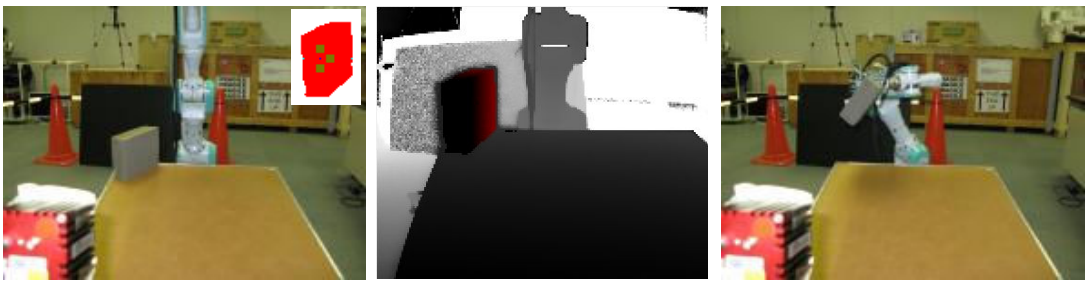
Push behavior created four different effect categories. A rollable object can be pushed out view; a non-rollable object's position and orientation can be changed in different amounts depending on the orientation and dimensions of the object, and the push direction. The prediction accuracy is 92% and the 3 most relevant features consist of the behavior parameter (α) and 2 components of normal vectors from left and right surfaces. As for *move-hand* behavior, 4 different categories were formed based on CP position change. The mean accuracy is 88% by only using the α parameter. Finally, 5 different categories were discovered in *release* behavior, since the shape features also change and contribute to the effect category formation.

9.6.4 Real Robot Results

The results obtained in the simulator were partially verified on the real robot platform. For this purpose, the effect category prediction system is transferred to the real robot. A box shaped object and an object with a handle are used to assess the ability in prediction of *lift effect* with *power-grasp* and *precision-grasp* behaviors, respectively.

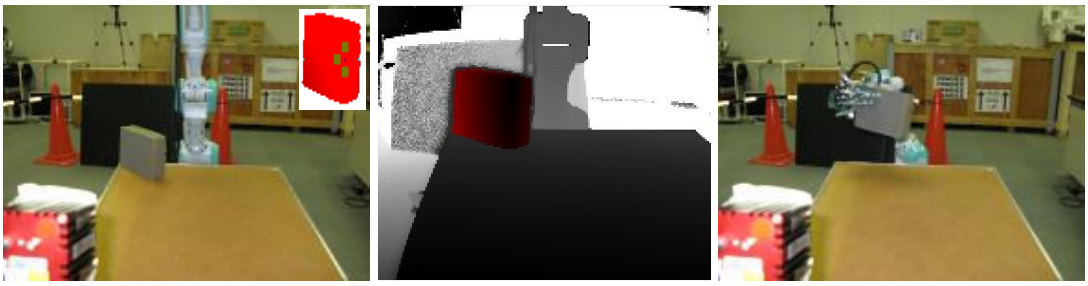
The box shaped object is placed in two different orientations as shown in Figure 9.9 and 9.10. The behaviors that are predicted to lift the objects from their narrow side are also parameterized with different angles.

The watering can is placed in two different orientations: in the first orientation, the CP is on



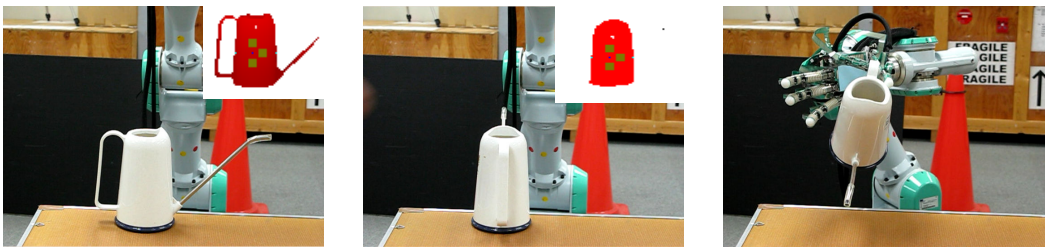
(a) Power-grasp [-30,40] (b) Range Image (c) Result of grasp

Figure 9.9: The object was grasped with an approach angle of 5°.



(a) Power-grasp [-70,20] (b) Range Image (c) Result of grasp

Figure 9.10: The object was grasped with an approach angle of −25°.



(a) No precision grasp (b) Precision grasp [-90,+90] (c) Result of precision grasp

Figure 9.11: The object is correctly predicted not to be liftable in (a). The same object when rotated is predicted to become liftable with any *precision-grasp* behavior. Thus, it is approached with 0° and lifted up.

its main body and in the other one, the CP is on the handle (Figure 9.11). The robot computes the features based on CP, so the results are different. In (a), no precision grasp is predicted to lift the object, where in (b) precision grasps from all directions are predicted to lift the object since the handle is reachable from all directions. When the average of these directions are used as the final parameter, the object is approached from behind and lifted up.

9.7 Conclusion

In this chapter, we proposed a method that allows a robot not only to discover *what* type of affordances are offered by the objects but also to learn *how* to actualize them. After robot's exploration, the effect of behavior parameters over discovered affordances are learned in relation with the object features and the generated effects. This learning enables the robot to predict the objects' next perceptual state based on the current object features and the behavior parameters. This prediction ability is used to satisfy particular goals, i.e. to reach desired final states. However, given an object and a behavior, many effects can be created by the same behavior depending on the parameters used. Thus, we proposed and tested a method to select the behavior parameters to reach desired goals.

The execution of *power* and *precision-grasp* behaviors on objects in different size and orientations (and with/without handles) had non-trivial dynamics when the dexterous robot hand was used in interactions. Still, our system could learn the affordances provided by these objects and could act upon the provided affordances by correctly parameterizing the grasp behaviors. For this purpose, we used an improved perceptual processing procedure that encodes the

affordance related properties around behavior contact point of the object.

The work presented in this chapter differs from the studies (e.g. [38]) that learns the grasp points on the objects in a number of ways. First, unlike these approaches, our method does not require the supervised labeling of its interactions (in these studies, typically these labels were automatically generated by hand-coded monitors that categorized the result of the execution as successful or not), and is completely unsupervised. Second, the proposed method is able to not only predict the type of effect that will be generated by a behavior for a certain type of parameter value, but also the change to be generated on the object as a result of execution. This property allows us to use these relations for making multi-step plans. Third, the method proposes a set of novel feature descriptors that work on the range images, rather than visual images or sparse depth information extracted from them.

9.8 Discussion

In this chapter, similar to previous chapters, the robot used an existing behavior repertoire that was assumed to be learned in previous developmental phases to discover affordances. In the next chapter, we will relax this last assumption, and propose a method that enables the robot to discover behavior primitives from one basic action and one basic reflex, using a crude tactile and visual perception.

CHAPTER 10

EMERGENCE OF BEHAVIOR PRIMITIVES FROM ONE BEHAVIOR

10.1 Introduction

Throughout this thesis, we assumed the existence of a behavior repertoire that was learned in a previous developmental phase. Thus, we designed supposedly learned behaviors using our intuition and understanding of natural systems in a pragmatic means so that the learned affordance prediction abilities based on those behaviors can be used in a goal-oriented way. For example while *push* behavior was simply 'reaching to the object center', grasp behavior was defined as 'reach to the object, close fingers, and lift the hand', as a combination three simple actions. Each behavior was encoded in different complexity and representation to satisfy designer's requirements.

Infants between 7-10 months have also acquired a set of behaviors that are qualitatively different and that can be used for different motivations such as grasping, dropping, reaching, shaking, etc. These actions can be considered as behavior primitives that are utilized to develop more advanced skills through exercise. There is evidence that some behavior primitives are represented as different modules in human's mind. For example, the 'transport' and 'grasp' primitives during reaching and holding the objects appear to be controlled by different regions of the human brain [124, p. 217]. Furthermore, there is a developmental order in maturation order of these areas. Thus, it's plausible that the infant starts from a small number of reflex like behaviors, and then progressively discovers and distinguishes new behaviors through use of old ones. Additionally, infants should be using nothing but their own perception and monitoring skills to differentiate and distinguish the behavior primitives. Lastly, a crude perception

Figure 10.1: The grasp reflex. A snapshot from an experiment by John B. Watson in 1919.

system should be employed to distinguish the basic behaviors, and a more advanced perception should be utilized to differentiate more complex behaviors and to discover more abstract concepts such as affordance relations.

This chapter focuses on design of such a robotic system which starts with only one basic reflex-like behavior (*swing-hand behavior*) and one basic reflex (*palmar-grasp reflex*). By exercising this behavior with different hand speeds (which corresponds to maturation of 5-month-old infants), and by using its crude tactile and vision sensors, the robot is able to distinguish a number of meaningful behavior primitives. In the next developmental step (7-10 months), one of these behaviors, namely grasp behavior, is exercised more on different objects and is monitored using a more complex visual and tactile perceptual system. The visual perceptual system is inspired from CIP neurons on the dorsal pathway of monkeys. CIP extracts visual data and forwards it to AIP area which is responsible from perception of graspability affordance [103, 114]. Similar to *surface orientation selective* neurons of CIP area [127, 150], the robot's visual perceptual representation includes direction and low-level curvature information of object surfaces.

In this chapter, progressively more complex action possibilities and affordances are discovered by the robot in three phases:

- In **Phase I**, referred as *touch-based behavior discovery phase*, a number of behavior primitives are discovered through infants most basic sense, touch.

- In **Phase II**, referred as *vision-based behavior discovery phase*, higher order behaviors are discovered by including a limited visual perception.

- In **Phase III**, referred as *affordance learning phase*, more complex touch and visual perception is employed to learn affordances using the discovered behavior primitives.

10.2 Framework Implementation

- **Behavior:** Initially, the robot is equipped with one behavior. In *behavior discovery phases*, through exercise of this behavior, it discovers a number of behavior primitives. Each behavior primitive is represented as $b_i(\boldsymbol{\alpha})$ where i corresponds to the index of the primitive and $\boldsymbol{\alpha}$ corresponds to parameter list. Since the behaviors are discovered from the *swing-hand* action by using the same method, they have a common encoding. In other words, each behavior primitive is encoded with a common set of parameters that are automatically instantiated during behavior discovery. The parameters are:

  - initial and final touch states,
  - initial and final hand velocities,
  - whether *grasp-reflex* is disabled, and
  - hand movement direction (towards object or towards robot).

- **Entity:** The *entity* corresponds to the feature vector which includes object features and robot's tactile sensor readings. The robot learns affordances by interacting with one object at a time. Thus, entity will be represented as $\boldsymbol{f}^{()}$. where $\boldsymbol{f}$ corresponds to the feature vector and () includes the list of the behaviors executed so far.

- **Effect category:** The robot discovers a variable number of *effect categories* ($E_{id}^{b_i}$) for each behavior b_i during it's interactions. An improved version of the hierarchical unsupervised categorization method is proposed for this purpose. Further, each effect category has a representative effect prototype vector ($\boldsymbol{f}_{\text{prototype},id}^{b_i}$), which is also found by the robot.

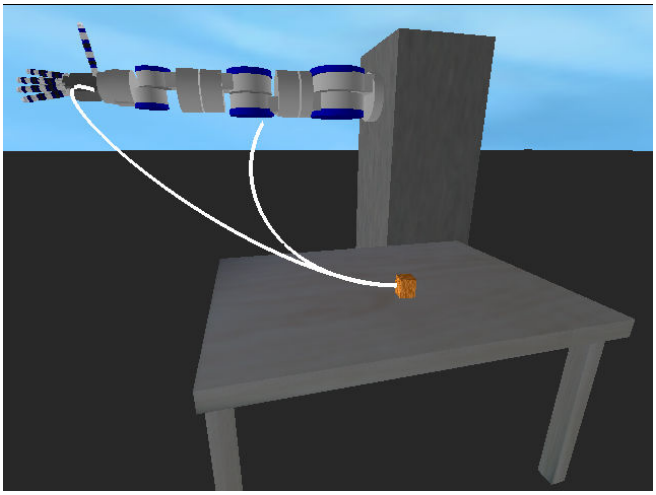


Figure 10.2: The trajectory of the basic *swing-hand* behavior.

- **Effect:** The robot perceives and represents the change in perception of the entities during its behavior executions. The *effect* feature vector ($\boldsymbol{f}^{b_i}_{\text{effect}}$) represents this change and is used to learn affordances and make predictions.

- **Affordance relation instance:** The affordance relation instance, which represents a sample interaction with the environment, will be represented as follows:

$$\{< \boldsymbol{f}^{b_i}_{\text{effect}}, \boldsymbol{f}^{()}, b_i(\alpha) >\}$$

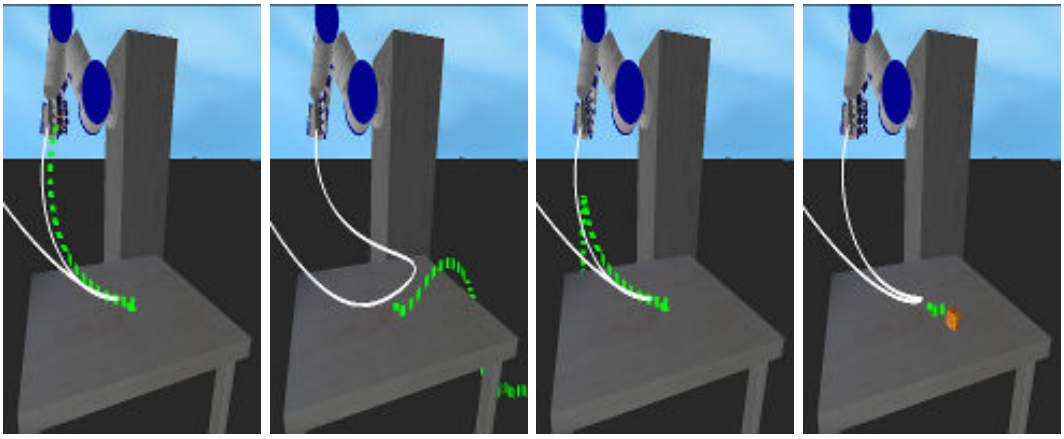
10.3 Experimental Setup

The anthropomorphic manipulator which is composed of Motoman robot arm and Gifu robot hand is used with infrared range camera.

10.3.1 Behaviors

"Even in the newborn infant, a basic neuro-muscular infrastructure for reaching and grasping is present. When an object is placed in the palm of a newborn infant, the tactile stimulation triggers a grasp reaction in which all digits are flexed around the object. Similarly, in newborns, reaching movements aimed towards objects within the center of the visual field are present. "[149, p. 235]

Swing-hand behavior: The robot is assumed to have the ability to reach and bring the objects



(a) Grasp (b) Hit (c) Drop (d) Tap

Figure 10.3: Robot-hand and object trajectories during *swing-hand* behavior with different velocities and hand states. The labels that explain situations are only given to ease the understanding of the figures and are not used in any phase of development. Corresponding sensor trajectories and behavior primitive segments are provided in Figure 10.4.

to it. *Swing-hand* behavior is used for this purpose where the robot reaches to the object and pulls back its hand. This behavior is implemented to generate a trajectory as in Figure 10.2. The hand that is either clenched or wide open prior to the behavior execution can reach to the object in different velocities. Due to a built-in *grasp-reflex*, if the robot feels anything in its palm using touch sensors, the hand is closed. Furthermore, at any moment, this reflex can be disabled randomly and in this case the robot hand is loosened even if there is an object inside.

The execution of the same *swing-hand* behavior over the same object with different parameters (velocity, disable-reflex, and initial hand state) produces different effects. Figure 10.3 shows hand and object trajectories during execution of the same behavior with four different parameter sets. In (a), the hand hit the object with a velocity of $V = 0.24$ cm/s, and due to the *grasp-reflex* the object was grasped and brought back. In (b), the high-velocity ($V = 0.42$ cm/s) collision between the hand and the object didn't allow the object to be grasped on time. In (c), the hitting/reaching velocity of the hand was same with (a), so the object was grasped. However, while pulling back the hand, the *grasp-reflex* was disabled (randomly) so the object was released. Finally, the hand was initially clenched in (d), so the object was tapped only.

How the objects are affected from the execution of the same behavior, depends on the free parameters of these behaviors. While *Swing-hand* behavior has one parameter (hand speed), the discovered behavior primitives may have more than one parameter.

10.3.2 Interactions

In **Phases I** and **II** (*behavior discovery phases*) the robot executes the *swing-hand* behavior on the same small object placed in a reachable position. Different hand speeds and grasp-reflex disabling timings generate different effects on the object and on the touch sensor. Based on these differences, the robot discovers a number of new behavior primitives.

In **Phase III** (*affordance learning phase*) the robot executes the discovered behavior primitives on a relatively diverse object set: boxes, cylinders and spheres of different size and orientation. The mechanism that is used to discover and distinguish behavior primitives and encode them in the same representation automatically imposes constraints in the execution of those primitives. If the sensors prior to behavior execution satisfies those automatically found constraints (encoded parameters), the corresponding behavior primitive can be executed. For example, the robot will distinguish *carry-object* and *move-empty-hand*, based on the initial (and final) touch senses. In other words, *carry-object* behavior has *initial-touch-sensor=ON* in its encoding and *move-empty-hand* has *initial-touch-sensor=OFF*. Based on this encoding, the robot can execute *carry-object* behavior only if the touch sensor is already *ON*. During the execution of each behavior, the robot observes the consequences of its actions.

10.3.3 Perception

10.3.3.1 Touch Perception

The sensory readings obtained from the distributed tactile sensors that cover the palm and fingers are processed as follows.

First, touch for each finger link and for the palm is detected and encoded as $5 \times 3 + 1 = 16$ binary touch signals. Then, their sum is normalized between $[0 - 1]$ to represent a general touch sensation for the hand, and called as *raw* touch signal. Since the readings are noisy, the *raw* touch trajectory is convolved with the first half of Gaussian window, $G(N = 50, \sigma = 0.5)$:

$$y_i = x_i \times e^{\frac{(i-M)^2}{2\sigma N}}$$

where $M = (N - 1)/2$. *Convolved* readings are used in perception during *affordance-learning phase*, **Phase III**.

On the other hand, in the **Phase I** and **II**, the robot has a crude sense of touch as mentioned before. This limited sense is represented by three touch states, which are computed from binary touch signal, to ease the representation of this change. In order to compute the binary signal, the *convolved* trajectory is discretized to *on/off* sensor using a threshold $t = 0.02$. Figure 10.4 shows a number of sample trajectories where the *raw* values are *convolved* and *binarized*.

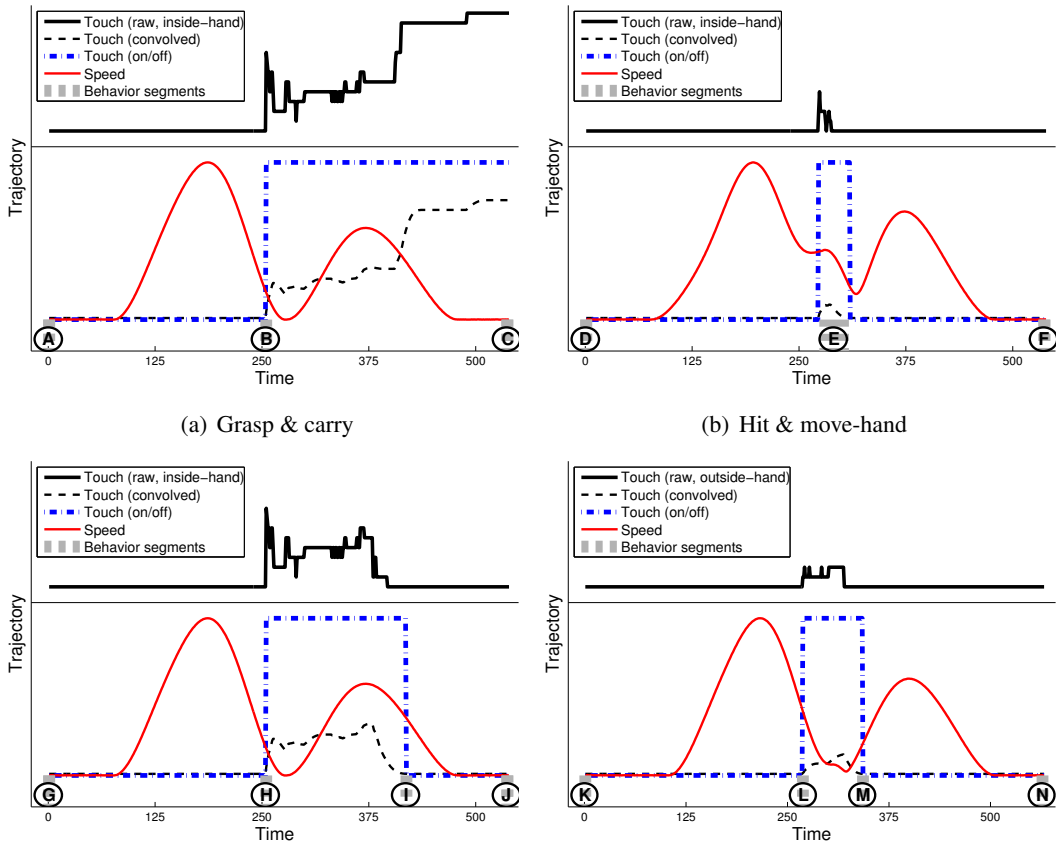
Touch States are defined based on the binary touch signal trajectory as follows:

- **on:** The touch sensor that is active during behavior execution.
- **off:** The touch sensor that is not active during behavior execution.
- **onf:** In some cases, the touch sensor is active for a short duration. Such cases are denoted by **on/off** or **onf** in short. There is such a case in Figure 10.4(b), change (E), where the robot hand hits the object with high velocity. A similar case in Figure 10.4(d) occurs, however since the reach velocity is lower, the object is dragged on the table rather then been hit. So the duration of touch is long in this case and hand state is represented by the consecutive **on** and **off** states instead of the **onf** state.

10.3.3.2 Visual Perception

Object detection: As in previous chapter, the confident pixels that are inside robot's workspace are first found. Then, Connected Component Labeling algorithm [63] is used to differentiate the objects. In order to reduce the effect of camera noise, boundary object pixels are discarded and median and Gaussian filters with 5×5 window sizes are applied. Finally, a feature vector for each object is computed using the positions obtained from the depth values of the corresponding object pixels as detailed in the next paragraph.

Object features: The object feature vector includes a binary feature for *object-visibility*, and a number of features related to location and shape of the object. The 3D position of the object center is used to represent location of the object. As for the shape features, inspired from CIP neurons in monkey brain, first, object surfaces are identified, then size, orientation, and curvature of each surface is computed. Object surfaces are found by grouping object



(a) Grasp & carry

(b) Hit & move-hand

(c) Grasp & release & move-hand

(d) Punch & drag & move-hand

Figure 10.4: Velocity, touch sensor trajectories and segmentation of *swing-hand* behavior with different parameters.

pixels with similar local orientations. For this purpose, for each detected object pixel a local normal vector is computed (as in Chapter 8) and a clustering algorithm is used to find 'normal vector clusters' that correspond to object surfaces. See Algorithm 6 for the details of this algorithm and Figure 10.5 for sample surfaces. After finding surfaces, for each cluster (surface), standard deviation of the normal vectors is computed in each coordinate axis to represent curvature. Additionally, for each surface, height and width are computed. At the end, the following feature vector represents the object shape perception:

$$\boldsymbol{S} = (w_1, h_1, \boldsymbol{\mu}_1, \boldsymbol{\sigma}_1, ...w_s, h_s, \boldsymbol{\mu}_s, \boldsymbol{\sigma}_s)$$

where w_1 and h_1 correspond to width and height of the first detected surface, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ refer to mean and standard deviation vectors (of size 3), respectively. s is the maximum number of surfaces, which is set to 3. Therefore, the object feature vector includes *object-visibility*, position and shape features, and has size of $1 + 3 + (1 + 1 + 3 + 3) \times 3 = 28$.

Algorithm 6 Surface identification through normal vector clustering

s_id = Index of the surface (cluster).

s_{normal} = Normal vector of the corresponding surface (the mean of the cluster).

$\{\boldsymbol{N}\}$: The set of normal vectors computed for each pixel.

$cluster(A, k)$: Cluster sample set A into k clusters. Return the list of cluster index and means.

$dist(s_i, s_j)$: Distance between normal vectors of surfaces s_i and s_j.

$threshold_{dist}$: A threshold to decide the similarity of surfaces.

$nSameSurfaceNeighbors(p, s_id)$: Number of pixels with same s_{id} in 8-neighborhood of p

1: $(\{s_{id}\}, \{s_{normal}\}) = cluster(\boldsymbol{N}, 3)$
2: **for** each surface pair $< s_i, s_j >$ in surface list $\{s_{id}\}$ **do**
3: **if** $dist(s_i, s_j) < threshold_{dist}$ **then**
4: Combine s_i and s_j in $\{s_{id}\}$
5: **end if**
6: **end for**
7: **for** each surface s_i in surface list $\{s_{id}\}$ **do**
8: **for** each pixel p in surface s_i **do**
9: Delete p from s_i if $nSameSurfaceNeighbors(p, s_i) < 3$
10: **end for**
11: **end for**

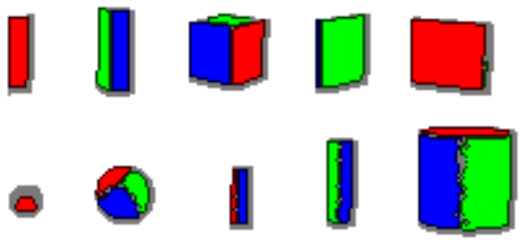


Figure 10.5: Identified surfaces through normal vector clustering. Upper row includes box objects and lower row is composed of spheres and cylinders. Each color represents a surface. Note that the pixels unconnected to other surface pixels are discarded using Connected Component Labeling [63] algorithm and are shown with gray color.

10.3.3.3 Entity Feature Vector Computation

Entity feature vector includes robot's visual and tactile percept in progressively increasing complexities in subsequent developmental phases:

In **Phase I**, entity feature vector is represented only with one feature, *touch state* (T):

$$\boldsymbol{f} = (T)$$

In **Phase II**, the most basic object visual feature, *object-visibility* (V) is added to the entity feature vector:

$$\boldsymbol{f} = (T, V)$$

In **Phase III**, the *convolved* touch signal is used as tactile feature instead of *touch-state*. Since the robot learns the object affordances in this phase, the object perception is more complex and includes position and shape related information:

$$\boldsymbol{f} = (C, V, \boldsymbol{P}, \boldsymbol{S})$$

where C, V, $\boldsymbol{P}$, and $\boldsymbol{S}$ represent *convolved* touch signal, *object visibility*, 3D object position, and object shape feature vector, respectively.

10.3.3.4 Effect Feature Vector Computation

The effect created by a behavior is defined as the difference between entities' final and initial feature vectors.

10.4 Discovering Behavior Primitives and Learning Affordances

10.4.1 Phase I: Emergence of Behavior Primitives Using Touch

In this phase, the robot observes its sensor trajectory during different executions of *swing-hand* behavior and extracts behavior primitives from common segments in these trajectories. The trajectories are segmented based on the changes in robot's touch sensors, i.e. a new segment starts when a change occurs in touch state.

For each *swing-hand* behavior execution, the trajectory is segmented when there is a change in touch state. Figure 10.4 shows obtained segment instances from behavior executions demonstrated in Figure 10.3. From different executions and different segmentations, there are segments with common characteristics. For example, the first segment obtained in (a) and (c) are similar since they correspond to the touch stage change of **off** $\rightarrow$ **on**. As another example, the last segments obtained in (b), (c), and (d) correspond to the common change (or no change) of **off** $\rightarrow$ **off**. Thus, based on touch state change, the experienced segment instances are grouped into generic segments, i.e. behavior primitives.

10.4.2 Phase II: Emergence of Higher Order Behavior Primitives Using Vision

In **Phase II**, the robot explores the environment using the behavior primitives discovered in **Phase I**, and it observes *object-visibility* besides *touch-state*. Thus, if the execution of the same behavior with different hand speeds generates different results in terms of object visibility, then two new behaviors emerge and the old one is deleted from repertoire. The number of new behaviors is two in each discovery since *is-visible* is a binary signal.

10.4.3 Phase III: Learning of Affordance Relations

In this phase, the robot learns affordance relations using the behavior primitives discovered in the previous phases. As in previous chapters, the robot interacts with objects, discovers effect categories and then learns the mapping between entity features and effect categories.

Effect category discovery Effect categories are discovered by clustering the effects using the channel-based hierarchical clustering algorithm described in Chapter 8. In the lower level, channel-specific effect categories are found by clustering in the space of each channel, discovering separate categories for visibility, position and shape. In the upper level, the channel-specific effect categories are combined to obtain all-channel effect categories using the Cartesian product operation. In both levels, if the number of samples of any effect category was lower than a threshold, the corresponding effect category was discarded.

This algorithm (in its original form as used in previous chapters) discards the effect categories based on how frequent they appear during interactions. Although an effect category is not common, it can still be significant for the robot. Thus, we replaced the strategy to discard effect categories with one method that is more robust since it checks the predictability of the effect category in deciding whether to discard or not. For this purpose, after finding a potential set of effect categories through clustering, the predictability performance of each category is checked separately by training classifiers. If the classifier is successful in distinguishing that effect category from others, that category is predictable. If one of the categories is not predictable in the potential set of effect categories, the clustering process is repeated with less number of maximum clusters. Algorithm 7 and Figure 10.6 provide the details of the channel-based effect category discovery method.

On the upper level, the same predictability mechanism is used to check each category. However, different from lower level, if a category is not predictable, it is discarded. Figure 10.6(b) presents this method.

Learning effect category prediction As in previous chapter, classifiers are trained to predict the effect category for a given entity feature vector, a behavior id and behavior's parameter by learning the $(\boldsymbol{f}^{()}, \alpha) \rightarrow E_{b_i,id}$ mapping. Specifically, we used a Support Vector Machine (SVM) classifier with Radial Basis Function (RBF) kernel to learn this mapping for each behavior b_i, where $(\boldsymbol{f}^{()}, \alpha)$ is given as the input, and the corresponding $E_{b_i,id}$ as the target category.

Algorithm 7 Channel based effect category discovery

$k_{\max}$: Maximum number of categories.

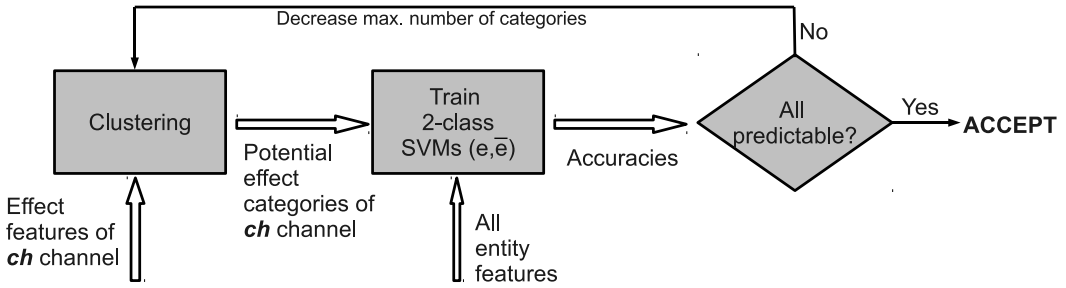
Reset($k_{\max}$): Reset $k_{\max}$ for new channel.

$f_{\text{effect}}(ch)$: Portion of feature vector limited to channel *ch*.

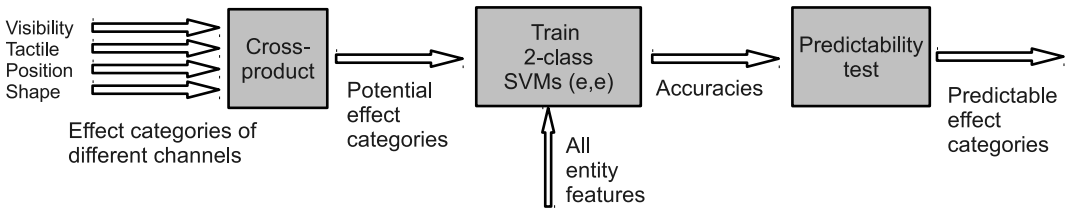
$Clusters(\{\boldsymbol{f}\}, k_{\max})$: Find between 1-$k_{\max}$ clusters with feature set $\{\boldsymbol{f}\}$.

$\{E_{id}\}^{i}_{ch}$: The set of effect categories found in channel *ch*.

1: **for** each channel *ch* in [*visibility*, *tactile*, *position*, *shape*] **do**
2: Reset($k_{\max}$)
3: **while** $k_{\max} \neq 1$ **do**
4: {STEP 1: Find optimal categorization by clustering n_{opt} times}
5: **for** $i = 1 : n_{\text{opt}}$ **do**
6: Find effect category set ($\{E_{id}\}^{i}_{ch}$) by $Cluster(\{f_{\text{effect}}(ch)\}, k_{\max})$.
7: Train classifier ($Predictor()^{i}$) to learn mapping $f \rightarrow E_{id}{}^{i}_{ch}$.
8: **end for**
9: Select effect category set ($\{E_{id}\}^{best}_{ch}$) with most accurate *Predictor*()
10: {STEP 2: Verify predictability of the optimal categories in $\{E_{id}\}^{best}_{ch}$}
11: **for** each effect category *e* in $\{E_{id}\}^{best}$ **do**
12: Assign same category ($\overline{a}$) to all effects except *e*
13: Train classifier to learn mapping $f \rightarrow e, \overline{e}$
14: **if** accuracy < threshold **then**
15: $k_{\max} = k_{\max} - 1$
16: **jump** (3) {Categorization does not allow prediction}
17: **else**
18: **continue** {Check predictability of next category}
19: **end if**
20: **end for**
21: **end while**
22: **end for**



(a) Effect category discovery for each channel *ch* in lower-level of hierarchical effect categorization method



(b) Each channel's effect category are combined in upper-level of hierarchical effect categorization method

Figure 10.6: Channel's effect category discovery based on categories' predictability.

10.5 Experiments

10.5.1 Discovered Behavior Primitives in Phase I

When only open-hand *swing-hand* behavior execution is considered, six different behavior primitives are discovered by the robot. Figure 10.7 gives these primitives with their touch state change characteristic. For each primitive, the initial and final velocity distributions are given on the left and right plots of the figure. At the bottom, the meaningful labels for these primitives are provided.

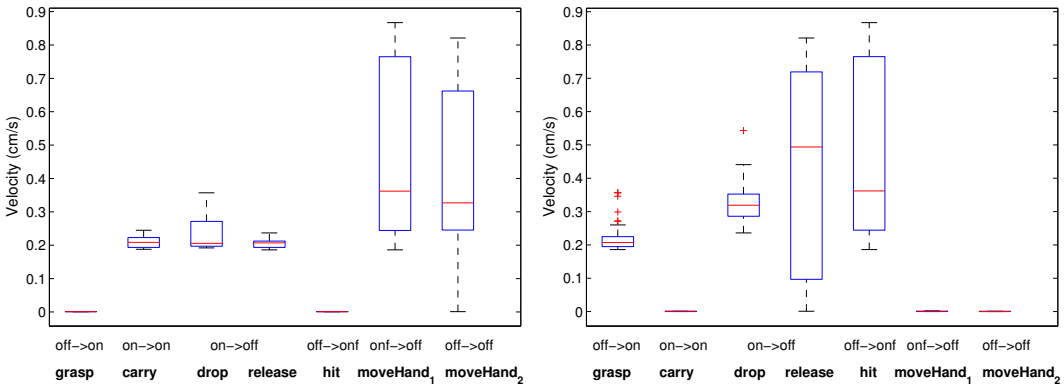
- **Hit** primitive corresponds to high velocity reach and hit to the object as shown by the first plot of (b). The touch sensor was activated for short time and the object was not grasped at the end.

- **MoveHand**$_1$ primitive corresponds to hand movement towards robot body without object after short duration touch to the object.

- **Grasp** primitive corresponds to slow velocity reach to the object and results in long period touch sensor activation, i.e. stable grasp.

- **Carry** primitive starts with slow velocity and stable grasp. When the primitive execution finished, the object is still at robot's hand indicated by **on** touch state.
- **MoveHand**$_2$ primitive corresponds to hand movement towards robot body without any object in the beginning and at the end.
- **Drop/release** primitive corresponds to hand movement to self with the object in the beginning of the behavior and no object at the end. The object falls as the result of unstable grasp in **drop** behavior in some situations, and as the result of disabled *grasp-reflex* in **release** behavior in other cases. We assumed that the robot can notice and and learn from the disabled grasp-reflexes.

Closed-hand *swing-hand* behavior is segmented to five different primitives, namely one tap, two drags, and 2 move-hands. Different from open-hand segments where the object drops due to unstable grasps, in closed-hand segments, the object may be dragged by robot's fist for certain time. However since the behavior segments are similar to previously found ones, they will not be carried to the next phase.



(a) Initial velocity distribution

(b) Final velocity distribution

Figure 10.7: The distribution of hand velocities in the beginning and at the end of behavior primitive executions.

Figure 10.7 gives the velocity distributions for all experienced segments. Consider the final velocity (reach velocity) distribution of **hit** and **grasp** behavior primitives. Ideally, the velocity of the hand for grasping should be always smaller than the velocity for hitting. However, as shown, the velocity ranges of these primitives are not clearly separable and contain overlapping parts because of the complex interaction dynamics and noise in the system. While the robot hand could **grasp** the object with an approach velocity of $0.36cm/s$, it could not grasp

Table 10.1: The behavior primitives and their encoding.

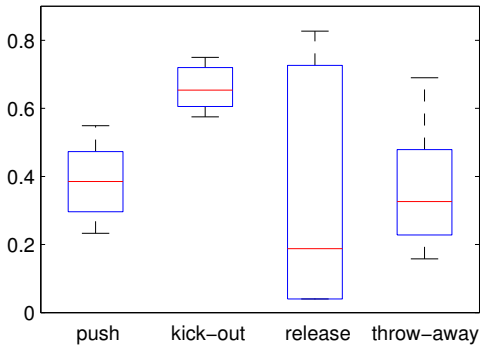
Name	Init Touch	Final Touch	Init Vel.	Final Vel.	GR	Dir.
Grasp	off	on	[0-0]	[0.19,0.22]	on	obj.
Carry	on	on	[0.19-0.21]	[0,0]	on	self.
Drop	on	off	[0.22-0.27]	[0.28,0.35]	on	self
Release	on	off	[0.19-0.21]	[0.10,0.70]	off	self
Hit	off	onf	[0-0]	[0.24-0.76]	on	obj.
MoveHand$_1$	onf	off	[0.25-0.46]	[0,0]	on	self
MoveHand$_2$	off	off	[0.25-0.65]	[0,0]	on	self

with $0.19cm/s$ in some situations. Thus, in order to increase the confidence we include the portion between first and third velocity quartiles into behavior primitive descriptions.

Table 10.1 gives the set of parameters that are automatically set for the corresponding discovered behavior primitive. As shown, if the robot hand which approaches to the object with $[0.19-0.21]cm/s$ velocity would **grasp** the object, however if the approach velocity is $[0.22-0.27]cm/s$, the object cannot be grasped and the robot hand would **hit** it. As another example, even if the touch state is **on** initially, if the hand velocity is high ($[0.22, 0.27]cm/s$), it corresponds to high-speed grasp attempt, i.e. unstable grasp, thus the object **drops** from the hand.

10.5.2 Discovered Behavior Primitives in Phase II

The seven behaviors obtained in previous phase are further exercised and the robot robot distinguished new behaviors based on differences in visibility change. From **release**, **release** and **throw-away** behaviors emerged. From **hit**, **push** and **kick-out** behaviors emerged. The new primitives are shown in Figure 10.8.



Name	Final Vel	Final Object
Push	[0.30,0.48]	visible
Kick-out	[0.61,0.72]	invisible
Release	[0.04,0.72]	visible
Throw-away	[0.23,0.48]	invisible

Figure 10.8: Distribution of hand velocities at the end of different primitive executions.

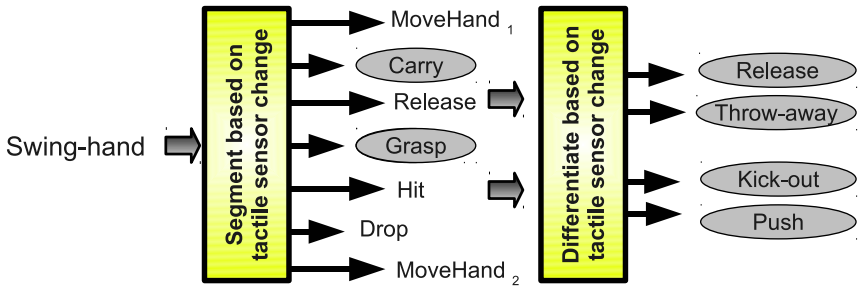


Figure 10.9: The summary of hierarchical behavior primitive discovery. The circled behaviors correspond to behaviors that can be used in next developmental phases.

10.5.3 Learned Affordances in Phase III

In this chapter, the affordances of **grasp** behavior is learned only. In the experiments, a table with 100×70 cm^2 surface area was placed with a distance of 40 cm in front of the robot. At the beginning of each exploration trial, one object of random size $[8cm - 40cm]$ was placed on a fixed reachable location at random orientation. Since the grasp affordances of boxes of different sizes and orientations is more difficult to learn and predict, large number of boxes are included into interactions. The robot simulated 2000, 400, and 400 **grasp** interactions with boxes, cylinders, and spheres, respectively. Through experiments, the *approach-direction* parameter (α) of grasp is kept random. The set of relation instances were used in learning. The X-means algorithm was used to find channel-specific effect categories[1], and Support Vector Machine (SVM) classifiers were employed to learn effect category prediction.

10.5.3.1 Discovered Effect Categories

The iterative version of the hierarchical clustering algorithm, detailed in Algorithm 7 is used to find the effect categories. The maximum number of categories, $k_{\max}$, is set to 5 for each channel. The results are given in Table 10.2.

- For visibility channel, X-means algorithm finds 2 categories. The first and second categories correspond to *disappear* and *stay-in-view*, respectively. When an SVM predictor is trained to differentiate the first category from the other categories (from the second one), the prediction accuracy is found to be high. The same is valid for second

[1] X-means implementation in Weka data mining software is used [94].

Table 10.2: The effect categories discovered by iterative hierarchical clustering. The potential categories, category prototype feature vectors, and the corresponding predictabilities are given for each channel.

Channel	Categories	Prototype vector	2-category accuracy	Predictable?	Accepted?
Visibility	2 categories	-1 0	90.6 % 90.6 %	√ √	√
Tactile	3 categories	0.40 0.11 0.00	79.73 % 64.42 % 68.82 %	√ *X* *X*	*X*
	2 categories	0.38 0.04	80.15 % 80.15 %	√ √	√
Position	3 categories	[9, 2, 1] cm [0, 0, 0] cm [12,-4,12] cm	67.00 % 81.14 % 79.20 %	*X* √ √	*X*
	2 categories	[2, 1, 0] cm [12, -3, 12] cm	78.2 % 78.2 %	√ √	√
Shape	3 categories	Large Large Small	73.47 % 69.77 % 71.27 %	*X* *X* *X*	*X*
	2 categories	Large Small	70.5 % 70.5 %	*X* *X*	*X*
	1 category	NA.	NA.	√	√

category as well. Thus, these categories are valid and transformed to higher-level for cross-product operation.

- For tactile channel, X-means algorithm first finds 3 categories. The value shown in prototype vector is the ratio of activated touch sensors on hand. Thus, the first category corresponds to high-activation (grasp) and second and third categories correspond to low-activation (finger touch or no touch). The performance of the SVM classifier that is trained to differentiate the first category from the others (graspable from others) is high (79.73%). However, the SVM classifiers cannot distinguish second and third categories well since the accuracies are around 65%. As a result, maximum number of categories are decreased to 2, and X-means algorithm is executed again. The new categories correspond to high-touch-activation and low-touch-activation, and they are predictable (with 80% accuracy), so they are transformed to higher-level.

- For position channel, X-means algorithm finds 3 potential categories, where two of them cannot be predicted. Thus, in the next iteration 2 categories are found and transformed to higher-level.

Table 10.3: Effect categories discovered for grasp behavior.

Effect id	Visibility	Tactile	Position	Shape	Comment
Effect 1	-1	0.04	no change	N.A.	Disappeared
Effect 2	-1	0.38	no change	N.A.	Grasped & disappeared
Effect 3	no change	0.38	no change	N.A.	Grasped
Effect 4	no change	0.04	[12, -4, 12] cm	N.A.	Pushed

- For shape channel, neither 3 category nor 2 category set is found to be predictable. Thus, there is no categorization in shape channel. The failure in discovering any meaningful shape change category is mainly due to robot's inability to track object's surfaces.

After effect categories are found for each channel, they are combined in the upper level by cross-product operation. Some of the new categories are discarded since they don't satisfy *predictability* criteria (see Figure 10.6(b)). At the end, 4 effect categories are shown to remain.

10.5.3.2 Effect Prediction Performance

After the discovery of effect categories, the mapping from the initial object features to these categories is learned for grasp behavior by multi-class Support Vector Machines (SVMs). The LibSVM [19] software package was used with optimized parameters of the RBF kernel through cross-validated grid-search in parameter space. 2200 simulated interactions were used in training and a separate set of simulated 600 interactions were used for testing. During training and testing, same number of samples from each category are used to avoid any effect of dominant category. Figure 10.10 gives the change in accuracy based on number of features used in training.

10.6 Conclusion

This chapter provides the most comprehensive work in terms of affordance learning and developmental progression of the robot. In this system, the manipulator robot initially possesses one basic action (*swing-hand*) and one basic reflex (*palmar-grasp*). In three developmental phases, it learns affordances for a set of self-discovered behaviors in a completely unsupervised way. In the first phase of development, it executes *swing-hand* action on a fixed small

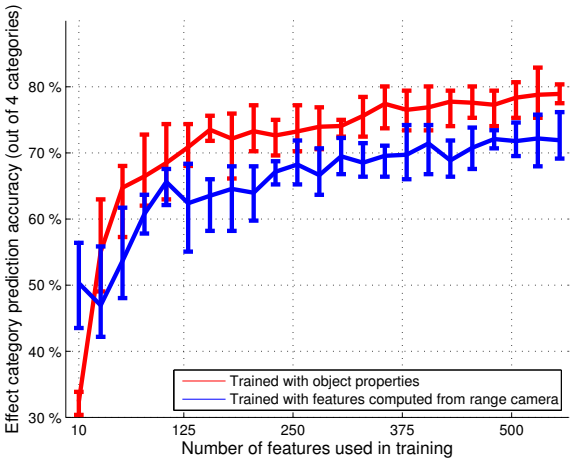


Figure 10.10: The change in prediction accuracy of classifiers trained with real object information and computed features.

object with different hand approach speeds. During action executions, it monitors the changes occurred in its limited tactile sense and automatically distinguishes behavior primitives by segmenting *swing-hand* action. In the second developmental phase, it starts using a limited visual perception, and exercises the discovered behaviors again on the same object. It monitors the changes occurred in visibility of the objects and distinguishes more behaviors from the existing behavior primitives.

Third developmental step corresponds to affordance learning, i.e. learning the relation between behavior parameters, object and robot's perceptual features and the effects generated. Thus, the representation and complexity of discovered behavior primitives are enriched by including more parameters (such as approach direction or direction of move). Then, the robot learned affordances of *grasp* behavior in terms of visual properties of objects, it's tactile sensors, approach direction and effect categories. In this chapter, we also formulated the most robust form of the hierarchical clustering algorithm that was used to discover effect categories.

CHAPTER 11

DISCUSSION

In this chapter, first, we will review the related robotic studies and emphasize our contributions to the field of Autonomous Robotics. Next, we will discuss the realized robotic framework and the experimental results within the field of Developmental Psychology.

11.1 Robotics

During the recent years, studies inspired by ideas in developmental psychology have increased considerably ([155, 92, 4, 134]). These studies typically use exploration, learning and embodiment to enable robots learn about their environment via exploration with minimal expert knowledge. In the rest of this section, we review related studies (see Table 11.1 for a summary) and discuss the contribution of this thesis to the field of Autonomous Robotics.

The pioneering studies that used motor babbling as a means of exploration for learning of affordances include [99, 46] and [132]. For example, [46] studied the learning of rollability affordance by executing different actions on different objects and observing their effects. [132] investigated tool affordances by discovering tool-behavior pairs that give the desired effects. In these studies, learning of the association between the visual features of the objects/tools and their effects (i.e. affordances) were not addressed; therefore the learned affordance knowledge could not be generalized to novel objects/tools.

Later this issue was addressed by [139, 43, 48], where the relations between the visual features of entities and their affordances were learned. However, in these studies, the affordance categories (e.g. liftability and traversability) were pre-defined, and learning was performed in a supervised way based on the success criteria defined over the effect of each action. Thus,

the affordances were not discovered by pure exploration and the robot only learned to predict the effects designed by the programmer.

[129, 60, 32] proposed the self-discovery of the affordances, where the effect categories were found through unsupervised clustering in the effect space. Furthermore, using these categories, the mappings of *object* → *effect categories* were learned. Thus, the robot was able to make predictions to choose actions that would fulfil a desired environment change.

All the aforementioned studies were deterministic and relied on one-directional mappings. [37, 65, 101] used probabilistic networks that capture the stochastic relations between objects, actions and effects. These networks allowed bi-directional relation learning and prediction. For example in [101], after training Bayesian networks, the robot could predict the object categories when effects and actions were given, or it could predict the effect categories when objects and actions were given. One drawback of this approach was that the object categories were created by unsupervised clustering in feature space without any reference to the interaction experience of the robot. Cognitive development in humans suggest that actions and the effects created by them are used to parse the perceptual space into categories which may be called 'objects' (entities). Therefore, it is not that the object categories exist in the environment and their relations with the effects and actions are learned; but rather the effects and actions define the object categories. In our work, we follow this full action based perception view by categorizing the object feature space based on the effects. This is central to the affordance concept.

In all of the studies mentioned above, the agent acquires the ability to make predictions about the effects it can create through active exploration of the environment. However, due to the 'effect' representation adopted in these studies, the systems described cannot predict more than one step ahead, which prohibits complex planning.

Table 11.1: Summary of related robotics literature.

The *Fixed objects* group does not use to object properties. The *Supervised* and *Unsupervised* groups learn one-directional mapping from object/environment properties to given and discovered effect categories, respectively. The *Probabilistic* group learns the probabilistic bi-directional relations and the *High-level rules* group assumes the existence of AI inference rules. The *Planning in perceptual space* group's learning is similar to the *Unsupervised* group, but can make multi-step predictions.

		Initial percept representation	**Effect/final percept representation**	**Learning**	**Learning Method**
Fixed objects	[46] [132]	Predefined-ids	Continuous Binary	initial → effect	Probabilistic inference Table-lookup
Supervised	[139] [43] [48]	Continuous	Binary	initial → effect	SVM GMM Decision tree
Unsupervised	[129] [60] [32]	Continuous Discrete (GWR)	Discrete (X-means) Discrete (GWR)	initial → effect	Decision tree Nearest-neighbor Feed-forward NN
Probabilistic	[37] [101] [65]	Discrete (X-means) Continuous	Continuous Discrete (X-means) Binary	(initial) ↔ (action) ↔ effect	Bayesian Belief Nets. Bayesian Nets. Relational Dep. Nets.
High-level rules	[117] [159] [100]	Binary Continuous Continuous	Binary Binary Continuous	Logical predicates	Kernel Perceptron Knowledge Base Optimize cost func.
Planning in perceptual space	[121] Current work	Discrete (SOM) Continuous	Discrete (SOM) Discrete (X-means)	initial → final effect	List of links SVM

In [117, 159, 100] on the other hand, after learning, the robots could make multi-step predictions using transition rules and hence were able to demonstrate complex planning. The transition rules were defined as actions linked by logical precondition and postcondition predicates. Their approach is different from the previous ones since sensorimotor experience of the robot was used to associate the predicates of the transition rules. These conditions were pre-defined binary functions of sensor readings in [159], where the robot learned to combine these conditions in the form of pre-conditions and effects through human assistance. [117] used pre-defined or pre-learned high-level object and environment properties as the predicates of the transition rules. On the other hand, [100] discovered these predicates from low-level sensory readings during a goal-free exploration and learning phase. Although objects could be categorized based on their shapes in the sensory level, this information was not used in effect prediction. Moreover, only position features were used to learn "simple affordances of the object" [100, p.886]. In short, in these approaches, the learned affordances were either simple or acquired through supervision. In addition, the mapping of these architectures to developmental psychology is not straightforward as logical inference mechanisms are assumed to be available to the learning agent.

In this thesis, we followed a similar approach to the studies [129, 60, 32] that were discussed above (*Unsupervised* group in Table 11.1). The main novelty of our approach is the encoding of the effects and objects in the same feature space. In contrast, in the other studies the effect representation were context and task dependent, and therefore did not correspond to the object feature space. Having the effects and objects encoded in the same space provides the ability to predict the next perceptual state by adding the current features to the predicted effect features. This enables the robot to make plans (without using high-level AI rule techniques) based on the structures that were learned in a completely bottom-up manner during its interaction with the environment.

From the planning viewpoint, [121] can be considered as the closest to our approach, where the robot learns the environment dynamics in its perceptual space and plans multi-step actions to achieve goals in a locomotion task. However, there are important differences that sets our work apart: First, the initial percept space is categorized in an unsupervised manner, i.e. irrespective of the interaction experience of the robot (as was the case for [101]). Second, (in [121]) the robot learned the *initial*→*final* mapping, whereas our system learns the *initial*→*effect* mapping which provides better generalization. For example, in our case, push-

ing a box located on the table would always generate the same effect regardless of its position (unless, of course the object is at the edge). Yet, at the same time, it is possible to obtain the final percept (i.e. the predicted position of the box). Generalization of the knowledge obtained via exploration is a critical issue when the world of the agent becomes more complex, i.e. when the number of actions and the type of environments that can be experienced becomes large. An adaptive agent needs to utilize its resources parsimoniously, and needs to be able to predict in situations that it never encountered before.

In summary, the points that set our work apart from the existing ones are (1) multi-step planning, (2) categorization of the perceptual space based on actions and their effects, (3) generalization of the knowledge obtained through exploration.

11.2 Cognitive Development

The development of artificial agents that learn through embodied interaction with the environment is rather recent in robotics. Yet, the concepts leading to these ideas have been studied in developmental psychology for years. The necessity of prediction capability for goal-directed action execution and planning goes back to 19th century. The ideomotor principle postulates that an agent must use his/her anticipation of an action's outcome to execute intentional actions [74]. Furthermore, according to this principle, these anticipations are represented as action-effect relations which are learned during the motor babbling phase through exploration of the environment [118]. Our work, among others, captures this basic observation, namely learning the effects of actions in the environment and representation of these experiences to be used in prediction and planning.

An unsettled discussion is whether the goals and predictions are encoded in the perceptual space of the agent [59] or not. This is an important issue both for a robot designer and a neuroscientist searching for neural correlates of intelligent behavior. Although, most would agree that a hierarchy of predictive mechanisms, ranging from sensory to abstract, is a prerequisite for intelligent behavior, we further argue that this by itself is not sufficient. The critical issue is that the goals and the predictions can be expressed in the low level perceptual space when needed. According to [40], this is supported by recent behavioral and neuropsychological findings. In the experiments presented with our system, the goals were specified directly in

the perceptual state of the robot. However, this is not the only possibility as the prediction mechanisms we employed represent the effects of actions in two levels: One is the affordance level where discrete and abstract items are predicted (i.e. effect categories). The second is the sensory level where the prediction takes place in the perceptual space (i.e. current perceptual state + effect category prototype). If we were to consider our robot as a 'high-level agent' and ask it to bring about the 'lifted effect' it would be able to tell whether there is, any object that affords this high level (non-perceptually specified) goal, and if, indeed, there is, it would be able to execute a behavior to bring about the desired effect.

In spite the ongoing debate on whether these anticipations are represented in the sensory-motor space or in a more abstract level, it is widely accepted that these mechanisms are used in planning [67]. According to Piaget (1952), human infants start to distinguish means-ends relation at 4-8 months, and start to use these relations until around 12 months for one step goal satisfaction. It is not implausible that a limited amount of anticipation skill be hardwired through evolution in humans and other animals; however, the majority of this skill must be acquired by the organisms through interaction with the environment. [42] and [66] argue that this ability cannot be innate and the human infant learns to use anticipation for goal-directed action execution in his/her early months of infancy. Infants use the learned action-effect relations to anticipate the results of their actions in a goal-directed way starting from 9 months [119, 138, 158]. Piaget argued that planning is only possible after development of symbolic representation at 18-24 months, although there is evidence that younger children are able to make multi-step plans. For example, 9-month olds are shown to generate a multi-step plan to reach a toy, by first removing the obstacle, then pulling the towel and grabbing the object placed on it [157, 23]. In our system, the prediction ability was demonstrated for multi-step planning without going through a gradual development. This, however, could be easily emulated by restricting the planner to plans of depth one and gradually increasing the allowed depth in the search for plans to achieve the desired progression. One can speculate that the inability of early infants to make complex plans is due to an immature working memory needed for planning. As the infant grows, the increasing memory that is available to the planning may allow complex plans. This argument regards the planning mechanism as fixed but requiring more memory as the sought plans become more complex; however this may not be the case as different planning mechanisms may coexists and develop along with the infant's cognitive development [36]. It is largely unknown whether symbolic manipulation ability is

necessary for complex planning, as Piaget argued. Our stance is that, computationally, there is no such necessity; even though a symbol manipulation machinery would be an invaluable tool for planning and other cognitive skills. Especially this would be more advantageous as the plans shift from physical to social domain. In the presented system, the plans are performed in the perceptual domain which allow the robot to naturally interact with objects it did not experience before. For example it would be able to make a bottle disappear from the table, even though it has no idea what a bottle is and has never seen it before.

One feature we introduced for specifying the goals automatically in our system has interesting relations to the so called 'goal emulation' in cognitive psychology. The term can be defined as an observer's learning that a particular goal can be achieved and setting about achieving it by its own [151]. Goal emulation is different from other social learning mechanisms such as mimicry and imitation and somewhat puzzling. Most animal mimicry is restricted to goal emulation, which is generally regarded as a simpler task than imitation. However, human infants who can imitate are unable to use goal emulation to learn new skills: Children show mimicry before 12 months of age but only start to pay attention to goals of the demonstrator only after that period [122, 17, 151]. 17-month-olds can use observed actions or their own action repertoire to achieve the observed goal depending on the context [50], and 18-month-olds can understand 'intended' goals of a demonstrator trying but failing to achieve his goal [98], 18-month-olds can learn tool use by observation. However, goal-emulation, i.e. executing a sequence of behaviors after observing only the goal state, develops rather late [10, 122] and only after 18-months-old infants are able to emulate action sequences for novel goals [69, 8]. For example, in [7, 8], 27-month-old infants were able to execute a 3-step plan to construct a rattle from two cups and a ball but 21-months-olds were not. The puzzling findings for human infant goal emulation could be due to the lack of motivation or the insufficient affordance experience with the toys used in the experiments. Many animals are known to make multi-step plans involving different types of objects. For example in [84], chimpanzees stack four boxes on top of each other to reach a banana that was hung out of their reach. They were also able to climb on a long stick instead of stacking the boxes when that stick was available in the environment. However, the same chimpanzees were not able to generate plans with those objects when the objects resided in another (accessible) room even if chimpanzees had explored that room recently. This observation leads [131] to conclude that non-linguistic animals use object affordances to make plans; and they start reasoning from the immediate environment (initial

state) to reach the goals, i.e. they use forward-chaining. Furthermore, the plan generation can be successful if they have learned the affordances of the objects before [58].

Within the light of above discussion, we can argue that our robot system when run in automatic goal setting mode is more similar to a chimpanzees rather than a human infant, as the goal is more important than the means for a chimpanzee. Although, chimpanzees utilize social learning mechanisms to develop various tool use skills, unlike humans, they are less sensitive to demonstrator's body movements and tend to emulate the goal more than to imitate the demonstrator [137]. In fact movements that do not have apparent targets such as another object or a body part has little imitation potential for chimpanzees [105].

One final similarity of our system's working with a chimpanzee's cognitive abilities is that chimpanzees have difficulties when manipulating objects in different multiplicities. It may be speculated that this could be due the lack of symbolic planning ability of chimpanzees. This is analogous to the case in our system: our system benefits from having the planning done in perceptual space in terms of generalization and robustness; but it faces difficulty in encoding goals in the same representation for different number of objects. In the current implementation, we overcame this by introducing a special goal setting mechanism inspired from the observation (unpublished video related to [105]) that when chimpanzees are asked to imitate an action involving two objects (put an object in a bowl), they appear to reproduce the spatial relation of the objects rather than the absolute spatial configuration shown to them (by holding the object and the bowl in both hands and bring them together in the air instead of on table).

We close this section by noting that although Piaget's requirement for symbolic manipulation ability for complex planning might be too strict, higher cognitive abilities, including multi-object and memory based planning requires the development of symbolic planning mechanisms irrespective of whether the symbols manifest themselves as linguistic constructs or not.

CHAPTER 12

CONCLUSION

12.1 Summary of the Results

In the Introduction Chapter, our approach was defined within a multi-disciplinary context at the junction of Autonomous Robotics, Developmental Sciences and Ecological Psychology (or the theory of affordances). Accordingly, in this section we will discuss the obtained results in the face of these three disciplines.

In **Chapter 3** [145], we gave a formal description of affordances learning framework. In this framework, the affordances were represented by (*effect*, *entity*, *behavior*) nested triplets. The affordance learning was performed in phases where similar action-effects were grouped as discrete effect categories first, relevant features necessary to predict effects were identified for each behavior next, and the mapping from relevant features of entities and behavior parameters to effect categories were learned at the end. In this chapter, we also described different control methods that use learned affordances in a goal-oriented way during actual robot execution.

In **Chapter 4**, the mobile and manipulator robot platforms, the range sensors and the simulation environments were described.

Chapter 5 [141, 139] studied the learning and perception of traversability affordances on a mobile robot equipped with range sensing ability.

- From robotics point of view, first, a new perspective to the navigation problem was developed where traversability was not limited to classical obstacle avoidance where the robot tries to avoid making any physical contact with the environment. Specifically, we

proposed a set of features for representing the shape and distance information on range images that are shown to provide a good degree of generalization, and a scalable method towards learning affordance relations. It was also shown that the robot does not need to detect the objects in the environment to detect their traversability. Instead, it could 'directly' perceive traversability affordances of the objects using low-level position and shape properties that were extracted from whole environment. The proposed method also showed that one can start with a large feature vector that contains all types of feature detectors that one can propose, and have it reduced down to only a fraction after training. In this sense, the robot could minimize the load on its perceptual processing after learning to achieve perceptual economy. At the end, using the learned affordance detection ability, the real robot could successfully navigate in an office environment cluttered with objects that it has never interacted before.

- From affordances point of view, we tested the robot's affordance perception in similar experiments conducted in Ecological Psychology such as detection of *climbability*, *go-under-ability*, and *go-through-ability*. We showed that three main attributes that are commonly associated with affordances, that is, affordances being *relative* to the environment, providing *perceptual economy*, and providing *general information*, are simply consequences of learning from the interactions of the robot with the environment.

Chapter 6 [140] studied a curiosity-based online learning algorithm that automatically chooses novel situations to interact based on previous experience.

- From robotics point of view, we proposed a two step learning process which consists of bootstrapping and curiosity-based learning phases. In the curiosity-based learning phase, we proposed a method that choose novel situations to interact based on the confidence of the affordance detector that was trained up-to that point. We showed that by adjusting the parameters of the system such as bootstrapping duration and curiosity threshold, the speed of the learning could be optimized and the robot could learn the traversability affordance using less exploration time.

- This method was inspired from 'intrinsic motivation' mechanism used by infants during exploration. Thus, from development point of view, it can be argued that infant's confidence and experience on affordances of an object affects its interest on 'playing'

with that particular object. In other words, the confidence on object affordances can be part of of the so called 'intrinsic motivation' mechanism.

Chapter 7 [143] and **8** [144, 142] studied unsupervised learning of affordances where the mobile and manipulator robots interact with the objects in their environment using a pre-coded repertoire of behaviors.

- From robotics point of view, we proposed a method that allows the robot to learn object affordance relations which can be used to predict the change in the percept of the object when a certain behavior is applied. The key aspect of this approach is that, the robot learns about its environment by discovering the effects it can generate through its actions using a novel feature-channel based hierarchical clustering algorithm. It then forms forward models [79] that enable it to predict the changes in the environment in terms of discrete effect categories as well as low level sensory changes. Furthermore, this prediction ability could then be used to develop plans using forward chaining. A robot that learned affordances through self-interaction and self-observation could make plans to achieve desired goals, emulate end states of demonstrated actions, monitor the plan execution and take corrective actions using the perceptual structures employed or discovered during learning. Using this method, the mobile robot with limited manipulation capabilities could generate and execute multi-step step plans such as 'drive-forward-left', 'drive-forward-right', and 'lift', in order to lift a novel unreachable object. On the other hand, the anthropomorphic robotic manipulator could emulate the observed goals by making multi-step plans. For example, if it observed an empty table as goal, then it could clear the table by pushing or lifting or dropping the objects. As another example, if the robot observed one object lifted in the air, it could bring other objects to the same position by pushing several times and lifting. It also had the ability to make plans with multi-objects. For example, if it observed two objects that are close to each other as the goal, then given two objects far from each other, it could generate plans to bring them closer. All these plans were made in robot's perceptual space and they were based on learned affordances and learned prediction ability.

- From developmental point of view, first (as discussed in detail in Chapter 11), the proposed learning system share crucial elements with infant development such as goal-free exploration and self-observation. Second, as supported by recent behavioral and neu-

ropsychological findings, the goals and multi-step prediction mechanisms can indeed be encoded in perceptual space of 12-month-old infants who has not developed symbolic representation yet. Our work also showed that multi-step plans can be generated and executed using unsupervised learned and bottom-up constructed knowledge with no symbolic structures or complex reasoning mechanisms. Third, similar to the non-linguistic animals, our system also uses object affordances to make plans and the robot starts reasoning from the immediate environment to reach the goals through forward-chaining. Fourth, we discussed that in terms of 'goal emulation', our robot system when run in automatic goal setting mode is more similar to a chimpanzees rather than a human infant, as the goal is more important than the means (action imitation) for a chimpanzee. One final similarity of our system's working with a chimpanzee's cognitive abilities was that chimpanzees have difficulties when manipulating objects in different multiplicities. We speculated that this could be due the lack of symbolic planning ability of chimpanzees (and our robot system).

Chapter 9 proposed a method that allowed the manipulation robot not only to discover *what* type of affordances were offered by the objects but also to learn *how* to parameterize its behaviors to act on the provided affordances. After robot's exploration, the effect of behavior parameters were learned in relation with the object features and the generated effects. This learning enabled the robot to predict the objects' next perceptual state based on the current object features and the behavior parameters. For execution, a control method that searches the behavior parameter space and selects a particular parameter to act on affordances was proposed. In the real robot experiments, the graspability affordances of different objects was detected. The robot correctly parameterized its precision or power grasp behaviors to approach and grasp different objects by approaching them from different directions.

- From robotics point of view, this chapter presented a method to learn the relation between the object affordances, behavior parameters, and obtained effect categories. This chapter also proposed a visual perceptual processing to represent objects using a combination of local and global position and shape (related) features. At the end, the real robot hand was able to detect liftability affordances, i.e. choose correct approach direction parameters for precision and power grasp behavior for mugs in different orientations. The execution of *power* and *precision-grasp* behaviors on objects in different

size and orientations (and with/without handles) was shown to have non-trivial dynamics when the dexterous robot hand was used in interactions. Still, our system could learn the affordances provided by these objects and could act upon the provided affordances by correctly parameterizing the grasp behaviors.

- From development point of view, we discussed that it takes 9 months for infants to reach for objects with correct hand-orientation and adjust their grip size based on object size before contact. As discussed before, the learning of basic *affordances* takes place mainly between 7-9 months. This suggests that visual properties of objects and behavior parameters are learned together indicating that findings of robot experiments is consistent with the developmental time-line of infants.

Chapter 10 provides the most comprehensive work in terms of affordance learning and developmental progression of the robot. In this system, the manipulator robot initially possessed one basic action (*swing-hand*) and one basic reflex (*palmar-grasp*). In three developmental phases, it learned affordances for a set of self-discovered behaviors in a completely unsupervised way. In the first phase of development, it executed *swing-hand* action on a fixed small object with different hand approach speeds. During action executions, the robot monitored the changes occurred in its limited tactile sensor and automatically formed behavior primitives such as *grasp*, *hit*, *carry-object*, *drop*, by segmenting *swing-hand* action. In the second developmental phase, it started using a limited visual perception, and exercised the discovered behaviors again on the same object. It monitored the changes occurred in visibility of the object and formed more behaviors from the existing behavior primitives. For example, by executing same *hit* action with different hand speeds, it discovered two new behaviors, namely *push* and *kick-out*. The third developmental step corresponded to affordance learning, i.e. learning the relation between behavior parameters, objects and perceptual representation of the environment, and the effects generated within this representation. The robot learned grasp related affordances in terms of visual properties of objects, tactile sensor readings, approach direction and effect categories. In this chapter, we also formulated the most robust form of the hierarchical clustering algorithm that is used to discover effect categories automatically.

Since this chapter is the most comprehensive work, we will relate our work to the common characteristics of Developmental Robotics as enumerated in the Introduction Chapter:

- The learning agent was embodied in a robot, and situated the an environment with many

object that it physically interacts. Cognitive development of the robot was the result of this embodiment and physical interactions.

- The development was incremental. In behavioral level, starting from one innate reflex like action, more complex behaviors were emerged based on previously acquired ones.Cognitive skills were also developed through time through discovering effect categories and acquiring prediction ability.

- The development was not task-dependent and it was completely driven by robot's perceptual system, innate action and reflexes and the objects in the environment. If there was no graspable objects in the environment, the *carry* behavior wouldn't have been emerged. Or as another example, if all the objects were rollable, *push* and *throw-away* behaviors wouldn't have emerged from *hit* since all objects would have rolled out off the table after any interaction.

- The complexity of sensory and motor system was initially limited and crude. In sensory level, progressively more complex perceptual processes and representations were used through development. In motor level, while initially the robot was *swinging* its hand with random speeds, after development, it was able to select approach directions and contact speeds in order to *grasp* the objects.

- The robot developed abstract concepts from its low-level continuous sensorimotor experience. Emerged behavior primitives, discovered effect categories are examples of such categories that were formed in behavioral and perceptual level.

From development point of view, the progression of robotic skills demonstrated is consistent with infant's development time-line. It is very likely that the infant starts from a small number of simple behaviors, and then progressively discovers and distinguishes new behaviors through use of former ones. Based on the robotic experimental results obtained, we think that exercising the innate hand movements in different speeds, monitoring the tactile sensation, and using the innate reflexes may have significant role in infant's discovery of behavior primitives.

12.2 Future Work

- Our learning algorithm enables predicting the next perceptual state given the current perceptual state and behavior parameters. However, the prediction is only one way because of the working principle of SVM Predictors. For example, it cannot find the behavior parameters given current and desired next states. In order to find the best behavior parameter, it needs to make a search in behavior parameter space. Similarly, it cannot find the current state given a desired state and behavior parameters. A method that represents affordances such that any component of the relation (entity, behavior or effect) can be predicted based on the other components can be useful.

- Throughout this thesis, different perceptual representations are used in affordances perception. We showed that in some cases such as for traversability, the robot does not need to detect objects in order to detect their affordances. In other cases, (e.g. related to manipulation), the robot requires object-dependent features. Finding a perceptual representation that can be used both for navigation and manipulation can be studied as a next step. This representation should also be consistent with human and monkey visual perception system which sends data to affordance detection areas. One way to combine all different representations is to include all the features to the perception system and expect the learning to discover the most relevant ones for the perception of different affordances.

- Our robotic developmental progression follows the principle of simple to complex skill acquisition. However, the transition between stages (such as from behavior discovery to affordance learning stage) are determined by the human designer. In addition, although based on infant development, the perceptual representation used in each stage is also decided by the human designer. Although the robot develops skills in an unsupervised fashion, human interventions make the system not strictly developmental. Additionally, while in our system the skill development is clearly separated from each other, in infant development there is no sharp distinction. Thus, this system can be improved (developmentally) by minimizing human intervention, i.e. by enabling the system to automatically find ways to decide these points, and by having a more smooth and probably overlapping transition between developmental phases.

- The manipulation affordances were learned by interacting with one object at a time. In

early stages of manipulation development, infants also has similar strategy: If they are playing with one object, they do not appear to be interested in other objects. However in later phases, they start playing with multiple objects, and learn the affordance relations between them. Our robot could generate and execute plans with multiple objects under the assumption that only one object is affected by the behavior execution. This assumption is too strict in real life and should be relaxed. Probably one can postulate that learning the object-object affordance relations is the subsequent stage after single object affordance learning.

- Our system uses classifiers that always make deterministic predictions. On the other hand, it is difficult to capture uncertainty and to make predictions in partially known environments by deterministic methods. Thus, probably the most valuable improvement to our work would be to integrate the stochastic nature of robot-object interaction as bi-directional relations while being faithful to the developmental stages of human infants and not sacrificing the planning ability demonstrated by our system.

REFERENCES

[1] K. E. Adolph, B. I. Bertenthal, S. M. Boker, E. C. Goldfield, and E. J. Gibson. Learning in the development of infant locomotion. *Monographs of the Society for Research in Child Development*, 62(3), 1997.

[2] R. Arkin. *Behavior-Based Robotics*. MIT Press, 1998.

[3] R. C. Arkin and T. Balch. AuRA: Principles and practice in review. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2):175–189, 1997.

[4] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida. Cognitive developmental robotics: a survey. *IEEE Transactions on Autonomous Mental Development*, 1(1):12–34, 2009.

[5] J. Atkinson, B. Hood, J. Wattam-Bell, S. Anker, and J. Tricklebank. Development of orientation discrimination in infancy. *Perception*, 17(5):587 – 595, 1988.

[6] M. Bajracharya, A. Howard, L. H. Matthies, B. Tang, and M. Turmon. Autonomous off-road navigation with end-to-end learning for the LAGR program. *Journal of Field Robotics*, 26(1):3–25, 2008.

[7] P. J. Bauer. Holding it all together: How enabling relations facilitate young children's event recall. *Cognitive Development*, 7:1–28, 1992.

[8] P. J. Bauer, J. A. Schwade, S. S. Wewerka, and K. Delaney. Planning ahead: Goal-directed problem solving by 2-year-olds. *Developmental psychology*, 35(5):1321–37, September 1999.

[9] G. Bekey. *Autonomous Robots*. MIT Press, 2005.

[10] F. Bellagamba. Re-enacting intended acts: Comparing 12- and 18-month-olds. *Development*, 22(2):277–282, 1999.

[11] A. Blum. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, December 1997.

[12] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[13] A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, September 2005.

[14] J. G. Bremner. *Infancy*. Blackwell Publishing, Malden, MA, USA, 1988. 2nd edition, 1994.

[15] A. Bur, A. Tapus, N. Ouerhani, R. Siegwar, and H. Hiigli. Robot navigation by panoramic vision and attention guided features. In *Proceedings of the 18th International Conference on Pattern Recognition*, pages 695–698, Hong Kong, 2006. IEEE.

[16] K. Capek. *R.U.R (Rossum's Universal Robots)*. Penguin Classsics, 2004.

[17] M. Carpenter, K. Nagell, and M. Tomasello. Social cognition, joint attention, and communicative competence from 9 to 15 months of age. *Monographs of the Society for Research in Child Development*, 63(4), 1998.

[18] M. Çakmak, M. R. Doğar, E. Uğur, and E. Şahin. Affordances as a framework for robot control. In *Proceedings of the 7th International Conference on Epigenetic Robotics, EpiRob'07*, Piscataway, NJ, USA, 2007.

[19] C-C. Chang and C-J. Lin. LIBSVM: A library for support vector machines. `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. Last visited on December 2010.

[20] A. Chemero. An outline of a theory of affordances. *Ecological Psychology*, 15(2):181–195, 2003.

[21] A. Chemero, C. Klein, and W. Cordeiro. Events as changes in the layout of affordances. *Ecological Psychology*, 15(1):19–28, 2003.

[22] A. Chemero and M. T. Turvey. Gibsonian affordances for roboticists. *Adaptive Behavior*, 15(4):473–480, 2007.

[23] Z. Chen, R. P. Sanchez, and T. Campbell. From beyond to within their grasp: The rudiments of analogical problem solving in 10- and 13-month-olds. *Developmental Psychology*, 33(5):790–801, 1997.

[24] T. Collett. Stereopsis in toads. *Nature*, 267:349–351, 1977.

[25] T. Collett. Do toads plan routes? A study of the detour behavior of bufo viridis. *Journal of Comparative Physiology*, 146:261–271, 1982.

[26] J. H. Connell. SSS: A hybrid architecture applied to robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2719– 2724, Los Alamitos, California, 1992.

[27] R. Cooper and D. W. Glasspool. Learning action affordances and action schemas. In *6th Neural Computation and Psychology Workshop, Connectionist Models of Learning, Development and Evolution*, Perspectives in Neural Computing, pages 133–142, London, 2001. Springer-Verlag.

[28] S. Cornus, G. Montagne, and M. Laurent. Perception of a stepping-across affordance. *Ecological Psychology*, 11(4):249–267, 1999.

[29] Yaskawa Electric Corporation. Yaskawa motoman robotics. `http://www.motoman.com/`. Last visited on December 2010.

[30] I. Cos-Aguilera, L. Canamero, and G. M. Hayes. Motivation-driven learning of object affordances: First experiments using a simulated Khepera robot. In *Proceedings of the 9th International Conference in Cognitive Modelling (ICCM'03)*, Bamberg, Germany, April 2003.

[31] I. Cos-Aguilera, L. Canamero, and G. M. Hayes. Using a SOFM to learn object affordances. In *Proceedings of the 5th Workshop of Physical Agents*, Girona, Catalonia, Spain, 3 2004.

[32] I. Cos-Aguilera, L. Canamero, and G. M. Hayes. Using a SOFM to learn object affordances. In *Proceedings of the 5th Workshop of Physical Agents*, Girona, Catalonia, Spain, March 2004.

[33] E. Şahin, M. Çakmak, M. R. Doğar, E. Ugur, and G. Üçoluk. To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior*, 15(4):447–472, 2007.

[34] J. C. Culham and K. F. Valyear. Human parietal cortex in action. *Current Opinion in Neurobiology*, 16:205–212, 2006.

[35] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1:131–156, 1997.

[36] R. DeLisi. A cognitive-developmental model of planning. In *Blueprints for thinking: The role of planning in cognitive development*, pages 79–109. Cambridge University Press, 1987.

[37] Y. Demiris and A. Dearden. From motor babbling to hierarchical learning by imitation: a robot developmental pathway. In *Proceedings of the 5th International Workshop on Epigenetic Robotics*, pages 31–37. Lund University, 2005.

[38] R. Detry, D. Kraft, A. G. Buch, N. Krüger, and J. Piater. Refining grasp affordance models by experience. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2010.

[39] M. R. Doğar, M. Çakmak, E. Ugur, and E. Şahin. From primitive behaviors to goal-directed behavior using affordances. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 729–734. IEEE, 2007.

[40] B. Elsner. Infants' imitation of goal-directed actions: the role of movements and action effects. *Acta psychologica*, 124(1):44–59, 2007.

[41] B. Elsner and G. Aschersleben. Do I get what you get? Learning about the effects of self-performed and observed actions in infancy. *Consciousness and Cognition*, 12:732–751, 2003.

[42] B. Elsner and B. Hommel. Effect anticipation and action control. *Journal of Experimental Psychology*, 27:229–240, 2003.

[43] E. Erdemir, C. B. Frankel, K. Kawamura, S. M. Gordon, S. Thornton, and B. Ulutas. Towards a cognitive robot that uses internal rehearsal to learn affordance relations. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2016–2021, 2008.

[44] R. E. Fikes and N. J. Nilson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4), 1971.

[45] P. Fitzpatrick and G. Metta. Grounding vision through experimental manipulation. *Philosophical Transactions of the Royal Society: Mathematical, Physical, and Engineering Sciences*, 361(1811):2165–2185, 2003.

[46] P. Fitzpatrick, G. Metta, L. Natale, A. Rao, and G. Sandini. Learning about objects through action: Initial steps towards artificial cognition. In *Proceedings of International Conference on Robotics and Automation*, pages 3140–3145. IEEE, 2003.

[47] D. Floreano, T. Kato, D. Marocco, and E. Sauser. Coevolution of active vision and feature selection. *Biological Cybernetics*, 90:218–28, March 2004.

[48] G. Fritz, L. Paletta, M. Kumar, G. Dorffner, R. Breithaupt, and R. Erich. Visual learning of affordance based cues. In S. Nolfi, G. Baldassarre, R. Calabretta, J. Hallam, D. Marocco, J-A. Meyer, and D. Parisi, editors, *From animals to animats 9: Proceedings of the Ninth International Conference on Simulation of Adaptive Behaviour (SAB)*, LNAI. Volume 4095., pages 52–64, Roma, Italy, 25–29 September 2006. Springer-Verlag, Berlin.

[49] G. Fritz, L. Paletta, M. Kumar, G. Dorffner, R. Breithaupt, and E. Rome. Visual learning of affordance based cues. In *Simulation of Adaptive Behavior (SAB)*, pages 52–64, 2006.

[50] G. Gergely, H. Bekkering, and I. Kiraly. Rational imitation in preverbal infants. *Nature*, 415:755, 2002.

[51] E. J. Gibson. Perceptual learning in development: Some basic concepts. *Ecological Psychology*, 12(4):295–302, 2000.

[52] E. J. Gibson. The world is so full of a number of things: On specification and perceptual learning. *Ecological Psychology*, 15(4):283–288, 2003.

[53] E. J. Gibson and R. D.Walk. The visual cliff. *Scientific American*, 202:64–71, 1960.

[54] E. J. Gibson, G. Riccio, M. A. Schmuckler, T. A. Stoffregen, D. Rosenberg, and J. Taromina. Detection of the traversability of surfaces by crawling and walking infants. *Journal of Experimental Psychology*, 13(4):533–544, 1987.

[55] J. J. Gibson. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1986.

[56] M. A. Goodale. Action without perception in human vision. *Cog Neuropsycho*, 25:891–919, 2008.

[57] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends Neurosci*, 15:20–25, 1992.

[58] J. L. Gould and C. G. Gould. *The Animal Mind*. Scientific American Library, 1994.

[59] A. G. Greenwald. Sensory feedback mechanisms in performance control: With special reference to the ideomotor mechanism. *Psychological Review*, 77:73–99, 1970.

[60] S. Griffith, J. Sinapov, M. Miller, and A. Stoytchev. Toward interactive learning of object categories by a robot: A case study with container and non-container objects. In *Proceedings of the 8th IEEE International Conference on Development and Learning (ICDL)*, pages 1–6, Shanghai, China, June 2009. IEEE.

[61] E. Guizzon. Autonomous vehicle driving from italy to china, blog in IEEE spectrum. `http://spectrum.ieee.org/automaton/robotics/robotics-software/autonomous-vehicle-driving-from-italy-to-china`, September 2010.

[62] E. Guizzon. Geminoid f gets job as robot actress, blog in IEEE spectrum. http://spectrum.ieee.org/automaton/robotics/humanoids/geminoid-f-takes-the-stage, November 2010.

[63] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision, Volume I*. Addison-Wesley, 1992.

[64] S. Harnad. The symbol grounding problem. *Physica D*, 42(1-2):335–346, 1990.

[65] S. Hart, R. Grupen, and D. Jensen. A relational representation for procedural task knowledge. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1280–1285. AAAI Press, 2005.

[66] B. Hommel. Acquisition and control of voluntary action, in voluntary action: Brains, minds, and sociality, 2003. In *Voluntary action: Brains, minds, and sociality*, pages 34–48. Oxford University Press, Oxford, UK, 2003.

[67] B. Hommel and B. Elsner. Acquisition, representation, anc control of action, in oxford handbook of human action, 2008. In *Oxford handbook of human action*, pages 371–398. Oxford University Press, New York, 2008.

[68] J. Hörnstein, L. Gustavsson, F. Lacerda, and J. Santos-Victor. Multimodal word learning from infant directed speech. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2748–2754, St. Louis, MO, USA, 2009. IEEE.

[69] C. Huang and T. Charman. Gradations of emulation learning in infants: imitation of actions on objects. *Journal of Experimental Psychology*, 92(3):276–302, 2005.

[70] Fraunhofer IAIS. KURT 3D. http://www.ais.fraunhofer.de/ARC/kurt3D/. Last visited on March 2009.

[71] Mesa Imaging. Sr 4000. http://www.mesa-imaging.ch/. Last visited on December 2010.

[72] D. Ingle and J. Cook. The effects of viewing distance upon size preference of frogs for prey. *Vision Research*, 17:1009–1019, 1977.

[73] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C Sullivan. The DARPA LAGR program: Goals, challenges, methodology, and phase I results. *Journal of Field Robotics*, 23(11–12):945 – 973, 2007.

[74] W. James. *The Principles of Psychology*. Dover Publications, New York, 1890.

[75] Y. Jiang and L. S. Mark. The effect of gap depth on the perception of whether a gap is crossable. *Perception and Psychophysics*, 56(6):691–700, 1994.

[76] K. S. Jones. What is an affordance? *Ecological Psychology*, 15(2):107–114, 2003.

[77] B. Jung and G. S. Sukhatme. Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Krose, editors, *Proceedings of the 8th International Conference on Intelligent Autonomous Systems*, pages 980–987, 2004.

[78] H. Kawasaki, T. Komatsu, and K. Uchiyama. Dexterous anthropomorphic robot hand with distributed tactile sensor: Gifu hand ii. *IEEE/ASME Transactions on Mechatronics*, 7(3):296–303, 2002.

[79] M. Kawato. Internal models for motor control and trajectory planning. *Current Opinion in Neurobiology*, 9:718–727, 1999.

[80] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. Bobick. Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 06)*, pages 303–309, Orlando, FL, May 2006.

[81] J. M. Kinsella-Shaw, B. Shaw, and M. T. Turvey. Perceiving walk-on-able slopes. *Ecological Psychology*, 4(4):223–239, 1992.

[82] K. Kira and L. A. Rendell. A practical approach to feature selection. In *Proceedings of the 9th international workshop on Machine Learning*, pages 249–256. Morgan Kaufmann Publishers Inc., 1992.

[83] V. Klingspor, Katharina Morik, and Anke D. Rieger. Learning concepts from sensor data of a mobile robot. *Machine Learning*, 23(2-3):305–332, 1996.

[84] W. Kohler. *The Mentality of Apes*. Harcourt Brace and World, New York, 1925.

[85] I. Kononenko. Estimating attributes: analysis and extensions of RELIEF. In *Proceedings of the European conference on machine learning on Machine Learning*, pages 171–182, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[86] D. Kraft, N. Pugeault, E. Başeski, M. Popovic, D. Kragic, S. Kalkan, F. Wörgötter, and N. Krüger. Birth of the object: Detection of objectness and extraction of object shape through object-action complexes. *International Journal of Humanoid Robotics*, 5(2):247–265, 2008.

[87] Y. Kuniyoshi and S. Sangawa. Early motor development from partially ordered neural-body dynamics: experiments with a cortico-spinal-musculo-skeletal model. *Biological Cybernetics*, 95(6):589–605, December 2006.

[88] F. Li and J. Kosecka. Probabilistic location recognition using reduced feature set. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3405–3410. IEEE, 2006.

[89] A. Lock and T. Collett. A toads devious approach to its prey: A study of some complex uses of depth vision. *Journal of Comparative Physiology*, 131:179–189, 1979.

[90] C. Lorken and J. Hertzberg. Grounding planning operators by affordances. In *Proceedings of the International Conference Cognitive Systems (CogSys 2008)*, Karlsruhe, Germany, April 2008.

[91] Dainichi Co. Ltd. Gifu Hand III. `http://www.kk-dainichi.co.jp/e/gifuhand.html`. Last visited on December 2010.

[92] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini. Developmental robotics: A survey. *Connection Science*, 15(4):151–190, December 2003.

[93] K. F. MacDorman. Responding to affordances: Learning and projecting a sensorimotor mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3253–3259, San Fransisco, California, USA, 2000.

[94] Machine Learning Group at University of Waikato. Weka 3: Data mining software in java. `http://www.cs.waikato.ac.nz/ml/weka/`. Last visited on December 2010.

[95] R. Marcilly and M. Luyat. The role of eye height in judgment of an affordance of passage under a barrier. *Current Psychology Letters: Behavior, Brain and Cognition*, 24(1), 2008.

[96] L. S. Mark. Eyeheight-scaled information about affordances: A study of sitting and stair climbing. *Journal of Experimental Psychology: Human Perception and Performance*, 13(3):361–370, 1987.

[97] MathWorks. Sequentialfs feature selection. `http://www.mathworks.com/access/helpdesk/help/toolbox/stats/sequentialfs.html`. Last visited on December 2010.

[98] A. N. Meltzoff. Understanding the intentions of others: re-enactment of intended acts by 18-month-old children. *Developmental Psychology*, 31:838–850, 1995.

[99] G. Metta and P. Fitzpatrick. Early integration of vision and manipulation. *Adaptive Behavior*, 11(2):109–128, 2003.

[100] J. Modayil and B. Kuipers. The initial development of object knowledge by a learning robot. *Robotics and Autonomous Systems*, 56(11):879–890, November 2008.

[101] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor. Learning object affordances: From sensory–motor maps to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[102] A. W. Moore and M. S. Lee. Efficient algorithms for minimizing cross validation error. In R. Greiner and D. Schuurmans, editors, *Proceedings of the 11th International Conference on Machine Learning*, pages 190–198. Morgan Kaufmann, 1994.

[103] A. Murata, V. Gallese, G. Luppino, M. Kaseda, and H. Sakata. Selectivity for the shape, size, and orientation of objects for grasping in neurons of monkey parietal are AIP. *Journal of Neuropyhsiology*, 83(5):2580–2601, 2000.

[104] R. R. Murphy. *An Introduction to AI Robotics*. The MIT Press, 2000.

[105] M. Myowa-Yamakoshi and T. Matsuzawa. Factors influencing imitation of manipulatory actions in chimpanzees (pan troglodytes). *Journal of comparative psychology*, 113(2):128–36, June 1999.

[106] Y. Nagai and K. J. Rohlfing. Can motionese tell infants and robots: What to imitate? In *Proceedings of the 4th International Symposium on Imitation in Animals and Artifacts*, pages 299–306. AISB, 2007.

[107] C. L. Nehaniv and K. Dautenhahn. Like me? Measures of correspondence and imitation. *Cybernetics and Systems*, 32:11–51, 2001.

[108] K. M. Newell, D. M. Scull, F. Tenenbaum, and S. Hardiman. Body scale and the development of prehension. *Developmental Psychobiology*, 22(1):1–13, 1989.

[109] S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, 2000.

[110] M. Ollis, W. H. Huang, M. Happold, and B. A. Stancil. Image-based path planning for outdoor mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2723–2728, California, 2008. IEEE.

[111] P-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, 2007.

[112] E. Oztop and M. A. Arbib. Schema design and implementation of the grasp-related mirror neuron system. *Biological Cybernetics*, 87:116–140, 2002.

[113] E. Oztop, N. S. Bradley, and M. A. Arbib. Infant grasp learning: A computational model. *Experimental Brain Research*, 158:1354–1361, 2004.

[114] E. Oztop, H. Imamizu, G. Cheng, and M. Kawato. A computational model of anterior intraparietal (AIP) neurons. *Neurocomputing*, 69:1354–1361, 2006.

[115] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the 17th International Conference on Machine Learning*, pages 727–734. Morgan Kaufmann, 2000.

[116] J. Peng and A. Peters. Extraction of salient features for mobile robot navigation via teleoperation. In *Proceedings of the 24th American Control Conference*, volume 7, pages 4903–4908. IEEE, 2005.

[117] R. Petrick, D. Kraft, K. Mourão, N. Pugeault, N. Krüger, and M. Steedman. Representation and integration: Combining robot control, high-level planning, and action learning. In *Proceedings of the 6th International Cognitive Robotics Workshop, 2008*, pages 32–41, July 2008.

[118] G. Pezzulo, G. Baldassarre, M. Butz, C. Castelfranchi, and J. Hoffmann. From actions to goals and vice-versa: Theoretical analysis and models of the ideomotor principle and tote, in anticipatory behavior in adaptive learning systems, 2007. In *Anticipatory Behavior in Adaptive Learning Systems*, pages 73–93. Springer Verlag, 2007.

[119] J. Piaget. *The Origins of Intelligence in Children*. International University Press, New York, USA, 1952.

[120] J. Piaget and B. Inhelder. *The Psychology of the Child*. Basic Books, New York, USA, 1966. Originally published in French as La Psychologie de l'enfant by Presses Universitaires de France, Paris, 1966.

[121] J. Pisokas and U. Nehmzow. Experiments in subsymbolic action planning with mobile robots. In *Adaptive Agents and Multi-Agent Systems II, Lecture Notes in Artificial Intelligence*, pages 80–87. Springer, 2005.

[122] J. Provasi, C. D. Dubon, and H. Bloch. Do 9- and 12-month-olds learn means-ends relation by observing? *Infant Behavior and Development*, 24(2):195–213, February 2001.

[123] L. Righetti and A. J. Ijspeert. Design methodologies for central pattern generators: an application to crawling humanoids. In G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, editors, *Proceedings of robotics: Science and systems*, pages 191–198. MIT Press, 2006.

[124] D. A. Rosenbaum. *Human Motor Control*. Academic Press, London, UK, 1991.

[125] H. Saltzman, M. Zacharatos, and E. R. Gruberg. Recognition of apertures in overhead transparent barriers by leopard frogs. *Brain, Behavior and Evolution*, 64(1):11–18, 2004.

[126] H. A. Sedgwick. *Visual Space Perception*, chapter 5. Wiley-Blackwell, Oxford, 2001.

[127] E. Shikata, Y. Tanaka, H. Nakamura, M. Taira, and H. Sakata. Selectivity of the parietal visual neurones in 3D orientation of surface of stereoscopic stimuli. *Neuroreport*, 7:2389–2394, 1996.

[128] M. Shneier, T. Chang, T. Hong, W. Shackleford, R. Bostelman, and J. S. Albus. Learning traversability models for autonomous mobile vehicles. *Autonomous Robots*, 24(1):69–86, 2008.

[129] J. Sinapov and A. Stoytchev. Detecting the functional similarities between tools using a hierarchical representation of outcomes. In *Proceedings of the 7th IEEE International Conference on Development and Learning*, pages 91–96. IEEE, August 2008.

[130] R. Smith. Open dynamics engine. `http://ode.org/`. Last visited on December 2010.

[131] M. Steedman. Plans, affordances, and combinatory grammar. *Linguistics and Philosophy*, 25, 2002.

[132] A. Stoytchev. Behavior-grounded representation of tool affordances. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 18–22, Barcelona, Spain, April 2005. IEEE.

[133] A. Stoytchev. Toward learning the binding affordances of objects: A behavior-grounded approach. In *Proceedings of AAAI Symposium on Developmental*, pages 17–22. AAAI, 2005.

[134] A. Stoytchev. Some basic principles of developmental robotics. *IEEE Transactions on Autonomous Mental Development*, 1(2):122–130, August 2009.

[135] R. Sun. Symbol grounding: A new look at an old idea. *Philosophical Psychology*, 13(149–172), 2000.

[136] A. Szokolszky. An interview with Eleanor Gibson. *Ecological Psychology*, 15(4):271–281, 2003.

[137] C. Tennie, J. Call, and M. Tomasello. Push or pull: Imitation vs. emulation in great apes and human children. *Ethology*, 112(12):1159–1169, 2006.

[138] M. Tomasello. *The Cultural Origins of Human Cognition*. Harvard University Press, Cambridge, 1999.

[139] E. Ugur and E. Şahin. Traversability: A case study for learning and perceiving affordances in robots. *Adaptive Behavior*, 18(3-4), 2010.

[140] E. Ugur, M. R. Doğar, M. Çakmak, and E. Şahin. Curiosity-driven learning of traversability affordance on a mobile robot. In *Proceedings of the IEEE International Conference on Development and Learning (ICDL 07)*, pages 13–18, London, UK, 2007.

[141] E. Ugur, M. R. Doğar, M. Çakmak, and E. Şahin. The learning and use of traversability affordance using range images on a mobile robot. In *Proceedings of ICRA'07*, pages 1721–1726, 2007.

[142] E. Ugur, E. Oztop, and E. Şahin. Goal emulation and planning in perceptual space using learned affordances. *Robotics and Autonomous Systems*, 2010. in revision.

[143] E. Ugur, E. Oztop, and E. Sahin. Learning object affordances for planing. In *Workshop on Approaches to Sensorimotor Learning on Humanoid Robots, International Conference on Robotics and Automation*, pages 38–39, 2009. Extended abstract.

[144] E. Ugur, E. Sahin, and E. Oztop. Affordance learning from range data for multi-step planning. In *Proceedings of the 9th International Conference on Epigenetic Robotics, EpiRob'09*, pages 177–184, Venice, Italy, 2009. Lund University Press.

[145] E. Ugur, E. Sahin, and E. Oztop. Predicting future object states using learned affordances. In *Proceedings of the 24th International Conference on Epigenetic Robotics, EpiRob'09*, pages 415–219, Guzelyurt, Northern Cyprus, 2009. IEEE Xplore.

[146] L. G. Ungerleider and M. Mishkin. Two cortical visual systems. In *Analysis of visual behavior*, pages 549–586. Cambridge MA: MIT Press, 1988.

[147] E. Uğur, M. R. Doğar, O. Soysal, M. Çakmak, and E. Şahin. MACSim: Physics-based simulation of the KURT3D robot platform for studying affordances. Technical report, METU, 2006. MACS Project Deliverable 1.2.1.

[148] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998.

[149] B. Vollmer and H. Forssberg. Development of grasping and object manipulation. In D.A. Nowak and J. Hermsdorfer, editors, *Sensorimotor Control of Grasping: Physiology and Pathophysiology*. Cambridge University Press, 2009.

[150] S. C. Vries, A. M. Kappers, and J. J. Koenderink. Influence of surface attitude and curvature scaling on discrimination of binocularly presented curved surfaces. *Vision Research*, 34:2409–2423, 1994.

[151] S. C. Want and P. L. Harris. How do children ape? Applying concepts from the study of non-human primates to the developmental study of imitation in children. *Developmental Science*, 5:1–13, 2002.

[152] W. H. Warren. Perceiving affordances: Visual guidance of stair climbing. *Journal of Experimental Psychology*, 105(5):683–703, 1984.

[153] W. H. Warren and S. Whang. Visual guidance of walking through apertures: Body-scaled information for affordances. *Journal of Experimental Psychology*, 13(3):371–383, 1987.

[154] W. H. Warren and S. Whang. Visual guidance of walking through apertures: body-scaled information for affordances. *Journal of Experimental Psychology*, 13(3):371–383, 1987.

[155] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291:599–600, 2001.

[156] R. White. Motivation reconsidered: The concept of competence. *Psychology Review*, 66:297–333, 1959.

[157] P. Willatts. The stage IV infant's solution of problems requiring the use of supports. *Infant Behavior and Development*, 7:125–134, 1984.

[158] P. Willatts. Development of means-end behavior in young infants: Pulling a support to retrieve a distant object. *Developmental Psychology*, 35:651–666, 1999.

[159] F. Wörgötter, A. Agostini, N. Krüger, N. Shylo, and B. Porr. Cognitive agents: A procedural perspective relying on the predictability of Object-Action-Complexes OACs. *Robotics and Autonomous Systems*, 57(4):420–432, April 2009.

[160] A. Yonas. Infants' response to optical information for collision. In Aslin R., J. Alberts, and M. Peterson, editors, *Development perception: Vol. 2, The visual system*. New York: of Academic, 1981.

[161] S. Zhang, X. Lihua, and M. D. Adams. Entropy based feature selection scheme for real time simultaneous localization and map building. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1175–1180. IEEE, 2005.

[162] J. Zlatev and C. Balkenius. Introduction: Why epigenetic robotics? In *Proceedings of the 1st International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 61–67. Lund University Cognitive Studies, 85., 2001.

VITA

P I

Surname, Name: Uğur, Emre

Date and Place of Birth: 12 September 1980, Ankara

Phone: (+81) 774 95 2403

Fax: (+81) 774 95 1236

E-mail: emre@atr.jp

Web: http://www.cns.atr.jp/˜emre

R I

Developmental and cognitive robotics

affordances, developmental robotics, sensory-motor learning, cognitive robotics, robot perception, swarm robotics

E

M.Sc., Dept. of Computer Engineering, METU, Turkey, Septembet 2006

- Thesis Title: Direct Perception of Traversability Affordance on Range Images Through Learning on a Mobile Robot
- Advisor: Erol Şahin
- CGPA: 3.79 / 4.00 (High Honor)

B.Sc., Dept. of Computer Engineering, METU, Turkey, June 2001

- CGPA: 3.46 / 4.00 (Honor)

A

- TUBITAK UBYT Paper Award (US $800), 2010

- TUBITAK Student Travel Grant (US $1000) for ICRA 2007
- TUBITAK UBYT Paper Award (US $275), 2006
- Middle East Technical University Honor Student Scholarship (covering tuition and a monthly stipend) (1998-2003)
- IsBank Award (1998)
- Ranked 13th among 1.5 million participants in the National University Entrance Exams (OYS) in Turkey (1998)
- Ranked 18th among 200 000 participants in the National School of Science Entrance Exam in Turkey (1995)

R

- Reviewer in IROS 2010, ISCIS 2009, ICDL 2010

E

Specialist **June 2009 -**

Advanced ICT Research Center, Biological ICT Group

Kobe National Institute of Information and Communication Technology (NICT), Kyoto, Japan

Intern Researcher **April 2009 -**

Computational Neuroscience Labs., Dept. of Communication and Cognitive Cybernetics

Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan

Research Assistant **September 2003 - April 2009**

Faculty of Engineering, Department of Computer Engineering, KOVAN Research Laboratory

Middle East Technical University, Ankara, Turkey

P

Journal Publications

E. Ugur, E. Oztop and E. Sahin, Goal emulation and planning in perceptual space using learned affordances, *Robotics and Autonomous Systems*, in revision.

E. Ugur and E. Sahin, Traversability: A case study for learning and perceiving affordances in robots, *Adaptive Behavior*, 18(3-4), pp. 258-284, 2010.

E. Sahin, M. Cakmak, M. R. Dogar, E. Ugur, and G. Ucoluk , To afford or not to afford: A new formalization of affordances towards affordance-based robot control, *Adaptive Behavior*, 15(4), pp. 447-472, 2007.

E. Sahin, S. Girgin, and E. Ugur , Area measurement of large closed regions with a mobile robot, *Autonomous Robots*, 21(3) No: 3, pp. 255-266, November 2006.

Conference Publications

E. Ugur, E. Sahin and E. Oztop, Affordance learning from range data for multi-step planning, *Ninth International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems (EpiRob)*, pp. 177-184, Venice, Italy, November, 2009.

E. Ugur, E. Sahin and E. Oztop, Predicting future object states using learned affordances, *International Symposium on Computer and Information Sciences (ISCIS 2009)*, Special Session on Cognitive Cybernetics and Brain Modeling, pp. 415-419, Northern Cyprus, September 2009.

M. R. Dogar, E. Ugur, M. Cakmak, and E. Sahin , Using Learned Affordances for Robotic Behavior Development, *IEEE Intl. Conf. on Robotics and Automation (ICRA 08)*, pp. 3802-3807, Pasadena, CA, USA, May 19-23, 2008.

M. Cakmak, M. R. Dogar, E. Ugur, and E. Sahin, Affordances as a Framework for Robot Control, *Seventh International Conference on Epigenetic Robotics*, Piscataway, NJ, USA, November 5-7, 2007.

E. Ugur, Ali E. Turgut, and E. Sahin, Dispersion of a swarm of robots based on realistic

wireless intensity signals, *22nd Intl. Symposium on Computer and Information Sciences (ISCIS'07)* , Ankara, Turkey, November 7-9, 2007.

E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin, Curiosity-driven Learning of Traversability Affordance on a Mobile Robot, *IEEE Intl. Conf. on Development and Learning (ICDL 07)*, pp. 13-18, London, UK., July 11-13 2007.

E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin, The learning and use of traversability affordance using range images on a mobile robot, *IEEE Intl. Conf. on Robotics and Automation (ICRA 07)*, pp. 1721-1726, Rome, Italy, April 10-14, 2007.

S

Action planning based on learned affordances, invited talk in *Motor Control: from Humans to Robots session, Motor Control Symposium*, Nagoya, Japan, 2010.

Learning affordances in mobile and manipulator robots, invited talk *at Dept. of Automatics, Biocybernetics, and Robotics*, Jozef Stefan Institute, Ljubljana, Slovenia, 2009.

The learning and use of traversability affordance on a mobile robot, *seminar at University of Edinburgh*, Institute for Perception, Action and Behavior, Edinburgh, UK, 2007.

B C

E. Rome, L. Paletta, E. Sahin, G. Dorffner, J. Hertzberg, G. Fritz, J. Irran, F. Kintzler, C. Lorken, S. May, E. Ugur, R. Breithaupt, The MACS project: An approach to affordance-based robot control Towards Affordance-based Robot Control, *Proceedings of Dagstuhl Seminar* 06231, Springer-Verlag, Berlin, pp. 173-210, Rome, E., Hertzberg, J. and Dorffner, G. (eds.), February 2008.

W P

B. Moore, E. Ugur, and E. Oztop, Biologically inspired robot grasping through human-in-the-loop robot control, *IROS 2010 Workshop on grasp planning and task learning by imitation*, 2010, Taiwan.

E. Ugur, E. Oztop, and Erol Sahin, Discovering action-oriented object meanings in an anthropomorphic robot platform, *Neuro 2010*, 2010, Kobe.

E. Ugur, E. Sahin, and E. Oztop, Use of range cameras for the perception of push and grasp affordances, *Computer Vision for Humanoid Robots in Real Environments Workshop, ICCV*, 2009, Kyoto.

E. Ugur, E. Oztop, and E. Sahin, Learning Affordance Relations in a Mobile Robot with Limited Manipulation Capabilities, *The 32st Annual Meeting of the Japan Neuroscience Society*, Nagoya, 2009.

E. Ugur, E. Oztop, and E. Sahin, Learning object affordances for planning, Workshop on Approaches to Sensorimotor Learning on Humanoid Robots, *International Conference on Robotics and Automation (ICRA 2009)*, pp. 38-39, Kobe, Japan, May, 2009. (Extended abstract)

T R

E. Sahin, M. Cakmak, M. R. Dogar, E. Ugur, and G. Ucoluk, To afford or not to afford: A new formalization of affordances towards affordance-based robot control, KOVAN Research Lab., Department of Computer Engineering, Middle East Technical University, Ankara, Turkey, 2006.

E. Sahin, E. Ugur and M. Cakmak, Evaluation of existing control architectures for using affordances, MACS Technical Report *MACS/2/2.1*, KOVAN Research Lab., Department of Computer Engineering, Middle East Technical University, Ankara, 2006.

E. Rome, E. Sahin, R. Breithaupt, J. Irran, F. Kintzler, L. Paletta, M. Cakmak, E. Ugur, G. Ucoluk, M. R. Dogar, P. Rudol, G. Fritz, G. Dorffner, P. Doherty, M. Wzorek, H. Surmann, C. Lorken, Development of an Affordance-based Control Architecture, MACS Technical Report *MACS/2/2.2*, Fraunhofer AIS, Sankt Augustin, 2006.

E. Ugur, Mehmet Remzi Dogar, Onur Soysal, M. Cakmak and E. Sahin, MACSim: Physics-based Simulation of the KURT3D Robot Platform for Studying Affordances, MACS Technical Report *MACS/1/2.1*, KOVAN Research Lab., Department of Computer Engineering, Middle East Technical University, Ankara, 2006.

E. Ugur, Mehmet Remzi Dogar, Onur Soysal, M. Cakmak, Ralph Breithaupt and E. Sahin, Modeling of the final demonstrator scenario in a physics-based simulator, MACS Technical Report *MACS/6/6.2*, KOVAN Research Lab., Department of Computer Engineering, Middle East Technical University, Ankara, 2006.

L S

Turkish/native
English/proficient in reading, writing, understanding, speaking
Japanese/beginner in reading including Kanji, understanding, speaking

P I

Hiking, climbing, reading