DEVELOPMENT OF SENSORIMOTOR ORGANIZATION GUIDED BY LEARNING PROGRESS AND PREDICTABILITY

by

Serkan Buğur

B.S., Computer Engineering, Boğaziçi University, 2015

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2019

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Assist. Prof. Emre Uğur for his support and guidance throughout my master's student. His very kind support, advice, encouragement, and motivation helped me a lot to be able to complete this master thesis work. I also would like to thank Assoc. Prof. Erhan Öztop and Dr. Yukie Nagai for their support and contribution to my research. My sincere gratitude to Assist. Prof. İnci Ayhan for kindly accepting to be in my thesis jury.

My sincerest gratitude to Erhan Çağırıcı, Hakan Girgin, Melisa İdil Şener, Mert İmre, Mert Tiftikci, Onur Poyraz for their friendship and support anytime, anywhere. I feel privileged to be able to work with the members of the COLORS research group. Many thanks to Ahmet Ercan Tekden, Mete Tuluhan Akbulut and Yunus Şeker for the academic discussions, and joyful coffee breaks. Last but not least, I would like to thank my dearest friends, Buğrahan Memiş, İsmail Güvenç, and Serkan Özer.

Finally, I would like to thank my parents, my brother & and his wife for their genuine support and encouragement during my whole life. I would have never been able to persevere in my education without them.

This research has received fundings from Bogazici Research Fund (BAP) project IMAGINE-COG++ (Grant No: 18A01P5), JST CREST "Cognitive Mirroring: Assisting people with developmental disorders by means of self-understanding and social sharing of cognitive processes", Japan (grant no: JPMJCR16E2).

ABSTRACT

DEVELOPMENT OF SENSORIMOTOR ORGANIZATION GUIDED BY LEARNING PROGRESS AND PREDICTABILITY

This thesis proposes an effective action parameter exploration mechanism that enables efficient discovery of robot actions through interacting with objects in a simulated table-top environment. For this, the robot organizes its action parameter space based on the generated effects in the environment and learns forward models for predicting consequences of its actions. Following the Intrinsic Motivation approach, the robot samples the action parameters from the regions that are expected to yield high learning progress (LP). In addition to the LP-based action sampling, our method uses a novel parameter space organization scheme to form regions that naturally correspond to qualitatively different action classes, which might be also called action primitives. The proposed method enabled the robot to discover a number of lateralized movement primitives and to acquire the capability of prediction the consequences of these primitives. Furthermore, our findings have some parallels with data from infant development, and might explain the reasons behind the earlier development of grasp compared to push action in infants, and the reasons behind the correspondence between development of action production and prediction.

ÖZET

DUYU-MOTOR ORGANİZASYONUNUN ÖĞRENME GELİŞİMİ VE ÖNGÖRÜLEBİLİRLİK PRENSİPLERİ ARACILIĞIYLA GELİŞİMİ

Bu tezde masa üstüne yerleştirilen nesneler üzerinde keşif yapan bir robotun eylem parametrelerini efektif bir şekilde düzenleyerek, farklı robot eylemlerini keşfedebileceği bir hesaplamalı model önerilmektedir. Bu amaç özelinde simüle edilmiş bir robot ortamı kullanılmış, ve robot eylem parametre uzayını, eylem sonucu gözlemlenen sonuçlara göre organize etmektedir. Aynı zamanda robot, kendi eylemlerinin sonuçlarını tahmin edebilmek adına farklı modeller de eğitmektedir. İçsel motivasyon yaklaşımıyla robot eylemlerini potansiyel olarak yüksek öğrenme gelişimi verecek bölgelerden seçmektedir. Öğrenme gelişimi tabanlı eylem seçimine ek olarak, yapılan çalışma sonucu oluşan eylem parametre bölgeleri farklı eylem sınıflarına tekabül etmektedir ve bunlara basit eylemler adı verilmektedir. Yapılan çalışma sonucu, robot zorluk seviyesi giderek artan basit eylemler öğrenmiş ve bu eylemlerin henüz tamamlanmadan sonuçlarını tahmin edebilme yeteneğini geliştirmiştir. Bunun yanında, elde edilen sonuçlar çocuk gelişimi alanındaki bulgularla paralellik göstermekte, çocuk gelişiminde kavrama eyleminin neden itme eylemine göre daha erken öğrenildiğine dair fikir verebilmektedir. Son olarak, bir eylemi yapabilecek yeteneklere sahip olmak ile aynı eylemin sonuç tahmininin yapılabilmesi arasındaki ilişkiyi de açıklayabilmektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

C_x	Effect category with number x
f	Transformation function
R	Real Number
R_x	Region with number x
R_t^{sel}	Selected region at learning step t
$M(t)$	Motor parameter space
$SM(t)$	Sensorimotor paramater space
t	Learning step
w	Smoothing paramater
$\gamma_i(t)$	Prediction accuracy of the region
τ	Time window
θ	Reach action parameter

LIST OF ACRONYMS/ABBREVIATIONS

3D	Three Dimensional
ANN	Artifical Neural Network
CB-IM	Competence-Based Intrinsic Motivation
CD	Corollary Discharge
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DOF	Degree of Freedom
GPU	Graphics Processing Unit
IAC	Intelligent Adaptive Curiosity
KB-IM	Knowledge-Based Intrinsic Motivation
LDA	Linear Discriminant Analysis
LP	Learning Progress
LPM	Learning Progress Motivation
LPPA	LP and Predictability Based Region Sampling and Splitting
LSTM	Long Short-Term Memory
MNS	Mirror Neuron System
ROS	Robot Operating System
V-REP	Virtual Robot Experimantation Platform

1. INTRODUCTION

Predicting the consequences of one's own actions is an important requirement for intelligent control and decision making in both biological and artificial systems. Neurophysiological data suggests that human brain benefits from internal forward models that continuously predict the outcomes of the generated motor commands for trajectory planning and movement control [1]. For higher-level cognitive functions, behavioral data suggest that forward prediction is used to generate plans to achieve different goals [2]. Infants start (learning of) predicting events and consequences of their actions in early ages. Predictive ability is argued to already exist in 3-month old infants who can smoothly track sinusoidal targets through their eyes [3], and in 6-month old infants who can track objects that move behind occluders [4] or predict target objects of grasp actions of others [5]. On the motor side, starting from a reflex like a grasp behavior in 0-2 months, intentional power grasp develops in 4-6 months [6], and for a typical infant, in order to reach for an object with correct hand orientation, 9 months should have been passed after birth [7]. Furthermore, infants start learning the causal relationship and the dynamics of the objects related to their actions while developing the motor skills correlated to the actions [8].

It is reasonable to say that infants continuously interact with the environment, perform some actions, observe the outcomes of their actions, and correlate the outcomes to the performed actions. How the motor and prediction capabilities develop, on the other hand, depends on the way infants interact with the environment. One prominent view posits that babies are endowed with an internal drive, named intrinsic motivation (IM) that allows them to actively choose the interactions they engage in for maximizing the speed and the extent of learning [9].

In this thesis, we propose a developmental progression applicable to a robot that interacts with its environment (Figure 1.1), which takes inspiration by infant development. Starting with one parametric reach action, the robot organizes its action parameter space based on the generated effects in the environment and learns forward

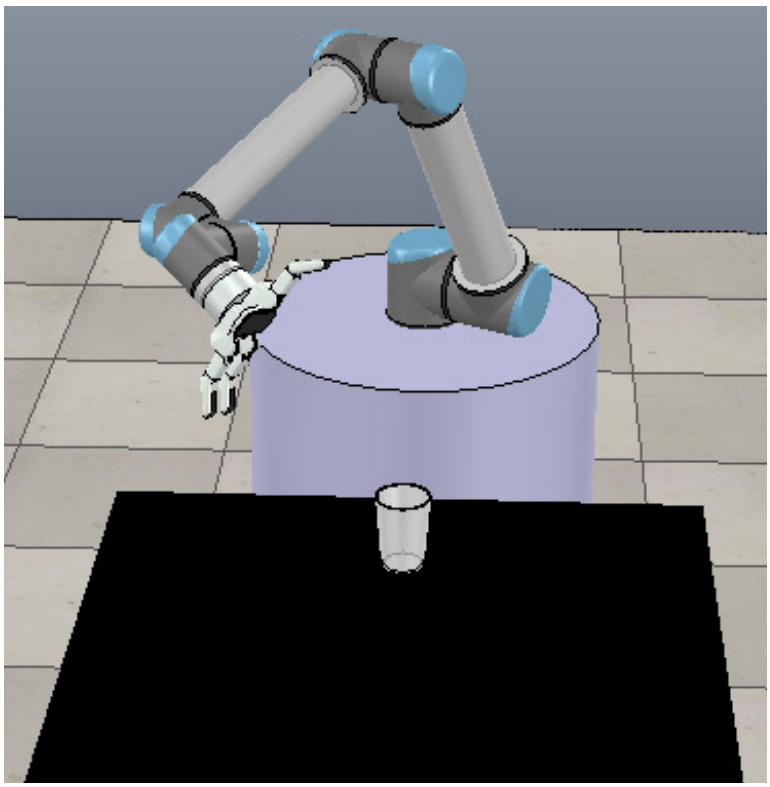


Figure 1.1. A simulated manipulator robot interacting with an object using its 3-fingered adaptive gripper.

models for predicting consequences of its actions. Following the IM paradigm, the robot samples the action parameters from the regions that are expected to yield high learning progress (LP). In addition to the LP-based action sampling, our method uses a novel parameter space organization scheme to form regions that naturally correspond to qualitatively different action classes, which might also be called action primitives.

We implemented the proposal on a simulated robotic arm and gripper (Figure 1.1), and found that lateralized grasp and push primitives emerge along with the continual sample-execute-learn-reorganize process. We observed an earlier developmental progression for the grasp action; i.e., forward models for predicting the external and the bodily consequences of grasp actions emerged earlier. To assess the parallels between actual infant development and the workings of our system, we compared our results with the behavioral data available in the literature on infant action prediction development. As a result, we found similar developmental timelines in motor and prediction capabilities of infants and our simulated system.

It is possible to summarize the contributions of this thesis as follows. First, the action parameter space is partitioned subject to the developing capability of forward prediction of action consequences, which leverage the benefit of LP-based action sampling for further speedup in learning. Second, the action parameter sampling is done through a special latent space that represents the action parameters with a topological structure shaped by the effects generated by the actions. Finally, our system provides a developmental progression that has parallels with the infant data [5], where the correspondence between some features of action prediction and execution can be observed.

Overview of the literature and the background information required to comprehend this thesis completely will be given in Chapter 2. In Chapter 3 the workstation we have used, and software entities will be introduced. In Chapter 4, previous works that inspired our study and gave us the motivation will be introduced. Chapter 5 provides the previous work we have been working on as a starting point for this thesis. Then the proposed model will be introduced in Chapter 6 whereas the details about the experimental setup will be given in Chapter 7 and the results will be provided in Chapter 8. Finally, discussions about this thesis will take place in Chapter 9 and the conclusions derived will be given in Chapter 10.

2. BACKGROUND

In this chapter, an overview of the critical software components we have used will be provided mostly from cognitive and robotics perspectives. This section is divided into several sections in which at each section, one of the software components will be explained in detail.

2.1. Artificial Neural Network

Artificial neural networks (ANN) are inspired by the physical phenomena of animal brains [10]. They are designed to copy how humans learn throughout the time into a machine learning system. Given the input and output data, the main aim is to learn a function that maps the given input to the right output. In order to handle this task, ANNs may have not only input and output layers but also some hidden layers. In each layer, depending on the structure, a different number of artificial neurons exist. The programmer of the ANN decides on the activation function for the layers. Therefore depending on the activation function, each of the artificial neurons in the network fires or not.

In most cases, ANNs are used in the field of pattern recognition and machine learning. Developing a specific algorithm for recognizing different patterns is a cumbersome task for a programmer. It requires knowledge about the distribution of the data as well as the correlation between the input and output data. However, in some cases, prior information about the data is hard to get. Therefore, instead of developing specific algorithms for each pattern, programmers started to use ANNs for pattern recognition.

ANNs are used in the machine learning field broadly. Classification, clustering, and prediction are the main categories in which ANNs are used. ANNs are used in the industry as well. From making autonomous cars to machine translation, fraud detection, to developmental learning in the robots, ANNs are commonly used.

2.2. Autoencoder

Autoencoders are used in unsupervised machine learning field in which ANNs are leveraged for the task of representation learning [11]. Autoencoders are useful when the input feature size is high, that prevents efficient and effective learning when a standard ANN is used. It is meaningful to squeeze the feature size if possible, without losing important information. Therefore a forward mapping from high dimensional data to low dimensional data and also a backward mapping (reconstruction) from low dimensional to high dimensional data are required. However, if the features of the input data were not correlated, the conventional methods such as Principal Component Analysis or Linear Discriminant Analysis would not be useful. However, autoencoders yield sufficient forward and backward mapping results even in the case of non-correlated input features. The most important features of an autoencoder are the followin g:

- Accuracy of mapping low dimensional data to high dimensional data is elevated, and autoencoders are sensitive to the changes in the input data.
- Prevents overfitting the training data by not memorizing the training data directly.
- Extracts the significant features finds the correlations between them. Also discards unimportant features which would introduce redundancy.

In the robotic domain, most of the time, the visual perception of the robot is provided via a camera. If traditional ANNs were to be used, the whole image would be input to the ANN. However, this would slow down the learning since the image size would be, and most of the time, only some part of the image changes. Therefore researchers use autoencoders in order to process visual data primarily.

2.3. Long Short-Term Memory

The machine learning applications have gained a lot of traction in recent years in several categories:

- Identification of images, or simply how to identify an image on the internet as containing a cat, for example.
- Sequence to sequence translation, or how to convert the speech into a text and how to translate one language to another.

The former tasks are mostly done with convolutional neural networks (CNN), and the latter is realized with recurrent neural networks (RNN), of which especially the Long Short-Term Memory (LSTM). Hochreiter and Schmidhuber [12] designed LSTM to tackle the challenge of long term dependencies. Because RNNs use recent information heavily, and not relies on long term dependencies. However, LSTMs keep the sequence information longer. In most cases, LSTMs are fed time series data and expected to predict the future of the data.

2.3.1. Structure of the LSTM

Different memory blocks called cells and gates are chained in a structured way in a typical LSTM. The cells keep the relevant information, whereas the gates decide on which information to keep, how to update the cells. Three types of gates exist, which are input, forget, and output gates.

2.3.1.1. Input Gate. The input gate is responsible for deciding what new information will be allowed to be stored in the memory cell. Some entries will be more relevant than the others. Those that are relevant will be stored in the memory cell that will be used for decision making at time t. First a sigmoid function filters the values to be kept using the inputs h_{t-1} and x_t. Then, using a tanh function, a vector is created, which will include all possible values from h_{t-1} and x_t.

2.3.1.2. Forget Gate. Forget gate is used to remove the information that is no longer needed. Input at the specific time, which is x_t and the most recent cell output, which is h_{t-1} are input to the gate. As like the feed forward phase of a typical ANN, these two inputs are fed to the network with an addition of bias term. Finally, the result of the multiplication is fed to an activation function in which the decision of forgetting or keeping the information is made.

2.3.1.3. Output Gate. The output h_t will depend on the information present in the memory cell as well as the incoming information x_t in the LSTM layer. Initially a sigmoid function filters the values to be kept using the inputs h_{t-1} and x_t. Then, using a tanh function, a vector is created, which will include all possible values from h_{t-1} and x_t. Finally, a multiplication operation is performed between the vector values and the sigmoid filtered value to get the related information. Output gate is responsible for the calculations and outputting the present information.

Some applications of LSTM include language modeling, machine translation, image captioning, robot navigation planning and mapping, handwriting generation, and question answering chatbots.

2.4. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is used commonly as a dimensionality reduction technique in the field of machine learning [13]. Projecting the data onto a lower dimensional space is the main goal without losing significant information. Mapping the data onto a lower dimension reduces the computational costs as well as is helpful to avoid overfitting. And also, in most cases speeds up the learning by correlating the features of the model with each other and reducing the feature size.

In the case of two predictors, LDA creates a linear boundary along a chosen axis in the variable space (plane, since there are two predictors). The predictors are generally multivariate vectors in this scenario. The axis is chosen such that the projections of

the variable along that axis falls the two sides of the axis such that the projections falling on one side represent one category of the independent vector. To find out the axis, training data is used. In that data which observation vector is falling in which category is already known. The analysis is done in the following sequence:

(i) Calculate means for every category of the group of observations.
(ii) Calculate variance-covariance matrices for individual groups.
(iii) Calculate projection y, which represents the best axis on which we will take the projections.
(iv) Calculate midpoint between the groups falling along the axis
(v) Calculate discriminant function and classify the training observations.

In terms of statistics, there are two underlying assumptions in LDA: First, data is Gaussian distributed with a group/category and second multivariate vectors show the same trend of variation within a group. LDA perform the best when these two assumptions hold.

3. EXPERIMENT PLATFORM

In this chapter, in order to better understand the implementation details and experiments, all the components of the platform will be explained. In section 3.1, the physical components will be explained, whereas in section 3.2, we will explain the software components.

3.1. Physical Components

3.1.1. UR10 Robot

In the experiments, we have used the UR10 robot of Universal Robots [14]. UR10 is a collaborative industrial robot arm designed for tasks which require high precision and reliability. It has a payload up to 10 kg. With a reach radius of up to 1300mm, the UR10 industrial robot is used in collaborative processes such as packaging, palletizing, assembly, and pick and place. It has a single 6 degrees of freedom (DOF) arm, meaning that the arm includes 6 joints. Moreover, each of the joints is of rotational joint type. Since in free space, at least 6 DOF is required to be able to reach every location in the working space, UR10 is capable of reaching every point in its reaching radius. In contrast to a human arm for all the positions in the workspace, there is only a single solution for the joint values to take. By this means UR10 has no kinematic redundancy.

3.1.2. Robot Gripper for Manipulation Tasks

A 3-finger adaptive robot gripper is attached to the robot. The gripper we use is developed by Robotiq [15] and is called *3-Finger Adaptive Robot Gripper* (Figure 3.1). This gripper has three fingers, each having three joints, as shown in Figure 3.1. This gripper is chosen for the ease of use and the flexibility to select between different built-in grip types. The gripper has 4 grip types, which are basic, wide, scissor, and pinch modes. According to the manual, it is possible to grip a huge number of objects by using the basic mode. However, for small objects, pinch and scissor modes are more

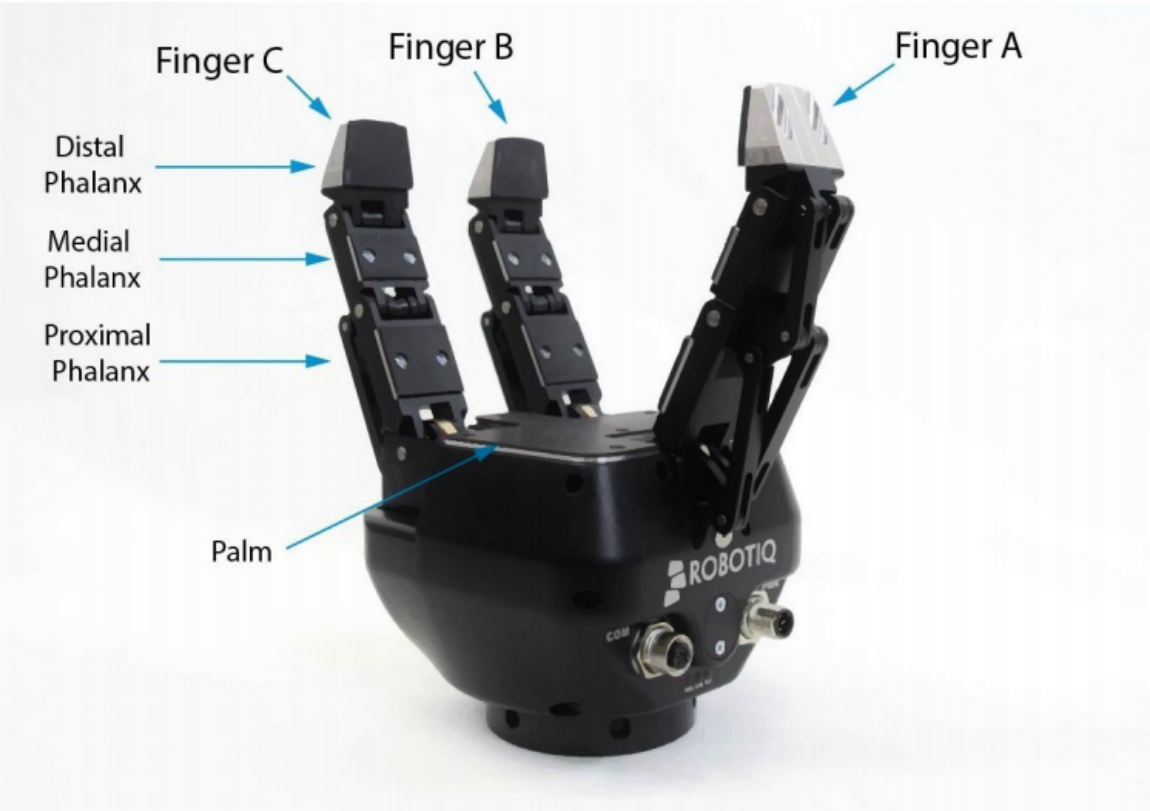


Figure 3.1. Three finger adaptive robot gripper. Depicted from [15].

useful. Using the pinch mode ensures that the object is gripped between two fingers. Scissor mode is more precise than the other three modes. A built-in force/torque sensor comes with the gripper which could be used to better adjust the grip depending on the object material, size, and texture. In our experiments, we have used the basic mode.

3.2. Software Components

3.2.1. Robot Operating System

The Robot Operating System (ROS) is a software framework meant to allow you to write applications which operate robotic hardware [16]. As its most fundamental level, it is an abstraction layer that offers hardware abstraction, so the guts of controlling the hardware components are not required to be known by the programmer. These look like software drivers and libraries. They work because the hardware has the hooks the drivers are looking for. To make it more useful, ROS then offers a set

of baseline and user-contributed functions and applications against that abstraction. ROS is open source so that it is possible to build a robotic setup that is compliant to the specifications or expand the architecture to improve the services it provides. These are built within a software architectural framework which allows you to pass machine state and messages between operating nodes. Executable softwares called nodes are communicated over topics. Topics are used to send or receive messages between nodes. It is possible to use predefined message types in ROS directly. Also, it is possible to create custom messages. ROS supports two widely used programming languages, which are C++ and Python. For kinematic control of the robot and executing the robot actions, we have used C++ whereas, in the learning part, we have used Python.

3.2.2. V-REP

V-REP is a 3D robot simulator which can be used to visualize robots controlled using ROS [17]. It is a collection of a scene, a model editor, a library for different kinds of models and meshes. V-REP Player is a computer-aided design tool that allows you to create, develop, and simulate any robotic project. With an integrated programming environment, V-REP Player is built around a control-command architecture. Therefore, it is possible to individually control each object or prototype from an embedded script, a plugin, a ROS node, or another custom solution. Also note that commands can be written in C / C ++, Python, Java, Lua, Matlab, Octave or Urbi.

3.2.3. External Library and Packages

Here the external libraries and packages used in the learning part of our model are presented and explained in detail.

3.2.3.1. Keras. Keras is a high-level neural network API, written in Python and interfaceable with TensorFlow, CNTK and Theano [18]. It has been developed with the aim of allowing rapid experiments. Researchers widely use Keras for the following reasons. It enables fast and easy prototyping (due to its ease of use, modularity,

and scalability). Supports both convolutional networks and recurrent networks as well as a combination of both. Works seamlessly on Central Processing Unit (CPU) and Graphics Processing Unit (GPU). The basic data structure of Keras is a model, a way of organizing layers. The simplest type of model is the sequential model, consisting of a linear stack of layers.

3.2.3.2. Scikit Learn. Scikit-learn is a free software machine learning library for the Python programming language [19]. Scikit-learn can be used as a middleware for prediction tasks. For example, a large number of web startups are using Scikit-learn to predict user buying behavior, offer product recommendations, or detect trends and misbehavior. Scikit Learn is used to extract the structure of complex data (texts, images) and to classify them using techniques corresponding to state of the art. It is easy to use, efficient, and accessible to non-data science experts. Scikit-learn does not come with a graphical interface, and it's a prediction engine. Scikit learn is developed in open source, and available under the BSD license.

4. RELATED WORK

4.1. Intrinsic Motivation

Intrinsic motivation is defined by Ryan and Deci [9] as "the inherent tendency to seek out novelty and challenges, to extend and exercise one's capacities, to explore and to learn". It has an internal locus of causality (i.e., the agent's perception of the cause of success or failure) and is regulated by interest, enjoyment and inherent satisfaction consequently seems to be one of the essential causes of self-determined behaviors. As the intrinsic motivation promotes the learning in humans, it took attention of the researchers from the artificial intelligence domain as an alternative to the use of external signals for learning.

In a detailed review of existing computational approaches to intrinsic motivation, Oudeyer, and Kaplan [20] divided the approaches into two main categories. The first one, *competence-based intrinsic motivation* (CB-IM) stems from the competency measures that are obtained by the achievement of the agent's goals and the second one, *knowledge-based intrinsic motivation* (KB-IM) relies on the discrepancy between reality and the agent's expectations. Our proposed work can be considered in the scope of KB-IM approaches.

In one of the pioneer work [21] in knowledge-based IM, Oudeyer *et al.* argued that in order to foster effective, efficient and compelling learning in high-dimensional search spaces, intrinsically motivated exploration strategies could be used. To this end, they introduced the Intelligent Adaptive Curiosity (IAC) framework that drives the agent to maximize the "learning progress" that maximizes the change in prediction performance of the learners. Oudeyer and Kaplan [20] described "Learning Progress Motivation (LPM)" within the KB-IM that motivates the agent to decrease the prediction errors by using *learning progress* as the intrinsic motivation signal.

In order to enhance sampling efficiency and facilitate open-ended learning without external rewards, Hester and Stone [22] combined two intrinsic motivation signals, particularly the variance of the predictions and novelty. Similar to [22], Sequeira *et al.* [23] combined numerical correspondences of some emotional appraisal dimensions, namely, novelty, motivation, control, and valence as a source of intrinsic motivation. Both of these studies combine the different components of the intrinsic reward linearly. While the former finds the weights of the different components empirically, the latter optimizes them according to the agent's fitness to the environment.

Santucci *et al.* [24] proposed an architecture called GRAIL that includes a competence based IM signal allowing the agent both to select from the existing goals and to generate new goals. In their experiments, a simulated humanoid robot that aims to learn how to reach targets that are located in different places in front of it.

Hart *et al.* [25] demonstrated a framework that allows hierarchical structuring of control knowledge by means of the intrinsic motivation signal. In their experiments, they showed three learning stages of a manipulator robot. One of these stages was intended to make a grasp action. Temel *et al.* [26] introduced "frustration" and "impulsiveness" concepts into an intrinsically motivated reinforcement learning setup and used their proposed method for the grasp-learning task. In their work, IM stems from frustration-based action selection mechanism that is regulated by the impulsiveness of the robot. In our previous work [27], we used learning progress to determine which action to explore next and diversity maximization to select the training objects in an affordance learning setup.

In general, the performance of the learning machines on the continuous search space highly depend on the state-action pair distribution. In most of the scenarios, unfortunately, the sensorimotor space has a highly heterogeneous distribution of the state-action pairs, and calculating learning progress on the whole search space causes the signal to be unstable. Therefore, grouping similar state-action pairs is suggested in [21] as a simple but powerful approach to deal with the instability problem. Starting from a single group, space was partitioned into groups based on the distribution of the

learning points. In our current study, instead of using point distribution, space is partitioned maximizing the performance of the learning machine.

4.2. Development of Reaching and Grasping

Six key computational ingredients for the development of reaching and grasping were identified in a detailed survey by Caligiore and Baldassare [28]: ecological active vision; motor babbling and associative learning processes; trial and error learning; hierarchical control architectures; embodiment; and finally the intrinsic motivation. They argue that intrinsic motivation leads the infant's cumulative learning of a large collection of skills in an adaptive manner. Gaussier *et al.* [29] emphasized processing of multi-modal information through their neuro-computational model in explaining reaching and grasping skills of infants. They proposed that the cortical memory may be organized as a reachable region map through the emergencies obtained from the body signals during the sensorimotor exploration. These body signals are the joints between the shoulder and wrist; tactile feedback from the hand; vision from the eye and, the sound of an object after contact. These aforementioned learning mechanisms may be modulated or bootstrapped by reinforcement learning guided by external rewarding stimuli, such as "joy of grasping" [30].

In our previous study [31], we proposed a three-staged developmental framework that is inspired from infant development. In the first stage, the robot was shown to discover movement primitives such as push, carry, and release through clustering the tactile feedback obtained during the execution of a swing behavior. The discovered grasp and push primitives were used to learn object affordances and goal-based action execution in the second stage and for learning a complex sequence of primitives through imitation and parental scaffolding in the last phase. Different from the previous work, the exploration of the robot is now guided by intrinsic motivation, and the primitives are formed in isolation in one-shot but learned in a hierarchical progression that is determined by the effect prediction capability.

4.3. Correspondence Between Action Production and Action Prediction

In our work, the action ability of our system develops together with the capability that enables prediction of action consequences. It is argued that others' actions can also be understood through a direct matching with self, which is believed to be realized by the mirror neuron system [32].

It is proven that mirror neurons are fired when a specific action is performed by a person as well as when the person monitors someone else performing the same or similar action.

Interestingly, behavioral data suggest that such a mapping is not hardwired and probably depends on the motor development of the agent. For example, Sommerville *et al.* [33] showed that the infants could infer the goals of others' actions only if they develop the capability of executing the corresponding actions themselves.

Kanakogi *et al.* [5] further supported this developmental account by conducting an experiment that involves examination of the infant's eye movements. In their experiments, infants were shown a hand reaching to objects in grasp posture, in push posture, and with an unfamiliar tool. The experiments showed that the infants exhibit predictive eye movement towards the target objects only after they learned to execute the grasp and push actions themselves, after 6 and 10 months, respectively. This data suggests that a direct correlation between action prediction and the capability to perform the same or similar action exists developmentally.

Motivated from infant studies, Copete *et al.* [34] implemented a predictive learning based computational model and showed that the development of action prediction is correlated with the development of producing the same or similar action for a robot. In our study, on the other hand, our robot actively explores its action space, discovers action regions during development, and selects which actions to learn through intrinsic motivation. Interestingly, such a developmental approach generated motor learning stages similar to infants. Furthermore, the findings of our study, where the robot learns

predicting the rest of the hand trajectories have parallels with this experimental data as we discuss in Chapter 9.

5. PRELIMINARY WORK

5.1. Motivation

Understanding and predicting the goals of others' actions play a crucial role in human interaction. Starting from the first year of life, infants learn to predict the goals of others' actions swiftly and with high accuracy even though the actions performed quickly with no apparent explanation. How do infants perceive the environment and associate goals to observed actions of others before completion? Which abilities should infants have learned in order for them to infer the goals of others' actions? Before starting, one caveat is that by "other's actions" we specifically refer to object-directed reaching actions performed by an individual other than the observer. An object is a visually observable stuff that an experimenter is able to grasp and push, such as a ball or a can.

Neuroscientific studies performed on humans and animals showed that action context consists of five broad categories which enable future action prediction: the experimenter performing the action, the observer, the environment, the target object, and the object approach [35].

For the remaining of this work, the observer refers to a model of an infant who watches the experimenter performing an action. Rajmohan *et al.* [36] stated that a direct matching process is required to understand others' actions, which is believed to be implemented by the mirror neuron system (MNS). Mirror neurons that are the core of MNS fire when a specific action is performed by a person as well as when the person monitors someone else performing the same or similar action. Sommerville *et al.* [37] supported the idea functioning of MNS by showing that only the infants with the capability of producing the observed action themselves, are capable of inferring the goal of the observed actions. Kanakogi *et al.* [38] further supported this idea by conducting an experiment and by examining the infant eye movements. They have divided the actions performed by the experimenters into three types. In the first type,

infants are shown that an experimenter reaches for an object and grasps the object (GH - goal directed action). In the second one, again, an experimenter reaches for an object. However, this time, the experimenter uses back of his/her hand and does not grasp the object (BH - non-goal directed action). In the third action type, the experimenter uses a mechanical claw to reach for and grasp the object (MC - inanimate goal directed action). Neither group of the infants has prior experience with the BH and MC conditions. Therefore, those types of actions were not predicted by infants. They also stated that for a four-month-old infant, the skills required to perform grasping actions are lacked; hence, they did not make any predictive gazes towards the target object of the experimenter's reaching action. However, starting from the age of six-months, infants could perform one-handed grasping and were able to make predictive gazes towards the target object of the GH condition before completion of the reaching action. They concluded their experiment by saying that action prediction and the skills required to perform the same or similar action are correlated developmentally. Figure 5.5(a) shows the results of their experiment. Furthermore, Copete *et al.* [39] implemented a computational model and showed that for a robot, a strong correspondence exists between action prediction ability development and the ability to produce the same or similar action. They also proved that introducing motor signals improves the prediction of others' goal-directed actions.

How a target object is approached plays a crucial role for the infants to attend to the action. Hogan *et al.* [40] proposed minimizing jerk, in which humans move their hands toward the target points by using the rate of change of acceleration. They stated that between the start and end points, a straight line hand trajectory is formed with a symmetric bell-shaped velocity throughout the motion. Daum *et al.* [41], conducted an experiment to show that infants older than six-months are capable of completing the trajectory of the experimenter's hand in the case of goal-directed action.

Besides the direction of motion, after birth and before the first year of life, infants start to attend to the posture of the hand of the experimenter. [38, 42] showed that infants made predictive gazes toward the target object in the case of grasping reach, whereas when the back of the hand is used, they did not respond. [43] also showed

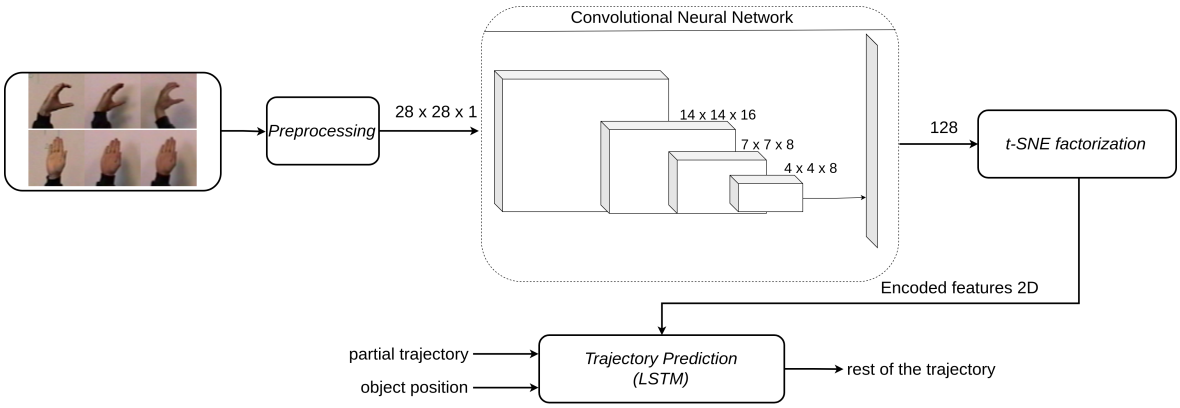


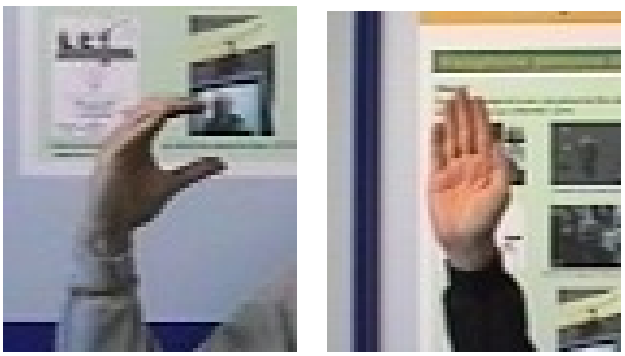
Figure 5.1. Preliminary work model architecture

that while an experimenter is performing a grasping action, infants made predictive gazes toward the target objects whereas, in the case of non-purposeful actions, infants followed the hand of the experimenter with a delay.

5.2. Model

In this preliminary work, we would like to show the prediction of others' actions developmentally, specifically for grasping and pushing actions. For both of the actions, reaching trajectories towards the target objects are similar [44]. However, infants learn to predict grasping actions of others earlier than pushing actions [38, 42]. Since the trajectories are similar in both actions, the hand posture seems to be the key to differentiate grasping from pushing. What we aim to demonstrate is that even the generated trajectories for both of the actions are similar, progressively as an infant learns from his/her experiences, he/she will predict the goals of others' grasping actions rather than the pushing actions.

We start by generating several trajectories following a basic parabola formula. We use a dataset generated by Sebastian Marcel [45]. The dataset includes six different hand posture images from ten people, including the grasping and pushing hand postures. An example of a grasping and pushing hand can be seen from Figure 5.2. We train an autoencoder with the images from grasping and pushing postures, and apply t-SNE factorization [46] for dimensionality reduction, hence reducing feature size to only two. Then we, train two LSTM (Long short-term memory) networks with the



(a) grasping hand (b) pushing hand

Figure 5.2. Examples from the dataset [45].

trajectories and hand posture features in order to predict both the next trajectory point and the target point of the trajectory. We assume that an object is placed at the end of each trajectory sequence. Therefore if the model predicts the end point of the trajectory as the target point with a small error margin, we say that it predicts the action correctly. Figure 5.1 shows the overall model architecture. In the following subsections, we will present more details of each module of the model.

5.2.1. Trajectory Generation

We generated 2000 2D trajectories in which 2D corresponds to x and y coordinates respectively, and each trajectory has 50 2D pairs in it. We randomly select a start and target point and fill the points in between those by applying the following formula:

$$ax^2 + bx + c \tag{5.1}$$

We generate all the trajectories by randomly selecting values for the parameters $a,\ b,\ c$, and start and target points.

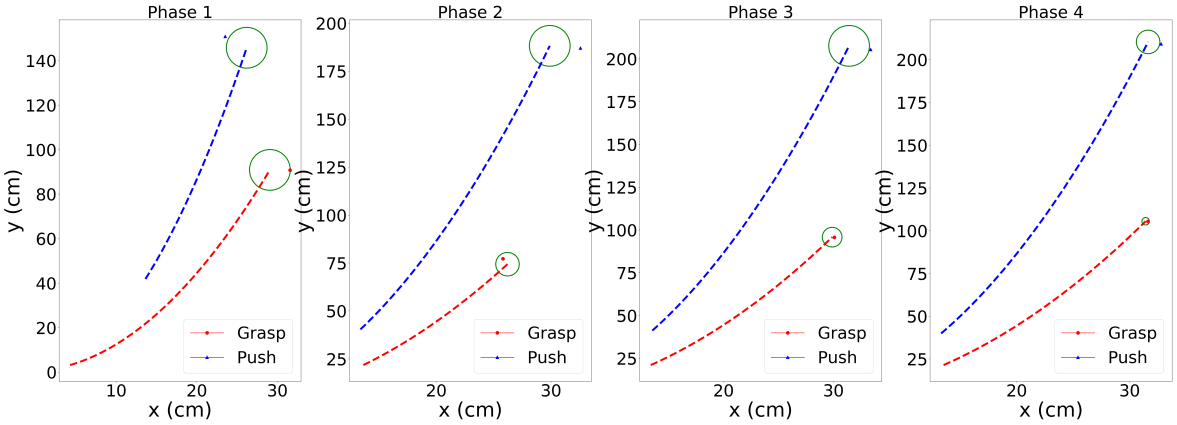


Figure 5.3. LSTM training procedure with changing variance. 4 different LSTM networks are trained.

5.2.2. Autoencoder and t-SNE Factorization

[38, 42] showed in their experiments that infants made predictive gazes only when the experimenter was performing a grasping action. Thus infants made predictions only when the hand of the experimenter is in grasping posture. Infants make the hand posture differentiation before the experimenter starts his/her motion of the action and decide whether he/she attends to the action or not. They have the ability to differentiate the hand postures and correlate them with the actions. In order to replicate this ability, we train an autoencoder and apply t-SNE factorization to each of the hand posture images. After the factorization, what we would like to observe is a clear separation between the two sets so that the resulting features could be used to differentiate between grasping and pushing hand successfully. The image dataset [45] we use has 500 training images, and 100 test images for each posture type and each image has a dimension of 120 x 120 x 3. We first apply a preprocessing on the images and convert each image to 28 x 28. We use convolution layers to reduce the image to 32 x 1. Finally, we apply t-SNE factorization to reduced image and get two features that represent the whole of the image. Figure 5.4 shows the results for the test sets of both grasping and pushing hand posture images. We see that two clusters are separated almost perfectly. Hence the two resulting features seem a good indicator to differentiate between the grasping and pushing hand postures.

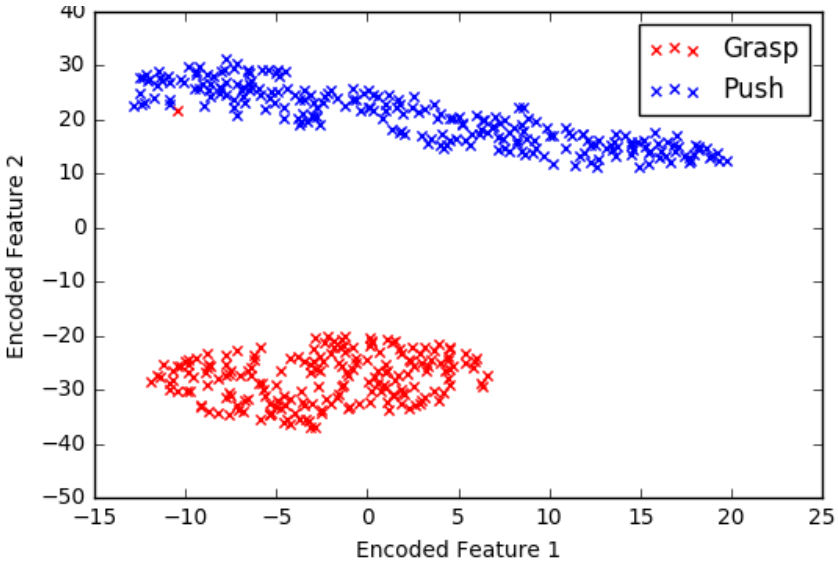


Figure 5.4. Autoencoder and t-SNE factorization clusters

5.2.3. Target Point Estimation with LSTM

Given a partial trajectory, we aim to both predict the future points of the trajectory and the final point of the trajectory, which corresponds to the target. Therefore we have trained two different LSTM networks, one for the future point prediction and one for the target point prediction. We divided each trajectory which has 50 2D points into sequences of 10 by sliding one by one. Each trajectory is then replaced with 40 new trajectories with length 10. Moreover, we refer to the divided trajectory sequence as the parent of the newly generated trajectory sequences. This operation is performed in order to predict future points better. The features obtained from t-SNE factorization were used with the trajectory sequences of 10 to predict the target point. Finally, we have divided the whole data set into two, which correspond to training and test sets. 80% of the data are selected randomly as the training set, whereas the remaining 20% are used as the test set.

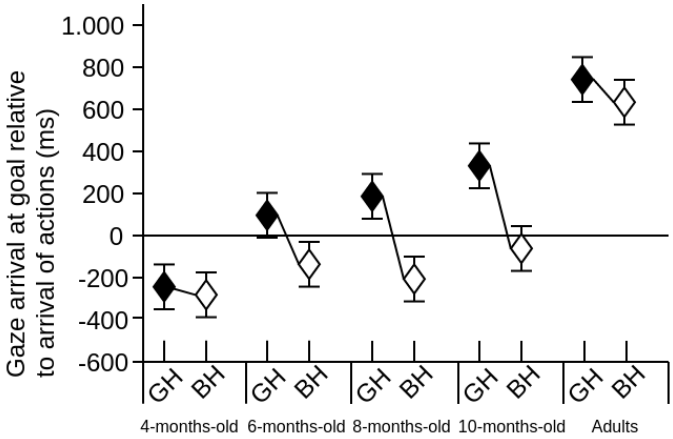
In order to show the development of the action prediction ability, we trained four different LSTM networks. We used the amount of variance to model the developmental progression. Going from the first to last network, we reduced the variances of the output value. When the variance is equal to 0, during the training, the output value is equal to the last point of the parent trajectory sequence. Hence we expect that the

target point prediction becomes more precise when the variance is lower. In order to replicate the experiment of Kanakogi *et al.* [38], we start with high variance for each of the actions which are grasping and pushing. In the second and third networks, we reduced the variance in grasping action whereas keeping the variance in the pushing action same. In the final network, we make the variance of grasping equal to 0 and reduced the variance in pushing as well, which corresponds to 10 months-old infants. Figure 5.3 refers to this idea.

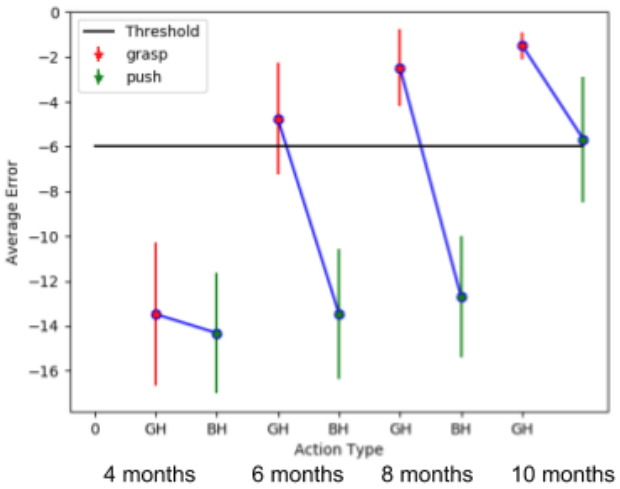
5.3. Results

In [38], the prediction accuracy with gaze arrival time is correlated. If the gaze of the infant arrives at the target object earlier than the completion of the hand trajectory, they conclude that an infant is able to predict the action. In our case, we correlate the prediction accuracy with average error for each action in the test set. As can be seen from Figure 5.5(b), the average errors decrease from the first network to the last network for grasping, suggesting that the network learns to predict the trajectory of the hand better. One caveat here is that in order to replicate the experiment results of [38], we have changed the signs of the average errors for each case. We set the error margin at -6 and say that if the average error is less than this value, it means that the prediction looks correct. For the pushing action, since the variance of the output values is higher compared to grasping action, the predictions look arbitrary, and it is not possible to observe the same learning progress as in the grasping action.

An example of the prediction results is shown in the Figure 5.6. All the predictions are made by looking at the first 10 points of the whole trajectory points. The little circles correspond to grasping predictions whereas the little triangles correspond to pushing predictions of the network. What is observed here is that the grasping action predictions overlap with the final point of the trajectory, which is expected since the model learns to predict grasping actions. However, in the case of pushing actions, prediction errors are higher than the error margin defined from the final point of the trajectories as expected. Because the variance of the output values is high; therefore, the model could not learn to push actions as precise as the grasping actions.



(a) Experiment results of [38], gaze arrival time at goal relative to arrival of actions for each age group.

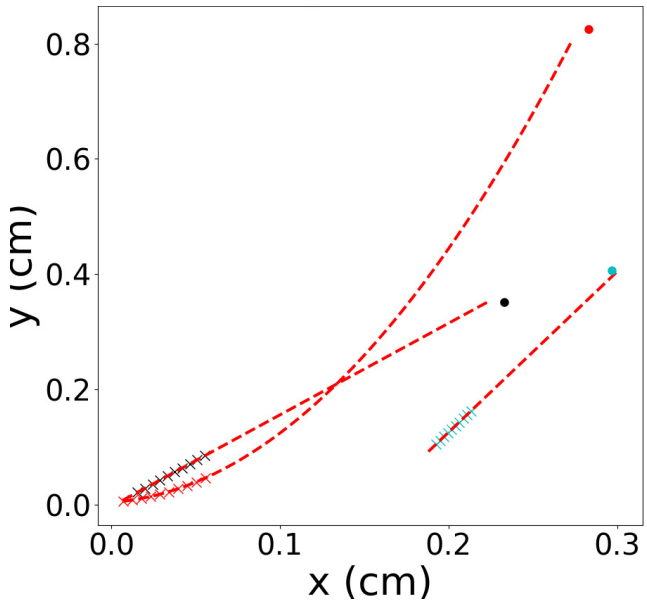


(b) Our results, the distance between the actual end point of the given trajectory and the predicted end point of the trajectory.

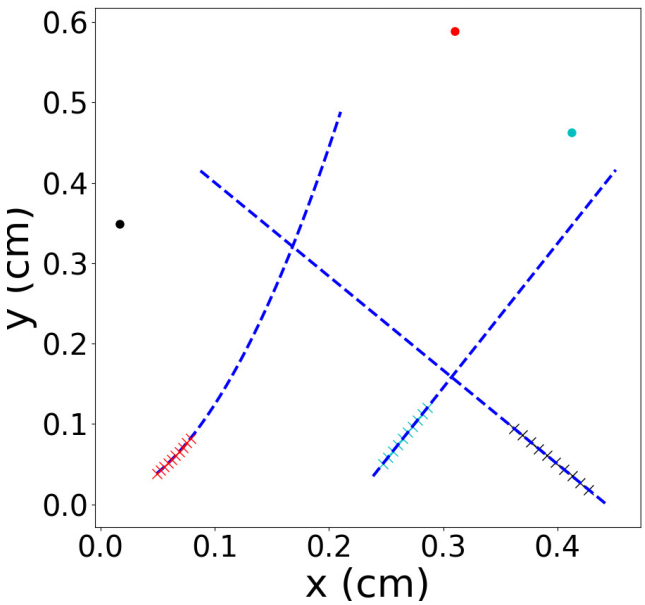
Figure 5.5. Preliminary work results.

5.4. Discussions

In this work, we replicate the experiment results of [38] by using the images of the hand postures and the artificial trajectory data. We train four different LSTM networks with changing variance in the output values to show the learning progress for two actions, which are grasping and pushing. We use changing variance in order to model the development progression of an infant. Younger infants have less experience



(a) Sample grasp predictions.



(b) Sample push predictions.

Figure 5.6. Prediction results for the grasping and pushing actions. Dashed lines correspond to sample trajectories, and the dot with the matching color is the prediction.

with both of the actions. Therefore, we expect their predictions of goal-directed actions of others will be less accurate compared to predictions of older infants. Going from the first network to last network, we reduce the variance in training for certain actions and anticipate that the predictions will be more accurate. We introduce an error margin and decide whether a prediction is successful or not by comparing the prediction error

with this margin. If the prediction error is smaller than the error margin, we count the prediction as successful.

In the future, instead of using variance as the metric for learning, we would like our model to emerge behaviors such as grasping and pushing. What we plan to do is by using a human hand, in a simulation environment, learning the parameters of actions via Reinforcement Learning in the same vein as [47]. In such a learning scenario, we expect the grasping action to emerge before then the pushing action. Because grasping creates more tactile stimulation and allows object manipulation with high prediction accuracy that is critical for planning, in particular, once the object is in hand, its pose can be predicted reliably, lending a higher utility for grasping over pushing.

During the reinforcement learning phase, after some episodes, we log all the data generated for all of the actions and replicate our work with the logged data. By doing this operation several times, after specific episodes, we plan to show the effect of the emergence of behaviors to prediction capability.

6. PROPOSED SYSTEM

In the proposed system, the robot is given a parameterized reach action which it can use to continually explore the parameter space while forming forward models to represent its action capabilities in terms of action effects and action contingencies. Importantly, during this process, the robot also re-organizes its motor space guided by the effects generated due to the interaction with the object. The main components of the proposed system are illustrated in Figure 6.1. (Bootstrapping phase takes the initial interactions, performs outcome clustering (C1, C2) and re-organizes the parameter space; resulting in two initial regions R1, R2, and the transformation function f. Region Re-organizer updates the regions and decides on the splits, LP-based Action Sampling calculates the LP for the regions and samples action ($M(t)$) parameters for the execution, and finally Forward Model for the region predicts both the changes in the object state and trajectory of the end-effector given the action parameter.) In the *bootstrapping phase*, the robot executes its reach action with random parameters, observes the effects generated in the environment and groups similar effects into effect categories. Based on the corresponding effect categories, the action parameter space is transformed into a latent parameter space and partitioned into regions. For each region, a *forward model* is trained to predict action effects. After the bootstrapping phase, the robot starts *LP-based action sampling*, i.e., sampling of action parameters from the regions with the highest learning progresses. The actions from the selected regions are executed by the robot, and the observed effects are used to update the corresponding forward models. If a particular region is explored too often, this region is divided into sub-regions, ensuring maximum prediction accuracy in the sub-regions as well. The proposed LP-based region sampling and prediction-accuracy based region splitting (LPPA) allows the robot to learn forward models along with the development efficiently. The following subsections provide the details of these processes.

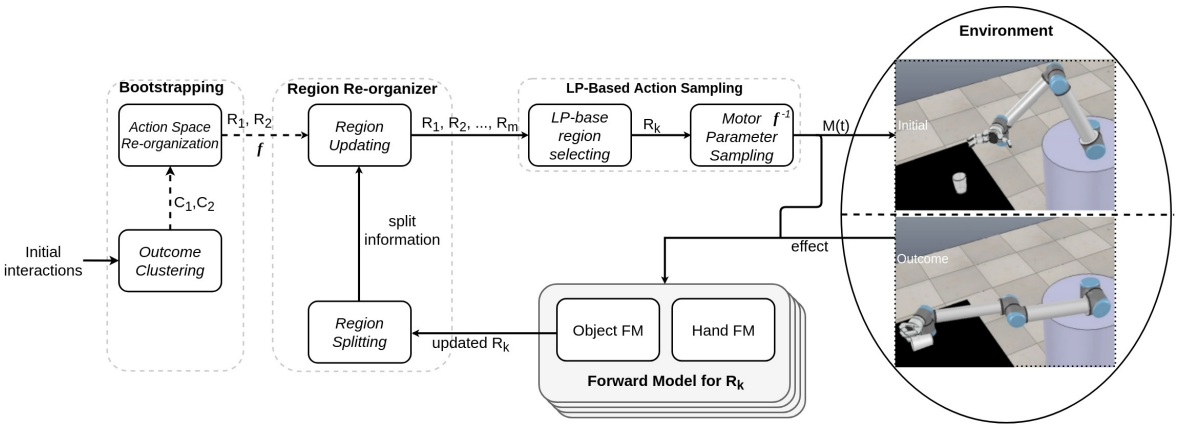


Figure 6.1. Overall model of the proposed system. The main components of the model are bootstrapping, region re-organization, LP-based action sampling and forward model predictions.

6.1. Bootstrapping Phase

For continual region formation and refinement, our system requires a bootstrapping phase that finds initial action parameter regions and an action space transformation function. These regions are formed by using the initial sensorimotor experience created by randomly sampled actions. Moreover, bootstrapping phase speeds up the learning and increases efficiency of the predictions by ensuring that the robot focuses on tasks that end up with an interaction with the object.

6.1.1. Effect Clustering

For differentiating actions with different effects, clustering is applied in the effect space, and as a result, qualitatively distinct effect categories (C_1, C_2 in Figure 6.1) are found. As the effect space might be high dimensional and composed of a diverse set of variables, a spectral clustering method is used for this purpose. Effect clustering is also used to group the action parameters: the parameters that cause the same effect are grouped, and the formed groups compose the first set of sub-regions (R_1, R_2 in Figure 6.1). The effect clustering is finally exploited to produce an operator that transforms the original action parameter space to a low dimensional space.

6.1.2. Action Space Re-organization

This module aims to re-organize action parameter space so as to facilitate effective splitting of the parameter space in the subsequent stages. One way to do this is to transform the original parameter space to a latent space where the similarity in effect space is preserved. For this, a supervised dimensionality reduction method, namely Linear Discriminant Analysis (LDA) is used where effect clustering results are used as the sample labels. Given the parameters and the corresponding effect categories, a projection function is computed to ensure well-separation of the effects in the formed lower dimensional latent parameter space. The formed latent parameter space defines the boundaries between the regions, and thus allows direct parameter sampling from the desired region.

6.2. Forward Model

The regions formed in the action parameter space may be considered as movement primitives that cause similar changes in the environment. As such, each primitive may be accompanied by predictive mechanisms for maintaining execution robustness, error detection, and recovery. Here, without digression these motor control aspects, we focus on the predictive capacity that is developed along with motor re-organization. We envision forward models that help motor control, namely object forward model and hand forward model. The former deals with the prediction of effects of the executed actions in the environment, whereas the latter learns to predict the continuation of an ongoing hand movement action.

6.2.1. Object Forward Model

In our setting, given the action parameters, the object forward model estimates the change in the position and the orientation of the object after the interaction. Instead of a single complex predictor, we adopt the notion that multiple local linear predictors are less costly to use, once the responsibility region of each local predictor is determined. Interested readers are referred to [48, 49].

6.2.2. Hand Forward Model

The hand forward model aims to predict the upcoming hand trajectory, given an initial portion of it. In our implementation, we propose individual models for each action defined by the initial parameter regions. In a biological system, such ability can be used for the benefit of the organism, for example, detecting deviations from a planned action, and can be developed, for example, via associative learning. In particular, a spatio-temporal Hebbian mechanism can yield such an ability [50]. In our study, instead of implementing an associative learning system, we adopted Long Short Term Memory (LSTM) network that emulates the desired functionality. In our system, the hand forward model works with the 3D trajectory data pertaining to the hand (end-effector) of the robot. Given an initial part of the hand trajectory, the network outputs the predicted trajectory of the hand.

6.3. LP-Based Action Sampling

In order to guide the robot towards a desirable difficulty level, which is neither too complicated nor too simple, LP-based selection is used. This is achieved by selecting the action within the regions in which the highest learning progress is achieved during the exploration. Therefore, after the bootstrapping phase, in each step, the robot selects the region where the learning progress is maximal. Recall that the regions are organized in the latent parameter space. For a region R, learning progress (LP) is defined by considering the average prediction accuracy increase of the forward model of the object for the corresponding region ($Pred_i$) as follows:

$$LP_i(t+1) = \overline{\gamma}_i(t+1) - \overline{\gamma}_i(t+1-\tau) \tag{6.1}$$

where $\overline{\gamma}_i(t+1)$ and $\overline{\gamma}_i(t+1-\tau)$ are defined as the average prediction accuracies of the forward model, consecutively for current and previous predictions, and τ represents

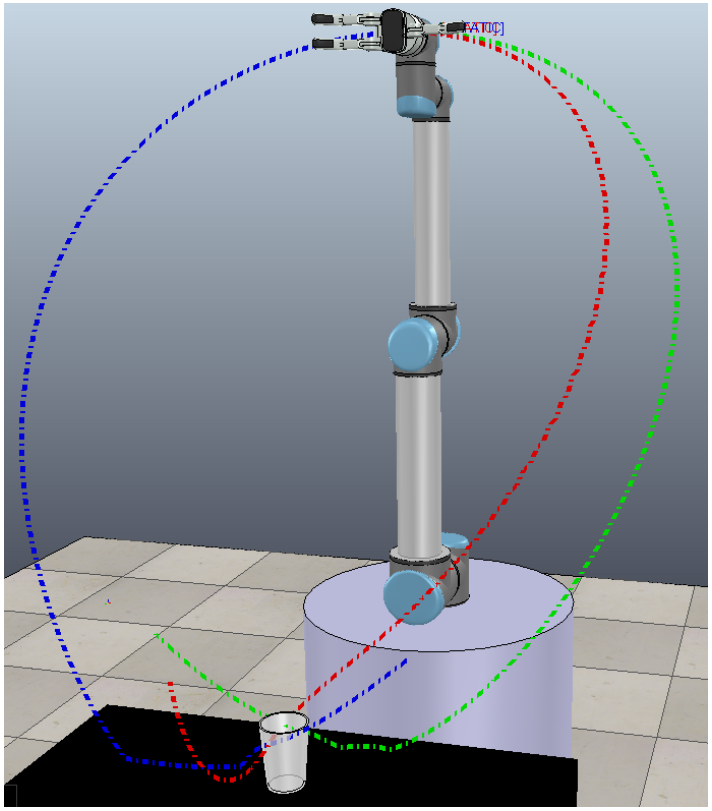


Figure 6.2. Action Trajectory Examples. Colored curves indicate three different trajectory examples that the end-effector of the robot followed in order to perform an action.

the time window which is set to 5.

Here a smoothing parameter ω is set to 25 empirically and used in the calculation of average prediction accuracy as follows:

$$\overline{\gamma}_i(t+1) = \frac{\sum_{j=0}^{\omega} \gamma_i(t+1-j)}{\omega+1}$$

where this formula only approximates the real accuracy and is used as a local measure. Prediction accuracy of the region $(\gamma_i(t))$ is equal to the ratio of the correct predictions on objects explored by the action at step t.

Since it is not possible for the robot to know in advance the outcome of the actions without actually executing them in a real setting, this local accuracy measure is used in our incremental learning setup.

Finally, the next region to be explored is selected based on the above learning progress criteria using ϵ-greedy strategy [51].

$$R_{\text{sel}}^{t} = \begin{cases} R_r & \text{if } \zeta < \epsilon \\ \arg\max_{R_i} LP_{R_i}(t) & \text{otherwise} \end{cases}$$

where R_{sel}^{t} denotes the selected region at learning step t, R_r corresponds to a random region, $0 \leq \zeta \leq 1$ is a uniform random number, and ϵ is set to 0.10.

6.4. Region Partitioning

The robot explores regions through the LP-based active selection mechanism described in Section 6.3. When the number of interactions sampled from a region exceeds a specified threshold, the corresponding region is split into two sub-regions as in [21]. Potential sub-regions are generated by sampling splitting points uniformly for each dimension in the latent parameter space. For each splitting point, the corresponding two potential sub-regions are formed, and training predictors calculate their prediction accuracies with the corresponding set of interactions. Finally, the pair with the maximum prediction accuracy is selected to replace the parent region in the rest of the development.

7. EXPERIMENTS

The experimental setup is created in V-REP simulator [52] to include a six degrees of freedom manipulator (UR10) with a 3-finger gripper (Robotiq), which can interact with a cylindrical object placed in a table-top environment. Since our focus is to investigate how a given basic skill (reaching) can lead to motor specialization when prediction and learning progress measures guide sensorimotor exploration and adaptation, we kept the object identity and location fixed in the experiments reported in this paper.

7.1. Reach Action

This basic action enables the robot to interact with the object from different approach directions, with different apertures and wrist orientations. Different means of interactions is achieved through the following parametrization:

$$(x, y, z, \theta, \supset)$$

Where (x, y, z) defines the approach vector towards the object, θ represents the wrist orientation, and $\supset$ corresponds to the gripper aperture width. In each interaction, the gripper starts moving from the home position provided in Figure 6.2 and follows a trajectory that passes through an approach position, the center of the object and a final position. The approach position is calculated by subtracting the approach vector from the object position. The final position is set to the symmetrically opposite side of the approach position with respect to the object center, but with higher elevation. The target joint angles that bring the gripper to the approach position, the object center, and the final position are calculated using inverse kinematics library[1] . Finally, a trajectory is generated in joint space through fitting a spline that connects these four points starting from the angles at the home position. While this trajectory is executed, the gripper angle θ and aperture width $\supset$ are set to the desired values. At

[1] www.orocos.org/kdl

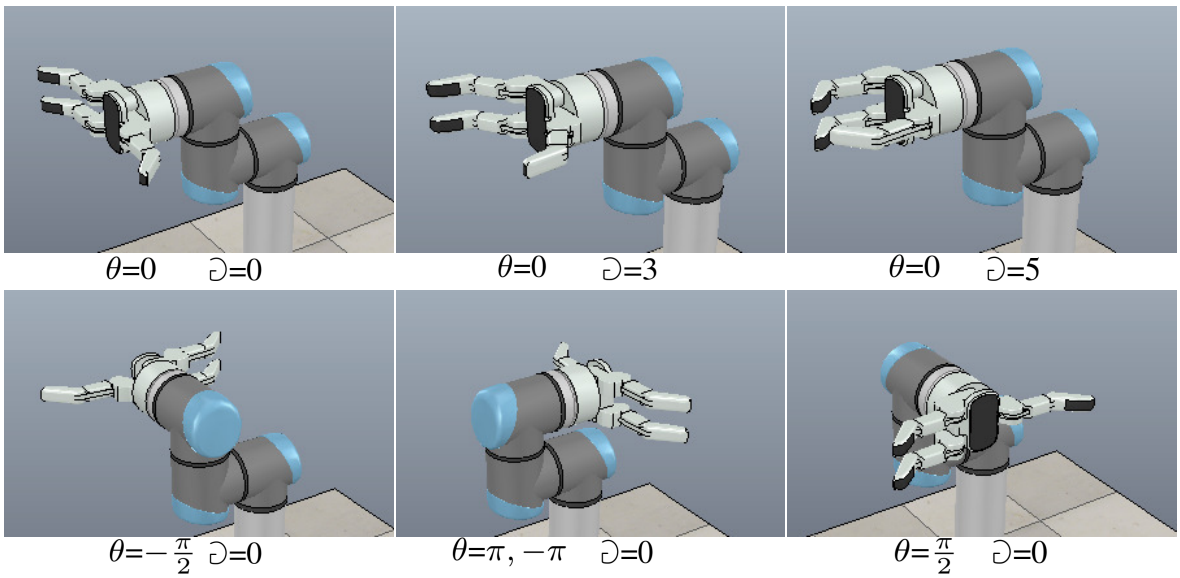


Figure 7.1. Gripper configuration parameters, θ = raw angle, ↄ = gripper closeness degree.

the time of contact of the gripper with the object, the fingers of the gripper are enclosed simulating grasp reflex. Finally, the robot arm is allowed to move to the final position after the gripper-object contact. Different reach trajectories with different parameters are overlaid on the snapshot of the robot at its home position in Figure 7.1. Note that the effects of the interactions are not shown in this figure; depending on the parameters the object might be pushed away towards different directions by different parts of the gripper or might be grasped by the gripper.

7.2. Data Collection

The action parameters are sampled from a uniform distribution from the following range:

- For the approach position $((x, y, z))$, an imaginary cylinder around the object with a radius of 30 cm and a height of 5 cm is generated. The cylinder is placed 5 cm above the table to avoid gripper-table collision during the execution. Finally, x, y and z are set to a random point on that cylinder.
- The gripper angle is set to a random value in the range of $[-\pi, \pi]$ radians. This effectively corresponds to a gripper orientation where the normal vector of gripper

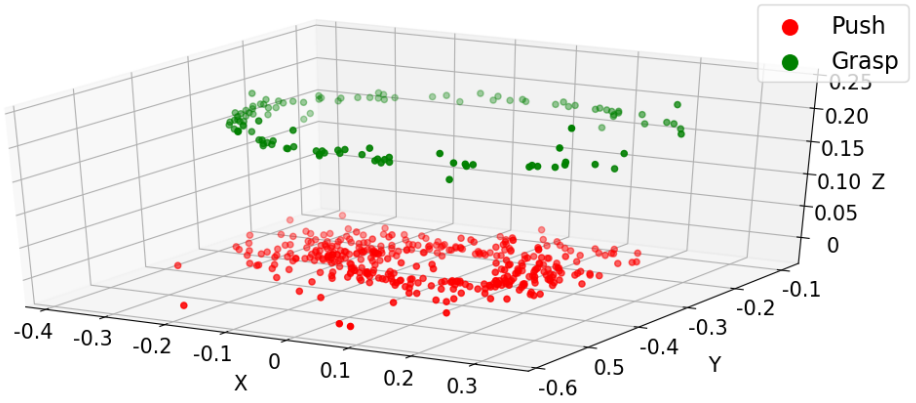


Figure 7.2. Effect clustering results are provided. Each point shows the displacement of the object in 3D as the result of the interaction with the robot. Two clusters were formed and referred to as *Push* and *Grasp* clusters.

palm is parallel to the table plane and is randomly rotated around the normal vector of the table plane (see Figure 7.1 (a-d)).

- The gripper aperture width is set to a random value in a range ([0-5]) that fully opens and closes the fingers at its so-called basic grasping mode as shown in Figure 7.1 (e-g).

In trajectory generation, the spline fit is performed using Bezier curves as Bezier curves are argued to well model human-like reaching motions [53]. As mentioned before, the spline is fit in joint space considering the home position, the approach position, the object center, and the final position.

In order to execute the action, 60 points are sampled from the corresponding spline and the corresponding joint angles are sent as target angles to the robot controller. The position and orientation of the object are extracted after the action execution. Finally, the change in the pose of the object is stored along with the action parameters in each interaction.

While calculating the prediction errors, the forward model uses the 3D position and 3D orientation changes of the object obtained from the simulator. In this work, we have used the precise position and orientation of the object for the calculations.

However, it is known that the visual sensory input processing of the infants is noisy [54]. The precise object position and orientation might not be available to an infant. Therefore, it would have been a proper approach to analyze the development of the visual sensory input processing accuracy of the inputs and model it according to infant data.

8. RESULTS

8.1. Developmental Progress in a Single Run

In this section, the exploration behavior of the robot is analyzed, the generated motor regions are presented, and the change in the performance by the proposed method is assessed. For this purpose, the agent was allowed to sample 3000 interactions, which correspond to 3000 sets of action parameters, based on the proposed LPPA method.

8.1.1. Initial Effect Clusters

In this subsection, we provide the effect clusters that were formed in the bootstrapping phase. The robot used 500 random interactions for this purpose, forming two initial regions by applying spectral clustering on the effect space. In Figure 7.2, the points correspond to the end position of the object subsequent to interactions of the robot, where colors indicate the clusters found. The interactions that increase the elevation of the object are clustered in the *Grasp* group represented by the green color, and the interactions that only change the position of the object on the table plane are clustered in the *Push* group, which is represented by the red color. Note that, the number of the points belonging to the *Grasp cluster* is almost half of the number of points belonging to *Push cluster*. The reason is that the class instances were unevenly distributed in the uniformly sampled data where the push effect was observed twice as much as the grasp effect.

8.1.2. Initial Regions

In this subsection, we presented the result of action space re-organization and the generated regions in the latent parameter space. For this, the 5D action parameter space was projected to 2D latent space using LDA method. Recall that the clusters obtained in the spectral clustering were used as labels for the projection performed in

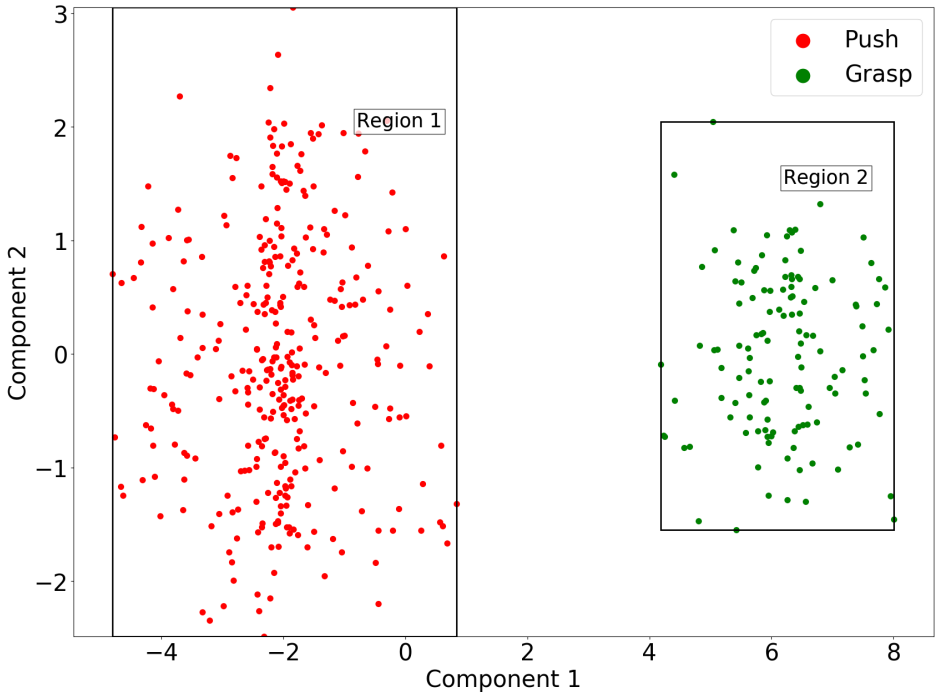


Figure 8.1. Bootstrapping phase interaction instances are plotted in the 2D the latent parameter space formed by the LDA. The red and green points correspond to the clusters formed through effect clustering, and referred to as *Push* and *Grasp* regions.

LDA. Further, the axis weighting used in LDA was based on the most discriminating component (i.e., the z component) of the cluster means found by the spectral clustering. In Figure 8.1, the interaction instances are plotted in the 2D Linear Discriminant space. Since z position was found to have a high weight in LDA computation, well-separable regions in the LD space are observed. The red and green colors represent the interactions from the first and second regions, referred to as *Push* and *Grasp* regions, respectively, in the rest of this section. The boundaries of these initial two regions are shown with the black bounding boxes.

8.1.3. Progressive Region Splitting Results

After the bootstrapping phase is completed, where the action parameter space was re-organized, and the first two regions were found, the robot executed 2500 actions following our LPPA algorithm. The regions generated during the exploration are provided by hierarchical region split tree in Figure 8.2. Note that given 2 dimensions

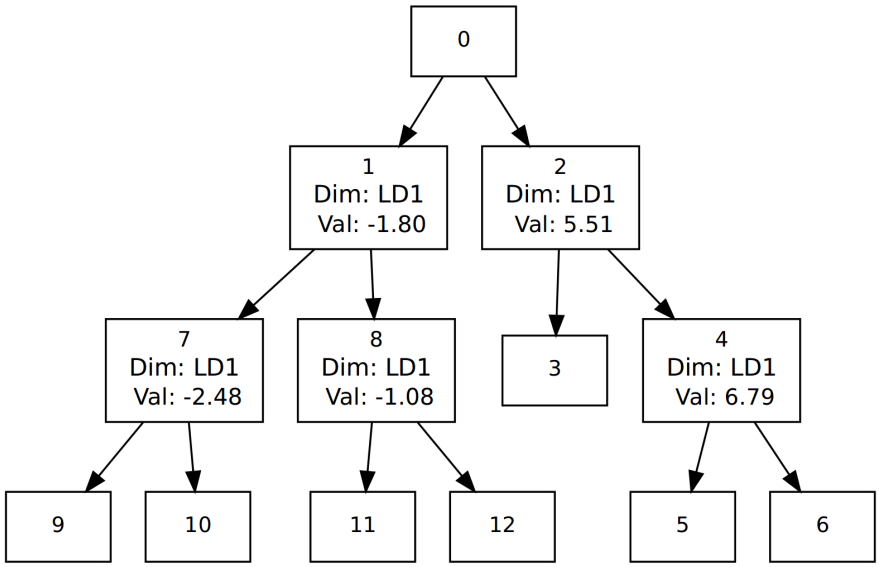


Figure 8.2. The visualization of the generated regions. The nodes 1 and 2 correspond to *Push* and *Grasp* regions, the child nodes correspond to their sub-regions.

and 30 cutting values, 60 pairs of potential child regions were obtained in each split step. Here the root node is the initial region and the second level nodes correspond to the *Push-Region* and the *Grasp-Region* obtained from the initial set of 500 random interactions; and the terminal nodes show the final regions obtained after making 3000 interactions. The boxes provide the region index and from which value and dimension the parent region were partitioned into the child regions. Figure 8.3 provides a visualization of the regions overlaid in the latent parameter space where each vertical bar shows the corresponding division boundary, and the region indices are shown with the numbers in the boxes. The *Grasp-Region* and *Push-Region* were divided two and three times, respectively, generating two and three regions at the end. These regions correspond to the terminal terminal nodes in Figure 8.2. Analyzing the region indices reveals the fact that, the splits in *Grasp-Region* occurred earlier than the splits in *Push-Region*.

8.1.4. Region Exploration

In this part, the region exploration strategy of our system and the order of emergence of the sub-regions are analyzed. Figure 8.4 demonstrates the exploration frequencies of the regions along the developmental timeline of the robot. The frequencies

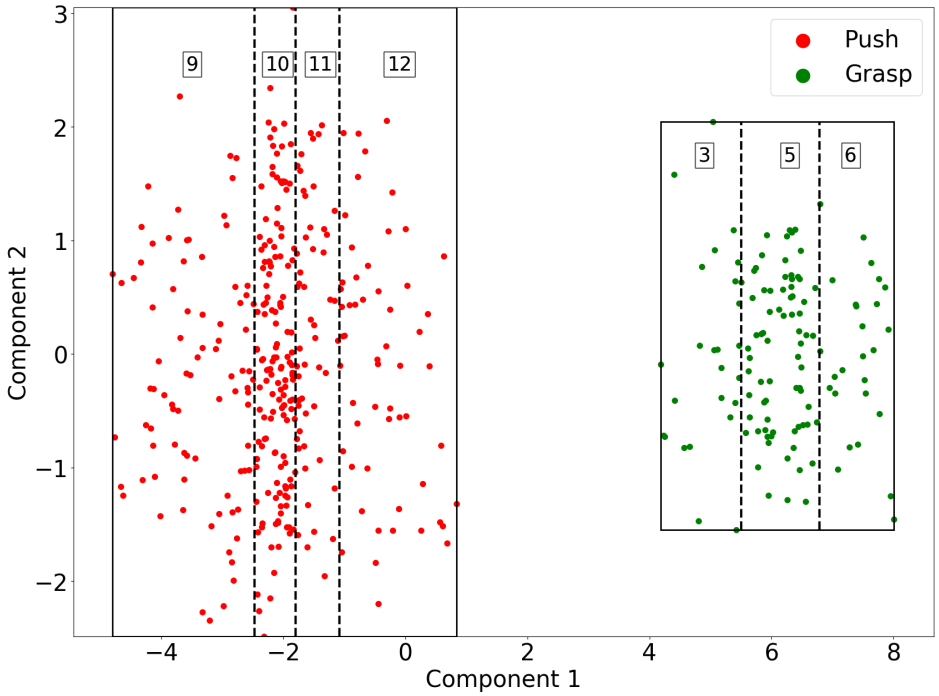


Figure 8.3. Terminal regions in LD space. Dashed lines represent the cutting value and dimension for the corresponding split. Region numbers are also shown in boxes, smaller the region number, earlier the split to form that region.

are provided from 500 interactions as there was no active sampling earlier. As discussed earlier, *Push-Region* was explored twice as much as *Grasp-Region* due to the uneven effect distribution with the uniform action parameter sampling for the initial 500 interactions. After this point, the learning progress in *Grasp-Region* was more substantial than the one in the *Push-Region*. Therefore, the actions were sampled from the grasp region, and after the maximum number of interactions were reached, the *Grasp-Region* was partitioned into two at around $t = 900$. One of the emerged regions were split into two again, and the frequency of exploration of these grasp regions gradually decreased due to low learning progress. The *push-region*, on the other hand, started being explored more after around $t = 1900$ and continued to be explored more until the end.

8.1.5. Detailed Inspection of the Generated Regions

The regions are further analyzed in order to understand why such splitting occurred, and what the final regions correspond to, as the visualization on the latent

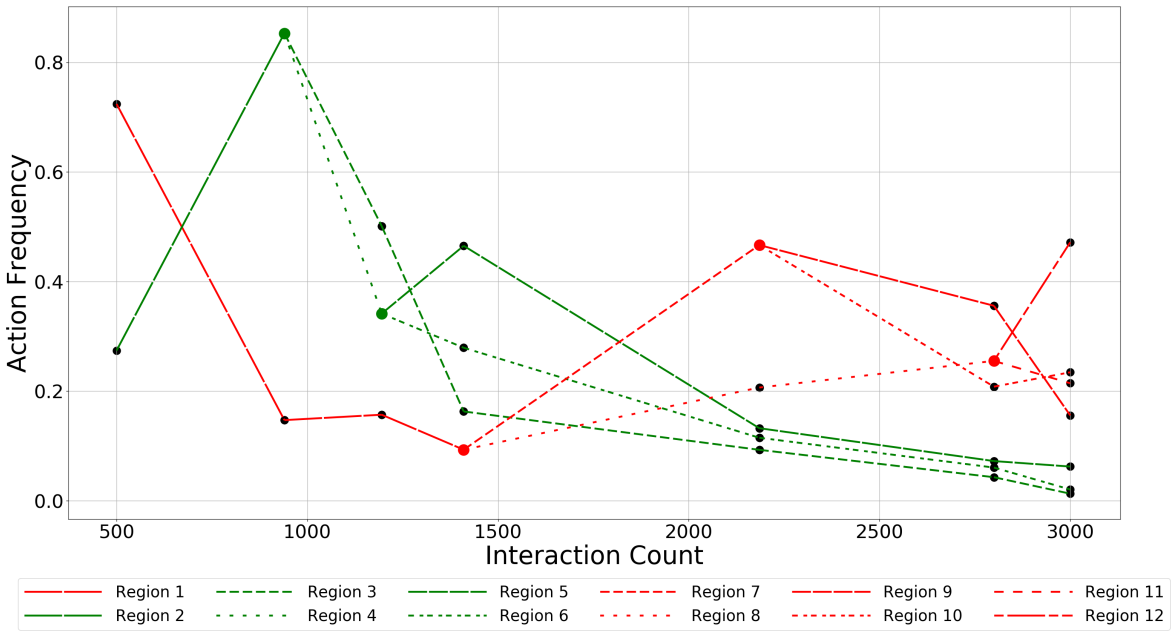


Figure 8.4. Action frequencies of all the regions over the 3000 interactions. Red lines correspond to pushing regions whereas green lines correspond to grasping regions.

parameter space does not directly provide this information. Consequently, the regions were back-projected into the original motor parameter space, and the mean values of each region is inspected. Figure 8.5 provides the mean motor parameter values for (x, y, θ) for each final region. For example, regions 3, 5, and 6 correspond to the regions that were emerged from the *Grasp-Region*. Based on the parameter values, regions 3 and 6 correspond to approach movement from the left and the right, respectively, with the wrist angles oriented towards the object. Region 5, on the other hand, is a mixture of left and right approaches. These results show that the system found *grasp-from-left* and *grasp-from right* primitives. There is similar lateralization for the push primitives as well, where regions 9 and 12 correspond to *push-from-left* and *approach-from-right* with the back side of the gripper. Figure 8.6 provide snapshots from interactions instances observed in regions 3, 6, 9 and 12. These interactions correspond to the mean point of the regions in the latent parameter space.

8.2. Analysis of Results From Independent Runs

In the previous section, the results of a single developmental progress were provided in detail. This section aims to assess the overall performance of the proposed

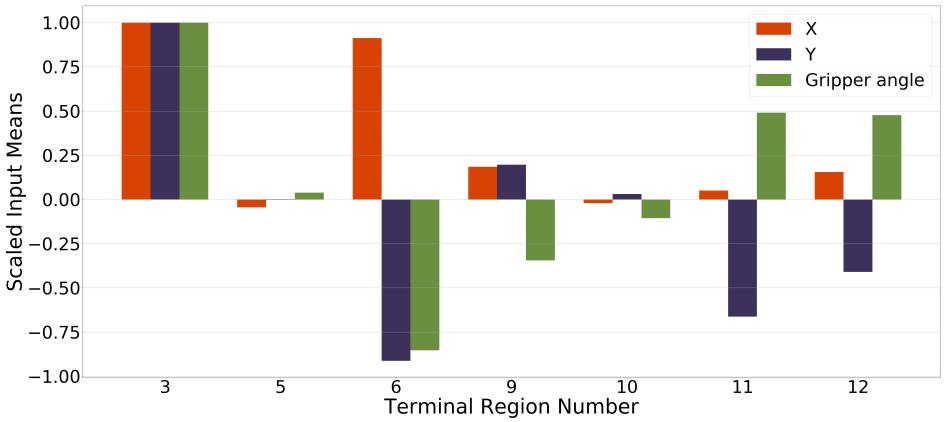


Figure 8.5. Scaled mean values for the action space of the terminal regions are shown for a single run.

method. Therefore, the developmental learning procedure is repeated 10 times with different initial seeds that practically correspond to different initial 500 interaction samples for bootstrapping phase, which at the end causes different sets of regions to emerge. In this section rather than providing the individual results of these 10 different runs, the statistics are provided as region groups in the rest of this section.

8.2.1. Developmental Order

Figure 8.7(a) provides the exploration frequencies of action parameters for regions that emerge from grasp and push regions. The squares and the bars provide the mean frequencies and the standard deviations for the 10 independent runs, respectively. In the beginning, push region motor parameter were sampled twice as much as the grasp motor parameters due to the uneven grasp/push distribution obtained from random parametrization. With our LPPA method, the robot focused on exploring the grasp region until around $t = 2500$. Figure 8.7(b) provides the learning progress statistics of the corresponding regions, and as shown at around $t = 2500$, the learning progress of push regions exceeded the learning progress of grasp regions. Therefore, all the runs changed their exploration strategy, focusing more on the push regions towards the end. Even though pushing action was performed more in the random exploration phase, the robot focused on grasping actions even in the first stages of the development due to the learning progress of the actions (Figure 8.7(b)). Learning progress is higher for

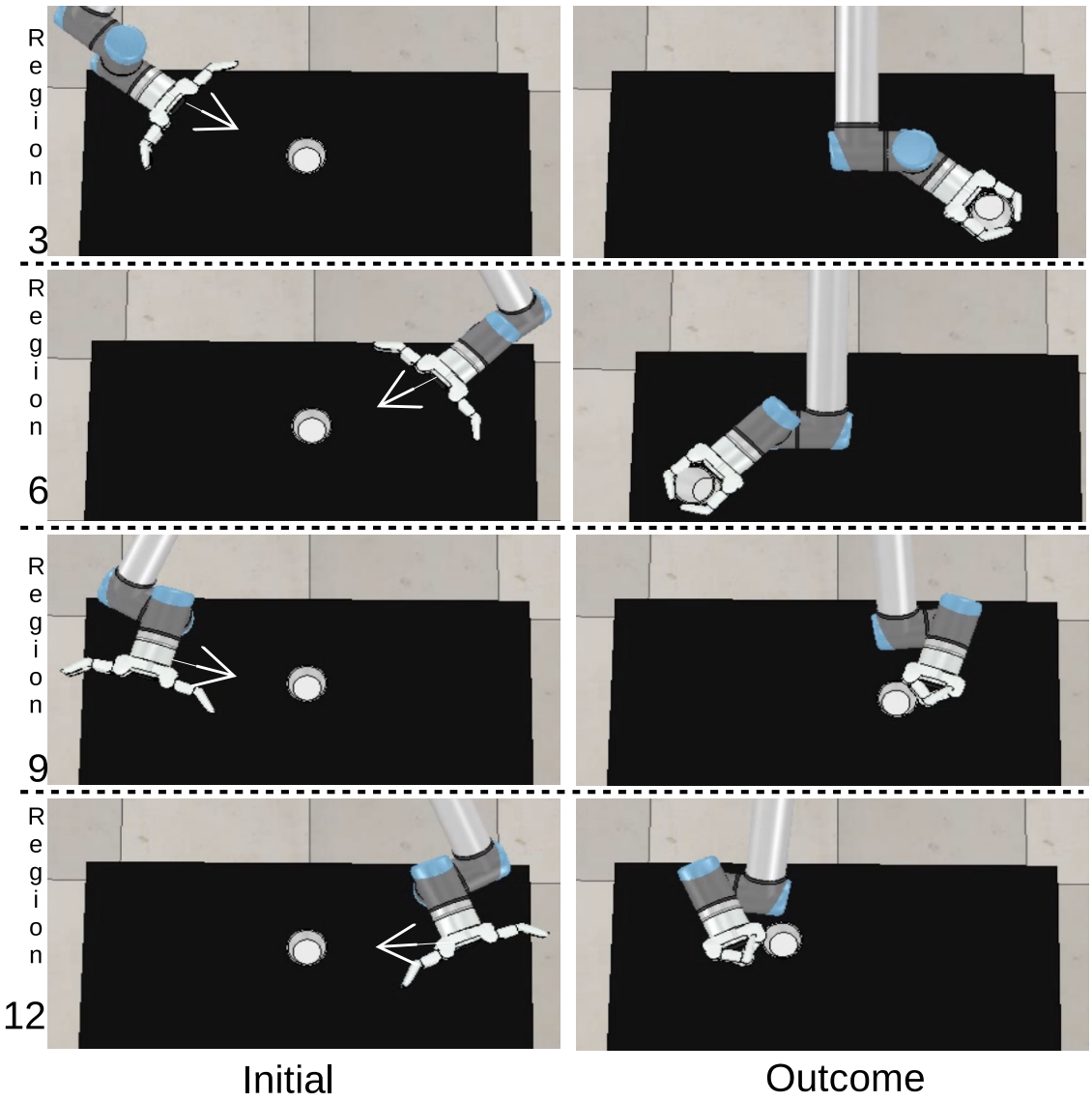
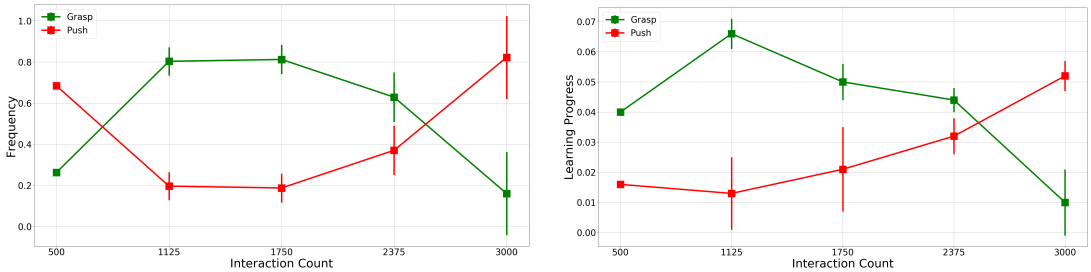


Figure 8.6. Snapshots of example robot executions from regions 3, 6, 9 and 12 are provided from top to bottom. The robot at the selected approach positions and at the final configurations are shown on the left and right respectively.

the grasping actions at the beginning of the exploration because of the dynamics of the environment. It is easier for the forward model to predict the outcomes of the grasping actions because the object will be in the hand of the robot. However, if the object is pushed, predicting the stochastic position and orientation of the object will be harder because of the rolling effect of the object.



(a) The exploration frequencies of actions. (b) The average LP fluctuation of the actions.

Figure 8.7. The average frequencies of the explored actions.

8.2.2. Efficiency in Learning

In this subsection, we compared the learning speed of our system with two alternative strategies. The first strategy, named as *random sampling*, randomly samples regions to explore and follows our prediction accuracy based sampling approach. The second strategy, names as *variance-based splitting*, samples the regions based on their LP, but uses variance based splitting criteria of Oudeyer *et al.* [21]. 10 independent runs were performed using all three strategies, including our LPPA, and their generalization performances are compared in Figure 8.8. As shown, the error decreased with more interactions in both variance-based splitting and LPPA, and decrease with LPPA exceeded the alternative model after 1500 interactions. This result shows that the prediction accuracy based region partitioning strategy employed by LPPA result in faster learning compared to the region splitting approach that only used the distribution of the data in the regions.

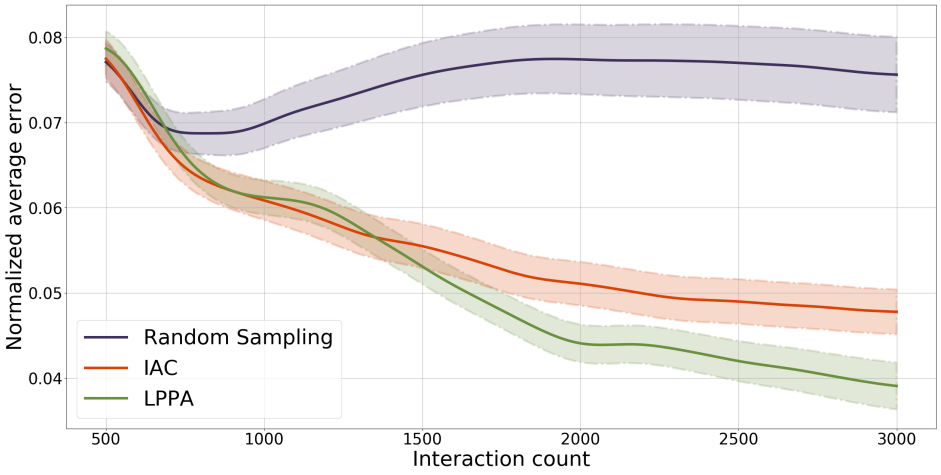


Figure 8.8. Comparison between the error plots for ten runs.

8.3. Development of Hand Forward Model

In this section, the change in hand forward model performance along the developmental timeline is analyzed. Given the initial 20 steps of the trajectory, the robot learns to predict the rest of the trajectory by training an LSTM network in each region. Figure 9.1 provides the prediction results for grasp and push regions along the developmental timeline. After the bootstrapping phase, the error in the grasp region had a sharp decrease compared to the error drop in the push region. Towards the end of the developmental timeline set as 3000 interactions, the performance of predicting the hand trajectory in push region becomes similar to the performance in grasp region. This progress in performance is the result of sampling from different regions in different amounts during the developmental timeline.

9. DISCUSSIONS

9.1. Predicting Own Action Consequences Mechanism in Brain

We have stated before that predicting the consequences of own actions is an important requirement for active learning. While performing an action, humans receive sensory input, processes them continuously and act upon those. How is it possible for a human to focus on a task while he is receiving some distractive sensory inputs such as an auditory signal while performing a reaching action towards an object? And also what mechanisms enable humans to distinguish whether a sensory input is caused internally or externally? Brain is continuously monitoring self-movements in order not to startle by the consequences of the sensory changes in the environment and the consequences of the self motor executions [55, 56]. Corollary Discharge (CD) signals provide internal feedback to the sensory areas of the brain, enabling the brain to prepare for the sensory and movement changes. A forward model of the oculomotor system is employed by the brain to process the sensory input and to generate predictions about the expected changes. This forward model uses CDs of the saccade. Finally, predictions of the forward model are being used to update the saccade.

Saccadic eye movements correspond to a quick an simultaneous movement of both of the eyes. It is observed most when a sudden and impactful sensory change in the environment occurs [56]. Wurtz *et al.* studied on the following question. How is it possible for a human brain to keep track of saccades? They also concluded that CDs are the key components and they are used to copy the saccading eye movements for prior prediction. They have proposed four distinct criteria in order to identify the CDs in brain.

9.2. Action Prediction and Production Correspondence

In Section 4, we introduced an infant study from literature in which the development of the ability to predict the target of certain actions (push or grasp) was

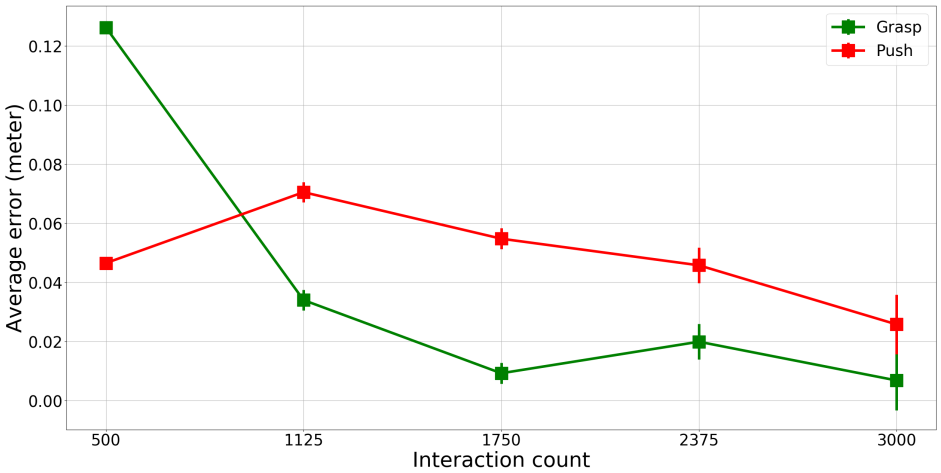


Figure 9.1. The average errors for trajectory prediction. The red plot represents errors corresponding to pushing actions whereas the green plot represents errors for grasping actions.

dependent upon or facilitated by the capacity to perform the action in consideration [5]. Figure 9.2 illustrates the results of that experiment where gaze arrival times to the target objects of ongoing actions are shown for different conditions and different ages. GH and BH represent the grasp-hand and back-hand (push) postures. Positive and negative gaze arrival times correspond to gaze arrival to the target object before and after the hand-object contact. As shown, the performance of predictive eye movement develops first for grasp movement compared to push movement. Our results (Figure 9.1) corroborate well with this finding. Although target prediction through gaze was not explicitly modeled in our method, the gaze movement can be assumed to be triggered by the result of trajectory prediction. Therefore, the inability to direct the gaze to the target of an ongoing action would indicate the inability to generate a correct trajectory prediction with our model. While the results measure different variables, the parallels in the results suggest that the action prediction depends on motor development in both cases, which in turn depends on the amount of experience.

9.3. Action Re-organization

As mentioned before, the robot is required to partition the parameter space into regions for further processing. However, in high dimensional parameter spaces with

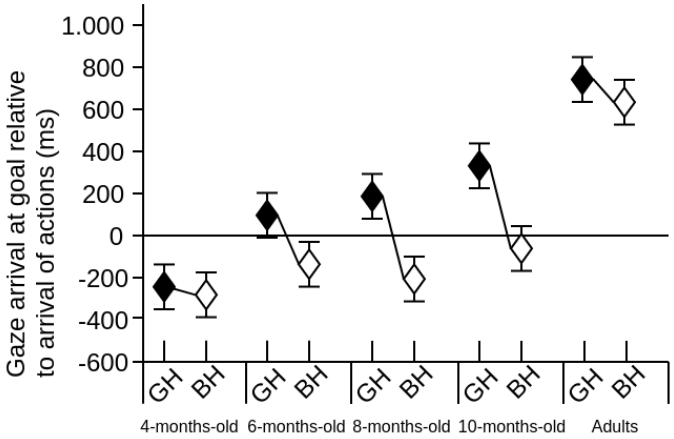


Figure 9.2. Adapted from [5], gaze arrival time at goal relative to the arrival of actions for each age group. Black and white diamonds indicate the grasping (GH) and back hand pushing (BH), respectively.

non-homogeneous component scales, the partitioning procedure is not straightforward. In [21, 57], the parameter space was partitioned by considering a single parameter at a time. These methods assumed that underlying regions in the parameter space are organized as hyper-rectangles. However, this assumption does not hold in many cases, especially when one considers high dimensional spaces with parameters having diverse types, semantics, and metrics. Therefore, in this study, we proposed to re-organize the parameter space by preserving the effect related similarity while reducing the effective dimensions. To this end, a supervised dimensionality reduction method that can impose the effect topology in the formed latent action space is used. This way, the parameters that caused different types of outcomes were ensured to be well-separated in the latent space.

9.4. Action Distribution

In the bootstrapping phase, the parameters of the sampled actions followed a uniform distribution. However, during development, we observed that the reach actions that approach from the opposite side of the object vanished. Further inspection revealed that because of the robot kinematics, the hand collided with the object while moving towards the approach position at the opposite side, pushing the object in a seemingly random way, thus making it impossible to predict the action consequences

for the corresponding action parameters. Consequently, the LP based sampling favored to explore more direct reach actions in both grasping and pushing. This is consistent with one of the main characteristics of reaching movements: straight paths towards the object [58].

10. CONCLUSION AND FUTURE WORK

In summary, the proposed LPPA method enabled the robot to explore its parameter space efficiently and effectively, to enable it to discover a number of specialized movement primitives for which predictive forward models are also built. Furthermore, the parallels between our results and the development of infant action prediction capacity suggest that the sensorimotor systems of developing infants may employ LPPA based mechanisms.

Finally, the proposed method establishes a strong connection between the action parameter and effect spaces. By considering the significant differences in the effect space, LPPA reorganizes the action parameter space. Thus, speeds up the learning by decreasing the time required for exploration and also enables the emergence of the actions by analyzing the action parameter space.

We are planning to extend this study by increasing the action space parameter count. The object information such as the location, type and color is kept same in this study. However, it would have yielded additional understanding for us if these parameters were changed. This study is not sufficient to answer whether the shape, type of the object effect the emergence of the actions. Moreover, depending on the location of the object, left or right handedness could emerge as well.

This thesis provides results for the simulated robot execution only. In the future, we are planning to use the real robot by transferring the knowledge gained by the simulated robot. For this purpose, a transformation from the simulated to real world is required. For the kinematics part, a deep inverse dynamics model could be employed. Furthermore, the robot force torque sensor could be used to control the robot robustly.

In this study, we assumed that the robot has already acquired the required skills of reaching an object by adjusting its joints and gripper configuration. However, for the robot, it would have been possible to first focus on controlling its joints and gripper,

then move on to more complex actions such as grasping and pushing. Most probably, this would increase the time required for the robot to learn all the actions mentioned. However, if we employ this strategy, it would simulate the development of an infant in a more similar way.

REFERENCES

1. Kawato, M., "Internal models for motor control and trajectory planning", *Current opinion in neurobiology*, Vol. 9, No. 6, pp. 718–727, 1999.

2. Steedman, M., "Plans, affordances, and combinatory grammar", *Linguistics and Philosophy*, Vol. 25, No. 5-6, pp. 723–753, 2002.

3. Von Hofsten, C. and K. Rosander, "Development of smooth pursuit tracking in young infants", *Vision research*, Vol. 37, No. 13, pp. 1799–1810, 1997.

4. Johnson, S. P., J. G. Bremner, A. Slater, U. Mason, K. Foster and A. Cheshire, "Infants' perception of object trajectories", *Child Development*, Vol. 74, No. 1, pp. 94–108, 2003.

5. Kanakogi, Y. and S. Itakura, "Developmental correspondence between action prediction and motor ability in early infancy.", *Nature communications*, Vol. 2, p. 341, 2011.

6. Gerber, R. J., T. Wilks and C. Erdie-Lalena, "Developmental milestones: motor development", *Pediatrics in review*, Vol. 31, No. 7, pp. 267–277, 2010.

7. Rosenbaum, D. A., *Human motor control*, Academic press, 2009.

8. Asada, M., K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino and C. Yoshida, "Cognitive developmental robotics: A survey", *IEEE transactions on autonomous mental development*, Vol. 1, No. 1, pp. 12–34, 2009.

9. Ryan, R. M. and E. L. Deci, "Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being.", *American psychologist*, Vol. 55, No. 1, p. 68, 2000.

10. Zurada, J. M., *Introduction to artificial neural systems*, Vol. 8, West publishing company St. Paul, 1992.

11. Finn, C., X. Y. Tan, Y. Duan, T. Darrell, S. Levine and P. Abbeel, "Deep spatial autoencoders for visuomotor learning", *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519, IEEE, 2016.

12. Hochreiter, S. and J. Schmidhuber, "Long short-term memory", *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

13. Fisher, R. A., "The use of multiple measurements in taxonomic problems", *Annals of eugenics*, Vol. 7, No. 2, pp. 179–188, 1936.

14. Universal Robots, *UR10 Website*, `https://www.universal-robots.com/products/ur10-robot/`, accessed at May 2019.

15. Robotic, *3-Finger Adaptive Gripper*, `https://robotiq.com/products/3-finger-adaptive-robot-gripper`, accessed at May 2019.

16. roswiki, *Robot Operating System*, `http://wiki.ros.org/`, accessed at May 2019.

17. Coppelia Robotics, *V-REP Robot Simulator*, `http://www.coppeliarobotics.com/`, accessed at May 2019.

18. Keras IO, *What is Keras*, `https://keras.io/`, accessed at May 2019.

19. Scikit Learn, *Scikit Learn*, `https://scikit-learn.org/stable/`, accessed at May 2019.

20. Oudeyer, P.-Y. and F. Kaplan, "What is intrinsic motivation? A typology of computational approaches", *Frontiers in neurorobotics*, Vol. 1, p. 6, 2009.

21. Oudeyer, P.-Y., F. Kaplan and V. V. Hafner, "Intrinsic Motivation Systems for Autonomous Mental Development", *IEEE Transactions on Evolutionary Compu-*

tation, Vol. 11, pp. 265–286, 2007.

22. Hester, T. and P. Stone, "Intrinsically motivated model learning for developing curious robots", *Artificial Intelligence*, Vol. 247, pp. 170–186, 2017.

23. Sequeira, P., F. S. Melo and A. Paiva, "Emotion-based intrinsic motivation for reinforcement learning agents", *International Conference on Affective Computing and Intelligent Interaction*, pp. 326–336, Springer, 2011.

24. Santucci, V. G., G. Baldassarre and M. Mirolli, "GRAIL: a goal-discovering robotic architecture for intrinsically-motivated learning", *IEEE Transactions on Cognitive and Developmental Systems*, Vol. 8, No. 3, pp. 214–231, 2016.

25. Hart, S., S. Sen and R. Grupen, "Intrinsically motivated hierarchical manipulation", *2008 IEEE International Conference on Robotics and Automation*, pp. 3814–3819, IEEE, 2008.

26. Temel, E., B. J. Grzyb and S. Sariel, "Learning Graspability of Unknown Objects via Intrinsic Motivation.", *AIC*, pp. 98–109, Citeseer, 2014.

27. Ugur, E. and J. Piater, "Emergent structuring of interdependent affordance learning tasks using intrinsic motivation and empirical feature selection", *IEEE Transactions on Cognitive and Developmental Systems*, Vol. 9, No. 4, pp. 328–340, 2017.

28. Caligiore, D. and G. Baldassarre, "The Development of Reaching and Grasping: Towards an Integrated Framework Based on a Critical Review of Computational and Robotic Models", *Reach-to-Grasp Behavior*, pp. 335–364, Routledge, 2018.

29. Gaussier, P. and A. Pitti, "Reaching and Grasping: what we can learn from psychology and robotics", *hal-01609659*, 2017.

30. Oztop, E., N. S. Bradley and M. A. Arbib, "Infant grasp learning: a computational model", *Experimental brain research*, Vol. 158, No. 4, pp. 480–503, 2004.

31. Ugur, E., Y. Nagai, E. Sahin and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese", *IEEE Transactions on Autonomous Mental Development*, Vol. 7, No. 2, pp. 119–139, 2015.

32. Rizzolatti, G. and L. Craighero, "The mirror-neuron system", *Annu. Rev. Neurosci.*, Vol. 27, pp. 169–192, 2004.

33. Sommerville, J. A., A. L. Woodward and A. Needham, "Action experience alters 3-month-old infants' perception of others' actions", *Cognition*, Vol. 96, pp. b1–b11, 2005.

34. Copete, J. L., Y. Nagai and M. Asada, "Motor development facilitates the prediction of others' actions through sensorimotor predictive learning", *2016 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pp. 223–229, 2016.

35. Robson, S. J. and V. A. Kuhlmeier, "Infants' understanding of object-directed action: An interdisciplinary synthesis", *Frontiers in psychology*, Vol. 7, p. 111, 2016.

36. Rajmohan, V. and E. Mohandas, "Mirror neuron system", *Indian journal of psychiatry*, Vol. 49, No. 1, p. 66, 2007.

37. Sommerville, J. A., A. L. Woodward and A. Needham, "Action experience alters 3-month-old infants' perception of others' actions", *Cognition*, Vol. 96, No. 1, pp. B1–B11, 2005.

38. Kanakogi, Y. and S. Itakura, "Developmental correspondence between action prediction and motor ability in early infancy", *Nature communications*, Vol. 2, p. 341, 2011.

39. Copete, J. L., Y. Nagai and M. Asada, "Motor development facilitates the predic-

tion of others' actions through sensorimotor predictive learning", *2016 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pp. 223–229, IEEE, 2016.

40. Flash, T. and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model", *Journal of neuroscience*, Vol. 5, No. 7, pp. 1688–1703, 1985.

41. Daum, M. M., W. Prinz and G. Aschersleben, "Encoding the goal of an object-directed but uncompleted reaching action in 6-and 9-month-old infants", *Developmental Science*, Vol. 11, No. 4, pp. 607–619, 2008.

42. Woodward, A. L., "Infants' ability to distinguish between purposeful and non-purposeful behaviors", *Infant Behavior and Development*, Vol. 22, No. 2, pp. 145–160, 1999.

43. Gredebäck, G., D. Stasiewicz, T. Falck-Ytter, C. V. Hofsten and K. Rosander, "Action type and goal type modulate goal-directed gaze shifts in 14-month-old infants.", *Developmental psychology*, Vol. 45 4, pp. 1190–4, 2009.

44. Hamlin, J., E. V. Hallinan and A. L. Woodward, "Do as I do: 7-month-old infants selectively reproduce others' goals.", *Developmental science*, Vol. 11 4, pp. 487–94, 2008.

45. Marcel, S., "Hand posture recognition in a body-face centered space", *CHI Extended Abstracts*, 1999.

46. van der Maaten, L., "Accelerating t-SNE using tree-based algorithms", *Journal of Machine Learning Research*, Vol. 15, pp. 3221–3245, 2014.

47. Oztop, E., N. S. Bradley and M. A. Arbib, "Infant grasp learning: a computational model", *Experimental Brain Research*, Vol. 158, pp. 480–503, 2004.

48. Schaal, S., C. G. Atkeson and S. Vijayakumar, "Scalable techniques from nonparametric statistics for real time robot learning", *Applied Intelligence*, Vol. 17, No. 1, pp. 49–60, 2002.

49. Jacobs, R. A., M. I. Jordan, S. J. Nowlan, G. E. Hinton *et al.*, "Adaptive mixtures of local experts.", *Neural computation*, Vol. 3, No. 1, pp. 79–87, 1991.

50. Chaminade, T., E. Oztop, G. Cheng and M. Kawato, "From self-observation to imitation: Visuomotor association on a robotic hand", *Brain research bulletin*, Vol. 75, No. 6, pp. 775–784, 2008.

51. Sutton, R. S. and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998, `http://www.cs.ualberta.ca/~sutton/book/the-book.html`.

52. Rohmer, E., S. P. N. Singh and M. Freese, "V-REP: A versatile and scalable robot simulation framework", *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1321–1326, 2013.

53. Faraway, J. J., M. P. Reed and J. Wang, "Modelling three-dimensional trajectories by using Bézier curves with application to hand motion", *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, Vol. 56, No. 5, pp. 571–585, 2007.

54. Berthier, N. E., "Learning to reach: a mathematical model.", *Developmental psychology*, Vol. 32, No. 5, p. 811, 1996.

55. Crapse, T. B. and M. A. Sommer, "Corollary discharge circuits in the primate brain", *Current opinion in neurobiology*, Vol. 18, No. 6, pp. 552–557, 2008.

56. Wurtz, R. H. and M. A. Sommer, "Identifying corollary discharges for movement in the primate brain", *Progress in brain research*, Vol. 144, pp. 47–60, 2004.

57. Sener, M. I. and E. Ugur, "Partitioning Sensorimotor Space by Predictability Principle in Intrinsic Motivation Systems", *2018 Joint IEEE International Conference*

on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), pp. 54–59, IEEE, 2018.

58. Karniel, A. and G. F. Inbar, "A model for learning human reaching movements", *Biological cybernetics*, Vol. 77, No. 3, pp. 173–183, 1997.