PERCEPTION, LEARNING AND USE OF TOOL AFFORDANCES ON HUMANOID ROBOTS

A THESIS
SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Yiğit Çalışkan
May, 2013

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Pınar Duygulu Şahin(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Erol Şahin(Co-Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Sinan Kalkan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Öznur Taştan

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

PERCEPTION, LEARNING AND USE OF TOOL AFFORDANCES ON HUMANOID ROBOTS

Yiğit Çalışkan
M.S. in Computer Engineering
Supervisor: Assist. Prof. Dr. Pınar Duygulu Şahin
Co-Supervisor: Assoc. Prof. Dr. Erol Şahin
May, 2013

Humans and some animals use different tools for different aims such as extending reach, amplifying mechanical force, create or augment signal value of social display, camouflage, bodily comfort and effective control of fluids. In robotics, tools are mostly used for extending the reach area of a robot. For this aim, the question "What kind of tool is better in which situation?" is very significant. The importance of affordance concept rises with this question. That is because, different tools afford variety of capabilities depending on target objects. Towards the aim of learning tool affordances, robots should experience effects by applying behaviors on different objects.

In this study, our goal is to teach the humanoid robot iCub, the affordances of tools by applying different behaviors on a variety of objects and observing the effects of these interactions. Using eye camera and Kinect, tool and object features are obtained for each interaction to construct the training data. Success of a behavior depends on the tool features, object position and properties and also the hand that the robot uses the tool with. As a result of the training of each behavior, the robot successfully predicts effects of different behaviors and infers the affordances when a tool is given and an object is shown. When an affordance is requested, the robot can apply the appropriate behavior given a tool and an object, the robot can select the best tool among different tools when a specific affordance is requested and an object is shown. This study also demonstrates how different positions and properties of objects affect the affordance and behavior results, and how affordance and behavior results are affected when a part of a tool is removed, modified or a new part is added.

Keywords: Affordance, Tool Affordance, Humanoid Robot, Tool Use.

ÖZET

İNSANSI ROBOTLARDA ALET SAĞLARLIĞI KAVRAMININ KULLANIMI, ALGILANMASI VE ÖĞRENİLMESİ

Yiğit Çalışkan
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Assist. Prof. Dr. Pınar Duygulu Şahin
Ortak Tez Yöneticisi: Assoc. Prof. Dr. Erol Şahin
Mayıs, 2013

İnsanlar ve bazı hayvanlar aletleri erişim alanını genişletmek, mekanik kuvvetlerini arttırmak, toplumsal değerini oluşturmak ve arttırmak, kamuflaj, vücudu rahatlatma ve sıvı kontrolleri gibi amaçlar için kullanırlar. Robotik alanında alet kullanımı genelde robotun erişim alanını arttırmak için kullanılır. Bu amaç doğrultusunda "Hangi alet hangi duruma uygundur?" sorusu çok önemlidir. Sağlarlık kavramının önemi bu soru ile ortaya çıkmaktadır. Çünkü farklı aletler, hedef nesneler üzerinde farklı kabiliyetlere sahip olabilirler. Alet sağlarlığının öğrenimi için robotlar farklı davranışları farklı nesneler üzerinde deneyerek ortaya çıkan sonuçları gözlemlemelidirler.

Bu çalışmadaki amaç, insansı robot iCub'a farklı davranışları nesneler üzerinde uygulatıp, çıkan sonuçları gözlemleterek alet sağlarlıklarını öğretmektir. Göz kamerası ve Kinect kullanarak, eğitim verisi oluşturmak için, her etkileşimde alet ve nesne nitelikleri elde edilir. Bir davranışın başarısı alet niteliklerine, nesnenin pozisyon ve özelliklerine ve robotun kullandığı ele bağlıdır. Davranış eğitimlerinin ardından, verilen bir alet ve nesneye göre, robot farklı davranışların sonuçlarını tahmin edip, sağlarlıkları çıkarabilmektedir. Herhangi bir sağlarlık istendiğinde, robot kendisine verilen alet ve nesneye göre uygun davranışı uygulayabilmektedir, herhangi bir nesne gösterildiğinde robot farklı aletler arasından en uygun aleti seçebilmektedir. Bu çalışma ayrıca nesnelerin farklı pozisyonları ve özelliklerinin sağlarlık ve davranış sonuçlarını nasıl etkilediğini ve bir aletin herhangi bir parçası çıkarıldığında, değiştirildiğinde veya yeni bir parça eklendiğinde sağlarlıkların ve davranış sonuçlarının nasıl etkilendiğini göstermektedir.

Anahtar sözcükler: Sağlarlık, Alet Sağlarlığı, İnsansı Robot, Alet Kullanımı.

Acknowledgement

First of all, I would like to thank to my advisor Pınar Duygulu Şahin and my co-advisor Erol Şahin for giving me a chance to work with them by providing a great work place full of brilliant people. Thank you for supporting and guiding me in this process. I also want to thank Sinan Kalkan for his guidance and ideas in hard times.

I would like to thank Hande for always being open for help even she has lots of work to do, she really helped me a lot in the last stage of my thesis. Thanks to Güner for always making me crazy while working, this work would finish earlier if Güner wouldn't be there. It would finish earlier but in a worse quality without the countless helps of Güner. Thanks for all your help. Thanks to Sertaç for always creating trouble between me and Güner. Good luck to you with Güner after I leave the laboratory. Thanks for your support and listening my complaints anytime. (Yes, you will be able to take my computer soon). Thanks to Fatih for being a good companion at nights, I am sure that you will be one of the best faculty members in SDÜ. Thanks to Asil for being a great companion. Thanks to Kadir for the helps in urgent situations. You all always supported me and you were great friends for me. I will miss working with you all. And thanks to all other Kovan members; Doruk, İlkay, Gaye, Akif, Buğra,... I also would like to thank Mustafa, my frisbee partner and a great friend but now in India. It would be better if you would not leave from Turkey. Hope you find what you want in there. I want to thank to all my other friends who supported me, especially Saygın.

My biggest thanks are to my family. They always supported me in this stage of my life. I know that they will continue their support after this stage. I owe them a lot. I could not do this without the sandwiches that my mom prepared for the dinners at the laboratory.

I would like to acknowledge the support of TÜBİTAK (The Scientific and Technological Research Council of Turkey) and this thesis is supported by TÜBİTAK under the project with id 109E033.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Merriam dictionary defines a tool as a handheld device that aids in accomplishing a task[1]. Tool use requires intelligence since it involves knowledge of tools, manufacturing ability and planning. For instance, when a human wants to pull back an object, he should know that he needs a T-shaped tool or a tool with hook, etcetera. With the absence of these types of tools, a human can manufacture tools by combining smaller parts. When he gets the tool, he needs to plan his movements to pull back the object with the tool. Therefore, humans and only some of the animals can use tools. A tool can be used in many situations for different aims. For instance, when a hammer is considered, depending on a situation it can be used to push something, crack something by hitting on it, etcetera. Although tool use is not important for industrial robots due to their automated jobs, it is very significant for intelligent robots which are gradually starting to appear in our daily life. Since these robots will interact with humans and the environment, they need to be experienced about "Which effects can rise when a specific action is applied?" and also "What kind of effects can rise in which situation?". These type of questions reveal the importance of affordance concept in robotics because, a robot with the knowledge of affordances will be able to deal with an unexpected situation in daily life. Related to the current study, combination of tool use and the affordance concept introduces "the tool

[1] Definition was taken from http://www.merriam-webster.com/

affordance" concept. In the next sections, tool use of humans and animals will be examined with detail and then affordance and tool affordance concept will be explained. At the end of the chapter, contribution of this study will be presented.

1.1 Tool Use in Animals

Most animals use different objects or their own limbs in order to achieve their goals. Few animals use tools to reach food, to hunt the prey, to scare the human etcetera.

In his book [2], Beck talks about different behaviors that are directly or indirectly related with tool use and propose the following definition for tool use:

Definition 1. *The external employment of an unattached or manipulable attached environmental object to alter more efficiently the form, position, or condition of another object, another organism, or the user itself, when the user holds and directly manipulates the tool during or prior to use and is responsible for the proper and effective orientation of the tool.*

In his book, Beck categorized tools based on their uses such as dropping, throwing, digging, rolling, kicking, reaching, cutting,... Each of these different modes of tool use produces different effects and extends the user's abilities by extending his reach, amplifying mechanical force, creating or augmenting signal value of social display, camouflage, bodily comfort and effective control of fluids.

Beck listed different modes of manufacturing tools which are detach, subtract, combine, add and reshape. St. Amant and Horton made an experiment about manufacturing tools on chimpanzees where chimpanzees combine two pieces of pipes together to reach a reward [3]. In their study, Bentley and Smith stated each of the manufacturing modes that Beck listed requires an active art of creation as opposed to mere object acquisition [4].

1.2 Tool Use in Early Stages of Humans

In humans, tool use ability is acquired through a developmental process and is a topic of active research. Guerin *et al.* proposed a three stage development [1] in first two years of infancy as shown in Figure 1.1 while Piaget proposed this development in six stages [5] and Fischer in four stages [6]. The first one is "behaviors with objects" stage where innate behaviors develop such as trying to grasp or reach, the second one is "behaviors with a single object" stage where interaction with single objects starts and lastly "object-object behaviors" stage where relationship between objects starts to be learned.

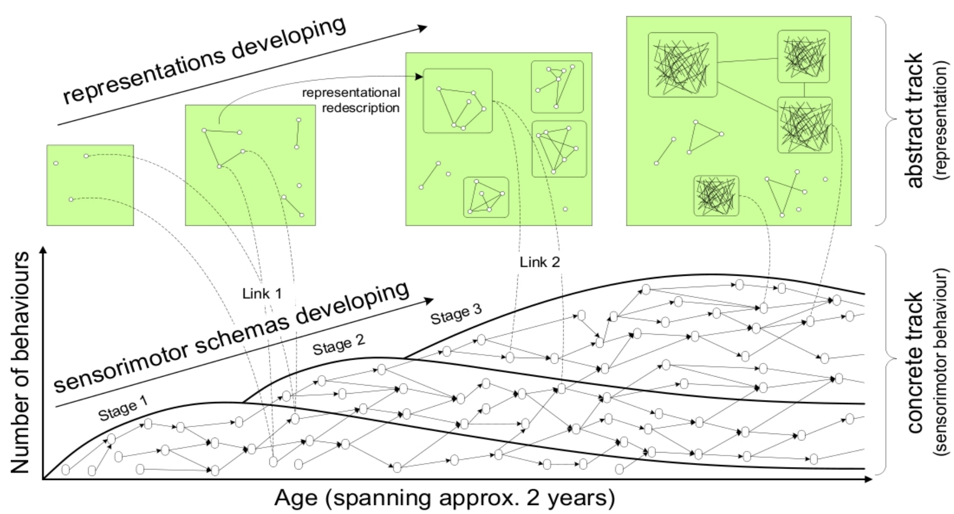


Figure 1.1: Developmental Stages of Tool Use (Taken from Guerin *et al.* [1])

The development takes place along two; namely concrete track and abstract track. The concrete track shows the development of sensorimotor schemas as observed from infant's behavior. A sensorimotor schema is a psychological term that gathers the perceptions and actions of a behavior in infant's repertoire. The abstract track shows the underlying representation that the infant uses. In the concrete track, each node represents a sensorimotor schema in other words an observable behavior. Each directed edges between nodes means "is a necessary precursor".

The concrete track is split into three consecutive overlapping stages as:

- **Behaviors Without Objects (Stage 1):** In this stage, innate behaviors develop that are assumed to calibrate vision and motor system leading to grasping ability. These behaviors act as precursor for reaching and manipulating objects. For instance, an infant may try to grasp an object without knowing he can succeed or not.

- **Behaviors with Single Objects (Stage 2):** In this stage, the infant starts interacting with single objects through repetitions, relationship between sensorimotor schemas are discovered step by step and effects of these actions become more predictable.

  In the abstract track, object and affordance representations start to be constructed. Generalization starts depending on the experiences such as to predict releasing a grasped object from high will result with a drop on the ground.

- **Object-Object Behaviors (Stage 3):** In this stage, sensorimotor schemas start to deal with the relationship between objects, representations of spatial locations and transforms start to be constructed in the abstract track. Simple planning indications can be seen in this stage since relationships between objects are started to be discovered. For example, having the awareness of pushing forward a short and long vertical boxes from top results with different effects.

As Guerin *et al.* indicated, development does not end at age two and continues in both tracks.

1.3 Affordances

The notion of affordances is introduced by J.J.Gibson to denote potential actions offered by an object to an organism [7]. As it can be seen from the Figure 1.2,

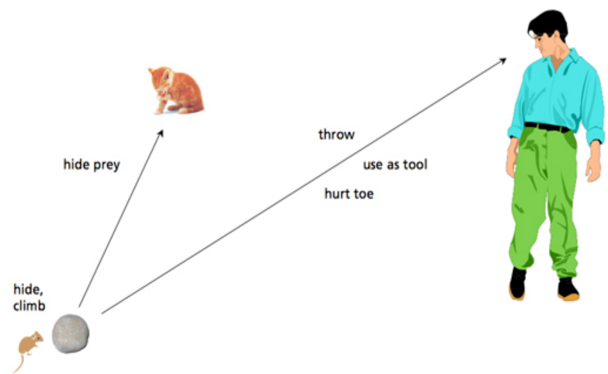


Figure 1.2: Affordances between different subjects (Image is taken from http://www.macs-eu.org/images/affordance-animals.jpg)

there are three subjects and between each subject, there are different affordances. For example, for the mouse, the rock is climbable, for the human, the same rock is a throwable object. However, if this rock is big and heavy, for the mouse this rock would not be climbable and for the human it would not be a throwable object. So, it can be inferred that affordances also depend on the different properties of an object.

By the time, some new ideas came up about affordance concept, one of them is the formalization of Sahin *et al.* about affordances. Sahin *et al.* offers the following formalization of affordances [8]:

(effect, (entity, behavior))

In this formalization, "entity" term is used as environmental member of the affordances, "behavior" term is used as fundamental perception-action control unit that is used to create physical interactions with the environment and lastly "effect" term is used as the resultant of the "behavior" that is applied on the "entity". Visualization of this formalization can be seen in Figure 1.3:

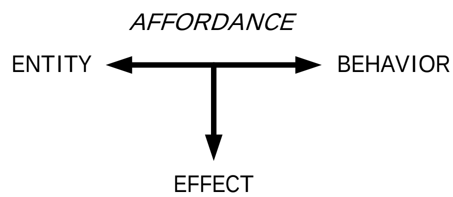


Figure 1.3: Visualization of Affordance Formalization

Agents discover different affordances by interacting with their environment. These interactions result with different effects. Repetition of these interactions leads learning of affordances. An example can be given as following: After many successful push behavior on a ball, an agent says that "The ball is pushable" after seeing the ball. Formal definition [8] can be seen in the following:

Definition 2. *An affordance is an acquired relation between a certain effect and an (entity, behavior) tuple, such that when the agent applies the behavior on the entity, the effect is generated.*

This formalization will be used to explain and build the tool affordance concept.

1.4 Tool Affordances in Robots

Beck categorized tool use into different modes where each mode grants user different properties such as extending reach, amplifying mechanical force, create or augment signal value of social display, camouflage, bodily comfort and effective control of fluids.

In robotics, tool use is mostly used for extending the reach area of the robot. This time, another question arises which is "What kind of tool is better in which situation?". The importance of affordance concept rises with this question. Affordance formalization can be adjusted to tool affordance as following:

$$\text{(effect, ((tool, object), behavior))}$$

As it can be seen from the formalization, the only difference is the addition of the tool which is a member of the environment. Using behaviors with different tools may result with different effects on different objects. For instance, trying to pull back an object with a stick or with a T-shaped tool results with different effects as in Figure 1.4 due to the existence of a tool part which helps one of the tools to pull back.

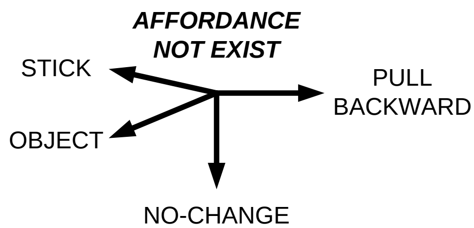


Figure 1.4: Pull Back an Object

Or trying to push forward a far object with a short stick results differently than pushing it using a long stick due to distance of the object as in Figure 1.5.

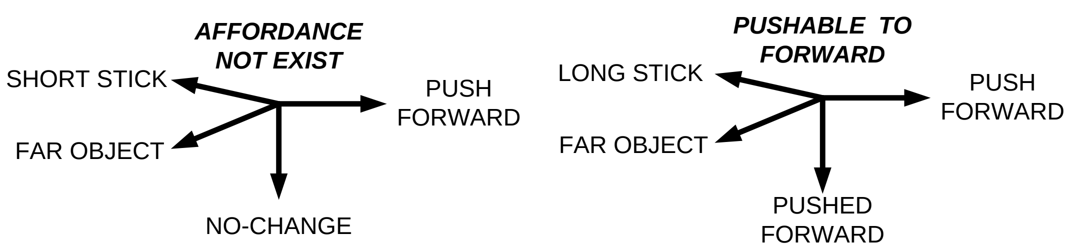


Figure 1.5: Push Forward a Far Object

Therefore, different tools afford variety of capabilities depending on target objects. Towards the aim of learning tool affordances, robots should experience effects by applying behaviors on different objects. By doing this, robot will be able to learn which tool affords what kind of effects when it is applied on different objects.

1.5 Contribution of the Thesis

Our aim is to teach humanoid robot, the affordances of tools by applying different behaviors on different objects in order to learn the different effects of different interactions. From these interactions, using the affordance formalization of Sahin *et al.* which is previously explained in affordances section, the robot will be able to predict the affordances of different tools with respect to a given object.

In this study, robot used different tools to extend his reach and it knows 11 different behaviors which can result with 5 different effects and that can reveal 4 different affordances. These behaviors are grounded in robot's repertoire to apply a requested behavior using a tool on an object.

This study showed that which parts of the tools are important for which behaviors, how an absent or a newly added part of a tool affects affordances and behavior effects. A new method was proposed to reveal important feature combinations specific to behaviors by combining tool's, object's size and object's position features. It was shown how affordances and behavior results are affected based on object's size, position changes and tool features. Lastly, it was shown that two similar/different tools by appearance, may be different/similar when affordances or behavior effects are considered.

Chapter 2

Related Studies

Related studies can be categorized into three. First one is the studies which examine tool use. These studies are mostly psychological studies. The second one is the studies which are about functionalities of the hand tools. Lastly, the studies which are about tool affordances, some of these studies use robots to illustrate their study. The last category includes the most similar studies to this study.

2.1 Studies on Tool Use

Pellicano *et al.* [9] suggested a model to explain the mechanism underlying choice of the most appropriate tool for a given goal. In this mechanism, authors claimed that there are stable affordances of each tool, for example a knife cuts, a stirrer stirs, etcetera. In addition to these stable affordances, there are also variable affordances which are the temporary characteristics of tools. In their model, if a canonical tool is not available for given goal, then these temporary characteristics of tools are activated for example stirring using a knife. According to their view, if this temporary tool is used repetitively for the given goal due to the absence of canonical tools, then this action-tool relation becomes a stable affordance but never gets the first priority of canonical tools.

Costantini *et al.* [10] explored the effects of active tool-use and tool-use observation on representation of reaching space. Six experiments were done on 150 participants 25 for each experiment. To examine the reach space of participants, in some of the experiments, participants were allowed to use the tools on the object and in some of them, they were only allowed to observe someone who was using the tool. As a result of the experiments, authors concluded that both active tool-use and also observation of tool-use shape the way of the individuals' mapping on objects.

Witt *et al.* [11] examined how a tool affects the perceived distance. Three different experiments were done each with different aims. In all experiments, distance of the target object was fixed and reachability distance varied. In some experiments participants were allowed to use tools and in experiments it was requested from participants to make verbal and visual judgments for the distance of the target object. As conclusion, authors argued that perception of an individual extends with a tool if he/she intends to use the tool otherwise it does not affect the range of perception.

2.2 Studies on Functionalities of the Tools

Shinchi *et al.* [12], discussed a computational model for object concept in their study and use hand tools for this aim. Then, they used the relationship between shape and function with a Bayesian network model, these hand tools were predicted. Like the previous work, Nakamura and Nagai [13] formed an object concept model again with Variational Bayes method [14] but in addition to the previous work, authors included grasping detection, contact areas and hand shape in the system and also they increased the number of functions. As a result of these studies, they were able to infer functionalities and usage of the objects based on only appearance.

Sinapov and Stoytchev [15], tried to find the similarities of the tools according to their functionalities. They experimented with six different tools and according to changes in environment that a tool produce, their robot learned models for each tool. At the end, in addition to the prediction of tool types, robot also found similarities between these six tools.

2.3 Studies on Tool Affordances

Caliskan *et al.* [16] tried to predict the affordances of hand tools by using interactive perception. Towards this end, they found the functional regions of the tools depending on the number of joints and then they extracted features from these functional regions. At the end, they classified tools into four affordance class as "can cut", "can push", "can pierce" and "can compress" by training models for each. This study lacks of object-tool relationship so predicted affordances may not be true according to the object that the tool encounters.

Sinapov and Stoytchev [17] described an approach for tool affordances in which a robot experimented randomly with its environment with the tools and learned which tool action affects the target object's position by how. Experiments were done in a simulator environment with six different tools, k-NN and decision trees were used for model creation. At the end, for the aim of finding affordances, they concluded that most predictive results were seen when the novel tools share similar local features with the tools which were previously seen.

Stoytchev [18] experimented with a mobile manipulator with five tools and hockey puck as target object to build a representation for tool affordance. This can be said as the most similar study to our study which will be proposed in this thesis. In this study, robot manipulator did different exploratory behaviors using the given tool on the puck, with these behaviors, position of the puck changes. Features were extracted using a camera from tool and puck and using these features from many trials, a representation was learned. Testing was done to test the quality of the representation and to see when the tool is deformed whether the system compensates this or not. As a result, they successfully manage to build a representation for tool affordances.

In our study, the main difference from the previous studies is the variety of object's properties and object's locations in addition to different tool features. This variety causes different tool affordances in many situations. Our study also has a variety of behaviors, some of them has the same effect but with different application style. This enabled the robot to try more than one behavior to succeed a requested effect.

Chapter 3

Experimental Setup

We have studied the learning and use of tool affordances on the iCub humanoid robot with 53 degrees of freedom as can be seen in Figure 3.1.

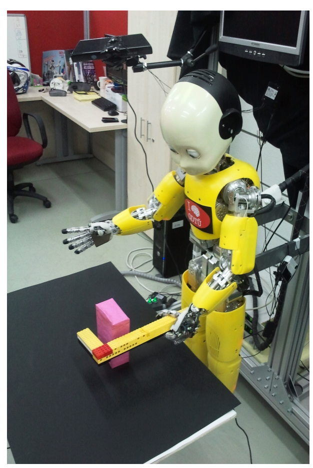


Figure 3.1: The iCub Humanoid Robot

iCub's head has 6 degrees of freedom, 3 in the neck and 3 for the eyes. Using the head and the eyes, iCub can look at the tool and process its image to extract the features.

iCub's torso/waist and arms are used to reach a tool and apply the behaviors. In each arm, there are 7 degrees of freedom and in his torso/waist there are total of 3 degrees of freedom.

iCub used its hands to grasp a tool and interact with an object. There are 9 degrees of freedom in each hand, these 9 dof are separated as 3 for the thumb, 2 for the index, 2 for the middle finger, 1 for the coupled ring and little finger, 1 for the adduction.

For the aim of grasping, iCub has to understand whether his hand contacted the tool or not. If his hand contacted then he closes his fingers. For this contact detection and grasping, tactile sensors are used which are placed on the palm of the hands and finger-tips as it can be seen in Figure 3.2.

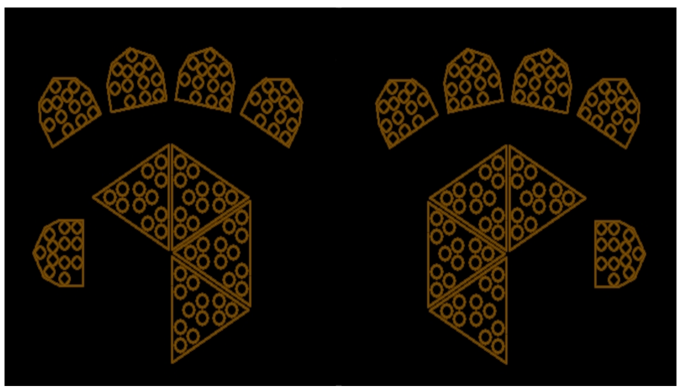


Figure 3.2: Sensors in the Palm

The sensors as shown in Figure 3.2, can have different settings such as binary mode-on/off, or their sensitivities can be changed.

3.1 Reference Frame of the Robot

Reference frame of robot, is the base frame of all related devices. 3D processing devices and kinematics of iCub take this frame as reference. Coordinate frame of the Kinects were transformed to this reference frame for simplifying and having common coordinates for 3D related works. This reference frame is located in the middle of the robot as it can be seen in Figure 3.3:

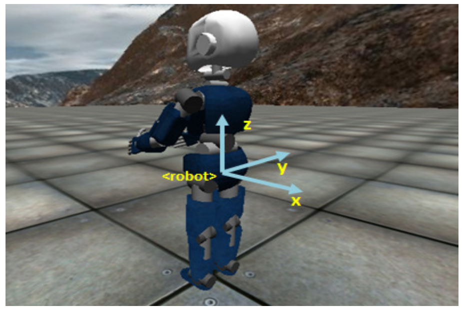


Figure 3.3: Reference Frame of the iCub

3.2 Available Space for Hands of iCub

In Figure 3.4, the table which iCub will interact with objects on can be seen. Available areas that iCub can move his hands inside can be seen with boundaries. The commands that move iCub's hands to outside this region are not allowed due to security reasons. This area is just shown for X and Y-directions, in Z-direction iCub has 28cm height to move his hands free.

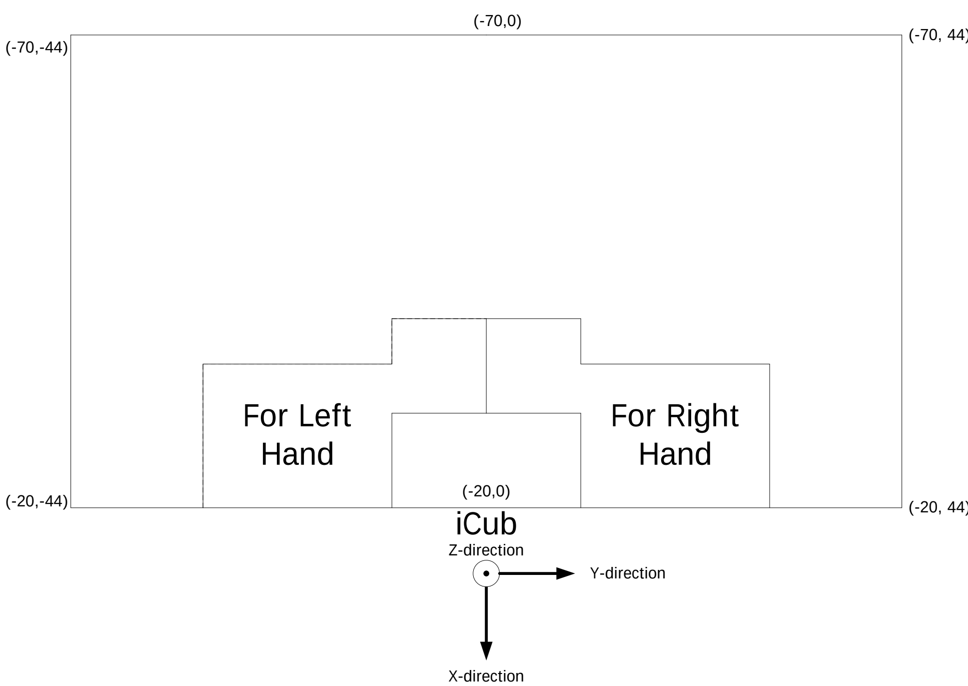


Figure 3.4: Available Space for Both Left and Right Hands on the Table

3.3 Perception Hardware

3.3.1 Visualeyez 3D Motion Tracking System

Visualeyez Motion Capture System is based on active optical technology which can track 3D motions with markers attached to the subject and is shown in Figure 3.5. It has support up to 512 markers with IDs with high accuracy and zero ID errors. In this work, this device was used for transformation of devices to iCub's reference frame.

Figure 3.5: Visualeyez 3D Motion Tracking System[1]

3.3.2 iCub Eye Camera

iCub has RGB cameras in his eyes and can get 640×480 resolution images with 30fps. Camera was calibrated in order to eliminate fish-eye effects and find intrinsic parameters. We used the images obtained from eyes to perceive the features of the tools.

[1]http://www.ptiphoenix.com/

3.3.3 Kinect RGB-D Camera

Kinect, an RGB-D camera, shown in Figure 3.6, can give 640×480 resolution images at 30Hz.

Figure 3.6: Kinect RGB-D Camera[2]

We used two Kinects to sense the tools and objects by processing point clouds. Since each Kinect has its own reference frame, this reference frame was transformed into robot's reference frame to have a common reference frame with iCub's kinematics. Transformation was done using markers of the Visualeyez placed on Kinect. The pose and the coordinate point of iCub's reference frame is known and using the markers from visualeyez, pose and the coordinate point of Kinect is also known in terms of iCub's reference frame. After this process, rotation and translation matrices are computed to make the transformation.

After each Kinect was transformed into iCub's reference frame, point clouds of each Kinect were combined in order to cover all sides of the tools and objects.

3.4 Construction of Tools

The tools are built of different colours of LEGO bricks. The handles of the tools are wrapped with a padding in order to ease grasping. In order to simplify grasping and segmentation of the tools from the table, a stand is prepared to

[2]http://www.xbox.com/en-US/kinect

put the tool on. A tool consists of a main part(at least a holding region), left part(optional) and right part(optional).

3.5 Software Development Tools, Platforms and Libraries

We used the following software tools and platforms to implement the perception learning and use of tools.

- **Yet Another Robot Platform(YARP),** a free and open software that consists of libraries, protocols and tools that keeps modules and also devices decoupled [19]. In this work, YARP middleware was used in order to control iCub's devices and also to maintain communication with different modules.
- **Open Source Computer Vision Library(OpenCV),** a library of functions for processing 2D images, was used for processing the image from iCub's camera to compute the features of the tools [20].
- **Point Cloud Library(PCL),** an open project for 2D/3D image/point cloud processing, includes algorithms for filtering, registration, feature estimation, reconstruction, segmentation and more [21]. In this work, PCL was used for 3D point cloud processing of tools and the objects.
- **Weka Data Mining Software,** a software tool consists of many data mining algorithms [22]. In this study, Weka was used for feature selection algorithms to reveal important tool and objects features.
- **Support Vector Machines(LibSVM),** supervised learning models with learning algorithms which is introduced Vapnik [23]. These learning models with associated learning algorithms help to analyze and recognize new input data. The main idea of SVM is creating a model that separates a set of points with a gap which is as wide as possible. When a newly seen datum is given as input to system, this input is mapped to the related point space

according to the model and depending on which side of the gap this mapped point is on, the input is labeled to that class.

Assume we have a training set as given below;

$$S = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m) | (x_i, y_i) \in R^n \times \{+1, -1\}\} \tag{3.1}$$

The aim is to find a hyperplane as in the following form;

$$w \cdot x + b = 0 \tag{3.2}$$

to separate the instances where x is training samples, w is the coefficient and b is the constant. Vapnik showed that the optimal hyperplane is the one which has the maximum $d_- + d_+$ where d_- denotes the shortest distance to hyperplane from a negative instance and d_+ denotes the shortest distance to hyperplane from a positive instance. Using this fact, w and b are found by solving optimization problems. After this optimization problem, it turns out w can be expressed with some of the training samples as given below:

$$w = \sum_i y_i \alpha_i x_i \tag{3.3}$$

x_i's are called are called *support vectors* and they lie on the margins of the shortest instances. At the end of the SVM training algorithm α_is and x_is are known. Using these, it is possible to classify an instance using:

$$class(x) = sgn(\sum_i \alpha_i \langle x, x_i \rangle + b) \tag{3.4}$$

Sometimes, training examples are not linearly separable, then this situation is solved using a mapping,

$$\phi : R^n \to F \tag{3.5}$$

where in feature space F instances are linearly separable.

To classify an instance, lots of dot product calculations need to be done. In order to prevent expensive calculation kernel trick is used as given:

$$k(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \tag{3.6}$$

In this work, we used LibSVM, an integrated software for support vector machines that supports multi-class classification [24]. In LibSVM, there variety of SVM types and kernel types, and also other parameters related to kernel and SVM type. To achieve a model that classifies well, user should find the good parameters.

Chapter 4

Perception

4.1 Tool Perception

The tools are perceived using visual images obtained from iCub cameras and point clouds obtained from Kinects. Tool features consist the features extracted from visual image and the features extracted from point clouds.

4.1.1 Tool Perception From Point Clouds

Tools are processed using combined point cloud from Kinects by applying the steps in the following Figure 4.1. At the end of these steps, tool features from Kinect RGB-D camera are obtained.

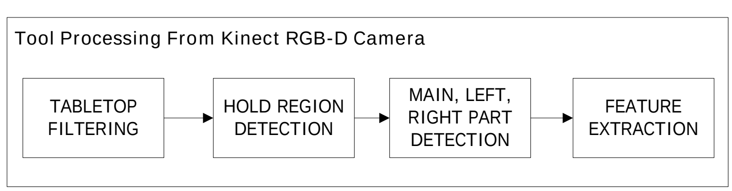
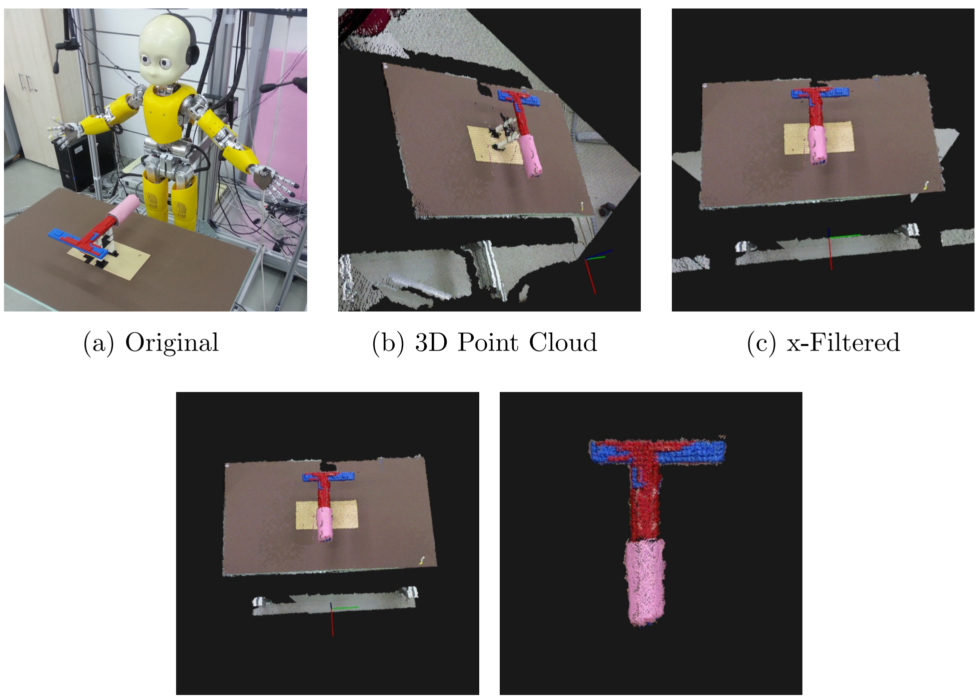


Figure 4.1: Tool Perception Steps Using Point Cloud From Kinects

- **Tabletop Filtering:** Using filters, point cloud of a tool is obtained by eliminating the points that belong table, environment, etcetera. The perceptual processing steps of a tool is shown in Figure 4.2. Figure 4.2(a) shows the sample original 3D world in front of the iCub. Figure 4.2(b) shows the sample point cloud of this 3D world from different perspective. Using passthrough filter, point cloud is filtered in x-dimension to capture the table in x dimension as it is shown in Figure 4.2(c). Then, point cloud is filtered in y and z dimensions in which samples are shown in Figure 4.2(d) and Figure 4.2(e) respectively.



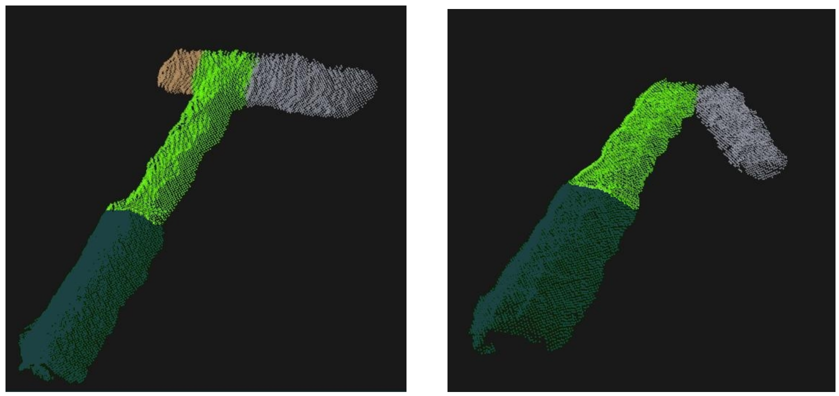
(a) Original (b) 3D Point Cloud (c) x-Filtered

(d) y-Filtered (e) z-Filtered

Figure 4.2: First Steps of the Tool Processing with Kinect

At the end of these perceptual processing steps, point cloud of a tool is obtained as in Figure 4.2(e).

- **Hold Region Detection:** Using point cloud of the tool, grasping region of the tool is determined. As discussed in the previous chapter, grasping is a challenging behavior for iCub and we assumed that each tool has a handle of 4cm×4cm×13cm built from soft padding. Sample hold regions can be seen in Figure 4.3(a) and Figure 4.3(b) with dark green colour. Target point for grasping is obtained by averaging points in hold region.



(a) (b)

Figure 4.3: Sample Hold Regions and Parts of the Tools

- **Detection of Tool Parts:** In this step, point cloud of a tool is divided into 3 different point clouds as point cloud of left part, point cloud of main part and point cloud of right part in order to easily extract features specific to these parts. Towards the aim of detection, using the points of hold region, two line equations are found. One line to separate right part from main and left parts. The other equation to separate left part from main and right parts. These two line equation divide point cloud of the tool into 3 different point clouds. As a result, parts of the tool are obtained.

  The sample point clouds of tool parts are shown in Figure 4.3(a) and Figure 4.3(b) using different colours. Light and dark green colored parts are together create the main part of the tool.

- **Feature Extraction:** From each tool, the following 7 features(Figure 4.4) are compiled.

  - main length of the tool (tool_main_length)
  - horizontal length of the left part (tool_left_part_to_main)
  - horizontal length of the right part (tool_right_part_to_main)
  - vertical upper length of the left part (tool_left_part_vertical_upper)

– vertical below length of the left part (tool_left_part_vertical_down)

– vertical upper length of the right part (tool_right_part_vertical_upper)

– vertical below length of the right part (tool_right_part_vertical_down)

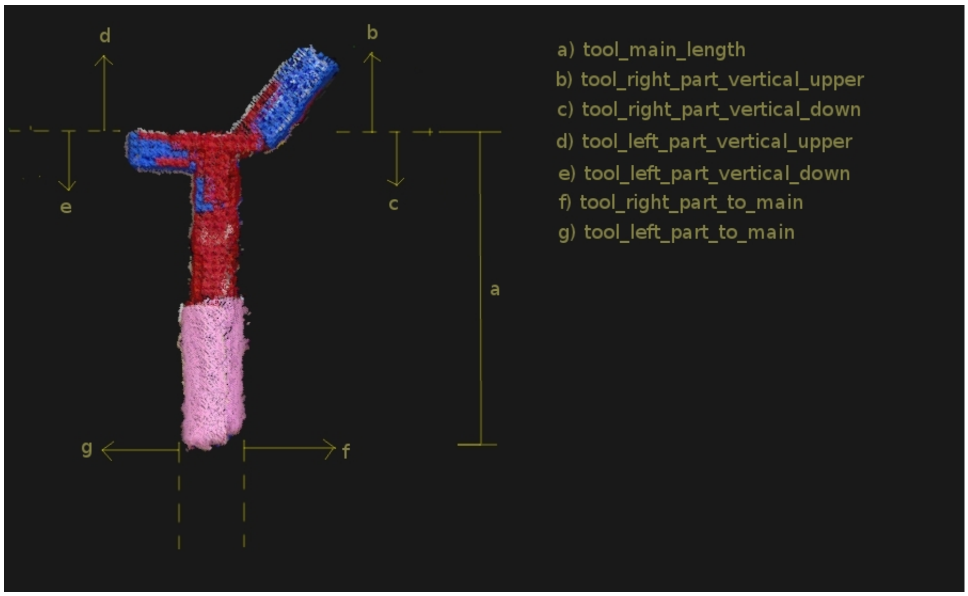


Figure 4.4: Kinect Tool Features

4.1.2 Tool Perception From Visual Images

The tool is also seen through the eye camera of iCub and the obtained image is processed to extract features using the following steps in Figure 4.5.

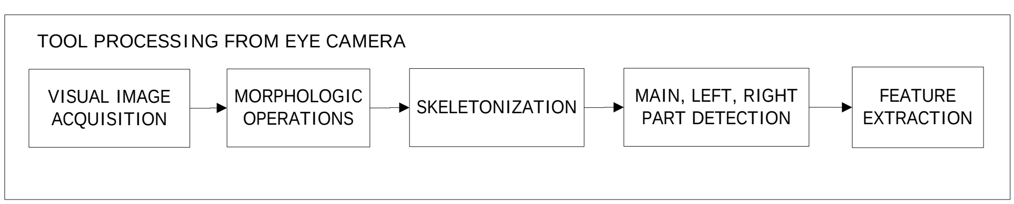


Figure 4.5: Tool Perception Steps Using Visual Image From Eye Camera

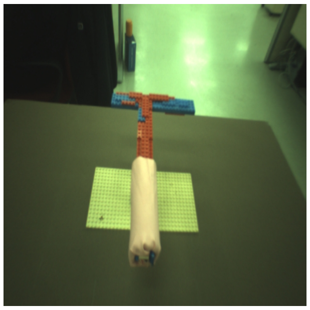
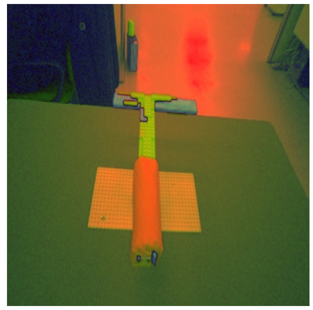
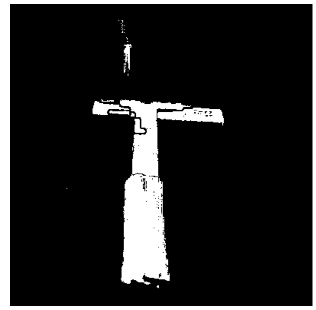
- **Visual Image Acquisition:** iCub looks at the tool and captures the visual image to start the image processing. A sample captured RGB image of the tool can be seen in Figure 4.6(a). The captured RGB images are transformed into HSV(Hue, saturation, value) images as in Figure 4.6(b) in order to easily apply color segmentation to get the tool itself.

As stated in previous chapter, tools were constructed using LEGO with different colours in order to use colour segmentation on the image. Tools consist of 3 colours as pink colour for handle of the tool, blue and/or red colour LEGO for the remaining parts of the tools. Using the channel interval values of these colours in HSV image, the tool is partially segmented as shown in Figure 4.6(c).

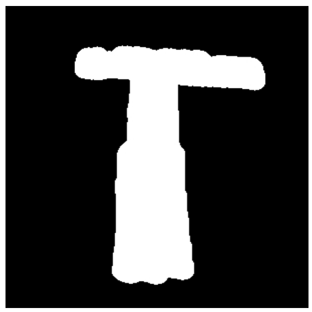
- **Morphological Operations:** In order to eliminate irregular blanks and noise after color segmentation due to illuminate changes, dilation and erosion morphological operations are applied on segmented image. Before morphological operations by comparing size of all blobs in the image, possible small blobs are eliminated and tool segment is obtained as biggest blob. Dilation operation is used to fill the irregular blanks as in Figure 4.6(d) and erosion operation is used to eliminate noises as in Figure 4.6(e). After these operations, the tool segment is obtained.

- **Skeletonization:** The tool segment is skeletonized in order to generate a morphological representation of the tool. For this process, Zhang *et al.*'s thinning algorithm is used [25] on binary images. Algorithm traces on white pixels and at each pixel, a list of conditions are checked to decide whether the pixel will be removed or not. These conditions are related to neighbour pixels such as existence of a specific pattern, multiplication result of specific neighbour pixels etcetera. The process finishes when the points were traced but none of them were removed. The conditions which are checked by the algorithm and flowchart of the algorithm can be seen in Appendix A.1.1. At the end of the process, a fully connected skeleton of the tool is obtained.

  Sample skeletons can be seen in Figure 4.7.

(a) RGB Image (b) HSV Image (c) Color Segmentation




(d) Dilation (e) Erosion

Figure 4.6: Processing Steps Before the Skeletonization

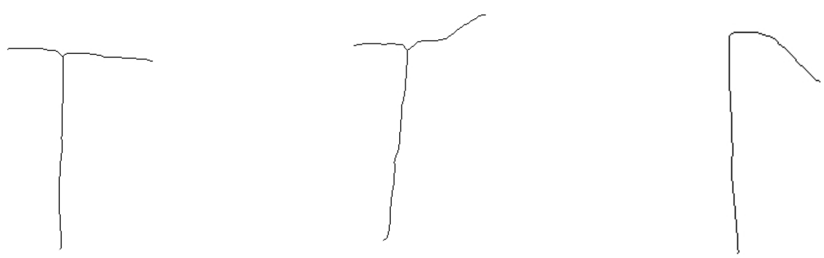
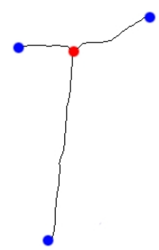
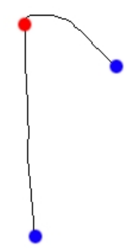


Figure 4.7: Skeletonized Tools

- **Detection of Tool Parts:** In this step, parts of the tool are detected by using special points on the skeleton in order to extract features. These special points are called "end points" and "middle points". Middle points are the points where a part finishes and another part or parts start. End points are the points where a part finishes and does not continue even as another part. Sample end and middle points can be seen in Figure 4.8(a). These special points are determined by using a recursive function which traces each point on the skeleton by looking its number of neighbours that are also points of the skeleton. If a point has only 1 neighbour then this point can be an end point, if it has 3 neighbours then it can be a middle point.



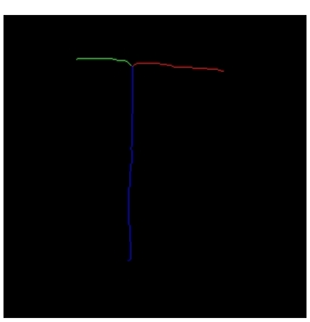
(a) Typical Case



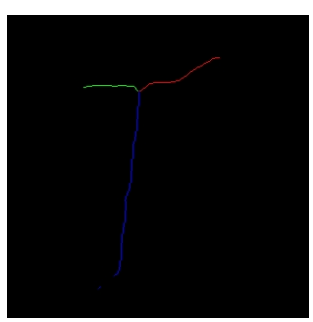
(b) Special Case

Figure 4.8: Red Dots Denote Middle Points, Blue Dots Denote End Points

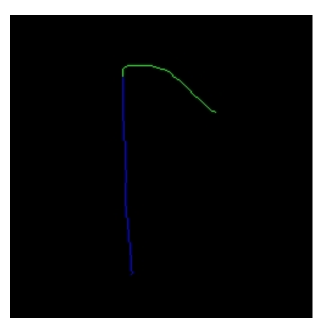
In our study, a skeleton must have at least 2 and at most 3 end points, for middle points it must have at most 1 middle points. If there is no middle points left, this means our tool consists of only main part or main part plus left or right part. In order to find this, slopes are computed between intervals of skeleton pixels, if in any of the intervals, slope passes a specified threshold, then this means a new part is started. This special case can be seen in Figure 4.8(b) and Figure 4.9(c). Samples for all these parts are shown in Figure 4.9.



(a)



(b)



(c)

Figure 4.9: Parts of the Tools

- **Feature Extraction:** In this step, using the end and middle points, tool features from eye camera are computed. This process includes transformation between iCub's eye reference frame and iCub's root frame. In order to find the corresponding 3D points of middle and end point pixels of the image, geometric distances between reference frame of the eye and the 3D points of middle and end points(these are obtained using Kinects) and pixel coordinates of these points in the image are used. Using the distances between 3D locations of end and middle points, following 5 features are computed:

  - length of the left part (lengthOfLeft)
  - length of the right part (lengthOfRight)
  - length of the main part (lenghtOfMain)
  - existence of left part (existenceOfLeftPart)
  - existence of right part (existenceOfRightPart)

  These features can be seen in Figure 4.10.

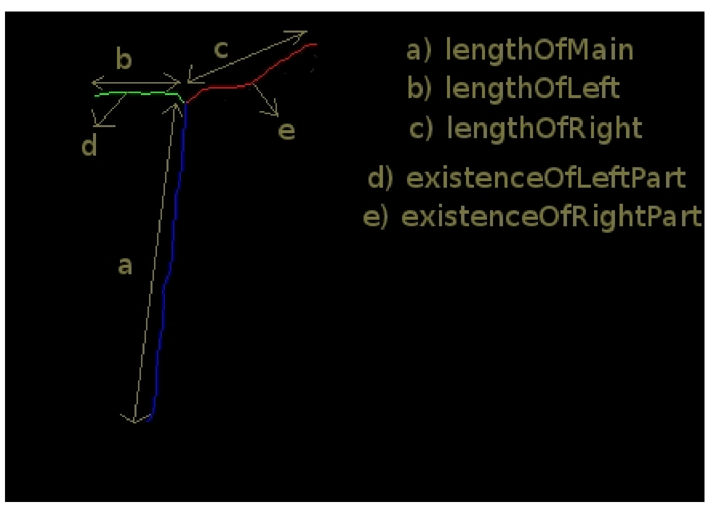


Figure 4.10: Eye Tool Features

4.2 Object Perception

iCub perceives the objects put on the table through the Kinects. The steps which are used to extract features from objects can be seen in the Figure 4.11. Objects can have different height, length and width and can be placed anywhere on the table.

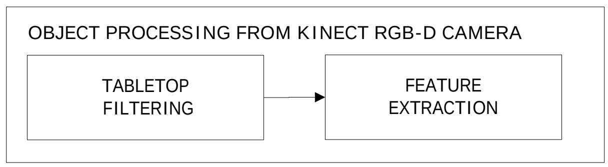


Figure 4.11: Object Perception Steps Using Point Cloud From Kinects

- **Tabletop Filtering:** We used the passthrough filter on each direction to segment the object from the background as can be seen in Figure 4.12.

Figure 4.12: Segmented Point Clouds From Sample Objects

- **Feature Extraction:** Features are extracted from the segmented point clouds as shown in Figure 4.13. Extracted 6 features are:
  - x dimension point of the object (object_point_x)
  - y dimension point of the object (object_point_y)
  - z dimension point of the object (object_point_z)
  - height of the object (object_height)

– width of the object (object_width)

– length of the object (object_length)

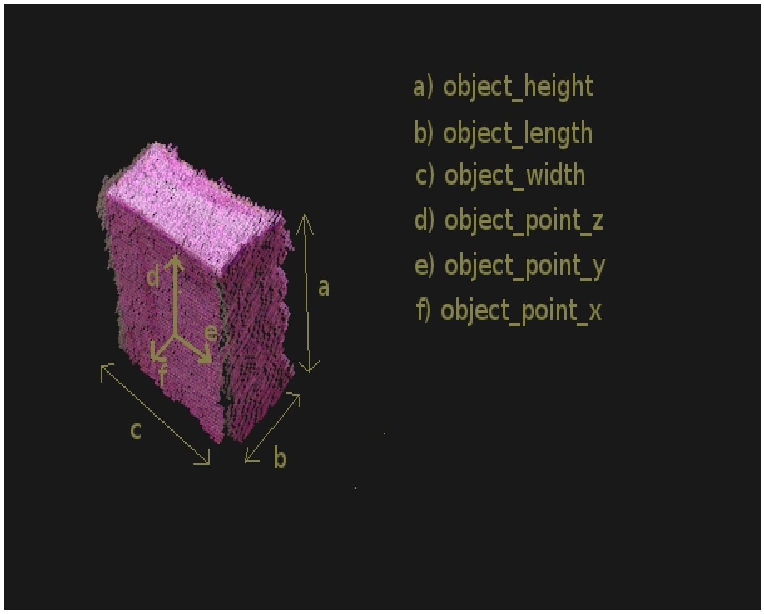


Figure 4.13: Features Obtained From Segmented Object From Kinect Range Data

Chapter 5

Behaviors

iCub interacts with the objects through a repertoire of behaviors. These behaviors can be classified into two groups as “behaviors with tools” and “behaviors without tools”. To apply these behaviors, iCub uses his hands. Therefore, which hand is used in an interaction is used as a feature.

5.1 Behaviors Without Tools

In the following behaviors, tools are not involved and these behaviors are either used to ease the perception or used to be prepared for behaviors with tools.

- **Tuck-arms behavior** used to ease perception and prevent occlusion on the table. Tuck arms position can be seen in Figure 5.1.
- **Grasp behavior** is used as a precursor to other behaviors. In order to grasp a tool, tactile sensors on palm and fingertips are used. Firstly, palm sensors are used to detect whether hand contacted to tool or not. Fingertip sensors are used to fully grasp the tool’s handle. A few examples of this grasp process are shown in Figure 5.2.

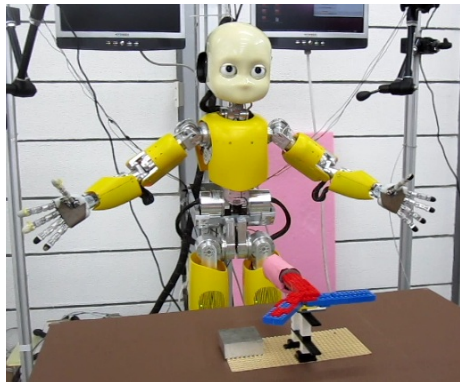


Figure 5.1: Tuck Arms Position

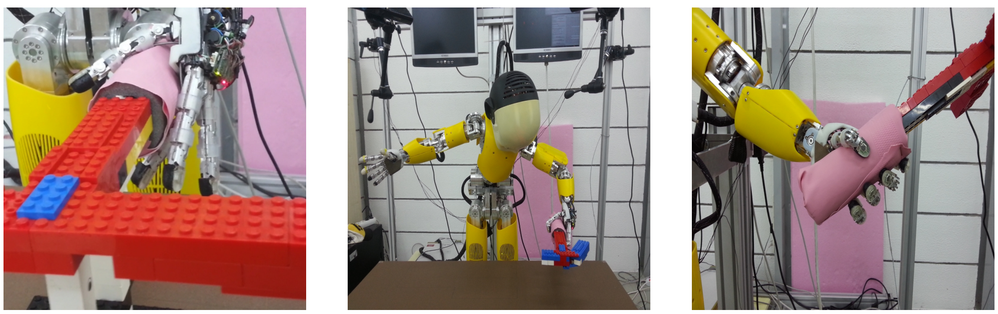


Figure 5.2: Grasping the Tool

5.2 Behaviors With Tools

After grasping the tool, iCub interacts with objects using the tools in order to learn tool affordances. There are 11 different behaviors that can result with 5 different effects. These 11 different behaviors can be grouped into 4 main behavior types. These behaviors and effects can be seen in the Figure 5.3.

5.2.1 Effects of Tool Behaviors

The effects are determined by whether center of mass moved towards the related way properly or not. These effect labels are given as supervised by the experimenter for the training phase.

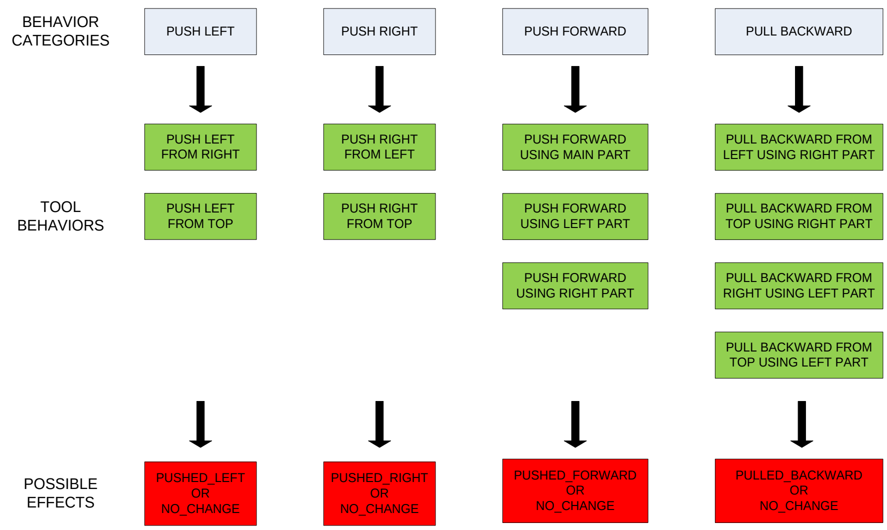


Figure 5.3: Behavior Categories, Tool Behaviors and Their Possible Effects

- **Pushed_Left:** If the center of mass of an object moved to left at the end of the interaction, it is labeled as pushed_left.
- **Pushed_Right:** If the center of mass of an object moved to right at the end of the interaction, it is labeled as pushed_right.
- **Pushed_Forward:** If the center of mass of an object moved away from the iCub at the end of the interaction, it is labeled as pushed_forward.
- **Pulled_Backward:** If the center of mass of an object moved closer to the iCub at the end of the interaction, it is labeled as pulled_backward.
- **No_Change:** If center of mass of an object stays still at the end of the interaction, it is labeled as no_change.

5.2.2 Tool Behaviors

- **Push Left From Right (PL-FR):** iCub brings the tool from the right of the object by considering horizontal length of left part and by aligning the left down or upper part tip with object's center and then moves his hand horizontally towards the object as it can be seen in Figure 5.4. As a result, the effect can be no_change or pushed_left.

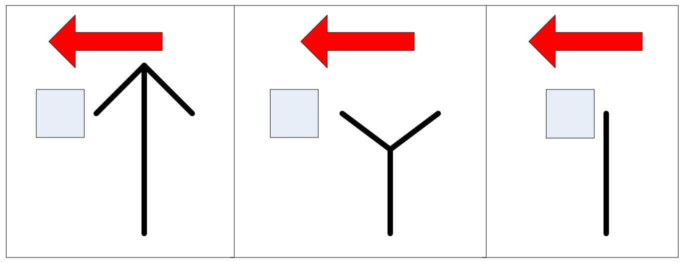


Figure 5.4: Push Left From Right

- **Push Left From Top (PL-FT):** iCub brings tool from top by considering vertical below length of left part and then lowers his hand and slides it to the left as shown in Figure 5.5. As a result, the effect can be no_change or pushed_left.

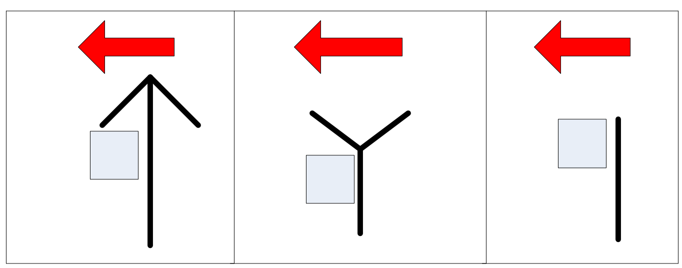


Figure 5.5: Push Left From Top

- **Push Right From Left (PR-FL):** This behavior is the symmetry of "Push Left From Right".
- **Push Right From Top (PR-FT):** This behavior is the symmetry of "Push Left From Top".
- **Push Forward Using Main Part (PF-UM):** iCub tries to push the given object to forward by using the main part of the tool as in Figure 5.6. As a result, the effect can be no_change or pushed_forward.

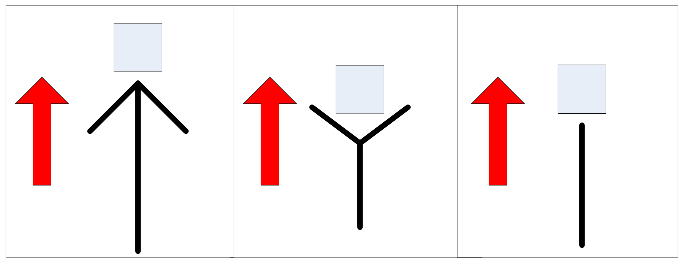


Figure 5.6: Push Forward Using Main Part

- **Push Forward Using Left Part (PF-UL):** iCub tries to push forward the object using the left part of the tool as it can be seen in Figure 5.7. The effect can be no_change or pushed_forward.

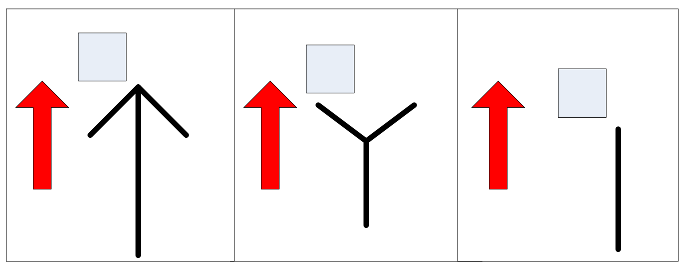


Figure 5.7: Push Forward Using Left Part

- **Push Forward Using Right Part (PF-UR):** This behavior is the symmetry of "Push Forward Using Left Part".
- **Pull Backward From Top Using Right Part (PB-FTUR):** iCub brings the tool from top by considering the vertical below length of the right part then he lowers and moves his hand backwards in this behavior as in Figure 5.8. The effect can be no_change or pulled_backward.

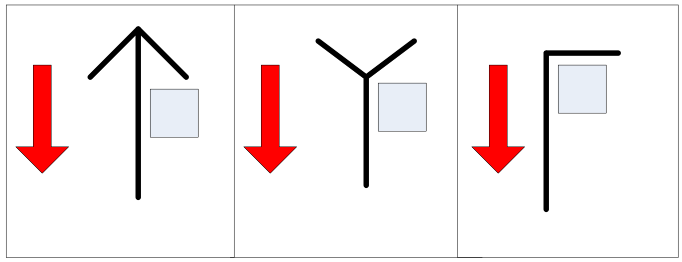


Figure 5.8: Pull Backward From Top Using Right Part

- **Pull Backward From Top Using Left Part (PB-FTUL):** This behavior is the symmetry of "Pull Backward From Top Using Right Part".
- **Pull Backward From Right Using Left Part (PB-FRUL):** iCub brings the tool from right of the object as in "Push Left From Right" behavior but considering the vertical below length of the left part. Afterwards, he moves his hand towards the object at least horizontal length of the left part then he moves his hand backwards to pull the object back as in Figure 5.9. The effect can be no_change or pulled_backward.

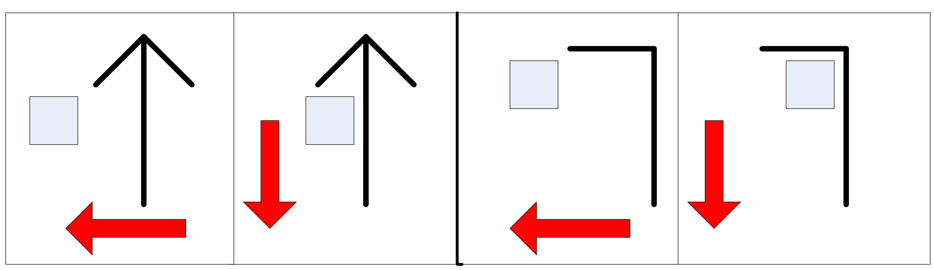


Figure 5.9: Pull Backward From Right Using Left Part

- **Pull Backward From Left Using Right Part (PB-FLUR):** This behavior is the symmetry of "Pull Backward From Right Using Left Part".

Chapter 6

Training of the System

Data collection system can be seen in the following diagram in Figure 6.1.

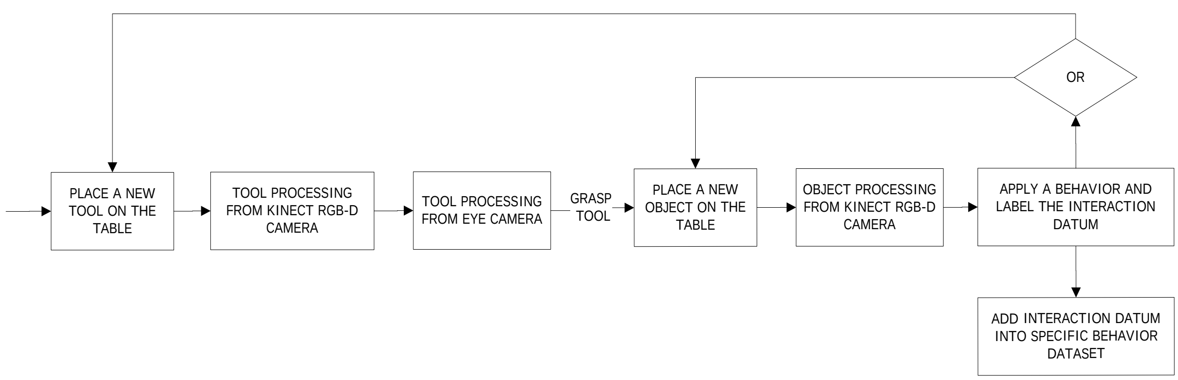


Figure 6.1: Data Collection System Diagram

6.1 Dataset

6.1.1 Tools and Objects Used For Training

iCub used tools which have different lengths of main part, different lengths of left or/and right parts with different angles as can be seen in Figure 6.2.

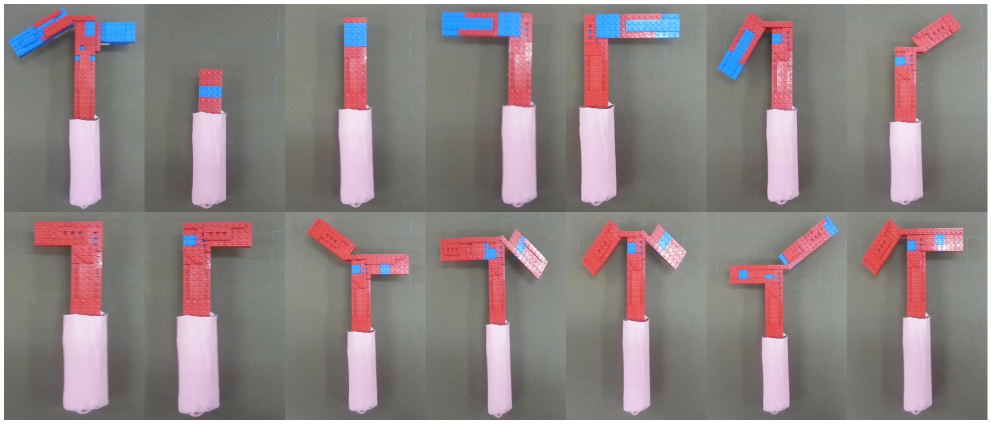


Figure 6.2: Samples of Training Tools

After grasping the tool, iCub applies his interaction behaviors on objects with different shape and sizes as shown Figure 6.3.

Figure 6.3: Training Objects

6.2 Training Phase and Results

Table 6.1 shows the number of interactions which are collected using iCub. Collection of features that creates an interaction datum can be seen in Figure 6.4.

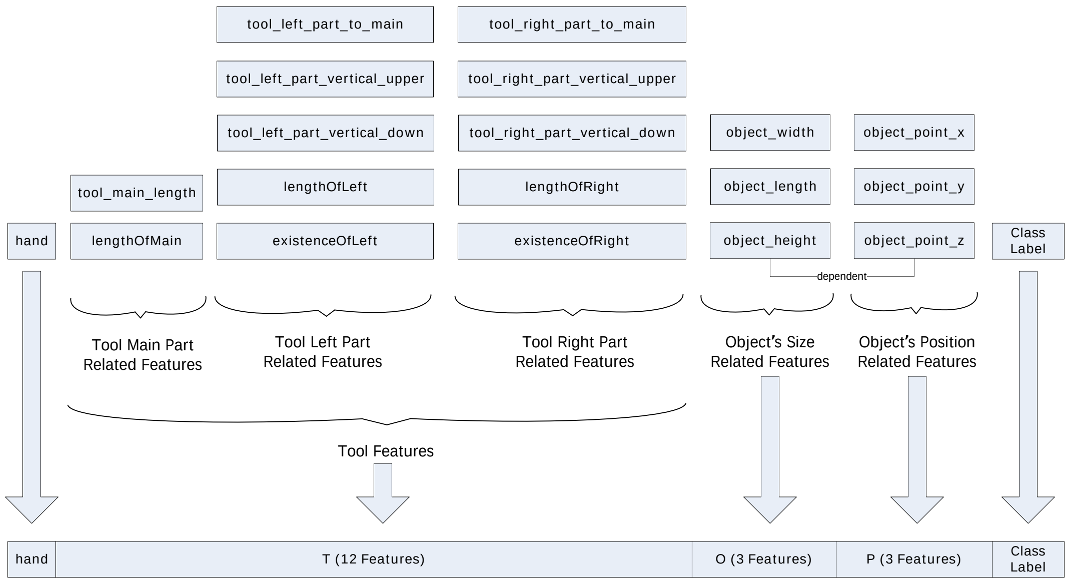


Figure 6.4: All Features That Creates Interaction Datum

Table 6.1: Number of Interactions Gathered From Each Behavior

Behavior	Number of Interactions
Push_Left (Bring From Right)	138
Push_Left (Bring From Top)	152
Push_Right (Bring From Left)	138
Push_Right (Bring From Top)	152
Push_Forward (Using Main Part)	240
Push_Forward (Using Left Part)	115
Push_Forward (Using Right Part)	115
Pull_Backward (Using Left Part, Bring From Right)	99
Pull_Backward (Using Right Part, Bring From Left)	99
Pull_Backward (Using Left Part, Bring From Top)	114
Pull_Backward (Using Right Part, Bring From Top)	114

In order to learn effective behavior models, collected interaction data should cover most of the possible situations for each behavior. As it can be seen in Figure 6.4, an interaction datum consists of a hand feature, tool features, object size features and object position features. When interaction data is being collected by an experimenter, it is not possible to cover all possible combination of feature values because most of the features are numeric features which causes many combination between features. These different combinations can result with different effects depending on a behavior. Therefore, interaction data in Table 6.1 are not enough to represent each of the behaviors. This reveals the need of having a big dataset that covers most of the possible combination of features.

6.2.1 Dataset Construction

As it was stated in previous part, in order to cover most of the possible combination of features, a big dataset should be constructed for each behavior. In this part, how dataset for a behavior is constructed will be explained. The whole dataset creation process can be seen in Figure 6.5. The steps for this process are:

Creating Basis Feature Sets: We used the collected data from the robot's interactions to create bigger behavior datasets.

First of all these 1476 interactions in Table 6.1 were combined into one dataset. This dataset was split into different datasets by grouping relevant features in the same dataset. As a result of grouping, we have tools' main part related set(includes 2 features with 1476 entries), tools' left part related set(includes 5 features with 1476 entries), tools' right part related set(includes 5 features with 1476 entries) and lastly objects' properties related set(includes 4 features with 1476 entries). Since hand, object_point_x and object_point_y features cannot be a part of tool creation or object selection phase because of this, they are not included in any of the set.

After feature sets were determined, among 1476 entries in each set, unique entries were selected by eliminating the duplicate entries. As a result of this selection, number of unique entries in these basis feature sets were given in Table 6.2.

Table 6.2: Number of Unique Entries in Basis Sets

Set	Number of Unique Entries
Tool's Main Length Related	19
Tool's Left Part Related	108
Tool's Right Part Related	108
Object's Properties Related	76

The table indicates that among 1476 interactions, 19 tools with different main parts are used, 76 different objects are used and 108 different type of left and right parts are used.

Construction of An Interaction Datum: In this step using basis feature sets, an interaction datum is created. Using tool related basis feature sets, one can construct any kind of tool and can select any object using object related feature set. Remaining features should be selected by the user which are hand feature and position features those indicate where the object will be placed on the table. At the end of this selection from feature sets, an unlabeled interaction datum is obtained.

Labeling Constructed Interaction Datum: This step labels the unlabeled interaction datum with proper effect. In previous chapters, boundaries that iCub can move its hands in and movements of each behavior were shown and explained. According to these constraints and iCub's kinematics, some conditions should be checked before the attempt of each behavior. These conditions determine whether iCub can accomplish the requested behavior on a shown object or not. If all conditions are satisfied, iCub is able to do the behavior on a given object with success. If a given interaction datum satisfies the conditions of the requested behavior then that instance is labeled as behavior specific effect, otherwise it is labeled as no_change.

Creating Behavior Datasets Using Constructed Interaction Data: In this part using labeled interaction data, behavior datasets were created. For this process, it is important to determine the number of interactions and determine how many of these interactions should be labeled as behavior specific effect and how many of them will be labeled as no_change. Since we have so many combinations with different tools, different objects and different positions, 10000 interactions were created for each behavior. Among these 10000 interactions, half of them belongs to behavior with left hand and the other half belongs to behavior with right hand. Among 5000 interactions, 1000 of them were labeled as behavior specific effect, 4000 of them were labeled as no_change. The reason behind this ratio is this: In each behavior, there are not so many variety of behavior specific effect situations. However, there may be many reasons for an instance to be labeled as no_change. Therefore, this ratio is selected between number of behavior specific effect and no_change labeled interactions. At the end, each behavior has 10000 instances with 2000 behavior specific effect labels and 8000 no_change labels.

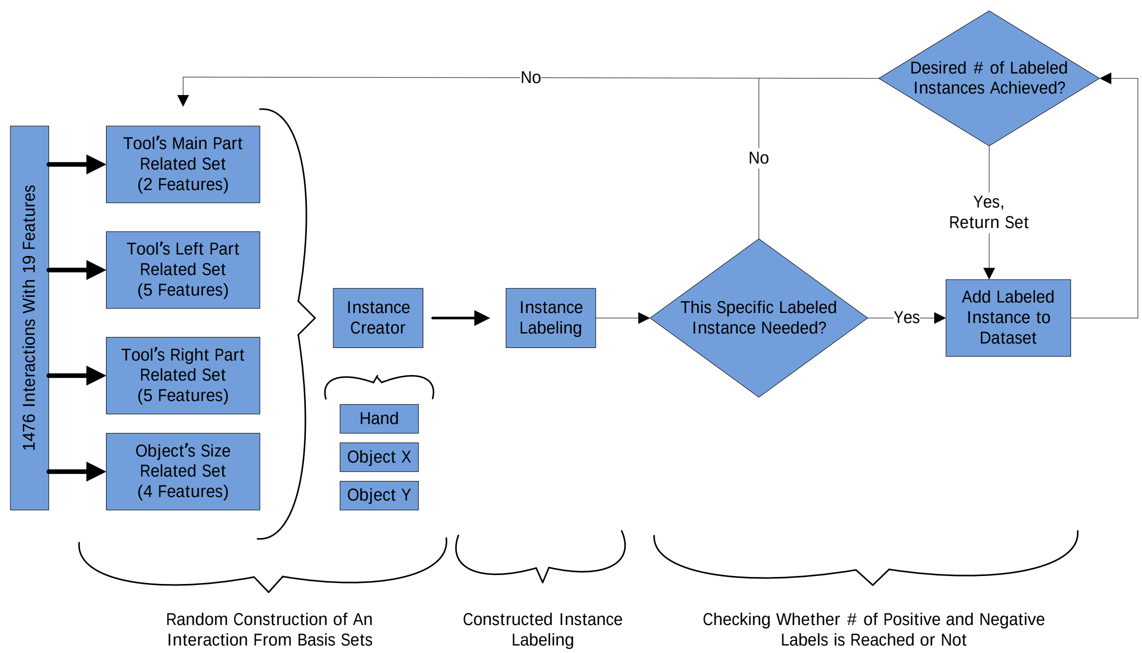


Figure 6.5: Dataset Construction Process

6.2.2 Combination of Tool and Object Features

After datasets for each behavior were created, features are combined with each other in order to to reveal complex relations between different features specific to each behaviors. Because, it is not possible to infer relationships between features from the result of ReliefF feature selection if the behaviors are not simple.

19 features in the dataset are combined in the following way. First of all, 19 features are split into sensible parts which are similar to basis feature sets. For this combination process, 19 features split into 4 groups. These are; a group which consists tool main part related features (2 features), a group consists tool's left and right part related features (8 features), a group consists of object properties (3 features) and lastly a group that consists position features of the object (3 features). Hand, existenceOfLeftPart and existenceOfRightPart features were not involved in the groups since they are nominal features.

Each of the features in these four groups are combined by having an integer coefficient(-1,0,+1) in order with all possible combinations. At the end of the process, the dataset ends up with 4168 features. The process of combination of core features can be seen in the Figure 6.6.

6.2.3 Feature Selection and Feature Elimination on Combined Dataset

Feature Ranking: In this part, 4168 features are ranked according to their contributions to separation of the labels. Towards this end, ReliefF feature selection method is applied on each behavior datasets as explained in Appendix A.2. The result of ReliefF method is a ranked features of dataset.

Feature Elimination: Among the ranked 4168 ranked features, unnecessary features are eliminated from the feature set. In order to do this, following way is used: "*If a combined feature will remain in the dataset, it must be ranked better than its subset of combined features*". Assume the following small sample set in

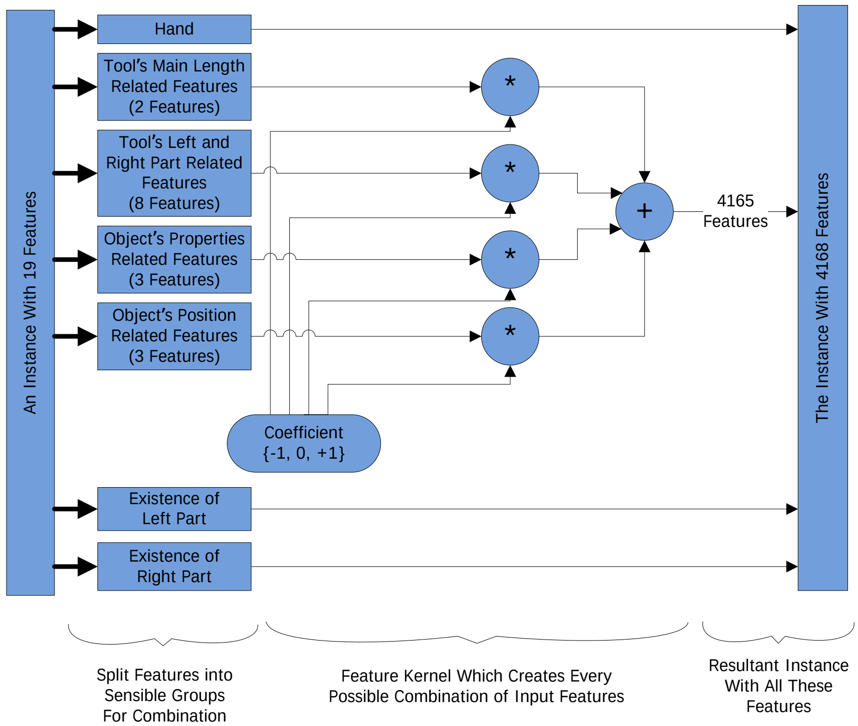


Figure 6.6: Feature Combination Process

Table 6.3:

Table 6.3: A Sample Ranked Feature Set Before Elimination

Ranking	Feature Name
1	+object_width
2	+tool_main_length+object_point_x
3	+object_width-object_point_y
4	+tool_main_length-tool_left_part_vertical_down+object_point_x
5	-object_point_y
6	-tool_left_part_vertical_down+object_point_x
7	+object_point_x
8	-tool_left_part_vertical_down
9	+tool_main_length-tool_left_part_vertical_down
10	+tool_main_length

Table 6.3 shows a sample dataset of features before the process of elimination. Ranked 1,5,7,8 and 10 features will remain in set because they are core features. For the 2nd ranked feature, it is ranked higher than its subsets which are 7th and 10th. So, the 2nd feature will remain in the set. 3rd feature will not be included since one of its subset feature is ranked 1st which is higher. This means *+object_width* feature is ranked very high but its combination with *-object_point_y* feature lowers its rank. Therefore, it will be removed from the set.

A similar situation is valid for *+tool_main_length-tool_left_part_vertical_down +object_point_x* feature, since this feature is a combination of 3 features, we should also look its 2-featured subsets. Since one of its subsets which is *+tool_main_length+object_point_x* is ranked higher, this feature will also be removed. 6th ranked feature is ranked higher than its subset so it will remain but 9th feature will not be able to remain in the set since one of its subsets is ranked higher than itself. The resultant set after elimination can be seen in Table- 6.4

Table 6.4: The Sample Ranked Feature Set After Elimination

Ranking	Feature Name
1	+object_width
2	+tool_main_length+object_point_x
5	-object_point_y
6	-tool_left_part_vertical_down+object_point_x
7	+object_point_x
8	-tool_left_part_vertical_down
10	+tool_main_length

6.2.4 Performance of Each Behaviors After Feature Selection and Elimination

After the selection and elimination process, the following number of features are remained in behavior datasets in Table 6.5.

Number of features for each behavior in Table 6.5 do not represent the behaviors in a best way. In order to find the best number of features that represents the behavior, support vector machine performances are examined at each number

Table 6.5: Number of Features Remained For Each Behavior After Feature Selection and Elimination

Behavior	Remaining Number of Features
Push Left From Right	20
Push Left From Top	21
Push Right From Left	22
Push Right From Top	21
Push Forward Using Main Part	19
Push Forward Using Left Part	21
Push Forward Using Right Part	21
Pull Backward From Top Using Right Part	21
Pull Backward From Top Using Left Part	21
Pull Backward From Right Using Left Part	22
Pull Backward From Left Using Right Part	22

of features by decreasing one by one. In these performance analysis, SVM with radial basis functioned kernel is used. Optimized cost and gamma parameters of the kernel at each number of features is found using grid search on c and gamma parameters. This process can be seen in Figure 6.7 and the result of these analysis for each behavior can be seen in the following Figure 6.8.

At the end of the process, an SVM model is trained for each behavior using the *c* and *gamma* parameters that gives best performance on behavior dataset with a specific number of features.

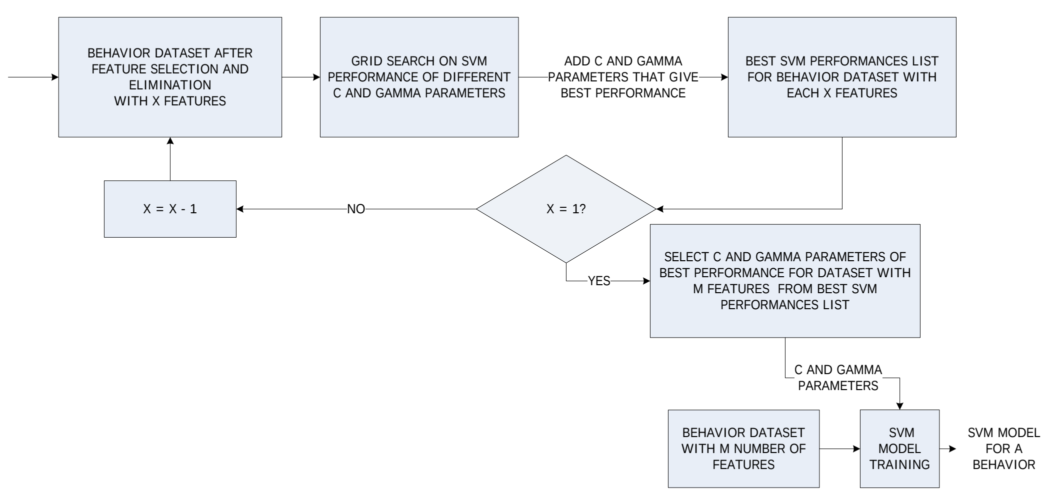


Figure 6.7: Training Behavior Models Using SVM

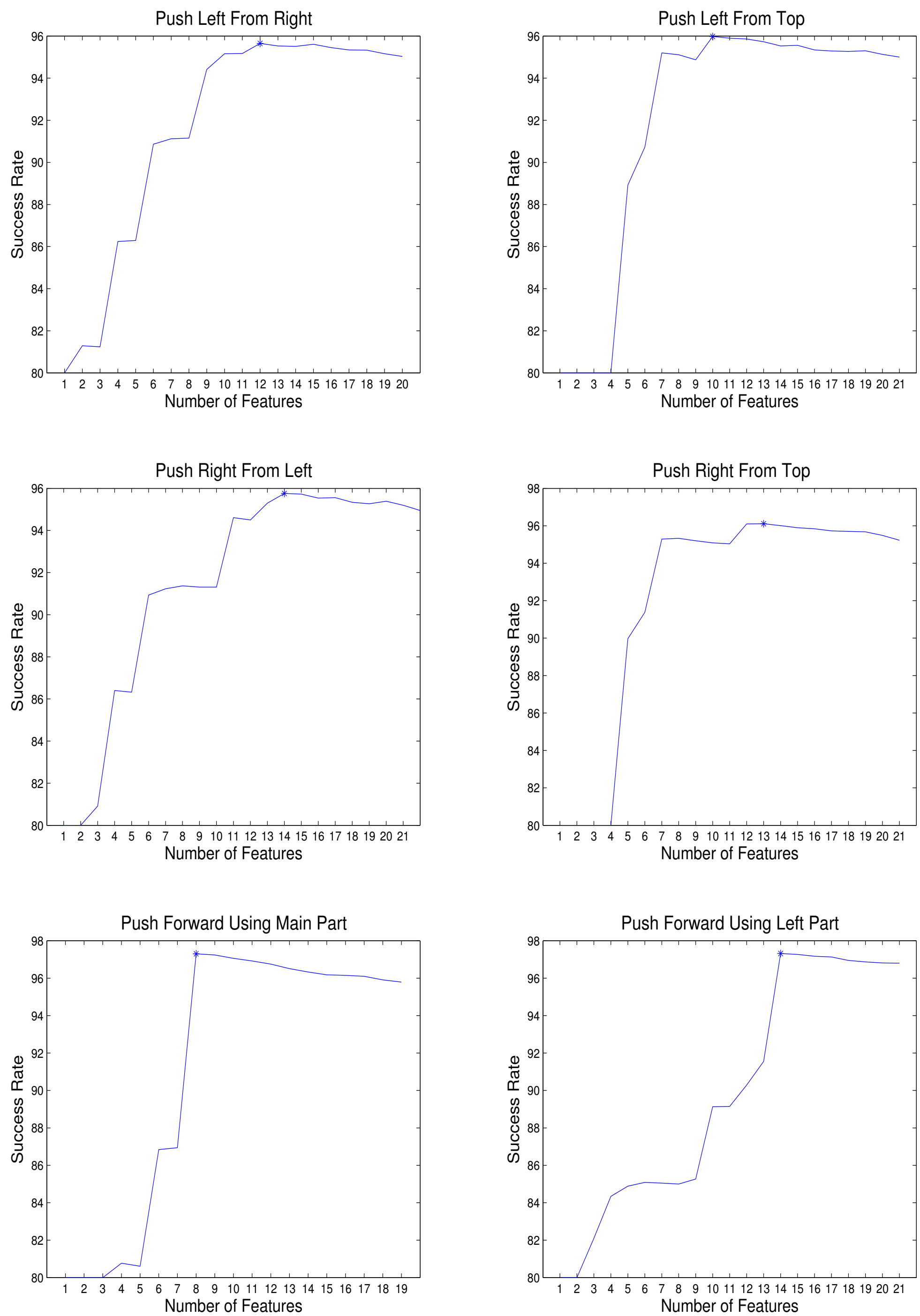


Figure 6.8: Performance of Each Behavior After ReliefF Feature Selection ('*' indicates number of features which represents that behavior best)

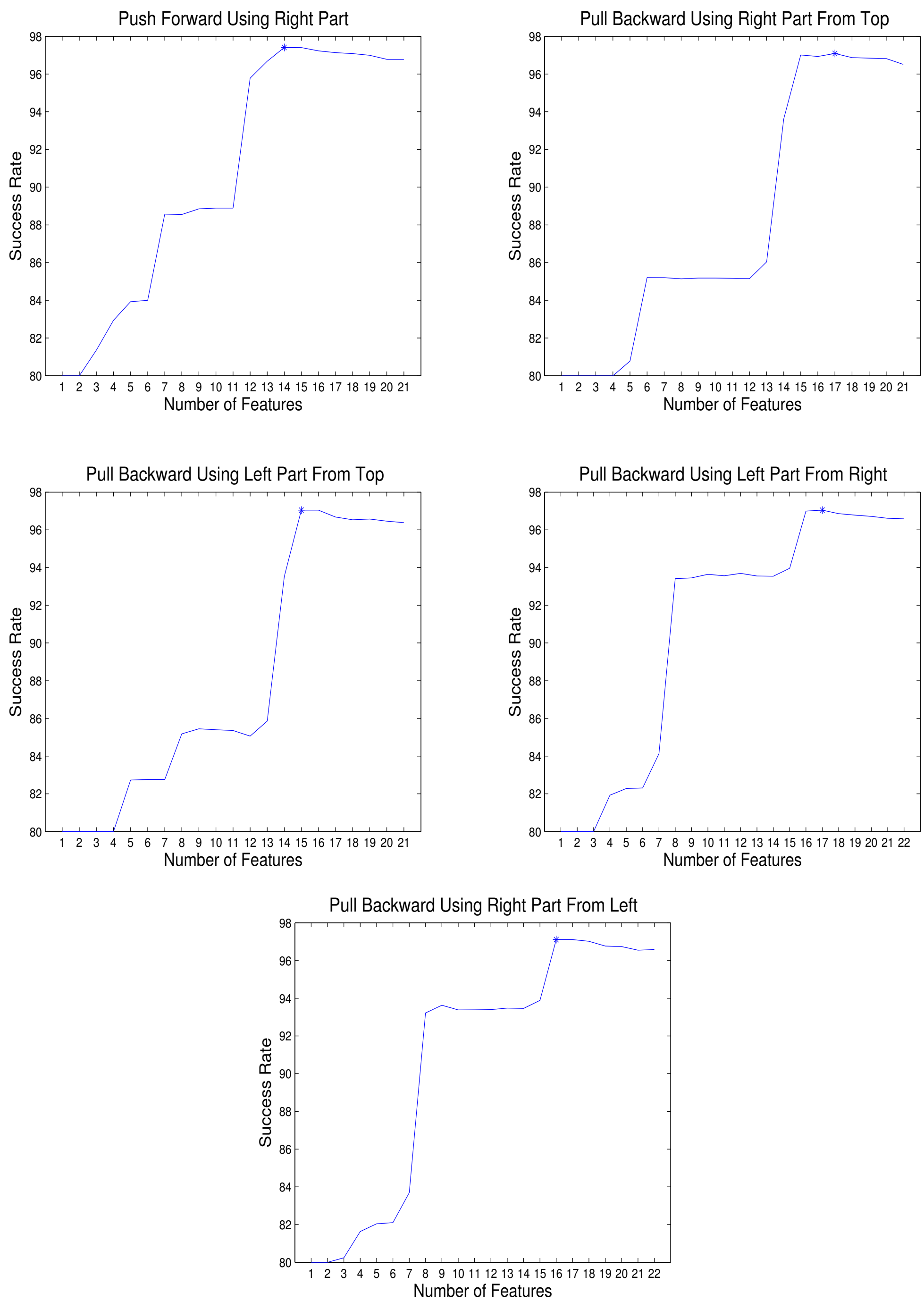


Figure 6.8: Performance of Each Behavior After ReliefF Feature Selection ('*' indicates number of features which represents that behavior best)

The best number of features in each behavior is marked with '*' on the graphs and in the following Table 6.6 number of features selected for each behavior and its performance on training set can be seen.

Table 6.6: Selected Number of Features and Their 5-Cross Validation Results

Behavior	Number of Features	Validation Result
Push Left From Right (PL-FR)	12	%95.65
Push Left From Top (PL-FT)	10	%95.98
Push Right From Left (PR-FL)	14	%95.76
Push Right From Top (PR-FT)	13	%96.11
Push Forward Using Main (PF-UM)	8	%97.3
Push Forward Using Left (PF-UL)	14	%97.32
Push Forward Using Right (PF-UR)	14	%97.41
Pull Backward From Top Using Right (PB-FTUR)	17	%97.09
Pull Backward From Top Using Left (PB-FTUL)	15	%97.04
Pull Backward From Right Using Left (PB-FRUL)	17	%97.04
Pull Backward From Left Using Right (PB-FLUR)	16	%97.11

6.2.5 Analysis of Each Behavior

In this section, one behavior for each effect will be analyzed out of 11 behaviors using their ranked features.

- **Push Left From Top (PL-FT):**

Table 6.7: Selected Features of *Push Left From Top*

Rank	Feature Name
0.139	hand
0.057	object_point_x
0.055	object_point_z
0.054	object_height
0.035	object_point_y
0.027	object_length
0.025	+tool_main_length-tool_left_part_vertical_down
0.025	+lenghtOfMain-tool_left_part_vertical_down
0.025	tool_left_part_vertical_down
0.024	object_width

In this behavior, iCub tries to avoid tool's left vertical down part by bringing the tool from top of the object and pushes it with the main part to the left. As can be seen in Table 6.7, hand feature is the most separating feature. Since the height of object is important for iCub to bring the tool from top of the object, object_height feature and related to this object_point_z feature ranked very high with object's distance feature object_point_x. object_point_y and object_length features are required for positioning the tool. The 7th and 8th features are combination of features. They denote the same thing which is the distance from starting point of the tool to object's furthest point. tool_left_part_vertical_down is a necessary feature to take care in order to avoid crash between tool and object. The last feature is the object_width which helps iCub to position the tool in horizontal direction. In this behavior, there are no features related to the right part due to the type of the behavior.

- **Push Right From Left (PR-FL):**

Table 6.8: Selected Features of *Push Right From Left*

Rank	Feature Name
0.138	hand
0.057	tool_right_part_vertical_down
0.057	object_point_y
0.051	+tool_right_part_vertical_upper+object_point_x
0.050	object_point_x
0.039	+object_height-object_point_z
0.039	+object_height+object_point_z
0.039	object_height
0.039	object_point_z
0.033	tool_right_part_vertical_upper
0.024	tool_main_length
0.024	lenghtOfMain
0.022	object_width
0.022	lengthOfRight

In this behavior, iCub pushes right an object by bringing the tool from left of the object using the tip of the right down or upper part of the tool. As can be seen in Table 6.8, the tool_right_part_vertical_down and tool_right_part_vertical_upper features should be among the top features with hand feature. object_point_y is ranked high since it is required to place the tool left of the object. As it is guessed tool_right_part_vertical_upper is among highly ranked features but as a combination of feature. This feature denotes the point where iCub should put the furthest point of the tool in x-direction. Then, object_point_x feature comes and necessary for reaching the object. After this, object height related combination and core features are listed. 6^{th} and 7^{th} features are combination of object_height and object_point_z features. It is normal to see these combination since these two features depends on each other. Although these features seem not so important it depends on which hand iCub uses. In order to not to hit the object, iCub may bring the tool from top to place the tool to left if it uses right hand. After these features, tool main part related features are listed with object_width and lengthOfRight features. As it can be seen eye features are not found so passive. Again in this behavior, there are no features related to left part because of type of the behavior.

- **Push Forward Using Main Part (PF-UM):**

Table 6.9: Selected Features of *Push Forward Using Main Part*

Rank	Feature Name
0.140	hand
0.137	object_point_x
0.038	object_length
0.038	lenghtOfMain
0.038	tool_main_length
0.037	object_point_z
0.036	object_height
0.024	object_point_y

Push forward using main part behavior is a simple behavior since for the success of the behavior length of the main tool and the distance of the object are the crucial features as it can be seen in Table 6.9. In addition, object's height thereby object's point z should be not so short or high. object_length is needed since it is related with reach. lenghtOfMain is similar to tool_main_length. object_point_y is required for alignment in y direction. As it can be seen from the table there are features related with left and right parts of the tools since behavior just uses main part.

- **Pull Backward From Top Using Left Part (PB-FTUL):**

Table 6.10: Selected Features of *Pull Backward From Top Using Left Part*

Rank	Feature Name
0.112	hand
0.063	+object_point_x-tool_left_part_vertical_down
0.063	object_point_x
0.060	object_width
0.055	+object_height-object_point_z
0.054	object_point_z
0.054	object_height
0.054	lengthOfLeft
0.052	tool_left_part_to_main
0.051	tool_left_part_vertical_down
0.038	existenceOfLeftPart
0.037	tool_left_part_vertical_upper
0.031	object_length
0.025	object_point_y
0.022	tool_main_length

This behavior is similar to *push left from top* behavior, but in this one as an extra movement iCub pulls the tool backwards which makes object_width and left part related lengths very important because of center of mass movement. As it can be seen in Table 6.10, hand is the highest ranked feature as it was in the previous behaviors. 2^{nd} feature is a combination of features which denotes the point where iCub should put the furthest point of tool's main length. object_point_x which denotes object's distance is important in order to reach the object and as it is said in first sentence, object_width is another important feature which can directly effect the success of the behavior. 5^{th} feature is a familiar combination of features related to height and also the following two features. The next 5 features are related left part of the tools which are the important features for pulling back the objects. Only tool_left_part_vertical_upper feature is not an effective one and listed at the end. object_length and object_point_y features are important features to align the tool according to object and tool_main_length feature is in the list which is effective to reach the object.

Chapter 7

Experiments

In Figure 7.1, there are five different tools and 3 different nails. One of these nails is fully nailed, the second one needs to be nailed and the third one is free. The tools are in order as a hammer, a claw hammer, a hammer-plier, a plier and lastly a nail remover. Uprootable, nailable and pushable affordances will be examined on these nails, respectively.

These tools and nails will be used as examples in order to define different concepts for the following tests as:

- **Same Tools:** If two tools have the same affordances by using same behaviors with same parts, these two tools are same tools. The hammer-plier and the claw hammer are same tools because they nail with their blunt parts, they can uproot the nail with their pincers and they can push the nail with their straight part.

- **Functionally Equivalent Tools:** If two tools have the same affordances but through using different behaviors with different parts then these two tools are functionally-equivalent tools. The nail remover and the plier are functionally equivalent tools. They cannot nail but they can push and uproot the nails using their different parts.

- **Equivalent Tools:** If two tools have the same affordances regardless of how they manage to achieve these affordances, these two tools are equivalent tools. All same tools and functionally equivalent tools are also equivalent tools. In Figure 7.1, the claw-hammer and the hammer-plier are one group of equivalent tools capable of 3 same affordance, the plier and the nail remover are another group of equivalent tools capable of 2 same affordance.

Figure 7.1: Different Tools versus Nails in Different Situations

In some of the tests, only one affordance was requested, in some all affordances were requested. Lastly, in different situations similarity between different tools including novel tools were examined.

In the tests, tools were denoted with 'T', objects were denoted with 'O' and object position was denoted with 'P' which will result as (T, O, P) tuple. In some tests, one or more tuple elements have possibility to be multiple which was denoted with a '*' next to a tuple element.

7.1 Novel Tools

In the following tests one or more novel tools were used which are shown in Figure 7.2.

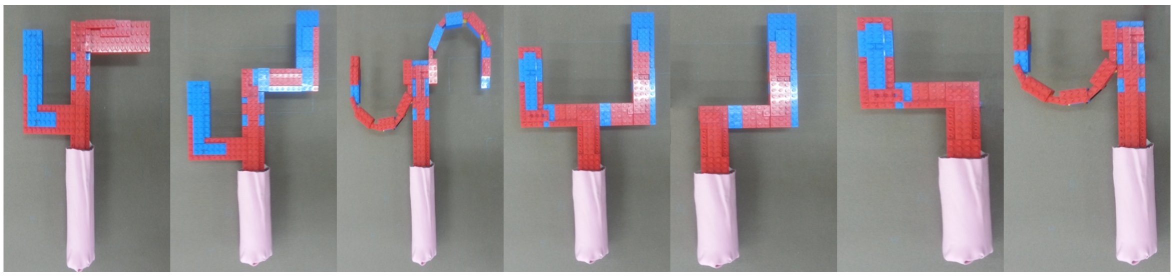
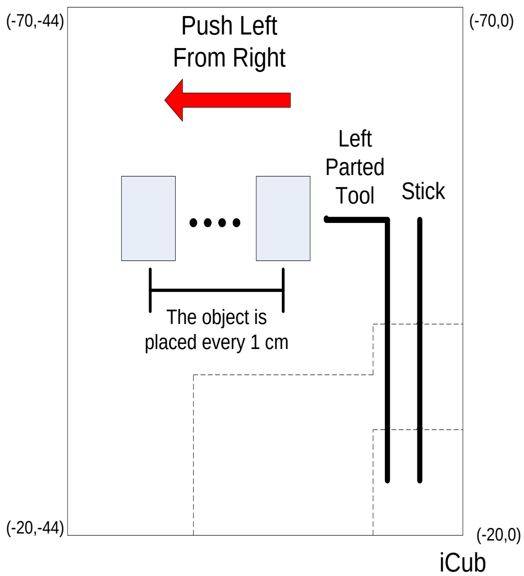


Figure 7.2: Samples of Novel Tools

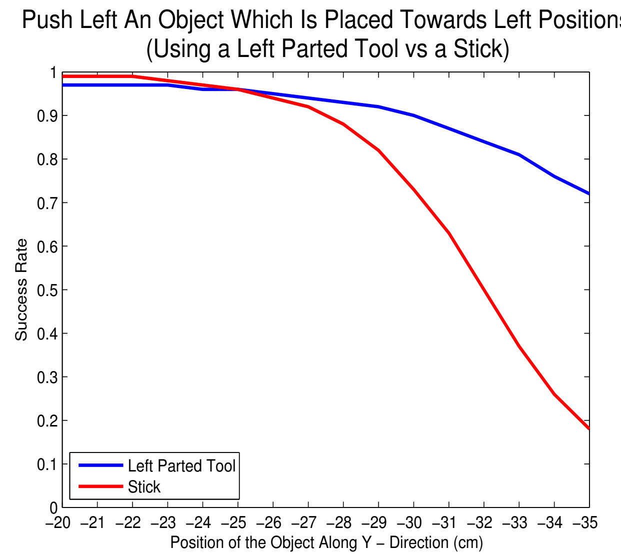
7.2 A Specific Affordance is Requested

In this test type, one affordance was requested on different tuples. The results were shown on graphs with illustrations of the tests in each part.

Pushed Left Affordance is Requested on (T*, O, P*) Tuple: The Figure 7.3(a) shows a stick versus a left parted tool which are used to push left an object which is gradually placed far away to the left. The plot in Figure 7.3(b) shows the success rate of these two tools using *push left from right* behavior on the objects in order to show the importance of existence of an extra part.



(a) Illustration of the Behavior



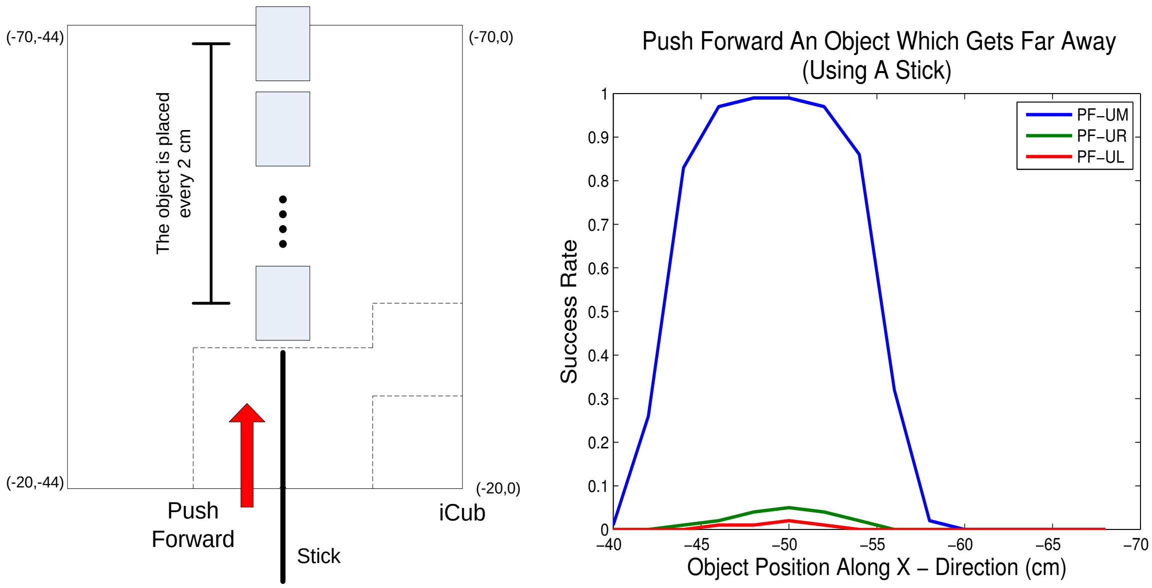
(b) Result Graph

Figure 7.3: Push Left Objects Which Are Gradually Placed Far Away to the Left

As it can be seen in the graph, the success rate of the stick is decreasing earlier than left straight parted tool towards left. The reason is the existence of left part. While using a stick iCub is not able to push left the object on the leftmost positions because the object starts to be placed out of iCub's reach area. However, using a left straight parted tool, although the object is placed out of reach area, with the help of left part iCub is still able to push the object left. One way or another, the object starts to get out of both of the tools' reach at the end.

With this experiment it was shown that existence of an extra part extends the reach and effect area.

Pushed Forward Affordance is Requested on (T, O, P*) Tuple: The Figure 7.4(a) shows a stick which is used to push forward an object that is placed a few different locations where these locations gradually get far away to forward. The plot in Figure 7.4(b) shows the effect of object's distance on success rate of push forward behaviors in order to show the importance of the relation between main part length of the tool and object's distance.



(a) Illustration of the Behavior

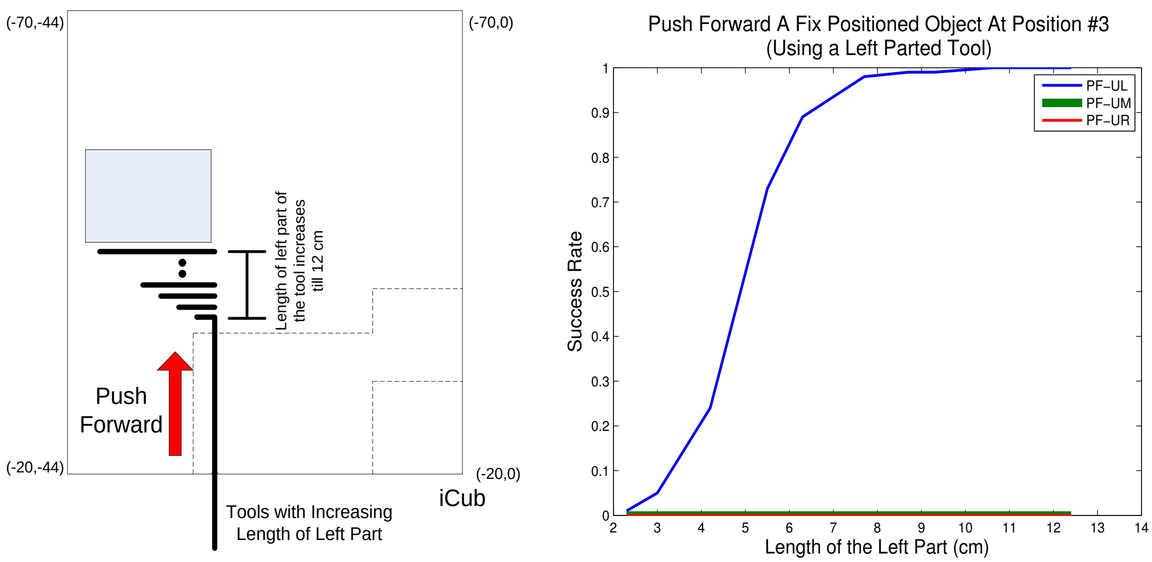
(b) Result Graph

Figure 7.4: Push Forward Objects Which Are Gradually Placed Far Away to Forward

It can be easily seen from the graph that since the stick does not have left or right parts, the success rate of both *push forward using left part* and *push forward using right part* behaviors are nearly zero. The success rate of *push forward using main part* behavior starts from lower rates and increases when object gets far away but to the end it starts to decrease. The rate is low at first positions because iCub is unable to bring his hand behind the object since the object is too close. When the distance starts to increase, iCub is able to bring the tool behind the object and can push the object forward. From some positions after, rate decreases because the object is getting far away and iCub cannot reach object.

Using this experiment it was shown that in order to push forward a far object, a long tool should be used and for closer objects it is enough to have shorter tools.

Pushed Forward Affordance is Requested on (T*, O, P) Tuple: The Figure 7.5(a) shows multiple tools with increasing length of left parts which are used to push forward a wide object put on leftmost of the table. The plot in Figure 7.5(b) shows the effect of length of left part on the success rate of push forward behaviors in order to show the relation between length of left part and the width of the object.



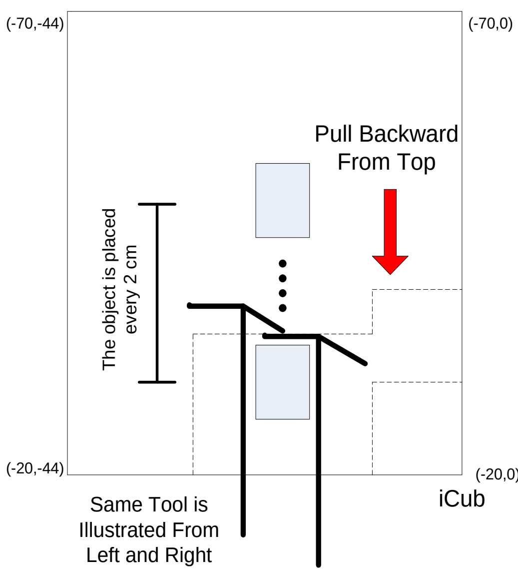
(a) Illustration of the Behavior (b) Result Graph

Figure 7.5: Push Forward An Object With Tools Having Increasing Length of Left Part

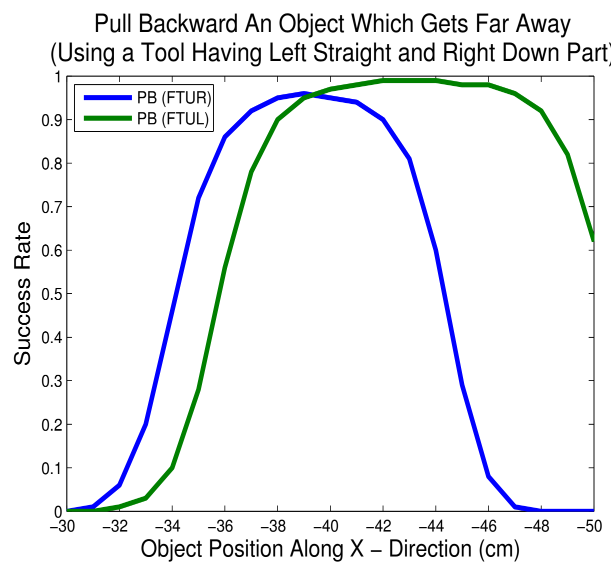
From the results it can be seen that success rate of *push forward using main part* and *push forward using right part* behaviors are always close to zero. The reason is this; since object is placed to a position which is far to the iCub's left side, it is not possible for iCub to align the tool's main part or right part according to center of the object. Therefore, iCub can only reach and push forward the object using left part of the tool. Length of left part starts from 2 cm to 12 cm. Success rate starts from lower since length of left part is not enough to result the behavior with pushed forward effect although tool can reach the object. Main reason is length of left part is shorter than half width of the object. With shorter length iCub can affect and move the object but center of mass does not move forward properly because of this it does not count as success. However, gradually left part increases and behavior starts to result with success.

As a result of this experiment, it was shown that in order to successfully push forward an object with left or right part, length of these parts should be equal or longer than half of the object's width.

Pulled Backward Affordance is Requested on (T, O, P*) Tuple: In Figure 7.6(a) an object is placed a few different positions from closer position towards far positions from iCub in -x direction and using a tool which has left straight part and right to the below part *pull backward from top using left part* and *pull backward from top using right part* behaviors are applied on the objects. The plot in Figure 7.6(b), shows the effect of length of left and right down parts on success rate.



(a) Illustration of the Behavior



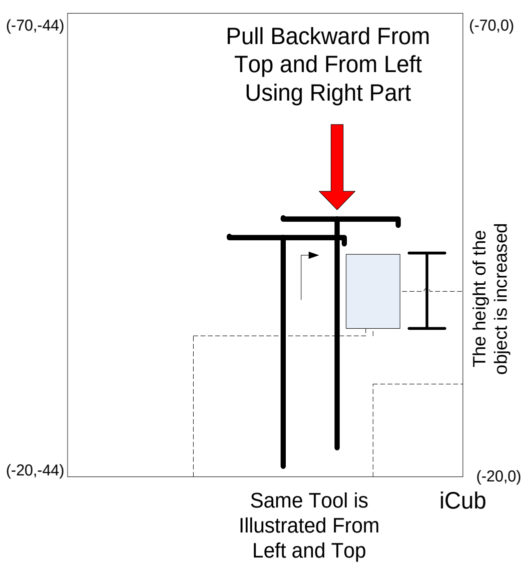
(b) Result Graph

Figure 7.6: Pull Backward Objects Which Are Gradually Placed Far Away to Forward

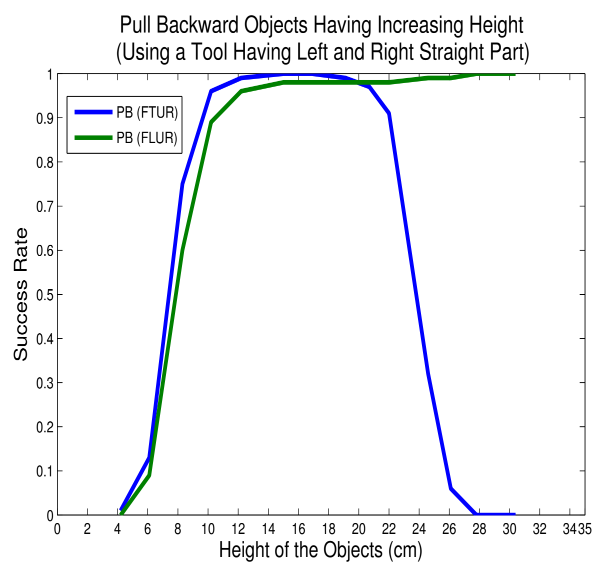
From the graph it can be seen two lines are really similar. However, *pull backward from top using left part* behavior's success rate starts to decrease later than *pull backward from top using right part* behavior. The reason is that since right part points to below, to pull back an object with right part iCub has to bring its hand further than the one while pulling back using left part. Because of this its rate decrease earlier. It is also expected for success rate of *pull backward* *from top using right part* to increase earlier than *pull backward from top using left part* at the beginning positions. As it can be seen, it is like that because of the same reasons but this time, the the part which is pointing below has advantage.

At the end, it was shown that longer left or right down part have better rates in early positions but worse in later positions when compared to shorter left or right down part.

Pulled Backward Affordance is Requested on (T, O*, P) Tuple: In Figure 7.7(a),using a tool with having left and right straight parts, *pull backward from left using right part* and *pull backward from top using right part* behaviors are applied on multiple objects which have increasing heights. The plot in Figure 7.7(b) shows the effect of height of the object on the success rate of these behaviors in order to show the difference of bringing the tool from left and from top.



(a) Illustration of the Behavior



(b) Result Graph

Figure 7.7: Pull Backward Objects Which Have Increasing Height

When the height is too low, iCub cannot bring his hand due to security conditions such as hitting his hand to the table. When the height of the object starts to increase success rates of both behaviors increase. After a point of height, success rate of *pull backward from top using right part* behavior starts to decrease because the height of the objects becomes challenging for iCub to bring the tool from top of the objects. However, using *pull backward from left using right part* behavior, iCub brings the tool from left at the level of center of object way more lower than from bringing from top. Therefore, success rate stays at high levels.

In this experiment, it was shown that increase in the height of an object can prevent to do "bring from top behaviors" but "bring from left" or "bring from right" behaviors are not affected from this increase.

7.3 All Affordances are Requested

In this test, 3 different tools (2 training type tools and 1 novel tool) were used on two different positions with two different objects represented as (T*, O*, P*) tuple. Test tools and locations with objects can be seen in Figure 7.8.

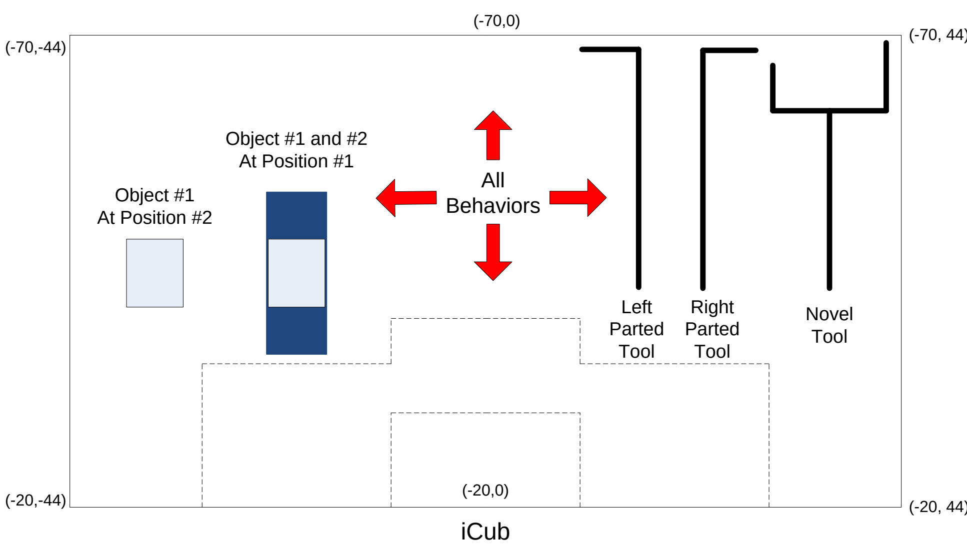


Figure 7.8: Tests With 3 Tools on 2 Objects at 2 Position

1st Case: All Affordances are Requested on Object #1 at Position #1:

Table 7.1: **T*:** Left Straight Parted Tool, Right Straight Parted Tool, Novel Tool **O:** Object #1, **P:** Position #1

(a) Behavior and Affordance Results of Tool With Left Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.98	0.87	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0.72	0.97	0	0	0	0	0	0	0	1
Pushed Forward	0	0	0	0	0.84	0.93	0.01	0	0	0	0	1
Pulled Backward	0	0	0	0	0	0	0	0	0.99	0.99	0	1

(b) Behavior and Affordance Results of Tool With Right Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.99	0.97	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0.01	0.8	0	0	0	0	0	0	0	1
Pushed Forward	0	0	0	0	0.84	0.03	0.39	0	0	0	0	1
Pulled Backward	0	0	0	0	0	0	0	0.96	0	0	0.08	1

(c) Behavior and Affordance Results of Novel Tool

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.95	0.36	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0	0	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0.98	0.23	0	0	0	0	0	1
Pulled Backward	0	0	0	0	0	0	0	0.19	0.99	1	0.1	1

In this experiment, all behaviors were applied on object #1 at position #1 in order to evaluate similarity between these 3 tools based on affordance and behavior results.

In Table 7.1(a), left parted tool is able to achieve all requested affordances. This is also valid for the right parted tool in the second Table 7.1(b). So, these two tools are equivalent tools when affordance results are considered, but when we look at with which behaviors these affordances are risen, they are different and this makes these tools functionally-equivalent. For instance, in first table pushed backward affordance is obtained with PB(FTUL) and PB(FRUL), in the second table this affordance is obtained with a different behavior which is PB(FTUR) as it can be seen with bold boundaries in results.

In Table 7.1(c), it is expected for novel tool to succeed in all affordances, but since its right part is long and when this long length combine with the y position of the object, iCub is not able to accomplish behaviors related with right part as it can be seen in Table 7.1(c). In addition, since tool's main length is shorter than other two tools, iCub cannot manage to push right from top, either. If the object would be a bit more at the right side on the table or right part would be shorter or main length of the tool would be longer then it could be expected for iCub to accomplish one or all these failed behaviors with this novel tool.

2nd Case: All Affordances are Requested on Object #1 at Position #2:

Table 7.2: **T*:** Left Straight Parted Tool, Right Straight Parted Tool, Novel Tool **O:** Object #1, **P:** Position #2

(a) Behavior and Affordance Results of Tool With Left Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	$\boxed{0.95}$	$\boxed{0}$	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0	0	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0	$\boxed{0.83}$	0	0	0	0	0	1
Pulled Backward	0	0	0	0	0	0	0	0	$\boxed{0.87}$	$\boxed{0.94}$	0	1

(b) Behavior and Affordance Results of Tool With Right Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	$\boxed{0.45}$	$\boxed{0}$	0	0	0	0	0	0	0	0	0	0
Pushed Right	0	0	0	0	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0	$\boxed{0}$	0	0	0	0	0	0
Pulled Backward	0	0	0	0	0	0	0	0	$\boxed{0}$	$\boxed{0}$	0	0

(c) Behavior and Affordance Results of Novel Tool

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	$\boxed{0.99}$	$\boxed{0}$	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0	0	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0	$\boxed{0.05}$	0	0	0	0	0	0
Pulled Backward	0	0	0	0	0	0	0	0	$\boxed{0.58}$	$\boxed{0.8}$	0	1

In this experiment, all of the behaviors were applied on object #1 at position #2 as it can be seen in Figure 7.8 in order to examine changes when the position is changed.

In this position we expect, left parted tools to succeed most of the behaviors because the object is placed left region of the table where iCub cannot move and align its hand according to center of the object.

In Table 7.2(a), it can be seen that iCub exploits the left part of the tool and can accomplish almost all affordances except *pushed right* affordance. As it is remembered, in the previous case first two tools were same when the affordances are considered. However, in this situation when the object is placed left, right parted tool lost the advantage of position. Therefore, in none of the affordances, iCub can succeed as it is indicated in Table 7.2(b). So, when both this situation and previous situation is examined, same tools were selected, the object was same, only position is changed and this change affected the second tool completely.

While in the first case these two tools were equivalent based on affordances but in the second case they are completely different because of just a position change. When it is examined behavior based, it can be seen that this difference is because of the left part of the first tool, in Table 7.2(a) left part related behaviors have high success rates which reflects overall affordances.

For the novel tool, it can be seen in Table 7.2(c), third tool is really similar to first tool based on affordances except pushed forward affordance. Again iCub exploits the left part of this novel tool, but because of its upper vertical length of left part is long, iCub has to move its hand way too backwards which is out of region. Therefore, iCub fails to push forward this object with left part.

As a result, it can be said that left parted tools are needed to apply a behavior on object #1 at position #2 as it can be seen from bold resulted behaviors from the tables.

3rd Case: All Affordances are Requested on Object #2 at Position #1:

Table 7.3: **T*:** Left Straight Parted Tool, Right Straight Parted Tool, Novel Tool **O:** Object #2, **P:** Position #1

(a) Behavior and Affordance Results of Tool With Left Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.98	0.01	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0.72	0.86	0	0	0	0	0	0	0	1
Pushed Forward	0	0	0	0	0	0.03	0	0	0	0	0	0
Pulled Backward	0	0	0	0	0	0	0	0	0.93	0.93	0	1

(b) Behavior and Affordance Results of Tool With Right Straight Part

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.99	0.34	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0.01	0.09	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0	0	0	0	0	0	0	0
Pulled Backward	0	0	0	0	0	0	0	0.05	0	0	0	0

(c) Behavior and Affordance Results of Novel Tool

	Push Left		Push Right		Push Forward			Pull Backward				
Effects	PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)	Affordance
Pushed Left	0.95	0	0	0	0	0	0	0	0	0	0	1
Pushed Right	0	0	0	0	0	0	0	0	0	0	0	0
Pushed Forward	0	0	0	0	0.4	0	0	0	0	0	0	0
Pulled Backward	0	0	0	0	0	0	0	0	0.63	0.94	0	1

In this third case, all of the behaviors were applied on object #2 at position #1 as it can be seen in Figure 7.8 in order to examine changes when the property of an object is changed while position is fixed.

This case is very similar to first case, this time position is fixed but object's length is increased by 10cm. From Table 7.3(a), due to the increase at the length of the object, using the first tool and second tool to push forward the object iCub needs to place its hand more way back comparing to first case. Therefore, iCub will not be able to place its hand out of the region. However, when the novel tool is considered since it is shorter as main length it has higher success rate at push forward behavior when push forward behaviors are examined in Table 7.3(c).

Since the object's length was increased, like push forward behaviors, pull backward behaviors were also affected. For example, assume right parted tool in first case, using this tool iCub was able to push the object right from top with 0.8 rate, but in the third case its rate decreased because of the increase of object's length.

In the first case left parted tool and right parted tool are able to pull back the object, in the third case, object's length is increased so it is expected for both tools to able to pull or not since they have the same main length. However, in the third case, while left parted tool still able to pull the object, right parted tool is not. This may be because of the distribution of the data in each behavior and also this instance may be encountered in boundary of the classes.

The results showed that increase in length of the object affects push forward, pull backward and also behaviors that bring the tool from top of the object were affected. When results of these behaviors in these tables are compared with the results of first case, this can be easily seen.

7.4 Similarity Between Tools

Similarity between 20 different tools (13 training type tools, 7 novel tools) will be examined in this test. Position and objects will be same as the previous test type (2 different object and position). 3 different similarity methods will be used.

First of all, simple feature based similarity will be examined, then similarity based on behaviors and lastly similarity based on affordances. They will not only examined in their own methods but also similarity changes between different methods will be examined. The similarities will be examined on the tests which are done on the following objects and positions in Figure 7.9. Object #1 has 11cm width, 19cm height and 7cm length. Object #2 has same width and height but having 17cm length.

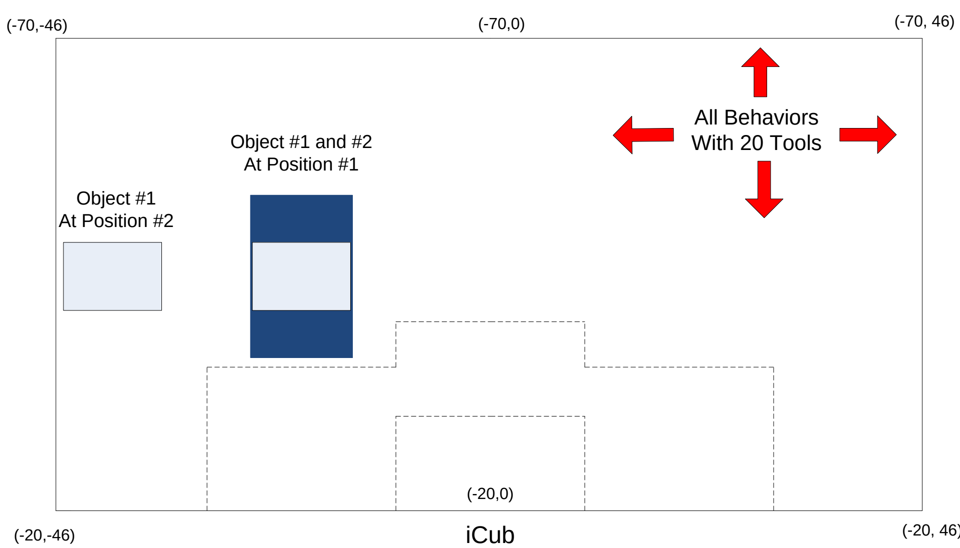


Figure 7.9: Tests With 20 Tools on 2 Objects at 2 Position

Feature Based Similarity:

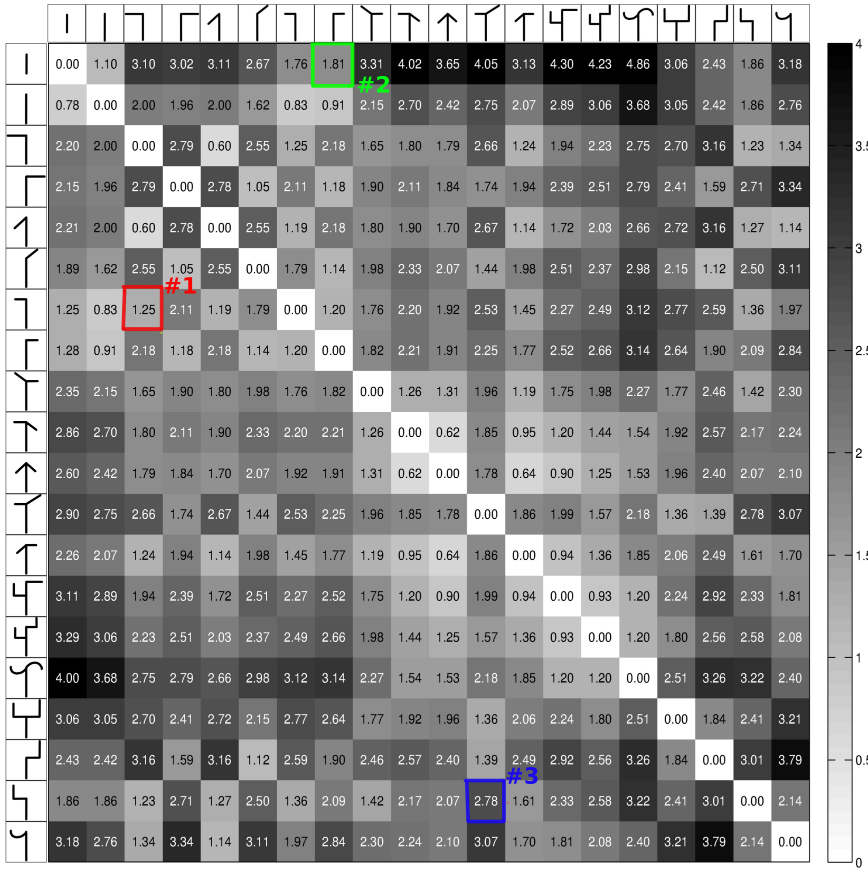


Figure 7.10: Distance Between Tools Based on Euclidean Distance of Features

Table 7.4: Feature Values of Each Tool in the Pairs

		Tool Related Features (cm)											
		main	left	right	left up	left down	right up	right down	eye main	eye left	eye right	left?	right?
#1		25.5	3.06	0	0	3.05	0	0	23.75	4.85	0	1	0
		25.29	10.88	0	0	3.97	0	0	23.86	11.28	0	1	0
#2		20.11	0	0	0	0	0	0	18.13	0	0	0	0
		25.36	0	3.01	0	0	0	3.97	23.48	0	5.24	0	1
#3		22.29	6.62	0	7.79	3.33	0	0	20.58	10.18	0	1	0
		25.13	5.59	10.76	0	3.16	9.96	0	23.82	6.26	14.24	1	1

In this method, the tools are compared according to their distance between features. In other words, this method compares tools according to their appearance. The features which are used for comparison do not consist of object's features since these are not related with affordances.

As it can be seen in Figure 7.10, 3 different pair of tools will be examined among these 20 tools. These pairs are indicated with colored rectangles and numbers next to them. According to values, tools in pair #1 can be said to similar, pair #2 can be said in the middle and lastly pair #3 can be said very different tools.

In the following tests, we will see how may similar or different tools by appearance become very different or similar based on behaviors and affordances. For the aim of comparison, the feature values are given in Table 7.4. Feature based similarity method depends on feature representation, but the following similarity methods depend on the results of the interactions.

1st Case: Behavior Based and Affordance Based Similarity on Object #1 at Position #1:

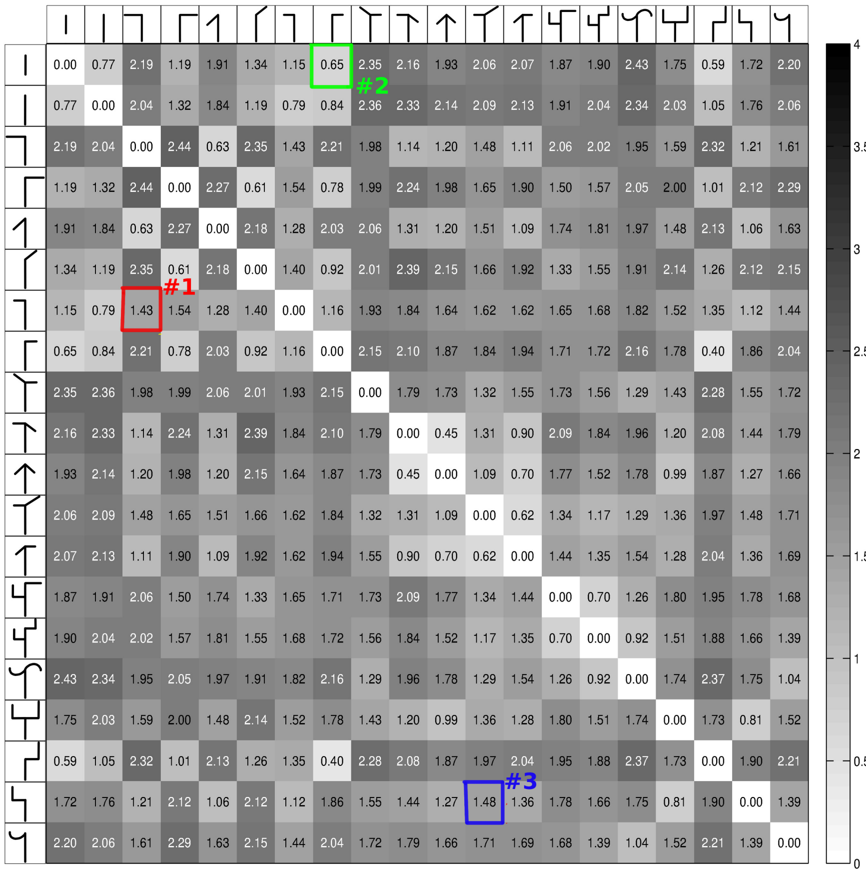


Figure 7.11: Distance Between Tools Based on Euclidean Distance of Behavior Results on Object #1 at Position#1

Table 7.5: Behavior Results (SVM Prediction) of the Pairs of Figure 7.11

		PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)
#1		0.98	0.97	0.48	0.76	0.83	0.05	0	0	0.43	0.48	0
		0.79	0.96	0.48	0.76	0.83	0.96	0	0	0.99	0.96	0
#2		0.99	0.68	0.37	0.16	0.98	0.01	0	0	0	0	0
		0.99	0.98	0.01	0.26	0.86	0	0.05	0.2	0	0	0.05
#3		0.67	0.61	0.49	0.45	0.98	0.09	0	0	0.94	0.96	0
		0.97	0.97	0	0.38	0.85	0.49	0	0.92	0.91	0.85	0

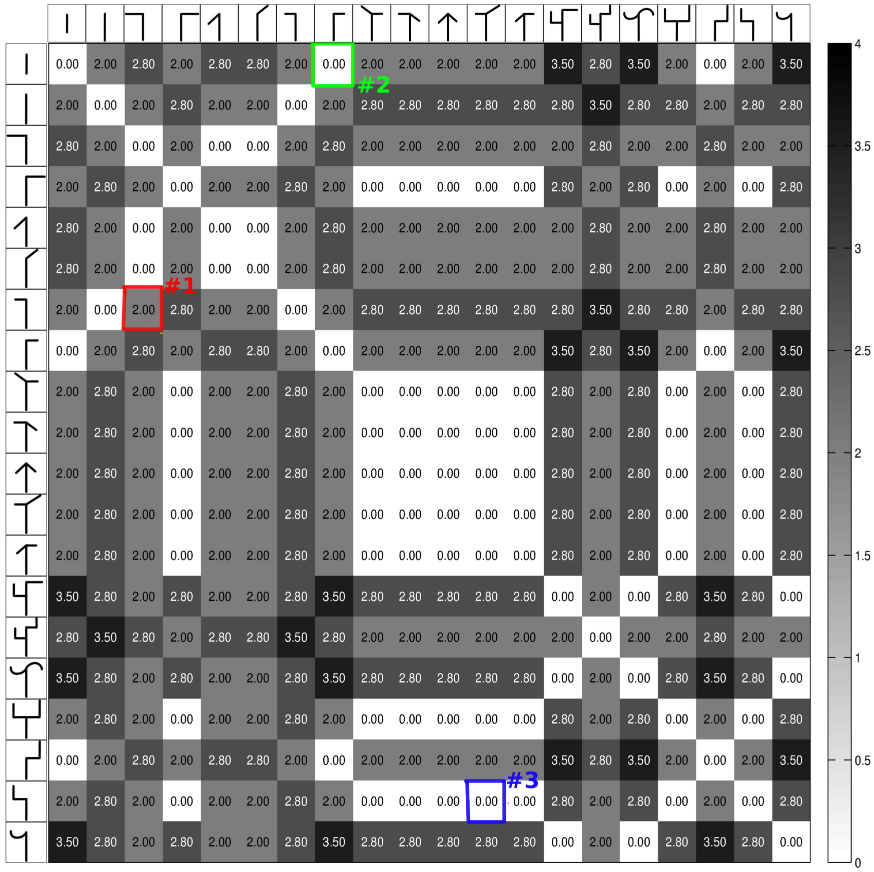


Figure 7.12: Distance Between Tools Based on Euclidean Distance of Affordance Results on Object #1 at Position#1

Table 7.6: Affordance Results of the Pairs of Figure 7.12

		Pushed Left	Pushed Right	Pushed Forward	Pulled Backward
#1		1	1	1	0
		1	1	1	1
#2		1	0	1	0
		1	0	1	0
#3		1	0	1	1
		1	0	1	1

In this part, similarity based on behaviors and affordances of multiple tools are examined on object #1 and position #1 as in Figure 7.9. When behavior based similarities are considered for all pairs, it can be said that tools in each pairs looks similar especially pair #2.

For pair #1, the difference between tools is caused by mostly due to of PF(UL), PB(FTUL) and PB(FRUL). Normally, it is also expected for these behaviors to result similar in each other. However, when length of left parts are compared from Table 7.4, it can be seen that there is almost 7cm difference. This affects the results of the behaviors because the tool with shorter left part is not able to succeed in push forward and pull backward behaviors since its left length is less than half of the object's width. This avoids center of object to move properly. When affordances are considered for this pair, they are found very similar, only the tool with shorter left part will never able to pull back the object, therefore only different affordance is the *pullable backward*.

For pair #2, as it can be seen in Table 7.6, when affordances are considered, these tools are equivalent tools and when behavioral results are considered, tools are same because they achieved same affordances using same behaviors with same tool parts. The tool with right part is expected to manage one of the pull back behaviors but since its length is shorter as in the tool in pair #1, it is not able to pull back the object. Stick has no left or right parts so there is no need to look pull back behaviors for the stick. So, that little difference rises from the rates that can be affected by this short right part.

For pair #3, according to behavioral results from Table 7.5, two tools are not very similar. However, affordance results of these behaviors were considered, these tools are functionally-equivalent because of the PB(FTUR) behavior. For this pair, as it can be seen in Figure 7.10, based on features, these two tools are really different but based on affordances, they are equivalent tools. The difference in the behavioral results are caused by the fact that one of the tool has a right part although this do not reflect the affordances like it makes difference in behavioral results.

2nd Case: Behavior Based and Affordance Based Similarity on Object #1 at Position #2:

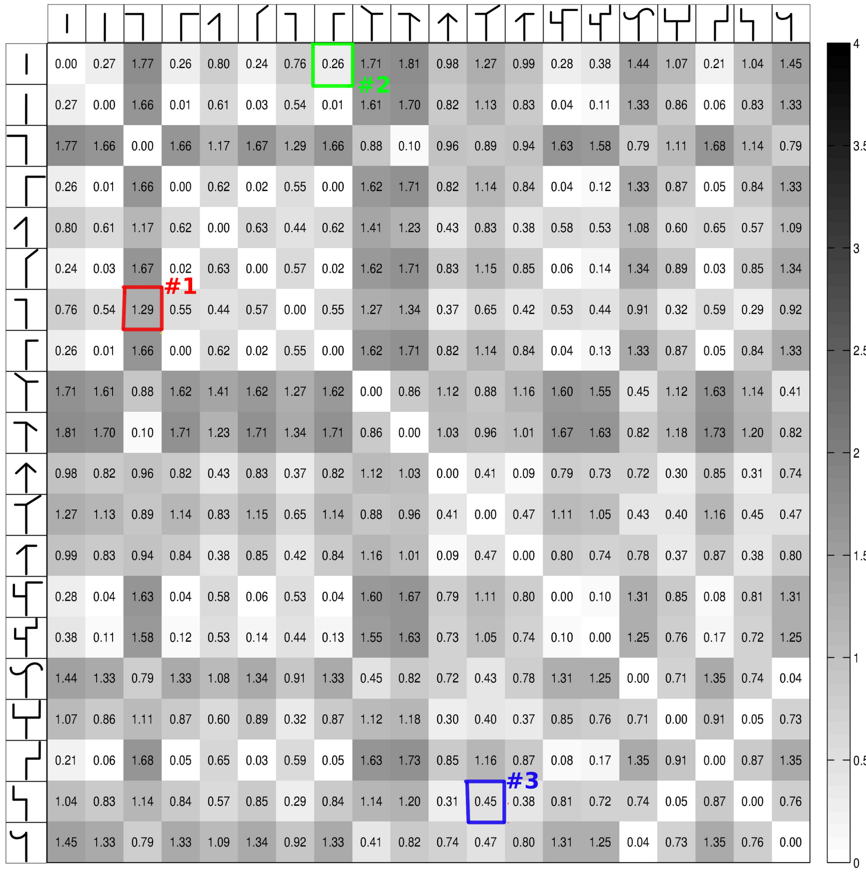


Figure 7.13: Distance Between Tools Based on Euclidean Distance of Behavior Results on Object #1 at Position#2

Table 7.7: Behavior Results (SVM Prediction) of the Pairs of Figure 7.13

		PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)
#1		0.81	0	0	0	0	0	0	0	0.02	0.32	0
		0.97	0	0	0	0	0.65	0	0	0.52	0.98	0
#2		0.27	0	0	0	0	0	0	0	0	0	0
		0.48	0	0	0	0	0	0	0	0	0	0
#3		0.98	0	0	0	0	0.01	0	0	0.04	0.49	0
		0.9	0	0	0	0	0.05	0	0	0.12	0.84	0

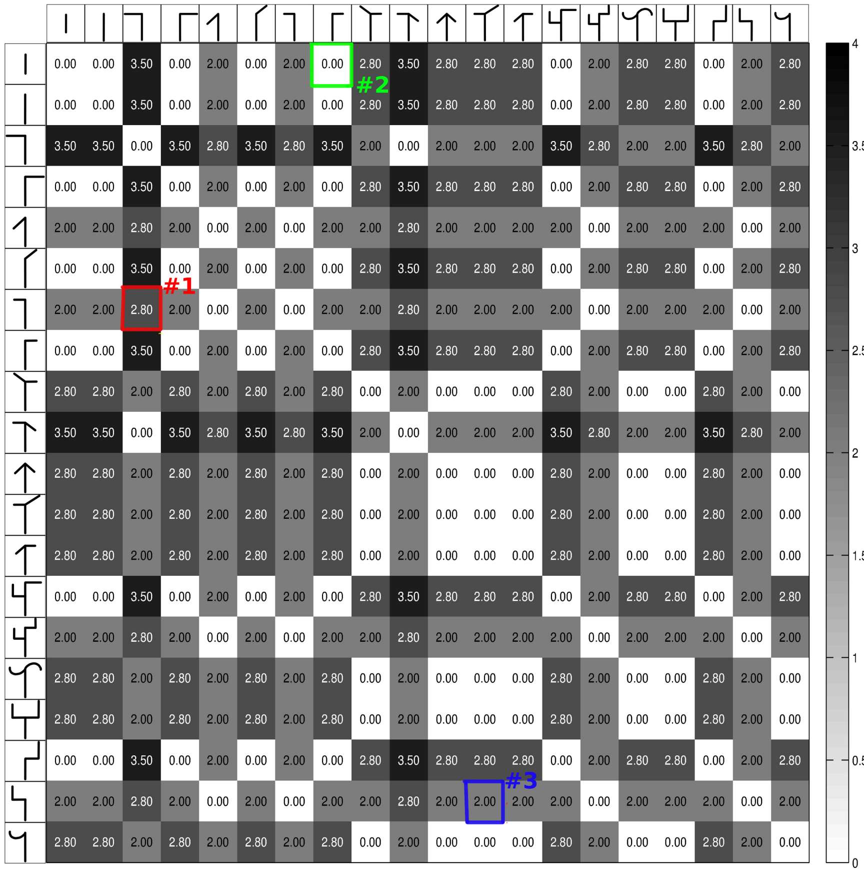


Figure 7.14: Distance Between Tools Based on Euclidean Distance of Affordance Results on Object #1 at Position#2

Table 7.8: Affordance Results of the Pairs of Figure 7.14

		Pushed Left	Pushed Right	Pushed Forward	Pulled Backward
#1		1	0	0	0
		1	0	1	1
#2		0	0	0	0
		0	0	0	0
#3		1	0	0	0
		1	0	0	1

In this case the same object in the first case was taken to the left side of the table as it can seen in Figure 7.9. Since it is on left side where iCub cannot reach or align his hand according to the center of the object, it is expected for left parted tools to have advantage.

For pair #1, when behavioral results are considered, there is not a big change on the similarity from the similarity of first case. For both tools, the success rates of push right behaviors decreased because of the location of the object. In the first case, the tool with shorter left part was able to push the object forward using main part but in this case there is no chance to push it with main part since it is out of the region. This causes the difference at *pushable forward* and *pullable backward* affordances as it can be seen from Table 7.8.

For pair #2, it can be said based on affordance result, these two tools are equivalent tools and based on behavioral results, these two tools are same. There is no left part in any of these two tools. Because of this, iCub cannot apply any behaviors on this object at this location.

For pair #3, when behavioral results are considered, comparing with first case the similarity between the tools were increased. The reason is one of the tools was able to push the object right in the first case but in this case it is for sure for both tools that it is not possible to push the object right at this location. In addition to this, since length of tools' left parts are enough to pull backward or push forward, it may be said that both tools should able to do these behaviors. However, object is not so close the region that iCub can bring his hand. Therefore both of the tools should fail for that behaviors. As it can be seen from Table 7.7, since instances are at the boundary for decision, one of tool's rate is so close to 0.50 percent and this is right but the other one is wrong with higher rate. This wrong result at boundary causes the difference although we expect for them to be same when affordance results are considered for this pair.

3rd Case: Behavior Based and Affordance Based Similarity on Object #2 at Position #1:

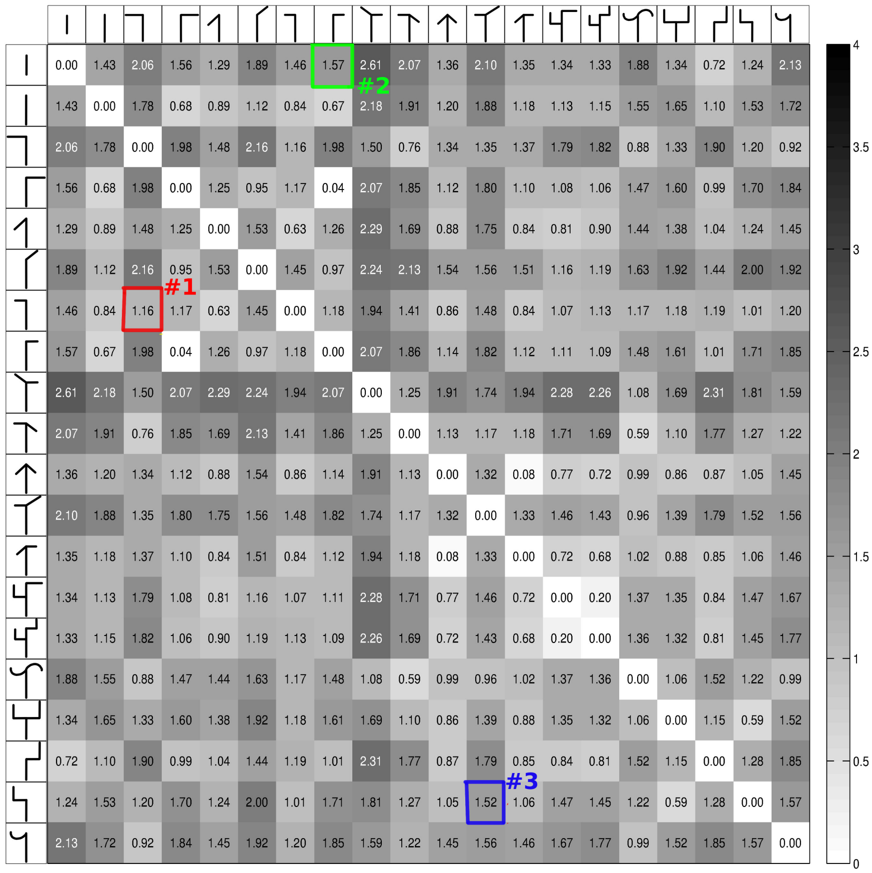


Figure 7.15: Distance Between Tools Based on Euclidean Distance of Behavior Results on Object #2 at Position#1

Table 7.9: Behavior Results (SVM Prediction) of the Pairs of Figure 7.15

		PL (FR)	PL (FT)	PR (FL)	PR (FT)	PF (UM)	PF (UL)	PF (UR)	PB (FTUR)	PB (FTUL)	PB (FRUL)	PB (FLUR)
#1		0.98	0.29	0.48	0.42	0	0	0	0	0.13	0.52	0
		0.79	0.11	0.48	0.41	0	0.11	0	0	0.95	0.94	0
#2		0.99	0.01	0.37	0	0.96	0	0	0	0	0	0
		0.99	0.82	0.01	0.01	0	0	0	0	0	0	0
#3		0.67	0	0.49	0.03	0.52	0	0	0	0.22	0.84	0
		0.97	0.22	0	0.07	0	0.01	0	0.87	0.65	0.87	0

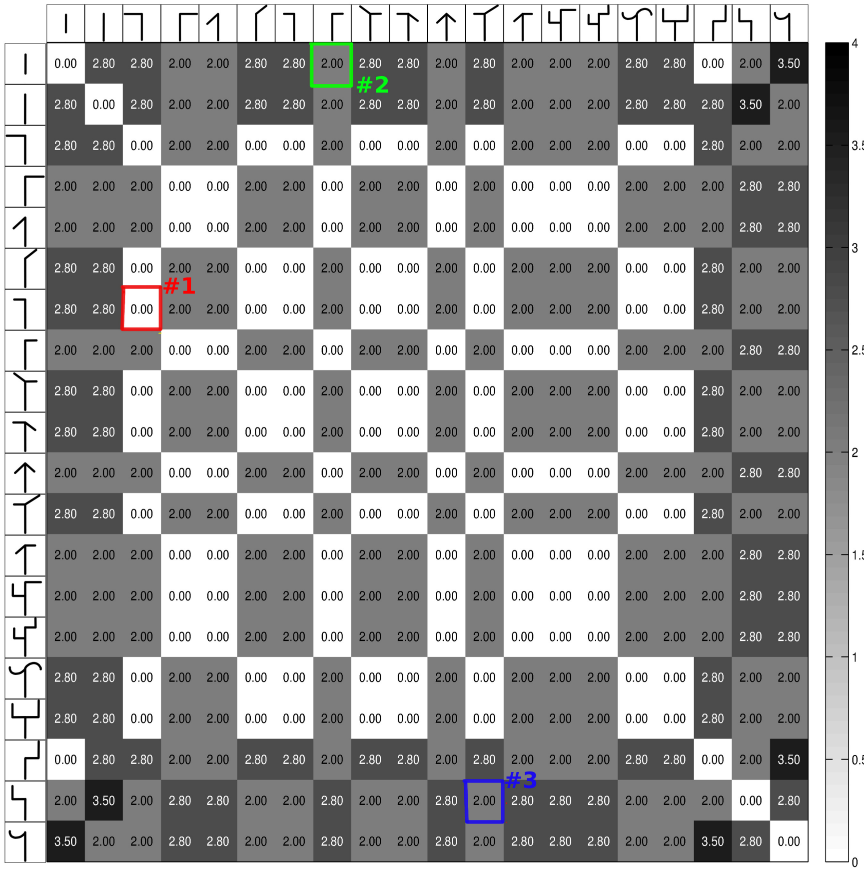


Figure 7.16: Distance Between Tools Based on Euclidean Distance of Affordance Results on Object #2 at Position#1

Table 7.10: Affordance Results of the Pairs of Figure 7.16

		Pushed Left	Pushed Right	Pushed Forward	Pulled Backward
#1		1	0	0	1
		1	0	0	1
#2		1	0	1	0
		1	0	0	0
#3		1	0	1	1
		1	0	0	1

In this third case, behaviors were applied on an object which is similar with the object that is in first and second case. The only difference is its length is increased by 10cm. Because of this increase, it is expected for push forward and pull backward behaviors to be affected and push left and push right behaviors to stay same. However, there are bring from top versions of push left and push right behavior so they can be affected from this length change of the object.

For pair #1, it can be seen from Table 7.9, the rates of push right and push left behaviors when compared to first case, the rates of PL(FR) and PR(FL) are unchanged but because of the length increase of the object the rates of PL(FT) and PR(FT) decreased as it is expected. Push forward behaviors are also affected from this change since iCub needs to bring his hand from more behind than in the first case which iCub cannot do because of the available region. When affordances are considered, *pushable right* and *pushable forward* affordances are seem to be affected as it can be seen from Table 7.10 when compared to first case. In this third case, this pair is equivalent based on affordances and functionally-equivalent when behavioral result are considered. The reason of being functionally-equivalent tools is PB(FTUL) behavior.

For pair #2, PL(FR) and PR(FL) behaviors were not affected although PL(FT) and PR(FT) are affected. However, affordances *pushable left* and *pushable right* are same with first case. When push forward behaviors are considered, it can be seen from Table 7.9, length increase of object affected the tool with right part and it lost this push forward ability. However, as it can be seen in Table 7.4, stick is shorter than the tool with right part, this shortness gives it an advantage and it is still able to push forward the object. They both still cannot pull the object backward. So, when affordances are considered these two similar tools in the first case are differentiated by *pushable forward* affordance in this third case.

For pair #3, like in all pairs in this case, the behaviors related with bring the tool from top are affected. Both tools have same rates for PL(FR) and PR(FL) but the rates of from top versions of these behavior decreased as it is guessed. Since the tool with just left part is shorter than the one with both parts, it is able to still push forward the object like the stick in pair #2. They can still manage to pull backward from left or right. So, when the affordances are considered, shortness of one tool makes the difference at *pushable forward* affordance.

7.5 Demonstration on iCub

- **Grasping the Tool:**

Figure 7.17: Grasping the Tool

- **Push Left From Right (PL-FR):**

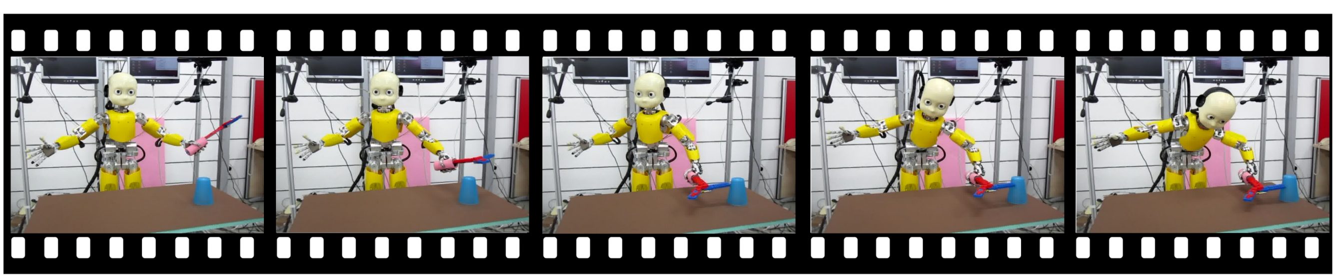


Figure 7.18: Push Left From Right Demonstration

• **Push Left From Top (PL-FT):**

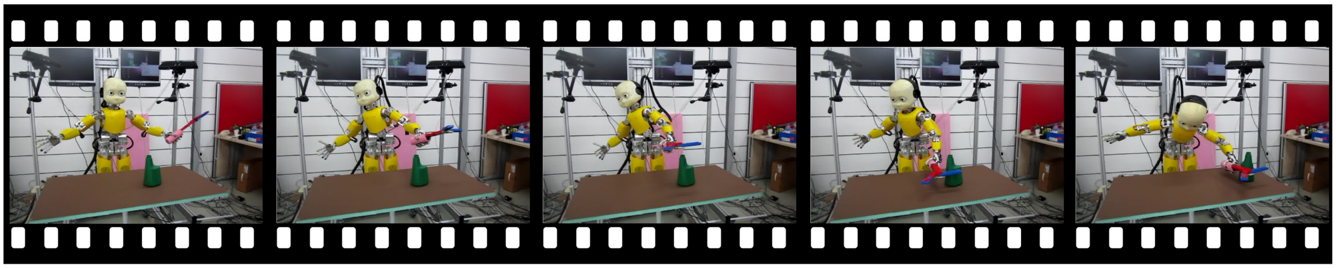


Figure 7.19: Push Left From Top Demonstration

• **Push Right From Left (PR-FL):**

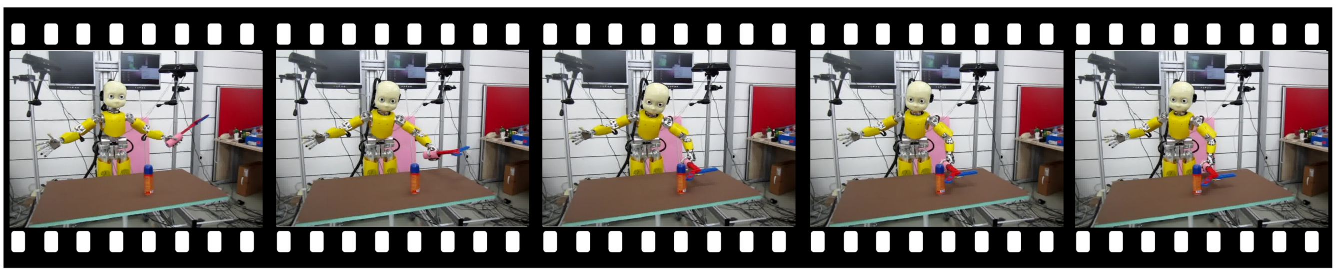


Figure 7.20: Push Right From Left Demonstration

• **Push Right From Top (PR-FT):**

Figure 7.21: Push Right From Top Demonstration

- **Push Forward Using Main Part (PF-UM):**

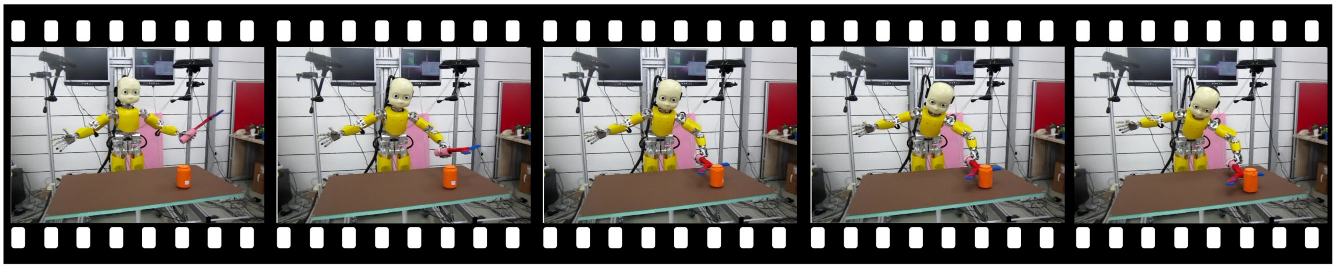


Figure 7.22: Push Forward Using Main Part Demonstration

- **Push Forward Using Left Part (PF-UL):**

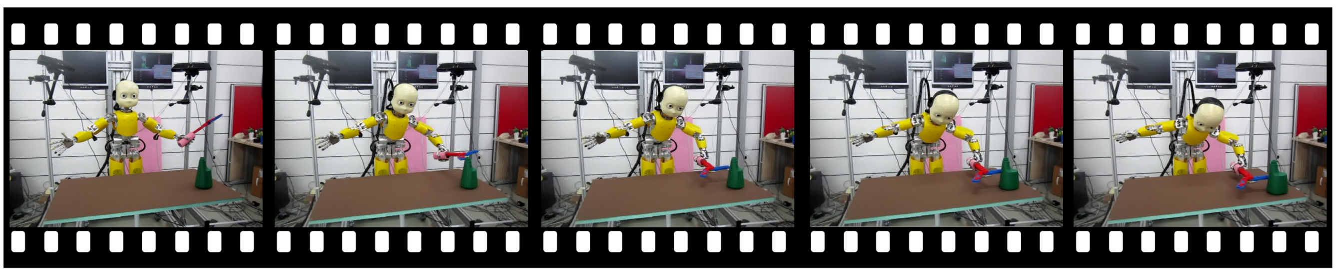


Figure 7.23: Push Forward Using Left Part Demonstration

- **Pull Backward From Left Using Right Part (PB-FLUR):**

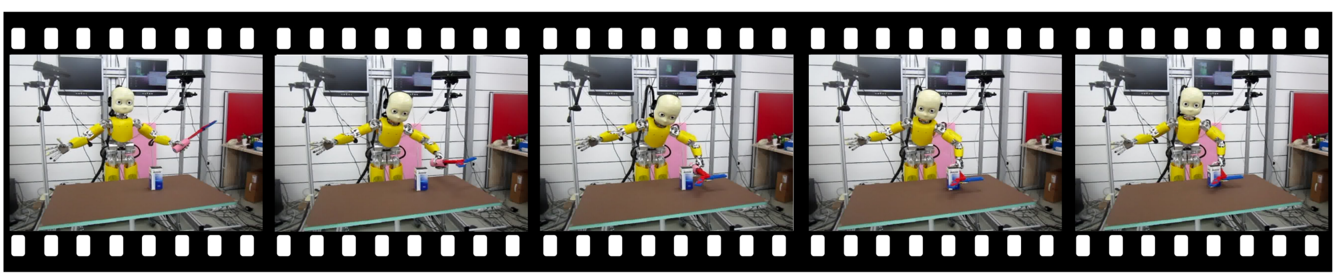


Figure 7.24: Pull Backward From Left Using Right Part Demonstration

- **Pull Backward From Top Using Right Part (PB-FTUL):**

Figure 7.25: Pull Backward From Top Using Right Part Demonstration

- **Irreguler Center of Mass Movement:**

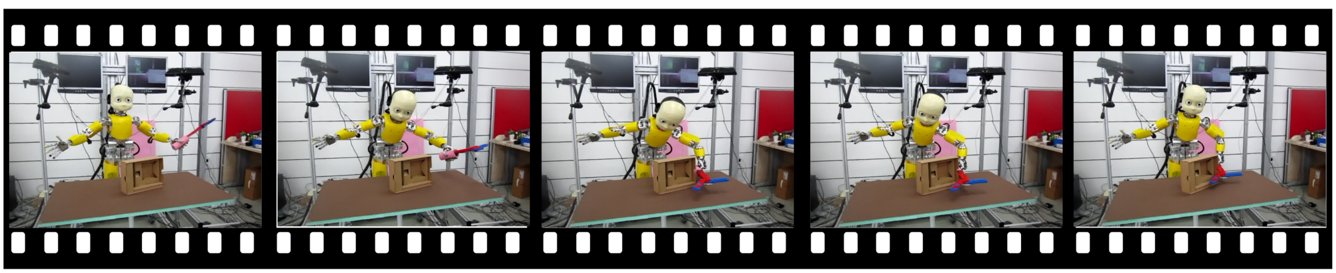


Figure 7.26: Irregular Center of Mass Movement Example

Chapter 8

Conclusion

In this work, we presented tool affordance concept on a humanoid robot platform. This concept is explained using the humanoid robot iCub, different tools which have different parts with different lengths and objects which iCub applied the behaviors on. For the aim of better understanding of behaviors and reveal dependencies between features, a feature kernel was used to combine different features. In order to find and select the feature set which will represent a behavior best, an elimination was done among ranked features after ReliefF feature selection. Using the remaining features, behavior models were trained. Using these behaviors model, we showed that:

- Robot successfully predicts effects of different behaviors when a tool and object is given.
- When a tool and object are given, robot can infer the affordances.
- When a novel tool is given and an object is shown, it can successfully apply behaviors and can predict the results.
- We showed that the robot can apply the suitable behavior when an affordance is requested given a tool and an object.
- We showed that the robot can select the best tool among different tools when an affordance is requested given an object.

- We showed that how the affordances and behavior results are affected when a part of the tool is removed, modified or a new part is added.
- We showed that how different positions and properties of the objects affect the affordances and behavior results.
- We showed that similar/different tools in a situation may become very different/similar in another situation when affordances or behavior results are considered.
- We showed that similar/different tools by appearance may become very different/similar in same situation when affordances or behavior results are considered.

8.1 Limitations of The System

- **Limited Area That iCub Can Move Its Hands:** iCub is not a mobile robot, it is fixated at his waist. Because of this immobilization and for security considerations, a region was determined for iCub to move its hands. However, this region was not so helpful for flexibility of some behaviors.
- **Tool Repertoire With up to One Branch Point:** Most of the tools were designed as having at most one branch point. However, this limitation can be easily extended to multiple branch point by some adjustments. However, this may require some work on shape analysis on both left and right part of the tools.
- **Hand Orientation:** While iCub was holding the tool and interacting with its environment, his hand was always in the same orientation. A changeable orientation may help system to compensate some problems due to properties of some objects. For instance, when iCub encounters with an object which is really short, because of the fix orientation, he needs to bring the tool to the same level with the object's center, but in order to prevent a crash between hand of the iCub and the table, this behavior fails. If the orientation would

be changeable, iCub could do his behavior from above by changing the orientation of holding the tool.

- **Extra Features For Objects:** The objects which were used for interactions are seen by iCub as a box with height, width and length at a location in the world. For example, surface features can be used while features were extracting from objects. These features may help iCub to apply behaviors on the objects more precisely, may gain iCub object representation models. This may lead combination of tool affordances with noun and adjective concepts.

- **Combination of Behaviors:** Combination of different behaviors may be developed in order to succeed in a requested behavior. For instance, given an object at a position, iCub may not pull back the given object at that position but may be he can pull back if the object is positioned to the left. So, iCub can succeed in this behavior by first pushing the object to left and then pulling back achieves the request.

8.2 Advantages of The System

- **Novel Tool Prediction:** In this work, one of our aims was to teach iCub the relation between tools and objects. By doing this, iCub does not abide only the tools he has seen. Therefore, when a novel tool is encountered whatever its properties are, iCub is able to use it and can predict its affordances when an object is given.

- **A Good Simulation of Tool Usage in Humans:** This work can be seen as a good simulation of tool usage in humans. Since, every possibility is in the system. For example, sometimes when we want to reach something using a tool, we can think that we can reach that object, but when we try to reach, we may not succeed. This is valid also for iCub, he may say "yes, I can push forward this object", but when he tries the object may not pushed forward. Another example, when we want to pull back an object, we search for a tool that has a branch. That is what iCub does using the trained models. When a stick is given to iCub and request from him to pull back the object, he has the experience that he cannot accomplish this because of the structure of the tool.

Bibliography

[1] F. Guerin, N. Kruger, and D. Kraft, “A survey of the ontogeny of tool use: fromsensorimotor experience to planning,” *Autonomous Mental Development, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2012.

[2] R. W. Shumaker, K. R. Walkup, and B. B. Beck, *Animal Tool Behavior: The Use and Manufacture of Tools by Animals*. The Johns Hopkins University Press, revised and updated edition ed., Apr. 2011.

[3] R. S. Amant and T. E. Horton, “Revisiting the definition of animal tool use,” *Animal Behaviour*, vol. 75, pp. 1199–1208, 2008.

[4] V. K. Bentley-Condit and E. O. Smith, “Animal tool use: current definitions and an updated comprehensive catalog,” *Behaviour*, vol. 147, 2010.

[5] J. Piaget, *The Origins of Intelligence in Children*. New York: Norton, 1952.

[6] K. W. Fischer, “A theory of cognitive development: The control and construction of hierarchies of skills,” *Psychological Review*, vol. 87, pp. 477–531, 1980.

[7] J. J. Gibson, *The Ecological Approach to Visual Perception*. 270 Madison Avenue, New York, NY I 0016: Taylor and Francis Group, LLC, 1986.

[8] E. Şahin, M. Çakmak, M. R. Doğar, E. Uğur, and G. Üçoluk, “To afford or not to afford: A new formalization of affordances towards affordance based robot control,” *Adaptive Behavior*, vol. 15, pp. 447–472, dec 2007.

[9] A. Pellicano, S. Thill, T. Ziemke, and F. Binkofski, "Affordances, adaptive tool use and grounded cognition," *Frontiers in Psychology*, vol. 2, no. Article 53, pp. 1–2, 2011.

[10] M. Costantini, E. Ambrosini, C. Sinigaglia, and V. Gallese, "Tool-use observation makes far objects ready-to-hand," *Neuropsychologia*, vol. 49, no. 9, pp. 2658–2663, 2011.

[11] J. K. Witt, D. R. Proffitt, and W. Epstein, "Tool use affects perceived distance, but only when you intend to use it," *Journal of Experimental Psychology-human Perception and Performance*, vol. 31, pp. 880–888, 2005.

[12] Y. Shinchi, Y. Sato, and T. Nagai, "Bayesian network model for object concept," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2, pp. II–473–II–476, april 2007.

[13] T. Nakamura and T. Nagai, "Object concept modeling based on the relationship among appearance, usage and functions," *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 5410–5415, oct. 2010.

[14] H. Attias, "Inferring parameters and structure of latent variable models by variational bayes," *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 21–30, 1999.

[15] J. Sinapov and A. Stoytchev, "Detecting the functional similarities between tools using a hierarchical representation of outcomes," *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on*, pp. 91–96, aug. 2008.

[16] Y. Caliskan, P. Duygulu, and E. Sahin, "Affordance prediction of hand tools using interactive perception," *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pp. 1–4, april 2012.

[17] J. Sinapov and A. Stoytchev, "Learning and generalization of behavior-grounded tool affordances," *Development and Learning, 2007. ICDL 2007. IEEE 6th International Conference on*, pp. 19–24, july 2007.

[18] A. Stoytchev, "Learning the affordances of tools using a behavior-grounded approach," *Proceedings of the 2006 international conference on Towards affordance-based robot control*, pp. 140–158, 2008.

[19] G. Metta, P. Fitzpatrick, and L. Natale, "Yarp: Yet another robot platform," *International Journal of Advanced Robotics Systems, special issue on Software Development and Integration in Robotics*, vol. 3, no. 1, 2006.

[20] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[21] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, 2011.

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, Nov. 2009.

[23] V. N. Vapnik, "The nature of statistical learning theory," 1995.

[24] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[25] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Commun. ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[26] K. Kira and L. A. Rendell, "A practical approach to feature selection," pp. 249–256, 1992.

[27] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Mach. Learn.*, vol. 53, pp. 23–69, Oct. 2003.

[28] I. Kononenko, "Estimating attributes: Analysis and extensions of relief," pp. 171–182, 1994.

Appendix A

Algorithms and Techniques

A.1 Details of the Skeletonization

A.1.1 Conditions for Removing a Contour Point

One of the important features of this algorithm [25] is to preserve the connectivity. For this aim this algorithm is divided into two subiterations. In the first subiteration, a contour point P_i is deleted if it satisfies the following conditions:

- $2 \leq B(P_1) \leq 6$
- $A(P_1) = 1$
- $P_2 * P_4 * P_6 = 0$
- $P_4 * P_6 * P_8 = 0$

In the second subiteration, first and second conditions stay same, only third and fourth condition changes into the following:

- $P_2 * P_4 * P_6 = 0$

- $P_2 * P_6 * P_8 = 0$

The P values in conditions are the pixels in 3×3 window which can be seen in Table A.1. Since, this process is applied on binary images a pixel value can be only 1 or 0. A(P_1) in the conditions is the number of 01 patterns in the order of P_1, P_2, P_3, ..., P_8, P_9. B(P_1) is the number of nonzero neighbours of P_1.

P_9 (i-1, j-1)	P_2 (i-1, j)	P_3 (i-1, j+1)
P_8 (i, j-1)	P_1 (i, j)	P_4 (i, j+1)
P_7 (i+1, j-1)	P_6 (i+1, j)	P_5 (i+1, j+1)

Table A.1: Nine Pixels in 3x3 Window

If any one of the conditions given above is not satisfied, then pixel P_1 is not deleted from the picture. The flowchart of the algorithm can be seen in Figure A.1.

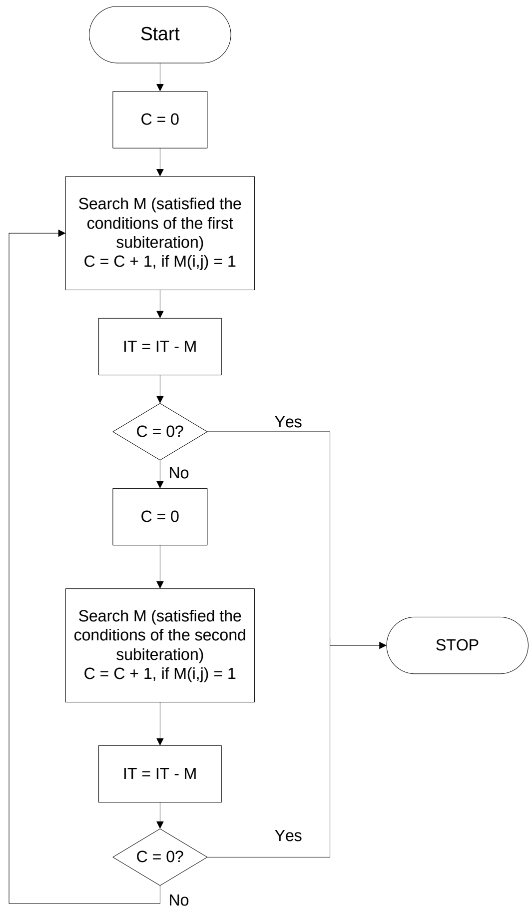


Figure A.1: Flowchart of the Thinning Algorithm

A.2 ReliefF

ReliefF is a multiclass quality feature estimator algorithm. This algorithm mostly used for preprocessing of data by ranking the quality features in the dataset. In addition, this feature estimator algorithm can detect the conditional dependencies between features. Its original algorithm which is called Relief was first presented by Kira and Rendell [26]. However, this version was just for classification problems with two classes. After this, extension of Relief which is called ReliefF was presented by Kononenko [27]. ReliefF is an improved version of Relief that is robust, can deal with noisy and inconsistent data. It is not limited with two class problems, it can deal with multi-class problems. The pseudocode of ReliefF algorithm can be seen in Algorithm 1.

Input : for each training instance including attribute values and the class value

Output: the vector W of estimations of the qualities of attributes

set all weights $W[A] := 0.0$;

for $i := 1$ **to** m **do**

  randomly select an instance R_i;

  find k nearest hits H_j;

  for *each class* $C \neq$ *class*(R_i) **do**

    from class C find k nearest misses $M_j(C)$;

  end

  for $A := 1$ **to** a **do**

    $W[A] := W[A] - \sum_{j=1}^{k} \text{diff}(A, R_i, H_j)/(m \cdot k) +$

    $\sum_{C \neq class(R_i)} [\frac{P(C)}{1-P(class(R_i))} \sum_{j=1}^{k} \text{diff}(A, R_i, M_j(C))]/(m \cdot k)$;

  end

end

Algorithm 1: Pseudocode of ReliefF Algorithm [27]

In ReliefF algorithm, first all weights of the attributes are initialized to 0. Then a random instance is selected from the set. According to k value, k nearest hit and also k nearest miss for each class which is not same with the class of R_i are selected. Basically, for an attribute a if value of another attribute in R_i is very different from attribute value a in one of instance which is hit, then this is not a desirable thing then weight of a is decreased. If value of an attribute in R_i is different from attribute value a in one of the instances which is miss, then this is a desirable thing so weight of a is increased. In the first case attribute a was separating the instances which are in the same class therefore its weight is decreased, but in the second case attribute a was separating the different classes which is good therefore weight of attribute a is increased. This is done k times for each class of misses which are different than the selected instance's class and k times with same class of instance's class. When the classification is a multi-class problem as it can be seen in Algorithm 1 probability estimates are also taken into account.

In this work, ReliefF algorithm is selected because it is an improved version of Relief and can be deal inconsistent data. Like most of the people, ReliefF algorithm is used in this work as a preprocessing of data. Features are ranked and some of the features are cut off from a determined threshold for using for the training of a model. In our work threshold is determined according to highest SVM performance. k value is selected as 10 which is found the best value for most of the problems [28].