Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Predicting User Intent During Teleoperation Using Neural Networks

Shao-Wen Wu

Course of Study:	INFOTECH
Examiner:	Prof. Dr. rer. nat. Marc Toussaint
Supervisor:	Ph.D. Jim Mainprice, M.Eng. Yoojin Oh
Commenced:	May 27, 2019
Completed:	November 27, 2019

Abstract

The goal of the thesis is to develop a method to predict user intent during robot teleoperation using machine learning methods. Teleoperation refers to controlling a robot through an interface when the robot is located distant from the user. Traditionally the robot can be teleoperated by giving commands that are directly mapped to robot actions. However, this can lead to intensive workload for the human operator and can be prone to operational mistakes. Teleoperation can be mitigated by combining user control and robot autonomy, where the robot can be semi-autonomously operated towards mid-level goals. Here, predicting the user intent can increase the performance of the robot by allowing the robot to anticipate and disambiguate user commands. In this thesis, we develop methods to predict the user intent while the user is controlling the robot using hand motion. In addition to predicting the intent, we model the uncertainty of how confident the robot is towards the prediction. We consider robot reaching tasks where human intent is represented as the object to be grasped or the final grasping position on the surface of the object. We use neural networks to predict the intent and estimate the uncertainty using Beta distribution and mixture multivariate Gaussian distribution and further learning the grasp densities. The positive results show that we can predict the user intent before the robot grasps and can be utilized in shared autonomy scenarios to provide better assistance.

Contents

List of Figures

List of Tables

Acronyms

LSTM Long Short-Term Memory

CNN Convolutional Neural Network

MDN Mixture Density Network

BiLSTM Bidirectional LSTM

ROS Robot Operating System

NLL negative log-likelihood

MLP Multi-Layer Perceptron

1 Introduction

Nowadays robots are becoming more capable and robust enough to substitute or assist humans to achieve certain complex tasks. With the deployment of robots, humans can achieve complex tasks that exceed their physical capabilities in dangerous conditions or inaccessible environments without risking their lives. However, the difference between the kinematic morphology between humans and robots makes it difficult to directly manipulate a robot. This can require extensive training hours and still become easily prone to human errors.

Humans are yet better than robots when it comes to interpreting complex and dynamic scenes and making high-level decisions whereas robots can take advantage of their high computational power to optimize low-level tasks. *Shared autonomy* is an active field of research [AS16], [DS13], [NZHS17] and is introduced to utilize the best of both worlds. Rather than the robot directly executing the human operator's commands, the robot can predict their intention and accomplish the task following an optimized trajectory such that the robot eventually performs what the human wants rather than what the human commands.

Shared Autonomy can be broadly divided into two phases: predicting user's target and blending the human commands with robot actions to provide assistance towards the predicted target [AS16], [DS13]. The robot predicts the goal that the user is operating towards through signals from user interface commands such as joysticks or keyboard commands. By knowing the goal, the robot can anticipate future commands and can compute an optimal path towards the goal. When the robot reaches a certain level of confidence of a goal, the robot can then provide assistance by blending user intent and robot autonomy to reach the target [FRPG04].

Predicting human intent is challenging, especially when human's intentions are not explicit. The intentions must instead be inferred through user interface commands. However, these commands can be indirect and noisy and can also contain errors due to controlling a high degree of freedom robot arm with limited number of controller inputs [AS16]. Other interaction methods can be utilized during teleoperation, such as using motion capture systems to capture motion commands [SBR12].

The thesis focuses on predicting the human operator's intentions while the operator is teleoperating the robot's end-effector through hand gestures in a virtual workspace. We not only infer the human intent but are also interested to know how we can model the confidence during prediction. Here, the human intent is represented as the object to be grasped and also the final grasping position on the surface of the object. We use neural networks to predict the intent and estimate the uncertainty using Beta distribution and mixture multivariate Gaussian distribution and further learning the grasp densities.

The thesis is structured as follows: In Chapter 2, we provide the background information of shared autonomy related to human intent prediction and uncertainty. In Chapter 3, we propose the developed human intent prediction models for pick-and-place tasks and the design of the neural

networks frameworks. We present the evaluation of the models in Chapter 4 and also the prediction results while executing the system at run-time. Finally, we discuss limitations and future possible improvements in Chapter 5 and conclude the thesis.

2 Background and Related Work

In this chapter, we provide explanations and understanding of the concept of human intent prediction in the field of robot teleoperation and shared autonomy. Furthermore, we emphasize uncertainty estimation methods using machine learning and grasp densities related to grasp affordances.

2.1 The Concept of Shared Autonomy

2.1.1 Robot Teleoperation

Teleoperation has been proven to be capable of dealing with complex manipulation tasks. Traditional *direct* teleoperation converts the low-dimensional control input to the non-anthropomorphic characteristics of high degree of freedom robot arms. However, the control needs high cognitive loads and adequate practice. Although *indirect* control interfaces (e.g., eye gaze) can provide high-level action commands and additional benefits. Users still implicitly communicating their aim to the robot because the *indirect* control is not intuitive and requires more awareness. In this case, combining the operator intention and capabilities of the robot is desirable [BLB14]. The autonomy or intelligence on the robot is one method to improve the performance of teleoperation. Some autonomy-based methods include adjustable autonomy [BSA+03], mixed initiatives [BDM02] and safeguarded control [KSCK96] have been proposed. Shared autonomy is an active approach in autonomy-based methods [NZHS17], [AA19], [JAP+18]. Share Autonomy integrates human intent with robot autonomy instead of only depend on control inputs from the operator.

2.1.2 Shared Autonomy

The shared autonomy system consists of two components:

1. Human intent prediction.
2. Operator and robot autonomy blending.

Unlike many common methods on human-robot interacted collaboration. Only robot actions can change the state in the scene in shared autonomy and goals are not entirely observable [NZHS17]. In other words, the robot does not know which goal users want to grasp a prior.

Human intent prediction

Huang and Mutlu propose to predict user intent by using the support vector machine (SVM) classifier to infer the next goal and confident score based on gaze patterns [HM16]. Admoni and Srinivasa use partially observable Markov decision process (POMDP) [AS16] to infer probability over user goals with eye gaze and joystick. Razin et al. use layered hidden Markov models with surface electromyography measurements to predict human intent [RPUF17]. Javdani et al. use hindsight optimization to solve the POMDP by modeling user intent in the latent space [JSB15]. Reddy et al. use model-free deep reinforcement learning with a separate recurrent LSTM network to predict the goal [RDL18]. Human intent prediction is one of the main determinants of shared autonomy. In this thesis, we focus on the first part of shared autonomy to predict human intent using neural networks.

2.2 Modeling Uncertainty with Probability Distributions

2.2.1 Uncertainty in Machine Learning

In most practical uses, people are satisfied with machine learning to simply produce predictions. Given an unseen input x, the model is expected to infer a certain output y. However, there is another estimation is interested to be known:

How confident is the model about a particular prediction?

The objective of machine learning is to minimize the expected loss. In the loss minimization, there are two types of uncertainties [DGQ+13]:

1. **Aleatoric uncertainty**: The natural randomness in a process associated with a physical system or environment, modeled by random variables or stochastic processes, namely, uncertainty in data.
2. **Epistemic uncertainty**: The scientific uncertainty in the model of the process due to incomplete knowledge about a physical system or environment, namely, uncertainty in the model.

The most common method to model uncertainty is the Bayesian approach [DD09], using statistical distributions by characterized weights instead of just predicting constant values. Likewise, the predicted result is as well as a statistical distribution which can be represented by means, standard deviations and densities. Bayesian models are slow to train in practical because sampling can be prohibitive with the high number of dimensions. Gal and Ghahramani in [GG16], [GG15] proposed modeling dropout to evaluate uncertainty concerning certain observed samples. In Bayesian neural networks, each weight is obtained from a certain distribution instead of having fixed weights. When dropout is applied to all weights in neural networks, essentially, each weight can be drawn from a Bernoulli distribution. In real-time performance, estimating epistemic uncertainty is time-consuming by processing multiple forward passes through networks and nearly impractical. An approach to estimate aleatoric uncertainty is using probability distribution to model the data from neural networks. In the following sections, we will introduce two probability distributions we used to model uncertainty in our models.

2.2.2 Beta Distribution

The Beta distribution is a continuous probability distribution that is commonly used to model uncertainty about the probability of success in the experiment. A distinct characteristic of Beta distribution is that it can be shifted and rescaled to create new distributions over any finite range. The Beta distribution has two positive shape parameters, denoted by α and β, that control the shape of the distribution and appear as exponents of the random variable. Equation 2.1 shows the Beta distribution and usually described by Gamma function as shown in Equation 2.2.

$$B(\alpha,\beta) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)} \tag{2.1}$$

$$\Gamma(x) = \int_{\infty}^{0} t^{x-1}e^{-t}dt \tag{2.2}$$

Assuming a probabilistic experiment has only two outcomes, success, and failure, which probability denote as X and *1-X* respectively. Suppose the X is unknown and all possible values are considered equally likely. The uncertainty can be described in the interval [0, 1] by assigning X as a uniform distribution because X is a probability, can only take values between 0 and 1. Next, assume we perform n times independent repetitions and observed k successes and *n-k* failures. The result of calculating the conditional distribution of X is a Beta distribution with parameters *k+1* and *n-k+1*. The probability density function for a Beta $X \sim \text{Beta}(\alpha, \beta)$ is shown in Equation 2.3, with condition $0 < x < 1$, otherwise $f(x) = 0$.

$$f(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} \tag{2.3}$$

The variance of Beta random variable X is shown in Equation 2.4.

$$VAR[X] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{2.4}$$

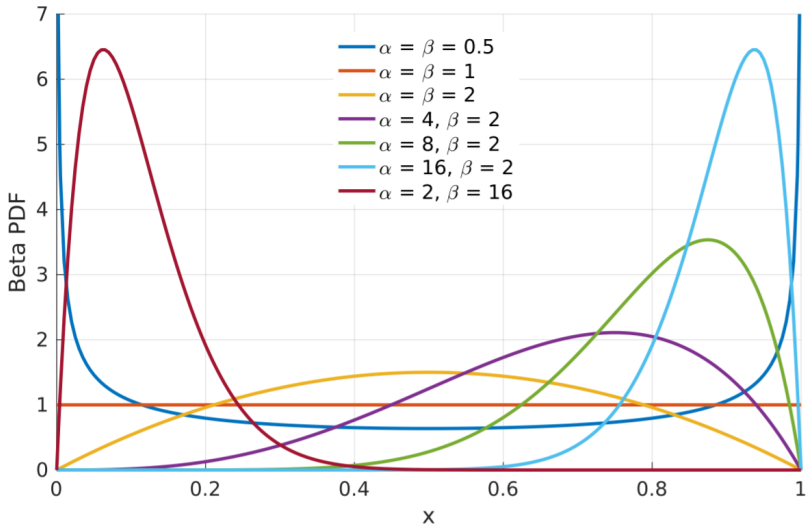


Figure 2.1: The probability density function of Beta distributions with different α and β [CMS17].

2.2.3 Mixture Density Networks

Mixture Density Network (MDN) is a type of neural networks proposed by Bishop [Bis94]. In general, the MDN is very similar to standard neural networks with the only difference that the output of the model is mapped to a mixture distribution, combining a neural network with a mixture density model. In this case, MDN has the ability to model conditional density functions. For each input x, we predict a probability density function of $P(Y = y|X = x)$, which is a probability-weighted sum of all Gaussian distribution kernels presented in Equation 2.5, $\mu_k(x)$ denotes the mean and $\sigma_k(x)$ denotes the standard deviation in each kernel. One restriction is that the sum of all probability density function should be one, to ensure the integration of probability density function is 100%. Another restriction is that the $\sigma_k(x)$ term must be strictly positive due to the exponential operator.

$$P(Y = y|X = x) = \sum_{k=0}^{K-1} \prod_{k}(x)\phi(y, \mu_k(x), \sigma_k(x)), \sum_{k=0}^{K-1} \prod_{k}(x) = 1 \tag{2.5}$$

The loss function is using negative log-likelihood (NLL) to minimize the logarithm of the likelihood of predicted mixture density distribution and ground truth as shown in Equation 2.6. Eventually, the output of MDN is a set of parameters contains mean μ, covariance σ and weight π for each Gaussian component.

$$Cost\ Function(y|x) = -log[\sum_{k=0}^{K-1} \prod_{k}(x)\phi(y, \mu_k(x), \sigma_k(x))] \tag{2.6}$$

One advantage is that there are variant distributions can be used in MDN, for example, Bishop [Bis94] uses 1D Gaussian distribution to perform the inverse problem involving robot inverse kinematics, Graves [Gra13] uses Long Short-Term Memory (LSTM) networks and 2D Gaussian distribution to predict the text sequence on handwriting synthesis and Ellefsen et al. combine autoencoder with

LSTM networks to investigate the role of different mixture components when networks predict the future using multivariate Gaussian distribution with standard diagonal covariance in [EMT19]. In this thesis, we also use multivariate Gaussian distribution to model the point of grasp uncertainty.

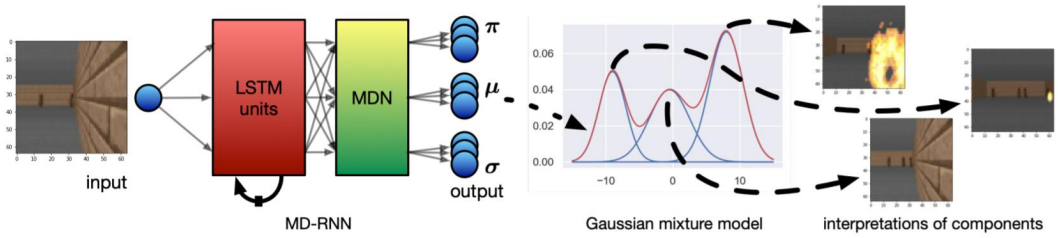


Figure 2.2: The LSTM-MDN model proposed by Ellefsen et al. [EMT19].

2.3 Learning Grasp Densities

Affordances can provide useful information about how robots can interact with objects when performing object manipulation tasks. Grasping previously unknown objects is challenging, the failure could be caused by small localization errors on the surface, especially when performing grasping on unmodeled objects. It is also demanding to have an intention when performing a grasp in cluttered and novel environments. In this section, we first introduce the concept of affordances and give a sense of affordances in grasping, then represent related previous works using vision-based techniques for learning grasp affordance densities.

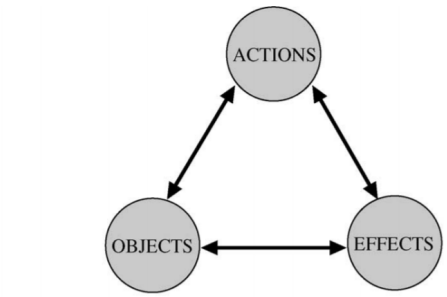
2.3.1 The Concept of Grasp Affordances

The original affordances were first introduced by a psychologist Gibson [Gib14] to explain how things in the environment can be perceived and the possibilities of how this information can be mapped to the action from the agent [ŞÇD+07]. Affordances are useful to create the relationships between available objects and actions performed by the agent. It became one of the active fields in the human-robot interaction area.

Affordances in Robotics

In the field of robotics, affordances describe the relationship between agent and agent's actions effected by its environment. Recently, many robotics researchers have focused on the learning of affordances in robotics since they obtain the essential object and environment properties. All these studies can be categorized into two major aspects [ŞÇD+07]:

1. Learning the results of certain actions in a given situation [FMN+03].
2. Learning invariant properties of environments that afford a certain behavior [CHC04].



inputs	outputs	function
(O, A)	E	Predict effect
(O, E)	A	Action recognition & planning
(A, E)	O	Object recognition & selection

Figure 2.3: Affordances relationships between actions, objects and effects described in [MLBS08].

Affordances in Grasping

Recently in the field of robot grasping, *grasp affordances* have appeared. In this case, the agent usually denotes to a gripper or arm parameters. Here, grasp affordances can be interpreted as the success (effect) of grasp solutions (action) performed to an object (environment) [DKK+11].

Most of the early research works use geometry representation to perform a stable grasp [BK00]. After that, several approaches were proposed extracting visual information to find the relation between action and perception based on 2D images [MSD02], [Jia95], using range information [TK02] or approximating the unknown shape of objects [MKCA03].

Localizing Graspable Geometries

The technique for localizing graspable geometries has been known as an approach to understanding affordances in grasping. Klingbeil et al. use raw depth data from a 3D sensor as input and search for geometries by a parallel-jaw gripper to autonomously grasp unknown objects [KRC+11]. Jiang et al. search regions in RGBD image using orientated grasping rectangle which takes into account orientation, location, and opening width [JMS11]. Fischinger and Vincze perform a 3 DOF search in a heightmap generated from a point cloud, they propose a new type Height Accumulated Features (HAF) and extend binary classification machine learning structure based on point cloud to generate grasp hypothesis [FV12]. Unlike using depth data or heightmap, another approach introduced by Ten Pas and Platt directly operates on the point cloud, which is allowed to be searched in different ways [TP16].

2.3.2 Related work

The vision-based method is an active research field for learning grasp affordance densities. Detry et al. in [DKK+11] use continuous probability density functions to model grasp affordances on previously unknown objects and infer object-relative grasp pose and success rates. The empirical evaluation shows how learning increases success probability. Grasp densities are obtained by non-parametric kernel density estimation method. Visual cues are also used to generate the grasp hypotheses, unlike most popular 3D point cloud approaches, they projected various grasp densities on the 2D color image. Saxena et al. [SWN08] also show a machine learning approach can be successfully performed to grasp novel objects using depth sensor sampled 2D images from a different angle of the same object to learn the shape representation.

The 2.5D or 3D vision-based techniques are the most popular approaches to be used in this field. Kokic et al. [KSHK17] using Convolutional Neural Network (CNN) to model affordance for task-specific grasping on the point cloud. The specific tasks are *cut, poke, pound, pour, and support*. They describe an object on the surface point cloud, the inputs are the point cloud and one of the tasks. In the first stage, they estimate object affordances and in parallel, they also classify object and estimate the orientation, each branch employs different CNN architecture, Figure 2.4 shows the pipeline of the process. To deal with influences with different tasks, they model a one-hot encoding mask for *contact constrain*, each object type has its affordance. Another *approach direction constraint* limits the robot hand approaching direction with estimated orientation.

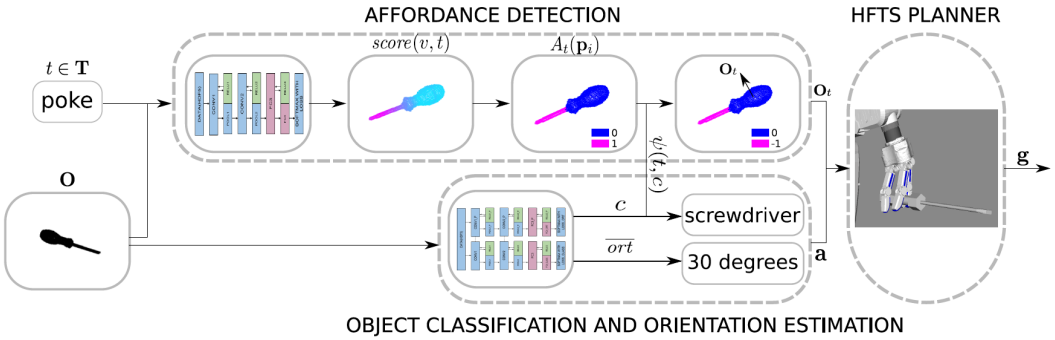


Figure 2.4: Affordances modeling for task-specific grasping [KSHK17]. The inputs are point cloud O and task t, the model in the first row detects object affordances, the output of this model indicates contact locations for grasping in a binary map O_t, the model in the second row classifies the type of object and infer the orientation.

In the paper presented by Boularias et al. [BKP11], the vision-based technique for predicting success rates on point cloud on the surface of the object is also used. The aim is to predict the success probability when performing a grasp at a given point on the surface using Markov Random Fields for the learning method to find a reasonable grasping point. The grasping points are manually labeled in the training point cloud as shown in Figure 2.5. They mentioned that the inadequate orientation of gripper would cause the most failed grasps.

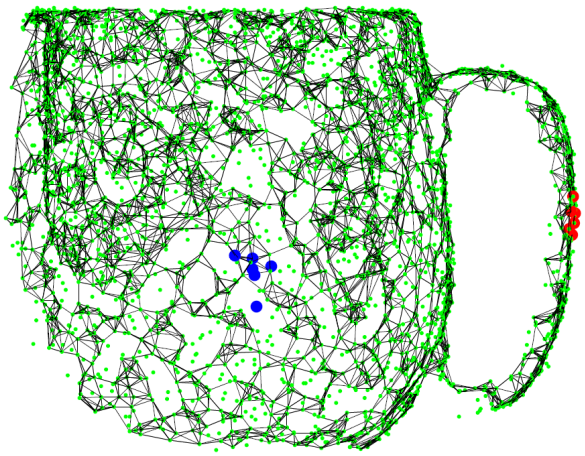


Figure 2.5: Grasping points are manually labeled in training, blue points indicate possible bad grasp locations, while red points denote likely good grasp locations [BKP11].

Unlike most research methods estimate possible grasp affordances on the whole object, Ten Pas and Platt [TP16] localize the important types of grasp affordances on the partial surface of the object. They propose the handle-like grasp affordances first localize handle area from point cloud then infer the success rates, which is very practical and can easily adapt to different scenarios.

3 Method

In this chapter, we propose our method to predict human intent and to estimate the uncertainty for sequential user input. Figure 3.1 shows the pipeline of our proposed model. We collect time-series data of human commands and virtual depth images from the robot's workspace. Since human intent and the robot's action are time-variant, we use an LSTM network to capture hidden representations. LSTM network is a type of recurrent neural network (RNN) that is capable of learning long-term dependencies. The output of the LSTM network is connected to probability models to estimate the uncertainty of prediction. The method in the thesis includes two models:

1. Low-dimensional: Target object prediction model.
2. High-dimensional: Point of grasp prediction model.

In the low-dimensional method, we model uncertainty using Beta distribution in the finite interval. The inference is a one-dimensional continuous probability on a line. In the high-dimensional method, the prediction in continuous space could be multi-valued. Fitting an uni-modal distribution to a multi-valued data can generate erroneous results. Therefore, we use MDN to model the uncertainty from the mixture distribution.

Before we introduce our intent prediction models, we first give a brief introduction of our system, experimental scenario and the way how we conducted the user study.

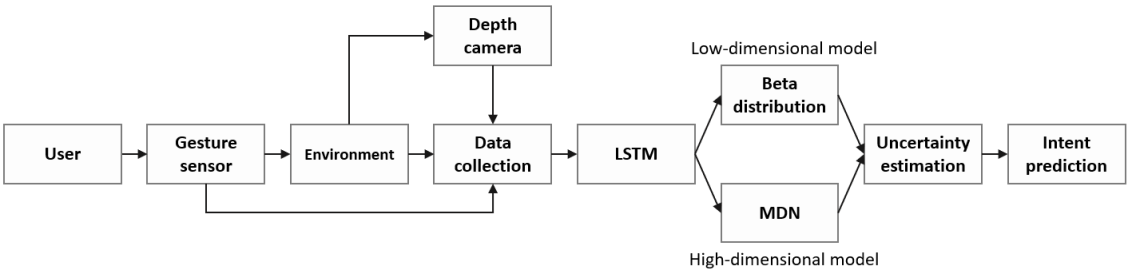


Figure 3.1: Pipeline of the general human intent prediction model.

3.1 Virtual Environment and Data Collection

We built a virtual environment using Robot Operating System (ROS) [1], and Gazebo for generating 2.5D depth images using a virtual depth camera. The human perceives the robot's workspace from a perspective view set up in Rviz, which is a 3D visualization tool for ROS and sampled at 60 frames per second.

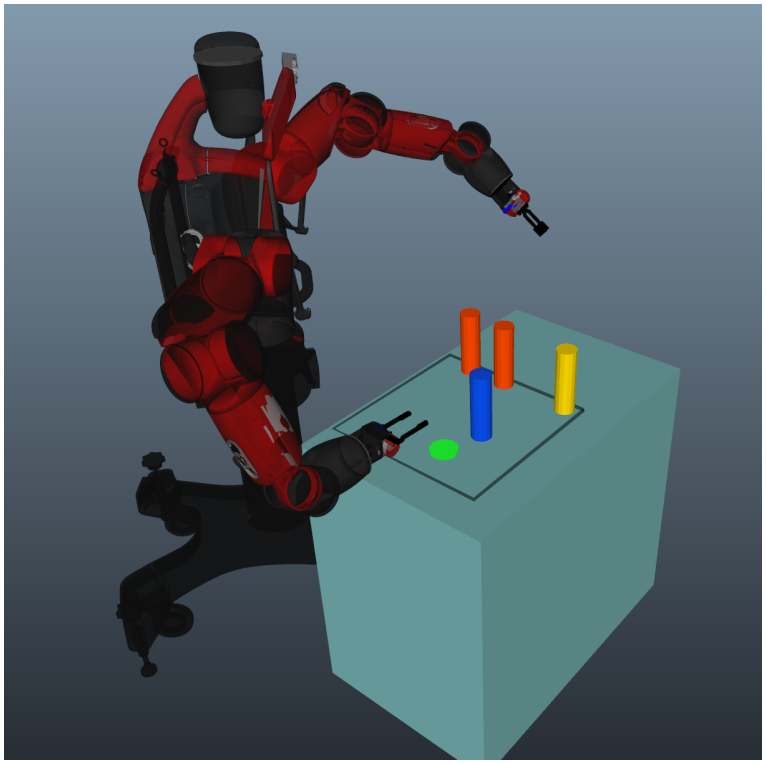
To interact with the virtual environment, we use the Leap Motion [2], a hand gesture motion-sensing hardware to teleoperate either a robot end-effector or the hand skeleton for each specific task respectively. To match the hand movement to the robot action, we track the velocity of the hand movement as well as the rotation angle of the wrist and update the robot's state at each time step.

We consider a teleoperation task to achieve pick-and-place motion by controlling the robot's end-effector or the hand skeleton. As shown in Figure 3.2, the environment consists of graspable objects (cylinders in red or yellow), obstacles to avoid (cylinders in blue), and a goal position (green circle). The task is to grasp a target object (yellow cylinder) among multiple objects (red cylinders) and retrieve the object to the goal position, participants are asked not to collide the obstacles during teleoperation. The robot does not have prior knowledge of the target object.

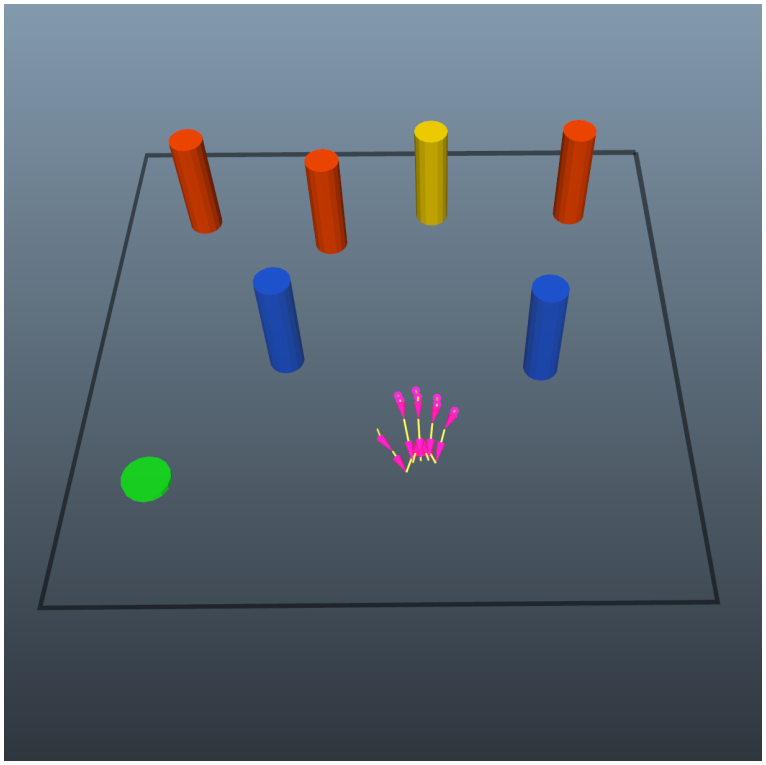
The two models we proposed in this thesis have a slightly different setup. The setup of the *target object prediction model* as shown in Figure 3.2a predicts which object at the scene will be the intended goal with the highest probability. The robot gripper can be manipulated over a 2D plane, in which the height of the gripper is fixed and predefined according to the placement of all the objects. To perform the more flexible movement for avoiding the collision with the obstacle in the scene, the gripper can be given rotation around the Z-axis. In this scenario, all graspable objects are aligned on a line.

The *point of grasp prediction model* goes a step further to predict which part of the goal will be the grasping point as illustrated in Figure 3.2b. Instead of manipulating the Baxter robot, we use the hand skeleton to perform the grasping task to have more various trajectories and fewer constraints of the robot's physical morphology. The hand skeleton is captured by the Leap Motion as well. Unlike the previous experiment on the robot, the hand skeleton has six degrees of freedom and has information at each segment of the finger (e.g., translation and rotation). Also, unlike the previous setup, the objects are not aligned over a line. To obtain information about the environment we set up a virtual camera to capture depth images of the scene.

[1] https://www.ros.org/
[2] https://www.leapmotion.com/



(a) Target object prediction scenario.



(b) Point of grasp prediction scenario.

Figure 3.2: Environment setup.

3.2 Target Object Prediction Model

In the target object prediction model, the aim is described as follows: *Predict the most likely goal and model the uncertainty as a one-dimensional probability distribution given time-series data*. When reaching an object, in the beginning the uncertainty over the objects should be relatively similar and as the robot reaches for an object the robot should gain more confidence about the object being the goal. We train the model to learn the parameters of Beta distribution and allocate the probability over the line where the objects locate. Figure 3.3 illustrates the uncertainty and trajectory changes during different time steps.

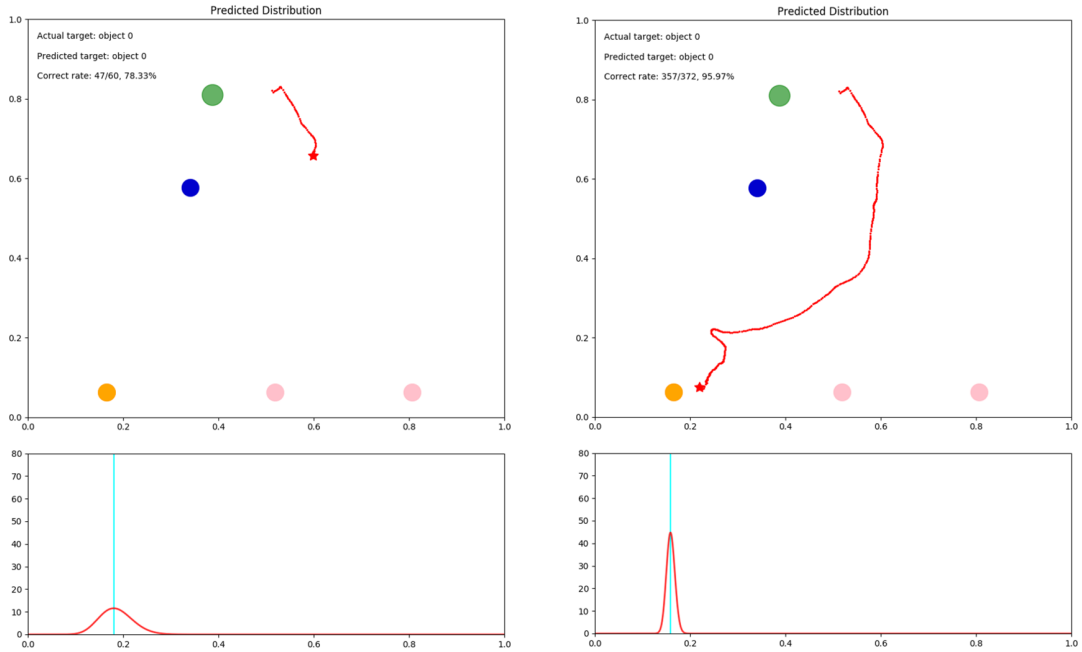


Figure 3.3: Target object inference in two time instances, in the earlier state of the episode the distribution is flat (left), while gripper approaching the goal, the distribution is getting steeper (right).

3.2.1 Features

The following features are used for the target object prediction model:

- **Inputs**:

1. Grab status: The status of whether the user has grasped the object or not. The feature is used to distinguish the state of picking and placing in the pick-and-place tasks.
2. Hand velocity: The velocity data of the XY-plane captured from the Leap Motion device, the user teleoperates the robot based on velocity control.

3. Hand position: The relative position based on the initial hand position in each time step. This feature is recorded directly from the Leap Motion device not from the robot's movement, in order not letting the robot's movement affect human intent. Only use data in the XY-plane.
4. Hand pointing angle: The yaw angle of palm from the Leap Motion, to give the model more clear cue of hand motions.

- **Outputs**: Parameters of Beta distribution (α and β).
- **Ground truth**: Parameters of Beta distribution generated based on the distance between the gripper position and the goal.

3.2.2 LSTM Model Design

We use LSTM to learn the latent space representation of the relationship between cumulative information and the environment. To model the distribution over the objects, we represent it using the Beta distribution. As shown in Figure 3.4, the model predicts a set of parameters that comprise the Beta distribution given sequential user commands. After modeling the distribution using the predicted parameters, the object is predicted as the target with the highest probability.

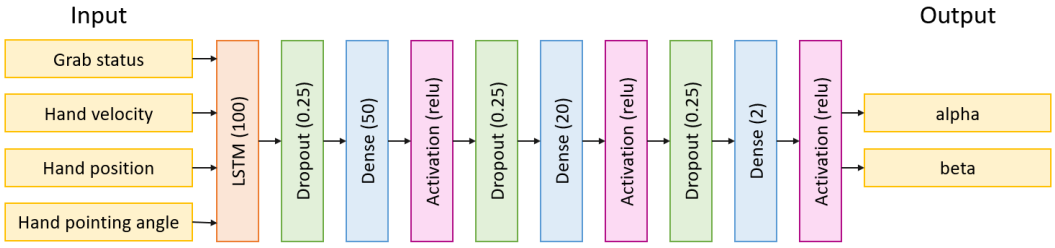


Figure 3.4: The architecture of the LSTM model.

The model takes as input sequential user command data captured from the Leap motion such as grab, hand velocity, hand position, and the angle in which the hand is pointing. Here, we use an LSTM layer to capture any relationships in the sequential data. The LSTM layer has 100 outputs followed by three dense layers comprising 50, 20, 2 outputs respectively. The outputs from the last Dense layer are two parameters of the Beta distribution, α and β. In order to train the network in a supervised manner, we represent a Beta distribution with variance (VAR) given the distance between the gripper and the object, as shown in Equation 3.1 and 3.2. To have a single-modal probability density function of the Beta distribution, a restriction is that k should be greater or equal to two. The variance of the Beta distribution is proportional to the reciprocal of k. The targets α and β are obtained through the relationship between the parameters and k.

$$k = C \times \frac{1}{dist(gripper, object)}, \; where \; k = \alpha + \beta \; and \; 2 \leq k, \; C \; is \; a \; constant \tag{3.1}$$

$$VAR[X] \propto k^{-1} \tag{3.2}$$

With these two parameters, we can estimate the uncertainty of the prediction by the Beta distribution and infer the maximum likelihood using the mode (Equation 3.3) given the position of the objects.

$$Mode = \frac{\alpha - 1}{\alpha + \beta - 2},\ for\ \alpha, \beta > 1 \tag{3.3}$$

3.2.3 Baseline

We use the goal prediction method in [DS13] as the baseline. Given initial gripper position, object positions, and current position, the baseline predicts the target object that maximizes its posterior probability. The baseline method is a very simplified memory-based prediction, which takes into account the trajectory from initial point to the current state.

We denote the gripper's initial point as S, a set of potential goals in each episode as G. U denotes the current state of the gripper, here means the velocity captured from the Leap Motion. We denote g^* as the most likely goal and $\xi_{S \to U}$ as the trajectory from the initial point to the current state.

The goal prediction can be formulated as shown in Equation 3.4, the predicted goal g^* is the maximizes posterior probability calculated by trajectory $\xi_{S \to U}$.

$$g^* = arg \max_{g \in G} P(g|\xi_{S \to U}) = arg \max_{g \in G} P(\xi_{S \to U}|g)P(g) \tag{3.4}$$

To calculate the $P(\xi_{S \to U}|g)$ term above, we use a cost function C_g to optimize the user inputs [ZMBD08]. Then the $P(\xi_{S \to U}|g)$ can be expressed as Equation 3.5.

$$P(\xi_{S \to U}|g) = \frac{exp(-C_g(\xi_{S \to U})) \int_{\xi_{U \to g}} exp(-C_g(\xi_{U \to g}))}{\int_{\xi_{S \to g}} exp(-C_g(\xi_{S \to g}))} \tag{3.5}$$

The integral over trajectories can be approximated by Laplace's method, and the cost function $C(\xi_{X \to Y})$ can be approximated by its second-order Taylor series expansion shown in Equation 3.6.

$$\xi^*_{X \to Y} = arg \min_{\xi_{X \to Y}} C(\xi_{X \to Y}) \tag{3.6}$$

In the end, the predicted goal can be approximated and formulated in Equation 3.7.

$$g^* = arg \max_{g \in G} \frac{exp(-C_g(\xi_{S \to U}) - C_g(\xi^*_{U \to g}))}{exp(-C_g(\xi^*_{S \to g}))} P(g) \tag{3.7}$$

3.2.4 Blended Model

We found out that the LSTM model which we previously proposed has a better inference performance when avoiding the obstacle collision, and the baseline method can have a more accurate prediction in beginning stages. To combine their strengths in each method, we further trained a blended model that can take advantage of each method regarding the given state. The structure of the blended model is shown in Figure 3.5 and is similar to the model structure we proposed in Figure 3.4, but combines the results of both methods and use Multi-Layer Perceptron (MLP) to distinguish the best prediction. The three inputs to the MLP is the output of the LSTM network, baseline predictions, and information of the environment. The MLP outputs a predicted object given each prediction results.

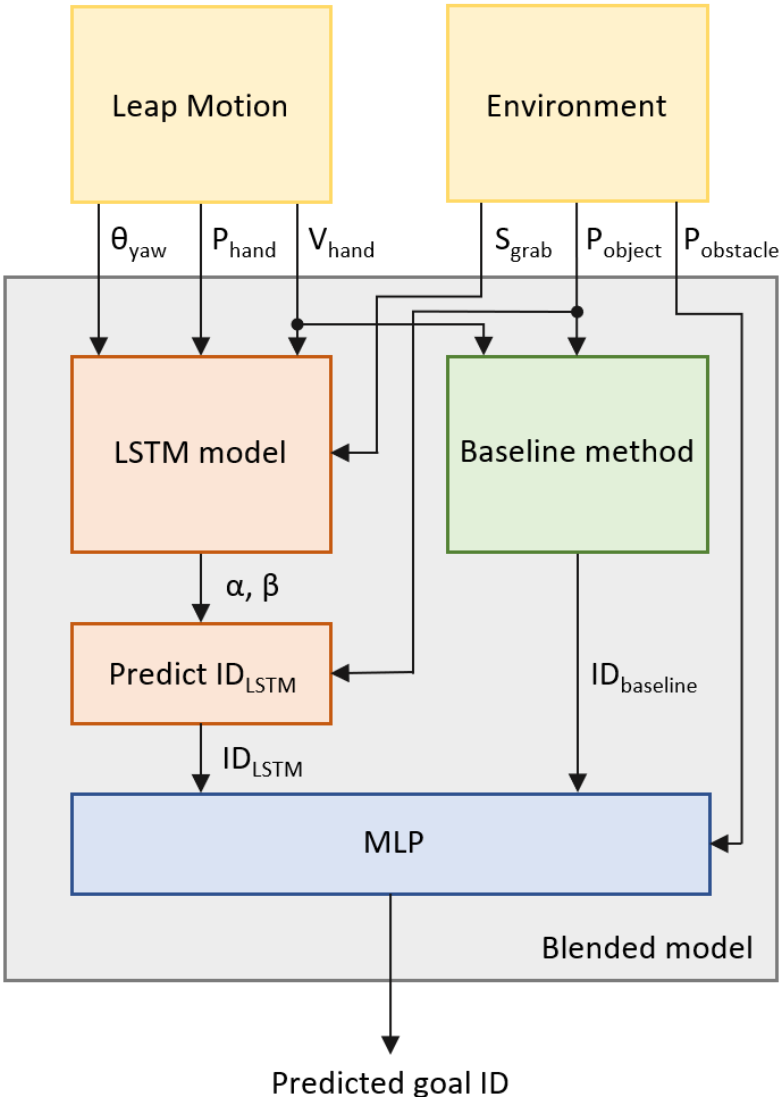


Figure 3.5: The architecture of the blended model.

3.3 Point of Grasp Prediction Model

When referring to grasp affordances in our pick-and-place task, the agent denotes the hand skeleton. Grasp affordances can be explained as the success effect of a grasp action performed to an object. Grasp density is concerned to the region where the hand would be in contact to the object. In this section, we propose a framework to predict the point of grasp on the most likely goal where the states of the objects are not given but must be inferred. We use virtual depth images as input to infer where the user will grasp and further predict the uncertainty for grasp densities of the object in the 2D manifold, which can be converted to points in 3D space.

The grasping point prediction framework can be divided and discussed in two parts:

1. CNN-BiLSTM autoencoder.
2. LSTM-MDN model.

3.3.1 CNN-BiLSTM Autoencoder

In this model, the states of objects and obstacles are not directly given. Instead, the network should interpret the scene given depth images of the environment. Thus, obtaining information from the scene is primacy. We use an autoencoder-like structure which consists of CNN layers to extract features from the image and Bidirectional LSTM (BiLSTM) layers to support prediction using temporal data.

Features

- **Inputs**: Sequential depth images of seven time steps.
- **Outputs**: Binary-like mask images.
- **Ground truth**: Binary images which segmented the object from the scene.

Model Structure

To deal with sequential images, we propose an autoencoder that includes BiLSTM structure as shown in Figure 3.6. The encoder part of the model is used in the following point of grasp prediction model to reduce the feature dimensions and learn stronger environment representations.

We first crop the raw depth image into 360×360 region of interest in the center, then downsample to 100×100 one channel image. The look-back size of the BiLSTM is seven as well. The autoencoder consists of the encoder and decoder. The encoder part compresses the input into a latent space representation. The decoder part aims to reconstruct the inputs to mask images from the latent space. We use CNN to extract and learn image features in both encoder and decoder parts. At the bottleneck, we use the repeat vector layer to give the decoder a sequential input. The outputs of the autoencoder are mask images, we trained the model with standard mean squared loss.

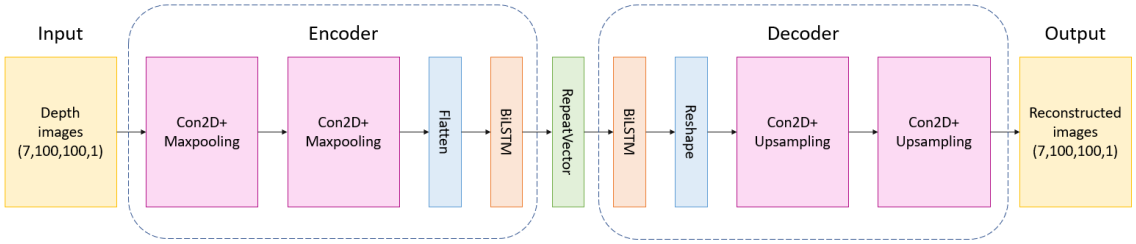


Figure 3.6: The architecture of the CNN-BiLSTM autoencoder.

3.3.2 LSTM-MDN Model

In this section, the LSTM-MDN model is proposed to infer the uncertainty of grasp densities over the intended goal object as well as the normal vector of the final grasping orientation. The time-series hand skeleton data is also used in this case, combining with the trained encoder in 3.3.1. The uncertainty of grasp density is modeled by MDN. The probability distribution for expected points of grasp is multi-valued; for instance, when the user's hand is far away from all the objects, the potential points of grasp could be multi-valued on the highly likely objects. The MDN is an approach to deal with this multi kernels distribution circumstance.

Features

- **Inputs**:
    1. Hand velocity: The velocity data of the XY-plane captured from the Leap Motion device.
    2. Hand position: The relative position to the initial point described in the target object prediction model.
    3. Palm vector: The three-dimensional unit normal vector from the center of palm, denotes the grasping direction of the action.
- **Outputs**:
    1. MDN parameters: A set of parameters of Gaussian distribution. For each two-dimensional Gaussian kernel, it contains five parameters: mean, variance and weight. In this thesis, we choose to use five kernels.
    2. Palm vector: A three-dimensional vector.
- **Ground truth**:
    1. Grasping position: The final contact point between the hand and the object.
    2. Palm vector: The final unit normal vector from hand.

Model Structure

We use a multi-task transfer learning technique to infer the probability distribution and the grasping orientation as illustrated in Figure 3.7. The model has two branches of inputs and outputs.

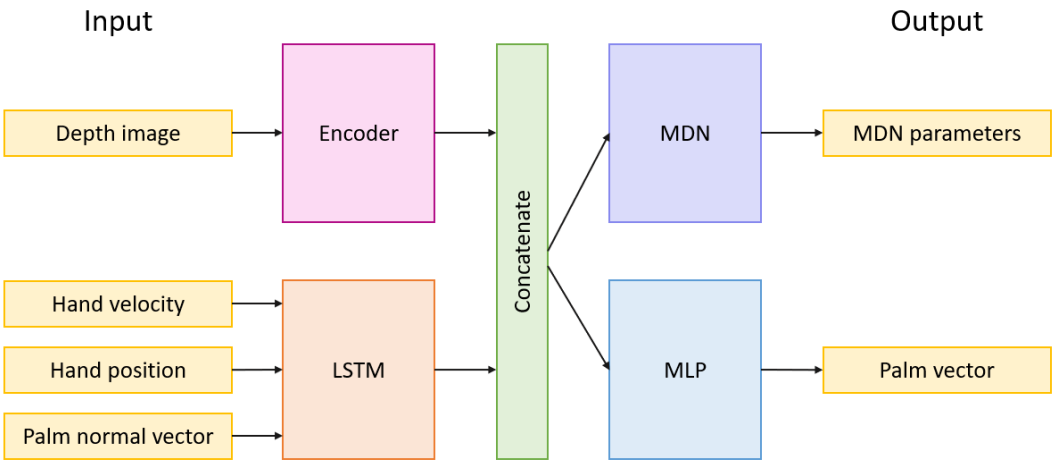


Figure 3.7: The architecture of the point of grasp prediction model.

The first input branch is the pre-trained encoder described in section 3.3.1, which learns the significant environment representations from 2.5D depth images. The weights of the encoder are taken from the autoencoder that was trained in advance to predict masks. The weights of the encoder are no longer trained in the model. The intuition is to capture the latent space and the network should learn to distinguish the objects from the background. The second branch takes the sequential features of the hand (look-back size = 7) as input and connected to an LSTM layer. The two branches of input are then concatenated. The model predicts two different data: the parameters of MDN and the normal vector of the hand.

We design MDN to map with multivariate Gaussian distribution with diagonal covariance. The number of kernels is chosen from observation, we found that using five kernels to represent the MDN is enough in this case. Thus, the output of MDN are five sets of Gaussian distribution parameters. The other output is the final contact vector when the hand grasped the object. Each output is optimized based on different loss functions, NLL of mixture Gaussian distribution for MDN and mean squared error for palm vector.

3.3.3 Baseline

To evaluate our model, we create a baseline method to predict the point of grasp based on hand movement. We calculate the point of intersection between an object and the line extended from the hand direction vector at each time step. The intuition is that the user will take a sub-optimal path towards objects and the prediction is depended on hand's moving direction. The baseline method does not take the grasping orientation into account. Thus, we only compare the grasping point with this method.

3.3.4 Block diagram

Figure 3.8 shows the architecture of the final prediction model. Notably, the outputs of the MDN are five sets of parameters in our model, we use mixture Gaussian distribution that derived from the collection of these predicted kernels. The density of the mixture distribution can also be visualized in RViz as the grasp probability.

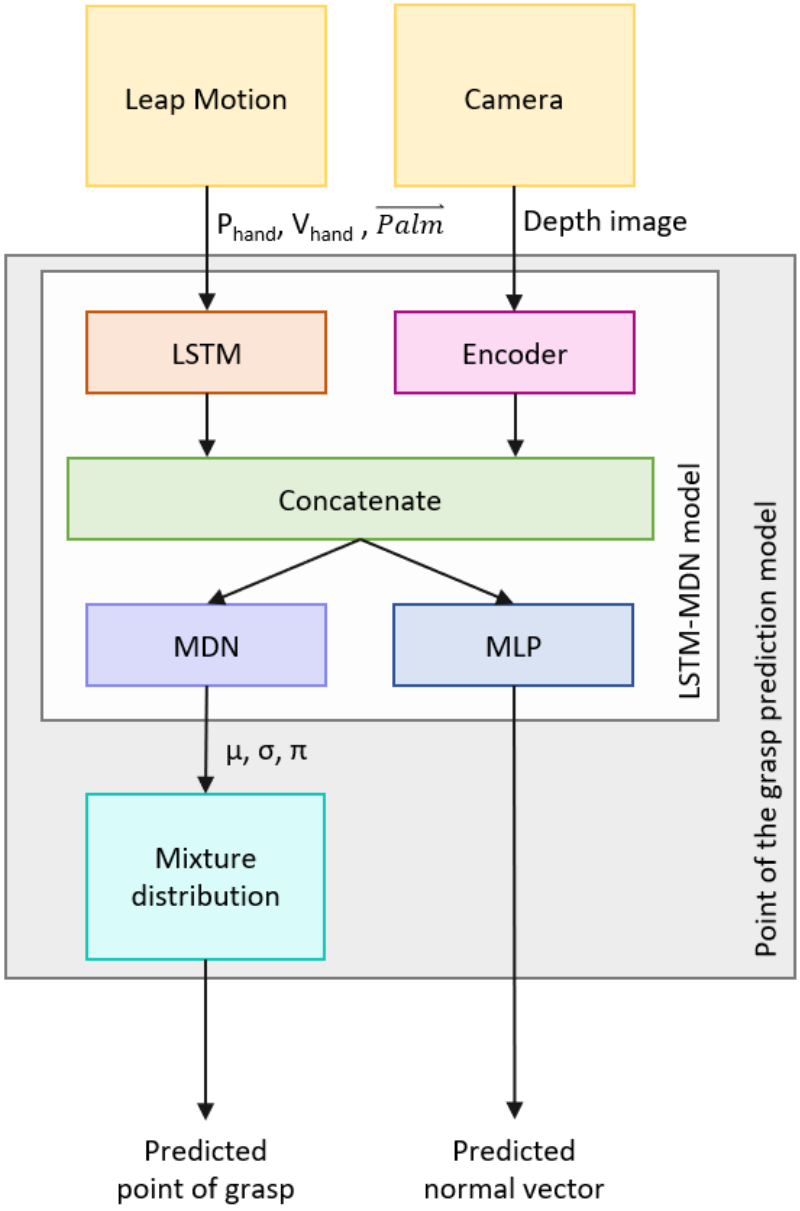


Figure 3.8: Block diagram of the point of grasp prediction model.

4 Evaluation

In this chapter, we evaluate and discuss the results of the methods. Section 4.1 shows the result of the *target object prediction model* proposed in section 3.2, section 4.2 shows the result of the *point of grasp prediction model* proposed in section 3.3.

To evaluate the *target object prediction model*, the user studies were conducted with 16 participants. Before each experiment, participants were allowed to practice the whole experiment in less than ten episodes to have a basic knowledge of the teleoperating system. The control inputs are captured by the Leap Motion sensor. Each participant was asked to perform a sequence of pick-and-place tasks for 30 episodes. The final effective data contains 467 episodes in 113,790 samples. To fairly validate the model performance, we classified the trajectories into five groups and collected ten episodes respectively, resulting in a total of 50 episodes.

In the *point of grasp prediction model*, the training data was collected from 60 different environment settings. In each setting, participants were asked to perform two grasp types (top grasp and side grasp) for all four objects separately. A mount of eight types of grasping categories was equally collected. The final effective data contains 467 episodes corresponding to 75,485 samples. The model was validated on 40 episodes.

We implemented our network using Keras library [1] and TensorFlow [2] framework as back-end.

4.1 Target Object Prediction Results

In this section, we explain the feature selection process and evaluate the results of each model. We assert not to use the action from the robot, but only use motion directly from hand movement detected by the sensor. With the predefined initial position of the gripper and the velocity data received from gesture sensor, we can calculate the user command for the robot action [AA19]. We intended to train the model using less environment information as possible. Thus, selecting a subset of relevant features is a key aspect. We conduct a feature selection procedure to discover the crucial features. Table 4.1 shows the results of different combinations of features and pre-processing methods. The results are obtained from the model where the yaw angle of the gripper is not considered. The difference between 7 and 13 features is whether the position of the objects is included or not. The absolute hand position uses the raw hand position data as input and the relative hand position refers to the current hand position subtracted by the initial position of the hand. As shown in Table 4.1, the model trained with the relative hand position and unknown object positions has the highest validation result.

[1] https://keras.io/
[2] https://www.tensorflow.org/

Method	Feature Numbers	Hand Position	Object Positions	Validation Result
Beta Distribution	13	absolute	known	83.38%
Beta Distribution	7	absolute	unknown	84.08%
Beta Distribution	13	relative	known	98.29%
Beta Distribution	7	relative	unknown	98.59%

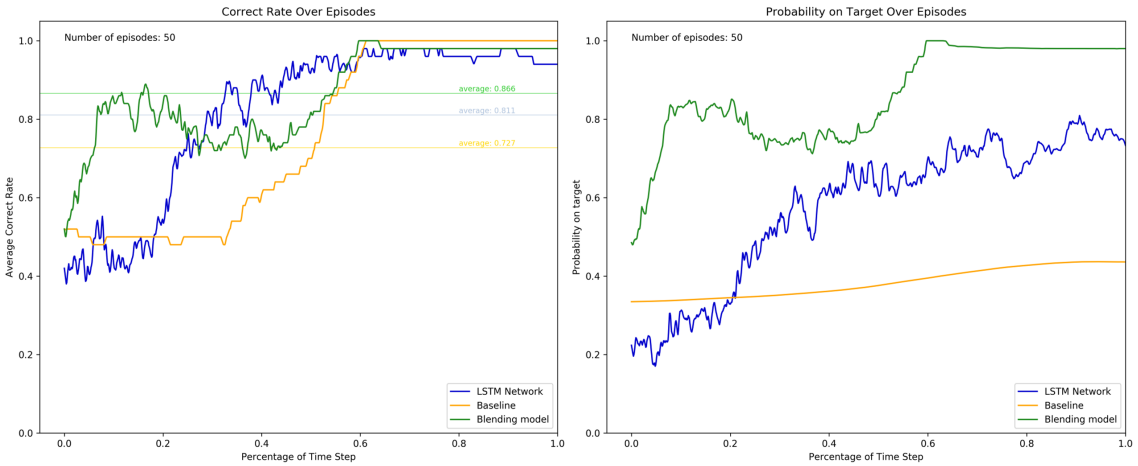
Table 4.1: Feature selection and comparison.

In Table 4.2, we show the training results of our two models. These two models contain dropout layers, therefore, the test set loss is lower than train set loss.

Model	Epochs	Batch size	Train set loss	Test set loss
LSTM model	279	512	0.0997	0.0805
Blended model	278	512	0.0163	0.0141

Table 4.2: Results obtained for the target object prediction models.

Figure 4.1a shows the correct rate of the three methods averaged over episodes. The baseline method (orange) appears to correctly predict the goal object during the last 40% of the episode duration usually the gripper has passed the obstacle. The LSTM network (blue) has the lowest correct rate at the beginning but increases drastically during 40 ~ 60% of the episode duration, at this period the user is usually trying to avoid the obstacle. This shows the LSTM network has the strength to deal with the obstacle avoidance situation. Finally, the blended model (green) combines the strengths of these methods to infer quicker in the first 20% of the episode duration. Even though the model may incorrectly predict during the middle of the episode, it still shows good performance at the beginning of the episode with the highest overall correct rate. Figure 4.1b illustrates the probabilities of the intended object over the episode.



(a) Average correct rates of the predicted target.

(b) Probabilities of the predicted target.

Figure 4.1: Prediction results of target object prediction models.

Table 4.3 shows the average correct rate of our LSTM intent prediction model is 81.1%, the baseline is 72.7% and our blended model has the highest correct rate with 86.6%.

Average correct rate		
LSTM model	Baseline method	Blended model
81.1%	72.7%	86.6%

Table 4.3: Results of target object average correct rate.

An example of the prediction over time in an episode is shown in figure 4.2. The model only has a few wrong steps of prediction while the gripper is avoiding the obstacle then gives a correct and stable prediction until the end.

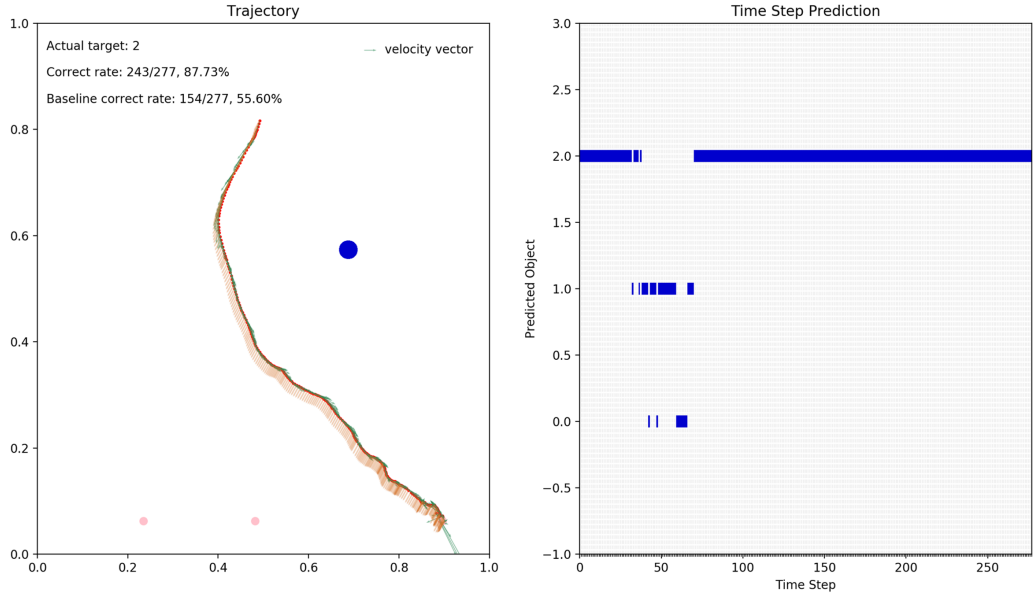


Figure 4.2: An example of prediction over time in an episode.
Left: Gripper trajectory. Right: Prediction in time step.

4.2 Point of Grasp Prediction Results

In the case of the *point of grasp prediction model*, we first evaluate results of the pre-trained autoencoder, then analyze results of the point of grasp and palm vector inference.

For the transfer learning technique, we trained an autoencoder-like network to extract relevant environment dependencies. Table 4.4 shows the autoencoder training results.

Model	Epochs	Batch size	Train set loss	Test set loss
Autoencoder	165	64	5.43E-05	1.20E-04

Table 4.4: Training results of the CNN-BiLSTM autoencoder.

Examples of the input depth image, ground truth, and corresponding reconstructed output image are shown in Figure 4.3. The hand is represented as a circle in the image. As can be seen from the figures, the output images are almost identical to the ground truth, which means the network has learned representation and is capable to reconstruct an image from the latent space.

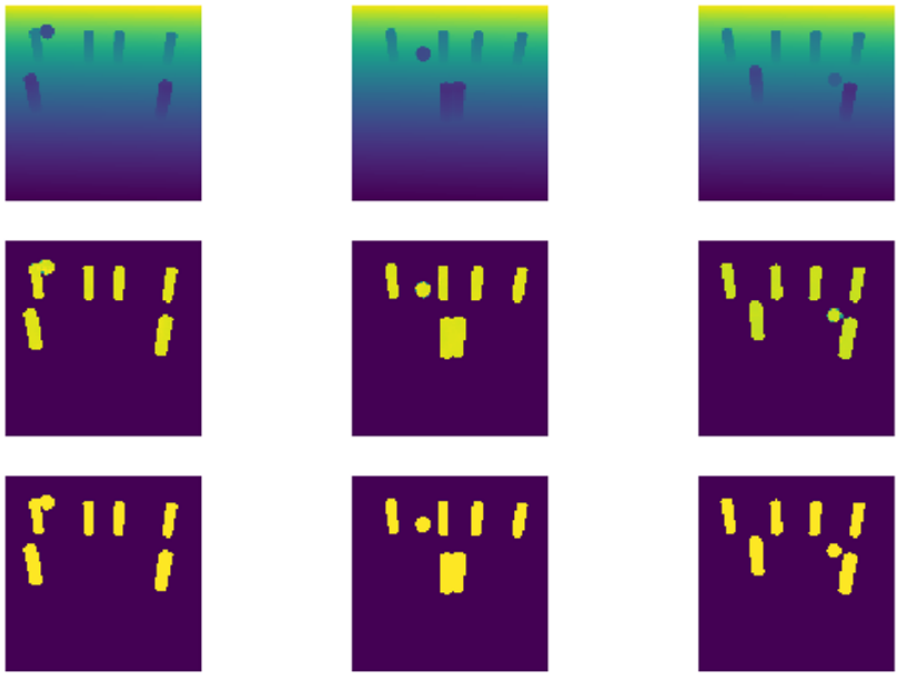


Figure 4.3: Reconstructed images of the autoencoder. The first row shows the input images, the second row shows the ground truth, the third row shows the reconstructed images.

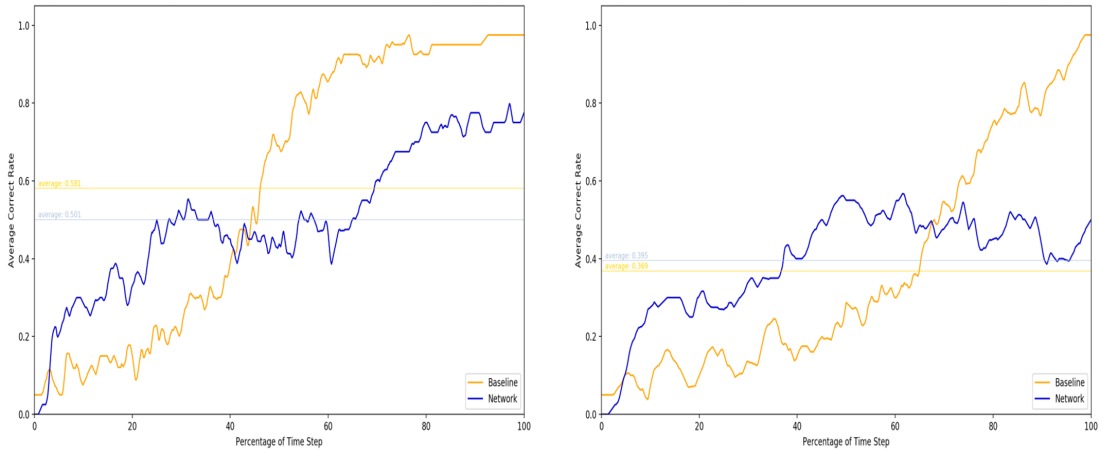
The model infers the point of a grasp as well as the normal vector of the palm. Table 4.5 shows the results with the lowest achieved loss. The output of the MDN branch is a set of multivariate Gaussian distributions. We sampled the mean of mixture Gaussian distribution and trained with NLL loss. The mean position corresponds to the predicted location of the point of grasp. The palm vector branch is trained by MSE loss using the normal vector of the final contact point as the ground truth.

			Train set loss		Test set loss	
Model	Epochs	Batch size	NLL	MSE	NLL	MSE
LSTM-MDN	45	32	-0.6562	2.23E-04	-0.2086	2.24E-04

Table 4.5: Training results of the LSTM-MDN model.

For evaluating the point of grasp inference, we first convert the mean point position of the mixture Gaussian distribution in the depth image to a 3D environment coordinate. We then divide each object into two segments for point of grasp detection. We have four objects in this scenario, equivalent to eight components in total. The baseline method is introduced in section 3.3.3. The validation set is captured fairly from four objects individually with two grasping poses, top grasp, and side grasp.

Figure 4.4 shows the result of the point of grasp prediction in two different postures. In figure 4.4a, we show the result throughout the whole episode from start to finish when the user is confident enough to grasp the object. Because our system does not have haptic feedback, the user needs to aim the goal by observing the hand skeleton gesture and object position in the visualization tool, which would cause noise and fluttering during this procedure. By observing the all episode time condition, our deep learning model has better performance at the beginning of an episode, while the baseline method lags when the hand is far away from the object. As long as the hand approaches the object and starts focusing on the grasping action, the correct rate of the baseline increases and eventually performs better than the deep learning model. To have a deeper understanding of the early steps in each episode, we also compare the correct rate from start until the time before the hand first contacts the object as shown in 4.4b. Table 4.6 shows the statistical data of correct rate results.



(a) Average correct rate **(b)** Average correct rate before contact

Figure 4.4: Correct rate of the grasping point prediction.

Correct rate of the point of grasp prediction			
All episode time		Duration before contact	
Network model	Baseline method	Network model	Baseline method
50.1%	58.1%	39.5%	36.9%

Table 4.6: Correct rate of the point of grasp prediction.

To evaluate the predicted vector, we calculate the cosine similarity between ground truth and the predicted vector. Figure 4.5 illustrates the result statistically analyzed on 40 validation data equally sampled from every object. The cyan color line shows the average trend over time, the red color line represents the overall average. The overall cosine similarity average is 0.948 corresponding to

18.55 degrees. In the last around 5% of episode time, the performance decreased to around 0.8. This is because our system does not contain the perception of physical properties when the user is aiming to the object and will cause a drifting phenomenon.

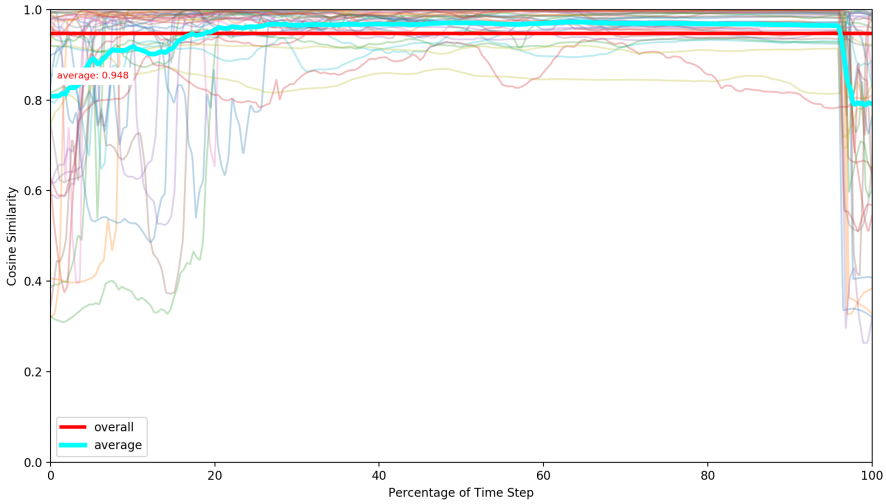


Figure 4.5: Cosine similarity of palm normal vector prediction.

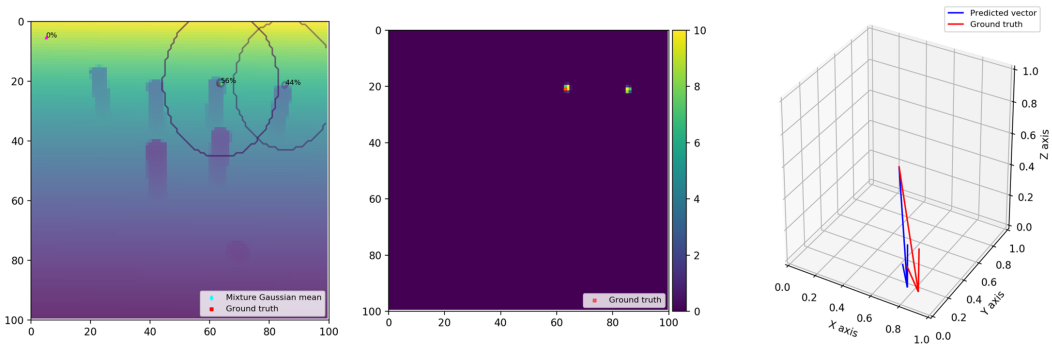
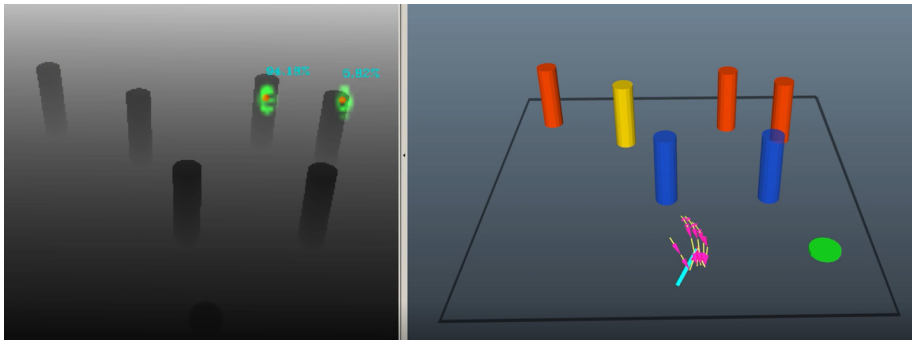
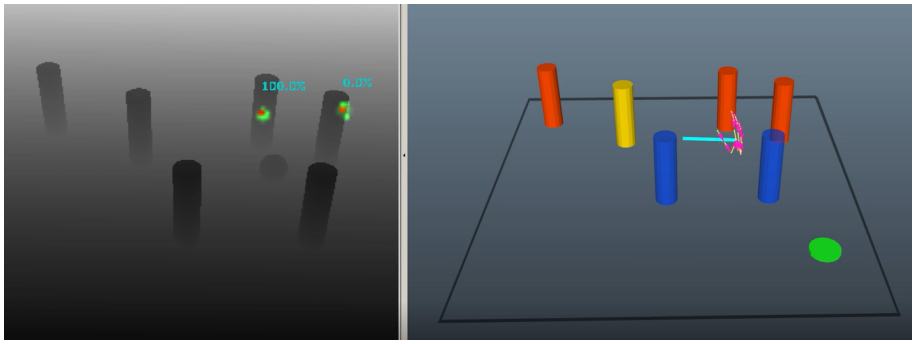


Figure 4.6: An example of the LSTM-MDN prediction result. Left: The prediction plotted on depth image, circles show the confident area of the predicted point of grasp. Middle: The probability value calculated from weight over each pixel. Right: The prediction of palm normal vector, the red arrow represents the ground truth, the blue arrow denotes the predicted vector.

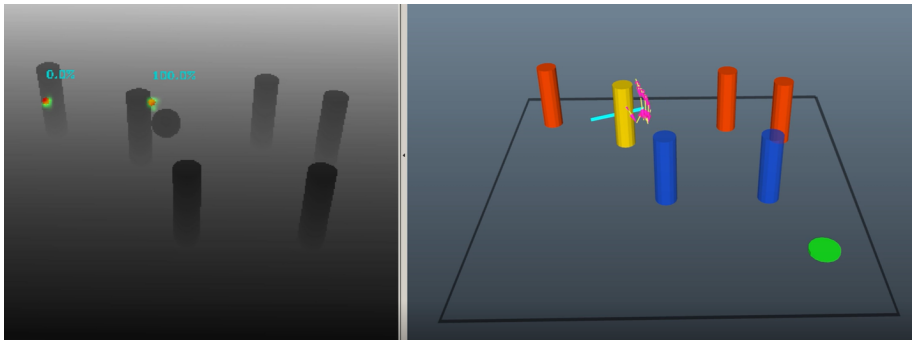
In Figure 4.7 we show the MDN changing trends in an episode, the time sequence is from (a) to (d). On the left side of each sub-figure is the depth image taken from the virtual depth camera and overlapped with the MDN result at run-time. The multivariate Gaussian distributions can be visualized as a green cluster accompanying with its weight. To make it simple and less noisy, we use a sphere to represent the hand skeleton in the view of the depth image. On the right side of each sub-figure, we present the virtual environment to see the actual movement in the scene, the cyan color line from the center of palm denotes the predicted normal vector.



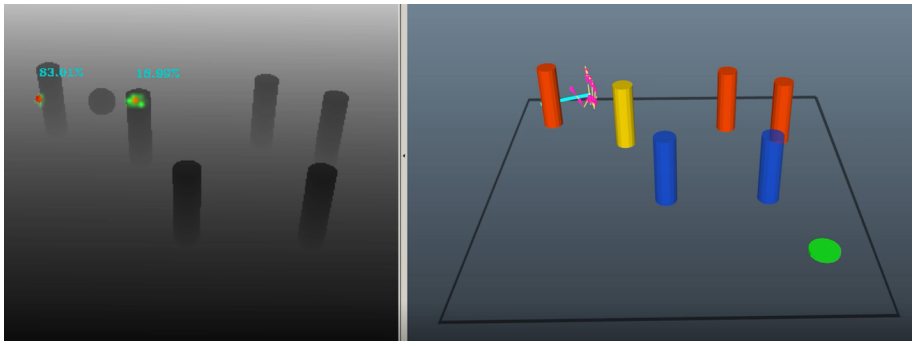
(a)



(b)



(c)



(d)

Figure 4.7: Grasp densities examples sampled in four time instances.

5 Conclusion

We proposed a framework using neural networks to predict human intent in pick-and-place scenarios capturing human control from the gesture sensor. We first estimate the low-dimensional uncertainty using the Beta distribution by capturing implicit intents. Further, we predict which object at the workspace is the most likely goal given the position of objects. Next, we model the high-dimensional grasp uncertainty using multivariate Gaussian probability distributions as possible graspable regions. We not only predict the human intent of grasping point on the surface of the object and also the grasping orientation. The models are implemented and evaluated at run-time.

5.1 Limitation

A limitation of the *target object prediction model* is that we model uncertainty with the Beta distribution, which is a one-dimensional probability distribution. This is sufficient when applying to the plane scenario where the objects are not overlapping. While moving to higher dimensional circumstances, multi-dimensional probability distribution should be considered. Another limitation is that the Beta distribution is a single-modal distribution it can not be applied to multi-valued situations or the network should be also trained as the MDN type of mixture model.

One limitation of our *point of grasp prediction model* is that we obtained the scene information and trained the neural networks only from one perspective of view, which will limit the range of activity, especially when we want to perform grasping on unmodeled objects. Also, the data we trained on the model only contains the right-hand movement. It would need more features or data when moving to both hands manipulation tasks.

5.2 Future Work

Future work includes performing the grasping tasks in cluttered and novel environments. In this case, the network first has to be capable of segment variant unseen objects from the scene and then predict the movability on the whole item. The grasping orientation can not only be obtained from the normal palm vector but also should include each orientation of the fingers to have a better understanding of grasp affordances. A hand haptic feedback device or multiple gesture sensors could be considered to improve the prediction precision since the fluttering would cause additional noise interference.

Bibliography

[AA19] R. M. Aronson, H. Admoni. "Semantic gaze labeling for human-robot shared manipulation". In: *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. ACM. 2019, p. 2 (cit. on pp. 15, 35).

[AS16] H. Admoni, S. Srinivasa. "Predicting user intent through eye gaze for shared autonomy". In: *2016 AAAI Fall Symposium Series*. 2016 (cit. on pp. 13, 16).

[BDM02] D. J. Bruemmer, D. D. Dudenhoeffer, J. L. Marble. "Dynamic-Autonomy for Urban Search and Rescue." In: *AAAI mobile robot competition*. 2002, pp. 33–37 (cit. on p. 15).

[Bis94] C. M. Bishop. "Mixture density networks". In: (1994) (cit. on p. 18).

[BK00] A. Bicchi, V. Kumar. "Robotic grasping and contact: A review". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 1. IEEE. 2000, pp. 348–353 (cit. on p. 20).

[BKP11] A. Boularias, O. Kroemer, J. Peters. "Learning robot grasping from 3-d images with markov random fields". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2011, pp. 1548–1553 (cit. on pp. 21, 22).

[BLB14] P. Birkenkampf, D. Leidner, C. Borst. "A knowledge-driven shared autonomy human-robot interface for tablet computers". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 152–159 (cit. on p. 15).

[BSA+03] J. M. Bradshaw, M. Sierhuis, A. Acquisti, P. Feltovich, R. Hoffman, R. Jeffers, D. Prescott, N. Suri, A. Uszok, R. Van Hoof. "Adjustable autonomy and human-agent teamwork in practice: An interim report on space applications". In: *Agent autonomy*. Springer, 2003, pp. 243–280 (cit. on p. 15).

[CHC04] I. Cos-Aguilera, G. Hayes, L. Canamero. "Using a SOFM to learn object affordances". In: *Procs 5th Workshop of Physical Agents (WAF'04)*. University of Edinburgh. 2004 (cit. on p. 19).

[CMS17] P.-W. Chou, D. Maturana, S. Scherer. "Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 834–843 (cit. on p. 18).

[DD09] A. Der Kiureghian, O. Ditlevsen. "Aleatory or epistemic? Does it matter?" In: *Structural Safety* 31.2 (2009), pp. 105–112 (cit. on p. 16).

[DGQ+13] J.-Y. Dantan, N. Gayton, A. J. Qureshi, M. Lemaire, A. Etienne. "Tolerance analysis approach based on the classification of uncertainty (aleatory/epistemic)". In: *Procedia CIRP* 10 (2013), pp. 287–293 (cit. on p. 16).

[DKK+11] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, J. Piater. "Learning grasp affordance densities". In: *Paladyn, Journal of Behavioral Robotics* 2.1 (2011), pp. 1–17 (cit. on pp. 20, 21).

[DS13] A. D. Dragan, S. S. Srinivasa. "A policy-blending formalism for shared control". In: *The International Journal of Robotics Research* 32.7 (2013), pp. 790–805 (cit. on pp. 13, 28).

[EMT19] K. O. Ellefsen, C. P. Martin, J. Torresen. "How do Mixture Density RNNs Predict the Future?" In: *arXiv preprint arXiv:1901.07859* (2019) (cit. on pp. 18, 19).

[FMN+03] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, G. Sandini. "Learning about objects through action-initial steps towards artificial cognition". In: *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*. Vol. 3. IEEE. 2003, pp. 3140–3145 (cit. on p. 19).

[FRPG04] A. Fagg, M. Rosenstein, R. Platt, R. Grupen. "Extracting user intent in mixed initiative teleoperator control". In: *AIAA 1st Intelligent Systems Technical Conference*. 2004, p. 6309 (cit. on p. 13).

[FV12] D. Fischinger, M. Vincze. "Empty the basket-a shape based learning approach for grasping piles of unknown objects". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 2051–2057 (cit. on p. 20).

[GG15] Y. Gal, Z. Ghahramani. "Bayesian convolutional neural networks with Bernoulli approximate variational inference". In: *arXiv preprint arXiv:1506.02158* (2015) (cit. on p. 16).

[GG16] Y. Gal, Z. Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning". In: *international conference on machine learning*. 2016, pp. 1050–1059 (cit. on p. 16).

[Gib14] J. J. Gibson. *The ecological approach to visual perception: classic edition*. Psychology Press, 2014 (cit. on p. 19).

[Gra13] A. Graves. "Generating sequences with recurrent neural networks". In: *arXiv preprint arXiv:1308.0850* (2013) (cit. on p. 18).

[HM16] C.-M. Huang, B. Mutlu. "Anticipatory robot control for efficient human-robot collaboration". In: *The eleventh ACM/IEEE international conference on human robot interaction*. IEEE Press. 2016, pp. 83–90 (cit. on p. 16).

[JAP+18] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, J. A. Bagnell. "Shared autonomy via hindsight optimization for teleoperation and teaming". In: *The International Journal of Robotics Research* 37.7 (2018), pp. 717–742 (cit. on p. 15).

[Jia95] Y.-B. Jia. "On computing optimal planar grasps". In: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 3. IEEE. 1995, pp. 427–434 (cit. on p. 20).

[JMS11] Y. Jiang, S. Moseson, A. Saxena. "Efficient grasping from rgbd images: Learning using a new rectangle representation". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3304–3311 (cit. on p. 20).

[JSB15] S. Javdani, S. S. Srinivasa, J. A. Bagnell. "Shared autonomy via hindsight optimization". In: *Robotics science and systems: online proceedings* 2015 (2015) (cit. on p. 16).

[KRC+11] E. Klingbeil, D. Rao, B. Carpenter, V. Ganapathi, A. Y. Ng, O. Khatib. "Grasping with application to an autonomous checkout robot". In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 2837–2844 (cit. on p. 20).

[KSCK96] E. Krotkov, R. Simmons, F. Cozman, S. Koenig. "Safeguarded teleoperation for lunar rovers: From human factors to field trials". In: *IEEE Planetary Rover Technology and Systems Workshop*. Vol. 21. 1996 (cit. on p. 15).

[KSHK17] M. Kokic, J. A. Stork, J. A. Haustein, D. Kragic. "Affordance detection for task-specific grasping using deep learning". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE. 2017, pp. 91–98 (cit. on p. 21).

[MKCA03] A. T. Miller, S. Knoop, H. I. Christensen, P. K. Allen. "Automatic grasp planning using shape primitives". In: (2003) (cit. on p. 20).

[MLBS08] L. Montesano, M. Lopes, A. Bernardino, J. Santos-Victor. "Learning object affordances: from sensory–motor coordination to imitation". In: *IEEE Transactions on Robotics* 24.1 (2008), pp. 15–26 (cit. on p. 20).

[MSD02] A. Morales, P. J. Sanz, A. P. Del Pobil. "Vision-based computation of three-finger grasps on unknown planar objects". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 2. IEEE. 2002, pp. 1711–1716 (cit. on p. 20).

[NZHS17] S. Nikolaidis, Y. X. Zhu, D. Hsu, S. Srinivasa. "Human-robot mutual adaptation in shared autonomy". In: *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*. IEEE. 2017, pp. 294–302 (cit. on pp. 13, 15).

[RDL18] S. Reddy, A. D. Dragan, S. Levine. "Shared autonomy via deep reinforcement learning". In: *arXiv preprint arXiv:1802.01744* (2018) (cit. on p. 16).

[RPUF17] Y. S. Razin, K. Pluckter, J. Ueda, K. Feigh. "Predicting task intent from surface electromyography using layered hidden Markov models". In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 1180–1185 (cit. on p. 16).

[SBR12] C. Stanton, A. Bogdanovych, E. Ratanasena. "Teleoperation of a humanoid robot using full-body motion capture, example movements, and machine learning". In: *Proc. Australasian Conference on Robotics and Automation*. 2012 (cit. on p. 13).

[ŞÇD+07] E. Şahin, M. Çakmak, M. R. Doğar, E. Uğur, G. Üçoluk. "To afford or not to afford: A new formalization of affordances toward affordance-based robot control". In: *Adaptive Behavior* 15.4 (2007), pp. 447–472 (cit. on p. 19).

[SWN08] A. Saxena, L. L. Wong, A. Y. Ng. "Learning grasp strategies with partial shape information." In: *AAAI*. Vol. 3. 2. 2008, pp. 1491–1494 (cit. on p. 21).

[TK02] G. Taylor, L. Kleeman. "Grasping unknown objects with a humanoid robot". In: *Proc. 2002 Australasian Conference on Robotics and Automation*. Vol. 27. 2002, p. 29 (cit. on p. 20).

[TP16] A. Ten Pas, R. Platt. "Localizing handle-like grasp affordances in 3d point clouds". In: *Experimental Robotics*. Springer. 2016, pp. 623–638 (cit. on pp. 20, 22).

[ZMBD08] B. D. Ziebart, A. Maas, J. A. Bagnell, A. K. Dey. "Maximum entropy inverse reinforcement learning". In: (2008) (cit. on p. 28).

All links were last followed on November 27, 2019.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

---

place, date, signature