Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Learning object affordances using human motion capture data

Niteesh Balachandra Midlagajni

Course of Study:	INFOTECH
Examiner:	Prof. Dr. rer. nat. Marc Toussaint
Supervisor:	Dr. Jim Mainprice, Philipp Kratzer, M.Sc.
Commenced:	April 23, 2019
Completed:	October 23, 2019

Abstract

When interacting with their environment, humans model the action possibilities directly in the product space of their own capabilities using the spatial configuration of their body and the environment. This idea of the existence of an intuitive and perceptual representation of the possibilities in an environment has been hypothesized and discussed by psychologist JJ Gibons, and is called affordances. The goal of this thesis is to build an algorithmic framework to learn and encode human object affordances from motion capture data. In this regard, we collect motion capture data, wherein, the human subjects perform pick and place activities in the scene. Using the collected data, we develop models using neural network architecture to learn graspability and placeability affordances, while also capturing the uncertainty in predictions. We achieve this by modeling affordances within the probabilistic framework of Deep Learning. Our models predict grasp densities and place densities accurately, in the sense that the ground truth is always within the confidence interval. Furthermore, we develop a system and integrate our models for real-time application, in order to produce affordance features in live setting and visualize the densities as heatmaps in real-time.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

CNN Convolution Neural Network. 21

LSTM Long Short-Term Memory. 35

MDN Mixture Density Network. 27

MSE Mean Squared Error. 24

NLL Negative Log Likelihood. 24

SDF Signed Distance Field. 39

vMF von Mises-Fisher. 28

1 Introduction

The pursuit of understanding the fundamentals of human intelligence, in terms of his ability to perceive and interact with the world around is a long seeking research question posed in the world of robotics. Robots might eventually blend in and operate in the same workspace as humans. Building a solid human motion prediction system that can be used by robots to effectively collaborate with humans, while not harming or interfering with them is necessary. One key aspect in this regard is decoding the manner in which humans model the action possibilities in their environment. When tasked with performing certain actions such as grasping an object in the scene, humans subconsciously tend to create a map in their mind about the future and final state they need to be in, to successfully perform that action, and plan their motion accordingly to reach that state. This idea that there exists an intuitive and perceptual representation of the possibilities in an environment is termed as Affordances and was introduced by Gibson[Gib66].

Affordances in a broader sense define the relationship between an agent and the environment in terms of the actions that can be undertaken on objects and the corresponding effects caused by it. In our scenario, we model affordances from the perspective of humans. Most of the work on human affordances have been mainly into solving computer vision problem involving RGB or RGB-D inputs [KGS12],[KS16],[RT16]. We explore the possibility of using motion capture data in our scenario.

The goal of this thesis is to design and implement a system to understand human object affordances using motion capture data. In particular, we concentrate on graspability and placeability affordances, and model them using neural network framework in continuous space. Additionally, we also capture the uncertainty in predictions. Modeling them this way gives action potentials in product space pertaining to how the objects can be interacted with. Based on the active affordance, for all the objects in the scene, our system outputs the probability density to undertake that affordance successfully. These features can be exploited to improve human motion prediction system and human-robot collaboration system, where the robots can predict and anticipate user intent, and act accordingly.

For grasp affordance, we model it as the dynamic human state in terms of the hand position. For place affordance, it is modeled as the position on the surface where the object is most likely to be placed. Using the output distribution maps of the final state, efficient planning can be performed for motion prediction.

We structure this thesis as follows: In Chapter 2, we provide a comprehensive explanation on the concept of affordance and discuss the state-of-the-art in affordance reasoning and other related work, followed by describing relevant and necessary background information required for modeling uncertainty in deep learning framework. In Chapter 3, we describe our motion capture environment and the data processing pipeline. Once all the necessary concepts are discussed, we move on to Chapter 4, where we propose our general affordance model, followed by the techniques involved in designing the machine learning frameworks and baselines based on heuristics to encode affordances. The evaluation of our frameworks is presented in Chapter 5, where custom metrics are used to showcase the results, followed by real-time performance evaluation. Based on the evaluation results, we discuss the limitations, possible improvements and finally conclude the thesis in Chapter 6.

2 Background and Related Work

In this chapter, we provide thorough definitions and understanding regarding the concept of affordances from various fields of research and link it to the work done in robotics field. We then discuss the related work done in the field of robotics corresponding to affordance learning, followed by a comprehensive overview on the topic of deep learning, with an emphasis on the uncertainty estimation techniques.

2.1 The Concept of Affordance

Affordances refer to the action possibilities of an object in a given environment, that depends not on only the environment, but also strongly depends on the perception and motor capabilities of an agent. To illustrate this definition, consider a tennis ball which is within the reach of biological agents. In the case of human beings, this object affords graspability, as the presence of hands allows us to grasp it. The same is not true in the case of four-legged animals, where their limitation in motor capabilities in terms of absence of hands, restricts them from perceiving grasp affordance. Affordances also change with respect to the configuration of the environment and the agent. The same graspability affordance which is active for humans if it is within the reach, is no longer active if the same ball is resting 20 ft above the ground and is generally out of reach from average built humans.

The concept of affordances took shape in the field of psychology once it was proposed by Gibson in [Gib66],[Gib86], and over time numerous explanations came up in different fields such as neuroscience, cognitive science and eventually caught up in the field of robotics. Jamone et al. present a survey on affordances elegantly in [JUC+18], where they provide a multidisciplinary perspective on affordances, namely in the field of psychology, neuroscience, and robotics. We summarize the contents of the survey in the following subsections.

2.1.1 Evidence from Psychology

Ecological Psychology view of affordances relates to the detection of action possibilities by animals offered by the objects in various conditions. This field of work provides information relating to the strong correlation between the affordance detection and the body dimensions of agents. For example, the climb-ability affordance of staircase with respect to humans, i.e. the ability to climb the stairs, does not depend on the size of the step, but depends on the ratio of the step with respect to the leg-length[War84]. However, it does not answer questions related to how affordances are perceived by animals.

In Psychophysics view, the relationship between the affordance perception and action execution is studied. For a given object, the reaction times of an action differs, depending upon the configuration of objects, in terms of its position and orientation. Direct perception of affordance, ultimately resulting in faster reaction time, is possible when the object is in preferred orientation within the peripersonal space of the humans[CAT+10]. Even if the representation of objects suggest an affordance action, the actual will to act upon it might be inhibited if there is no conscious will to undertake that action.

Developmental psychology view discusses the mechanisms involved in learning affordances, in terms of how they are perceived and developed over time. The human ability to perceive affordances in an environment is not fixed, it changes over their lifetime. New motor skills pave way for new action possibilities. A simple example can be seen in infants at early stages, where once they learn to sit, their hands are freed up, with which they start manipulating the surrounding objects, eventually learning affordance related to it[SAJ10]. This body of research also suggests that exploration of such actions by infants increases the predictability and efficiency of affordance perception and eventually, learns to start predicting the effects of undertaking certain actions.

In summary, ecological psychology discusses the affordances as the perception of action possibilities of objects in an environment, which is studied in adult humans, at a fixed age in their lifetime. Developmental psychology answers an important question related to how affordances are perceived, by looking into the development stages of infants. Psychophysics view talks about how depending on the environment configuration, activation of affordances differs.

2.1.2 Evidence from Neuroscience

The claims regarding affordances made in the field of psychology are verified from the physiological standpoint within the field of neuroscience. A number of studies have tried to understand affordance perception by observing the neural representation in brain. In the cerebral cortex of primates, there are two separate pathways for visual information processing[Nor02], namely the ventral pathway and the dorsal pathway. The ventral pathway is responsible for complete object recognition and has a longer latency, while the dorsal pathway is responsible for edge detection, depth processing, surface, and axis representations and has a shorter latency. A prominent research[Nor01] suggests that the dorsal pathway is responsible for affordance perception. This is inline with the psychological view, which says that the affordance detection is fast. Additionally, this body of research also suggests that the object recognition and semantic reasoning are not required for affordance perception. This idea, that the object recognition is not necessary for affordance perception is further solidified by the works[PG97][Hum01], in which they show that people can perceive affordances related to objects, despite not being able to recognize those objects.

2.1.3 Affordances in Robotics

A number of works in robotics research were influenced by affordances. Following the concepts of affordances discussed related to psychology and neuroscience, if affordance learning has to be achieved within a robot in the true sense, developmental psychology perspective must be followed. Accordingly, the below mentioned properties are to be satisfied

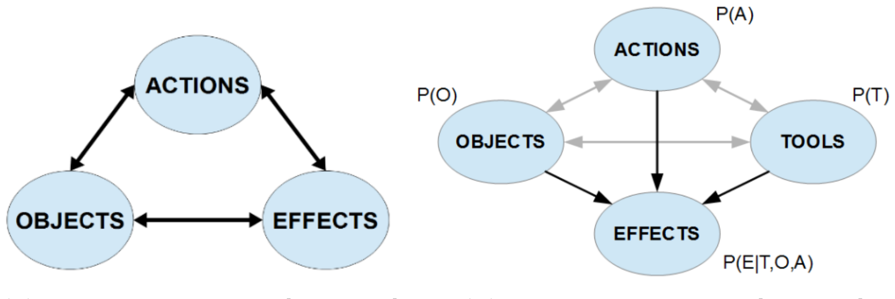
- In order to perceive affordances, robots must learn from its own sensorimotor interactions with the world
- Object representation for affordance learning should be achieved using low level object features, in order to generalize the affordance reasoning on novel objects
- Each affordance should be represented as a set, consisting of object features, action and effect, wherein taking a possible action based on the object features, results in a specific effect
- Many objects have multiple affordances associated with them and the robot should be able to perceive and choose the right affordance at a given moment, depending upon the environment, object and robot configuration

A number of papers in robotics[MLBS08][GSJB14][DJKS16] closely follow the points mentioned above. In the next section, we discuss these works while also discussing other prominent researches done in affordance modeling.

2.2 Related Work

Montesano et al. [MLBS08] model *grasping, tapping* and *touching* actions, and represent objects using a set of visual descriptors. The effects caused by performing an action, in terms of change in position, velocity and tactile information related to contact, is captured by the agent by processing the data from its sensors. Bayesian networks(BN) are used to encode the probabilistic dependencies between actions, object features and the effects caused by it in discrete space. Figure 2.1(a) shows the affordance model used. Once the BN is trained, they can predict different entities conditioned on the inputs. For instance, their model can predict the resulting effect(E), that can occur on taking an action(A) on object(O), or predict action(A) to be taken, in order to achieve effect(E) on object(O) and so on.

Goncalves et al. [GSJB14] extend the work in[MLBS08] by modeling tools that can be used to interact with objects. In particular, they are interested in multi-object affordances, where object interaction using tools are learnt as affordances. They use objects of different shapes, such as sphere, cube and cylinder. The tools used in their scenario are a stick and a rake. Using these tools and objects, they model actions such as *tap from left, tap from right, push* and *draw.* The structure of their architecture is shown in figure 2.1(b). Probabilistic dependencies are learnt between the objects, tools, actions, and effects using BN. Interesting inferences are possible, for instance, in planning, given an object(cylinder)



(a) Affordance model in[MLBS08] **(b)** Affordance model in [GSJB14]

Figure 2.1: Affordance model following developmental psychology. Images are taken from [GSJB14].

and a desired effect(bring closer), the model can infer about the possible tool to be used and the action to be performed in order to achieve it. In the best case scenario, the model returns tool as rake and action as *draw* with a high probability.

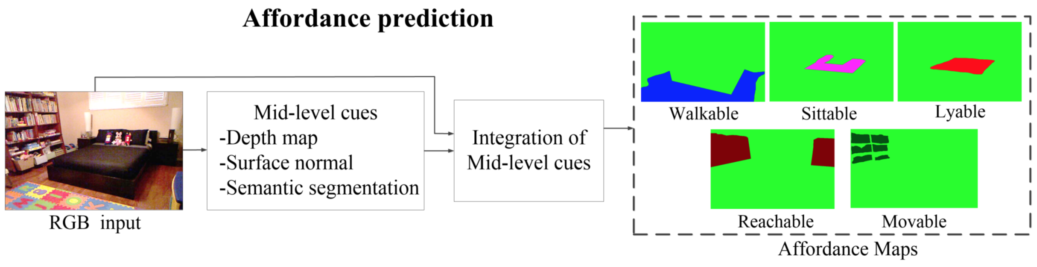
Dehban et al. in [DJKS16], which is the extension of [GSJB14], use denoising Autoencoder[VLBM08] to encode similar relationships in continuous space, rather than encoding them in discrete space. The input to denoising Autoencoder is the corrupted version of the object and tool features, action and effects, where one of these four modalities is forced to a value of 0.5 by random Bernoulli experiment during training. The output of the network is the uncorrupted version of the input. During inference, they can query a required modality by corrupting the corresponding input vector, feeding correct inputs for the other three modalities, and then, observing the output vector produced for the required modality.

Since grasp operation is one of the most common manipulation tasks in robotics, there are numerous work done specifically on grasp affordance. Detry et al. [DBP+09] work is one such example, where grasp action is learnt as an affordance and is parameterized as 6D pose of gripper, relative to the object. Objects are represented using the Early-Cognitive-Vision system, which creates short edge segments in 3D space from stereo images. Grasp affordances are modeled in a probabilistic framework as success conditional grasp densities. They use Kernel Density Estimation(KDE) method, which is a non-parametric density estimation technique, and use two kernels to represent gripper pose; one, for representing translation and the other, to represent orientation. Empirical densities are learnt by allowing the robot to perform grasp actions multiple times, and learn from only the successful grasps. Similar to this work, we model grasps as densities from human demonstration, in terms of the 3D position of the hand.

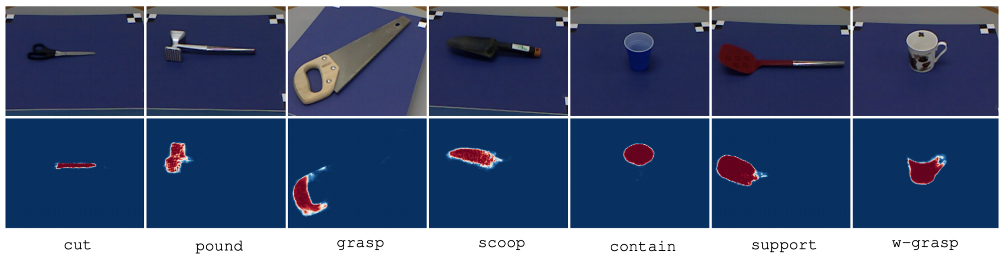
2.2.1 Visual Affordances

Visual affordance is a research field that deals with learning affordances as a computer vision problem. Hassanin et al. [HKT18] survey provides a good overview on the related work done in this field. Roy et al. in [RT16], use NYUv2 dataset, which is extended to include pixel-wise affordance ground truth. With RGB images used as inputs, a Convolution Neural Network (CNN) based architecture is designed to extract semantic segmentation relating to a number of affordances in the scene. Figure 2.1(a) shows the model architecture used for this approach and the corresponding output affordance maps obtained. Nguyen et al. in [NKCT16], model 7 affordances namely: *contain, support, cup, w-grasp, scoop, grasp* and *pound*. UMD dataset with RGBD data is used, and the network is designed using autoencoder architecture, with the final output producing k channel image of probabilities, k being the number of affordance classes. Figure 2.1(b) shows the detection results of their model.

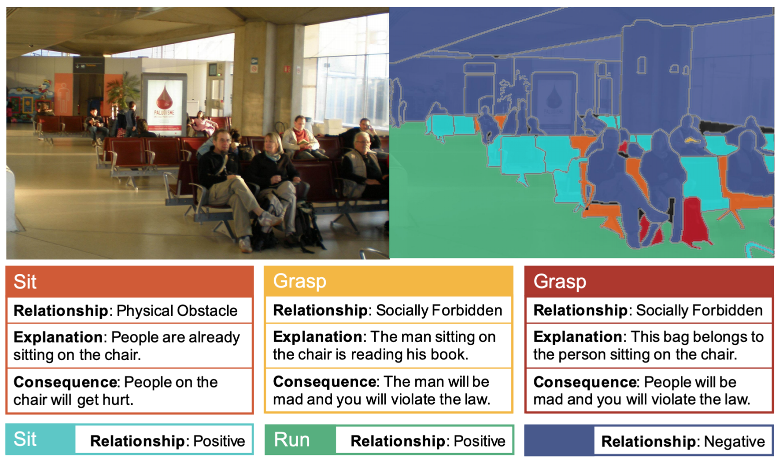
Chuang et al. in [CLTF17], use spatial gated graph neural network for affordance reasoning. They use the ADE30k dataset, and build affordance features on top of it for actions such as *run, grasp* and *sit*. For every object in the scene, the model outputs whether an affordance can be taken or not. If it cannot be taken, it provides an explanation as to why it is not possible, along with the consequence of taking that action. An example showcasing their affordance reasoning and model architecture is shown in figure 2.1(c) and figure 2.1(d) respectively. In contrast to all these works, we learn affordances without using visual inputs and instead, rely only on the analytical representation of the scene.



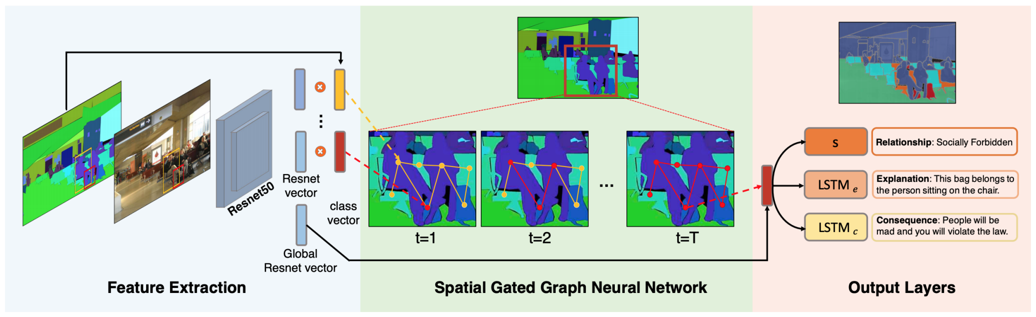
(a) Visual affordance prediction model used in [RT16]. It reasons over possible human actions in a scene.



(b) Outputs of visual affordance model used in [NKCT16].


Sit
Relationship: Physical Obstacle
Explanation: People are already sitting on the chair.
Consequence: People on the chair will get hurt.
Grasp
Relationship: Socially Forbidden
Explanation: The man sitting on the chair is reading his book.
Consequence: The man will be mad and you will violate the law.
Grasp
Relationship: Socially Forbidden
Explanation: This bag belongs to the person sitting on the chair.
Consequence: People will be mad and you will violate the law.
Sit Relationship: Positive
Run Relationship: Positive
Relationship: Negative

(c) Affordance reasoning in a given scene employed in [CLTF17]. For affordances in the scene which has negative relationship, an explanation and consequence of taking that affordance is given and learnt.


Resnet50
Resnet vector
class vector
Global Resnet vector
Feature Extraction
t=1
t=2
t=T
Spatial Gated Graph Neural Network
S
LSTM e
LSTM c
Relationship: Socially Forbidden
Explanation: This bag belongs to the person sitting on the chair.
Consequence: People will be mad and you will violate the law.
Output Layers

(d) Model architecture used in [CLTF17]

Figure 2.2: Examples of Interesting research done in visual affordance field.

2.2.2 Human Affordances

Human affordance modeling plays an important part in Human Robot Interaction (HRI). As the name suggests, the idea is to learn affordances as actions that objects afford to humans. Using such reasoning, robots can better collaborate with humans in a shared space. Popular works in this regard are [KGS12] and [KS16]. Koppula et al. in [KS16], use RGB-D video as input, track full body human pose and anticipate human activities using object affordances. Considering a particular affordance is active, they define affordances as potential functions depending upon how the object will be interacted with. The potentials are modeled as a combination of distance potential and relative angular potential, using Gaussian and von Mises distribution respectively. These distributions are learnt using maximum likelihood estimation. Depending on the affordance, the interactions considered also varies. For affordances that depend on target object such as *pourable*, the potentials are defined with respect to target object position. For affordances such as *placeable* and *openable*, that depends on the environment, potentials are defined with respect to the closest surface and head orientation. For other affordances such as *drinkable* and *reachable*, that do not depend on the environment or target location, and has to do more with the body itself, the potentials are defined with respect to the skeleton. Heatmaps are generated for each affordance by evaluating points in space with the help of the defined potential functions. Using the affordance features, they detect and anticipate human activities using a model presented by them called Augmented Temporal Conditional Random Field(ATCRF).

Our work is very closely related to [KS16] in terms of learning affordances, with an exception that we completely rely on an analytical representation of the world using motion capture setup. We use deep learning methods to implicitly learn the underlying distribution parameters representing affordances as potentials. Using deep learning to characterize uncertainty is a relatively new topic, details of which are provided in the next section.

2.3 Uncertainty in Deep Learning

Before we discuss the uncertainty modeling in deep learning, we review the concepts involved in solving regression problems using deep learning. The universal approximation theorem states that a feed forward neural network containing at least one hidden layer with nonlinear activation function, and a linear output layer(no activation), can approximate any continuous function with arbitrary accuracy.

Consider a supervised machine learning problem consisting of a dataset $D = \{x_i, y_i\}_{i=1}^N$, with x_i as input and y_i as ground truth. The true distribution underlying the data $p(x, y)$ is not known. Neural networks are used to predict the approximation $q(y|x; \theta)$ of the true distribution. This is achieved by the learning parameters of the model that minimizes the Kullback-Leibler(KL) divergence between the true distribution and the approximation, $D_{KL}(p||q)$. Under the assumption that the data generating distribution is fixed, minimizing the KL divergence is analogous to minimizing the cross entropy $H(p, q)$. Furthermore, if the true distribution $p(x, y)$ is considered to be the empirical distribution, then the cost function in terms of cross entropy reduces to equation 2.1, where θ represents the parameters of the network.

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} - \ln q(y_i|x_i; \theta) \tag{2.1}$$

In most of the regression problems, a Gaussian distribution of spherical structure is placed over the output with the neural network estimating the mean of the distribution. If $f(x_i; \theta)$ is the output of the neural network, then

$$q(y_i|x_i; \theta) \sim \mathcal{N}(f(x_i; \theta), \sigma^2 I) \tag{2.2}$$

Equation 2.2 reduces equation 2.1 to Negative Log Likelihood (NLL) of Gaussian distribution which is given by equation 2.3. Since traditional neural networks model point estimates, the variance parameter of the distribution is ignored that reduces the formulation to equation 2.4, which corresponds to the Mean Squared Error (MSE) loss.

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\sigma^2} \left\| y_i - f(x_i; \theta) \right\|^2 + \frac{1}{2} \ln \sigma^2 \tag{2.3}$$

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left\| y_i - f(x_i; \theta) \right\|^2 \tag{2.4}$$

Despite learning only point estimates from the data using MSE, deep learning models are powerful predictors and have been vastly successful in solving a variety of problems spanning a wide spectrum of fields.

2.3.1 Why Uncertainty?

If deep learning models act as powerful predictors, a question arises as to why uncertainty quantification is required. One way to think about the predictions from deep learning models is to reason them as always providing overconfident predictions. Suppose a trained deep learning model is fed with erroneous data because of a corrupted sensor, the model will still take a random guess with full confidence, which in crucial cases, might lead to drastic consequences. However, if the same is done in a Bayesian framework setting with uncertainty quantification, the resulting predictions will have high uncertainty, and the system can digress from taking the normal flow of operation.

Uncertainty information is vital for the safety aspect of robots of the future, sharing same workspace as humans. With uncertainty information, agents can qualitatively decide whether a particular action, in a given scenario can be taken or not. Considering the recent trend in the success of deep learning, research in machine intelligence for robotic field will inevitably shift towards using deep learning methods. Ultimately, when using neural network models in the decision process of an intelligent system, it is important for it to have a notion of confidence.

2.3.2 Types of Uncertainty

Uncertainty information, in general, is broadly categorized into two types: aleatoric and epistemic uncertainty [DD09].

- **Aleatoric uncertainty** specifies the inherent noise in the data. It is further classified into homoscedastic uncertainty, where the observation noise is constant for every input, and heteroscedastic uncertainty, where noise is assumed to be a function of inputs. Aleatoric uncertainty is modeled by placing a distribution over the network outputs.
- **Epistemic uncertainty** describes the uncertainty arising due to model parameters. Epistemic uncertainty generally exists due to the lack of data. With a sufficiently large dataset, this uncertainty in model parameters vanishes.

2.3.3 Popular Works in Uncertainty Modeling in Deep Learning

In recent years, there has been gaining interest in modeling uncertainty within deep learning framework. Most of the formalism takes Bayesian approach, where a prior distribution is placed over the network weights and posterior distribution to extract predictive uncertainty over the parameters, is computed using the training data. However, as exact Bayesian inference is computationally intractable for neural networks, a variety of appropriation methods have been used to approximate the posterior. We discuss two popular approaches that follow Bayesian formalism to provide good uncertainty estimates, while requiring only minor changes in the implementation side to the standard neural network modeling.

Gal et al. in [GG15],[KG17],[Gal16] proposed to use Dropouts[SHK+14] as Bayesian approximation. They model aleatoric uncertainty by including the variance in output predictions. Concretely, in case of regression, instead of minimizing the standard MSE loss, they consider output observations as sampling from a Gaussian distribution with both mean and variance parameters, and minimize the NLL given by equation 2.5. Notice that the variance parameter is a function of the input data, ergo modeling heteroscedastic aleatoric uncertainty. Using NLL instead of MSE for neural network was first formulated by Nix et al. in [NW94].

$$L_{NN}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2\sigma(x_i)^2}\left\|y_i - f(x_i)\right\|^2 + \frac{1}{2}\ln\sigma(x_i)^2 \tag{2.5}$$

To capture epistemic uncertainty, they place a prior distribution over the weights, such as a Gaussian prior $W \sim \mathcal{N}(0, I)$, and train the network. To evaluate the posterior, approximate inference technique in the form of dropout variational inference is used. To elaborate, dropout layers exist after every layer in the network, and these dropouts are kept active during test time. Multiple forward passes with the dropouts active is seen as sampling from approximate posterior, and is termed as Monte Carlo dropout(MC dropout). If $\{\hat{y}_t, \hat{\sigma}_t^2\}_{t=1}^{T}$ are a set of T samples obtained from T stochastic forward passes of the network, where $\hat{y}_t$ is the mean and $\hat{\sigma}_t^2$ is the variance obtained from t^{th} forward pass, then the total predictive variance of output y that includes both aleatoric and epistemic uncertainty is approximated using:

$$Var(y) \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T}\hat{y}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t^2 \tag{2.6}$$

Lakshminarayanan et al. [LPB16] take a similar approach to model aleatoric uncertainty by learning the variance implicitly from the data using the cost function in Equation 2.5. This technique differs in the way of capturing epistemic uncertainty by using the technique of ensembles. Randomization based approach is used to achieve ensembles. In particular, they use bootstrap aggregating strategy. However, unlike a traditional bootstrap method where individual models learn from a randomized subset of the data, they use the entire training set for each network with random shuffling and random initialization of parameters. The authors observed better performance when using the entire dataset instead of bagging. Additionally, they employ Adversarial training[GSS14] to smooth out predicted distributions.

The overall results from this work matched or outperformed the results obtained in [GG15]. If $\{\hat{y}_m, \hat{\sigma}_m^2\}_{m=1}^{M}$ are a set of M samples obtained from the output of M different networks belonging to the ensembles, where $\hat{y}_m$ is the mean output and $\hat{\sigma}_m^2$ is the variance output from the m^{th} network, then the mean and predictive variance of the ensembles are given by equation 2.7 and 2.8 respectively.

$$\mu = \frac{1}{M}\sum_{m=1}^{M} \hat{y}_m \tag{2.7}$$

$$\sigma^2 = \frac{1}{M}\sum_{m=1}^{M} (\hat{\sigma}_m^2 + \hat{y}_m^2) - \mu^2 \tag{2.8}$$

Both the approaches involve querying the model multiple times which inadvertently increases the computation time. In scenarios where real time performance is desired, capturing epistemic uncertainty is infeasible. Learning aleatoric uncertainty does not have much impact during test or inference phase. An obvious extension of aleatoric uncertainty modeling is to use complex distribution such as mixture models to learn the data and a class of neural networks exist to model them, as described in the next subsection.

2.3.4 Mixture Density Network

In specific problems pertaining to regression in continuous space, the ground truth mapping might be multi-valued and the underlying distribution could be multi-modal. In such scenarios, fitting a standard uni-modal Gaussian will not explain the data and will often give erroneous results. Mixture Density Network (MDN) are a class of neural network model that was introduced by C. M. Bishop[Bis94], which is designed to solve such issues. The conditional probability density of the target data in a mixture model is represented by equation 2.9.

$$p(t|x) = \sum_{i=1}^{m} \alpha_i(x)\phi_i(t|x) \tag{2.9}$$

where m indicates the count of the components in mixture model, $\alpha_i(x)$ are called the mixing coefficients, considered as prior probabilities(conditioned on data), for the output t being generated from i^{th} component of the mixture. $\phi_i(x)$ are functions representing conditional density of t for the i^{th} kernel. Most commonly used kernels are Gaussian, which is given by $\phi_i(x) \sim \mathcal{N}(\mu_i(x), \sigma_i(x)^2)$, where $\mu_i(x)$ and , $\sigma_i^2(x)$ are the mean and variance of i^{th} Gaussian kernel. MDN network is shown in figure 2.3 , where z is the output of MDN network. In case of using Gaussian kernel with spherical covariance, the total number of output parameters are $(c+2)\times m$ with $t \in \mathbb{R}^c$. If Gaussian kernel of diagonal covariance is used, then the total number of parameters increases to $(2c+1)\times m$.

Output parameters of MDN network should additionally satisfy the following constraints:

- Mixing coefficients must satisfy equation 2.10 in order to consider them as probabilities. This is achieved by using softmax activation for output neurons corresponding to mixing coefficients

$$\sum_{i=1}^{m} \alpha_i(x) = 1 \tag{2.10}$$

- Output neurons representing variances $\sigma_i^2(x)$ should always be positive. This is possible by using exponential activation for the corresponding neurons
- Output neurons corresponding to mean of Gaussian kernels have no activation and remain similar to the standard regression setting

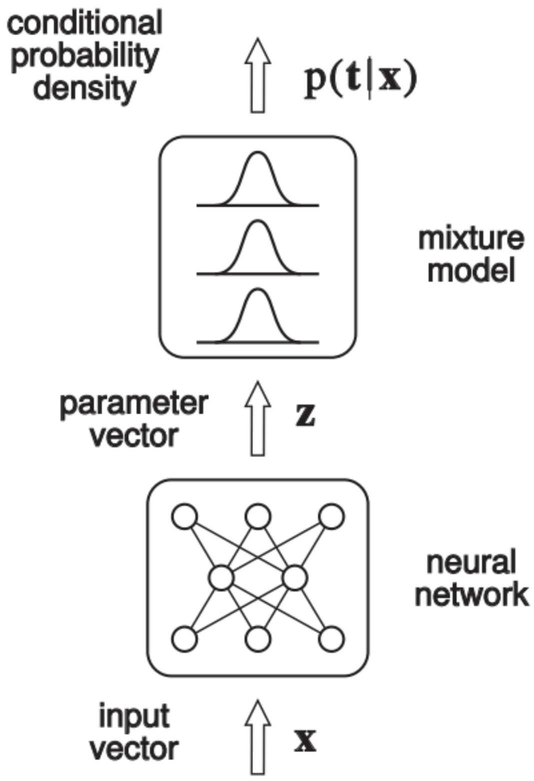


Figure 2.3: MDN network taken from [Bis94]. z denotes output parameters from the network corresponding to a Gaussian Mixture Model.

The loss function for MDN is negative log likelihood of equation 2.9. From the output parameters of MDN network, a mixture model based on equation 2.9 is constructed and the negative log likelihood is evaluated at ground truth data.

In the original paper, several optimizations are mentioned in order to efficiently calculate the gradients during back propagation. However, due to the advent in recent deep learning libraries, auto-differentiation can take care of optimizations to compute gradients efficiently. Selecting the number of kernels for MDN is completely specific to data and can be treated as hyperparameter optimization problem. As a guideline, the author mentions to consider the mixing coefficients to be at least the number of mappings possible, by observing the data. Choosing higher number of kernels will not have much impact, as the network will most likely switch off all the additional modes.

Learning aleatoric uncertainty from data is not restricted to using only Gaussian distribution. In practice, we can choose a distribution that best explains the dataset. For instance, if the ground truth of a dataset contains only unit norm vectors, instead of employing

Gaussian, we can model it using a distribution that is defined only for unit vectors. One such distribution is von Mises-Fisher (vMF), which is used to model one of our networks. A short introduction is given in the next subsection.

2.3.5 Von Mises-Fisher distribution

In the field of Directional statistics that deals with directions using unit vectors, vMF is one of the most commonly used distributions. A good overview of directional statistics is given by S. Sra in [Sra16]. vMF distribution can be considered as being analogous to multivariate Gaussian distribution on a hypersphere in $\mathbb{R}^p$. For a vector x with unit norm, von Mises-Fisher distribution is defined by equation 2.11,

$$p_{vmf}(x; \mu, \kappa) = C_p(\kappa) \exp(\kappa \mu^T x) \tag{2.11}$$

where $\mu \in \mathbb{R}^p$, $||\mu|| = 1$ is the mean direction and $\kappa \geq 0$ is the concentration parameter, which defines the spread of distribution on the surface the hypershere in the direction of μ. $C_p(\kappa)$ is the normalizing constant given by equation 2.12,

$$C_p(\kappa) = \frac{\kappa^{p/2-1}}{(2\pi)^{p/2} I_{p/2-1}(\kappa)}, \tag{2.12}$$

where I_s denotes the modified Bessel function of the first kind at order s. When $p = 3$, $C_p(\kappa)$ reduces to Equation 2.13.

$$C_p(\kappa) = \frac{\kappa}{4\pi \sinh \kappa} \tag{2.13}$$

When $\kappa = 0$, the density reduces to uniform distribution on the hypersphere and when $\kappa \to \infty$, vMF distribution reduces to point density. An example showcasing this scenario is shown in figure 2.4.

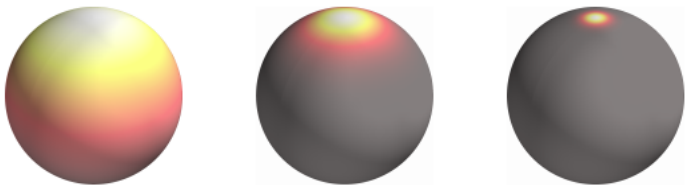


Figure 2.4: Example of vMF distribution on $\mathbb{S}^2$ taken from [Jul17]. κ is 1, 10 and 100 respectively for the three spheres.

Modeling vMF distribution using neural network

A deep learning model to learn vMF distribution should have its final layer output as the parameters of vMF distribution. If c is the dimension of ground truth data, then the number of output neurons representing vMF model to learn the data are $(c + 1)$, where the additional 1 neuron corresponds to the kappa term κ. The output layers should satisfy the following constraints:

- Output neurons corresponding to the mean direction μ should satisfy the unit norm constraint
- Output neurons representing kappa κ should always be positive. This is possible by using exponential activation for the corresponding neuron

The cost function for training this vMF model is obtained by substituting vMF distribution as the approximating distribution $q(y|x;\theta)$ in equation 2.1, i.e. NLL of vMF distribution is the loss against which the model has to be trained.

2.3.6 Summary

We discussed in this section how in a typical regression setting of deep learning, Gaussian distribution is assumed as the data generating distribution, and how upon simplification, we arrive at the MSE loss by ignoring the variance parameter. We then looked at why uncertainty information might be necessary even in deep learning field and discussed the recent advancement done in this regard. From the two types of uncertainty that can be extracted within deep learning, capturing epistemic uncertainty is computationally expensive and involves multiple sampling of a given model. Since, real-time performance is required from our system, we develop our models considering only aleatoric uncertainty.

3 Data Collection and Preprocessing

Before we propose our affordance models, we give a short introduction of the motion capture system and the environment used in our scenario. This is followed by the preprocessing techniques used to extract affordances from the collected data.

3.1 Motion Capture Setup

Motion capture system used in our setup is from Optitrack[1]. The environment has a total size of 4×4 meters. A snapshot of the setup is shown in figure 3.1. For data handling, we use Robot Operating System(ROS)[2] environment, and 3D visualization is fulfilled using RViz[3] tool .

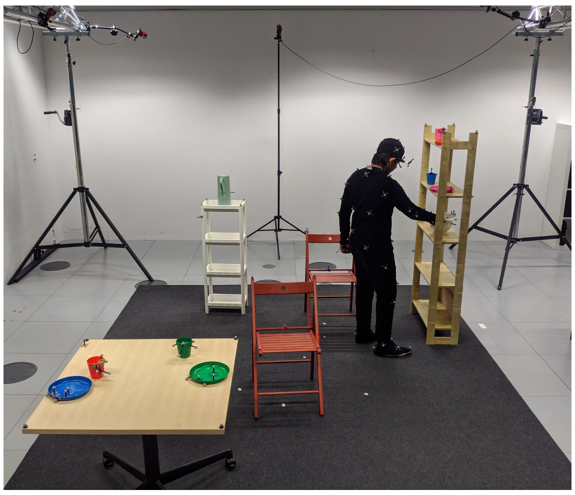


Figure 3.1: Motion capture environment with a subject wearing the tracking suit.

The human subjects were asked to wear a motion capture suit that contained 50 markers over it. There are objects in the scene that are each attached with markers for tracking and can be categorized into two types. The first type of objects are the ones that the users can directly interact with, such as cups, plates, jug and bowl. The second type of objects remain stationary in a given recording session and also acts as supporting bodies over which the first type of objects can be placed, namely a table, big shelf and small shelf. We model affordances for the first type of objects.

[1]https://optitrack.com/
[2]https://www.ros.org/
[3]https://github.com/ros-visualization/rviz

For each object, the motion capture system assigns a unique identifier and provides tracking information as pose in 7D, consisting of 3D position and 4D quaternion. For the human skeleton, a total of 21 joint information is passed, which is then stitched together as a human model and visualized in RViz tool.

The data recorded is part of the long term human motion prediction system data set. Participants were asked to perform tasks related to setting up the table and clearing them. In the collected data, the users were subject to two affordances, namely graspability and placeability. For instance, setting the table for one person involved the subject to locate relevant objects lying on shelves, grasp it, move to the table and place it.

3.2 Affordance Preprocessing

For affordance modeling, we are interested in extracting the state of the system when aforementioned actions are undertaken. The objects in the scene move only when they are displaced by subjects using grasp and place actions. Following this, a simple auto-segmentation algorithm is used to extract the grasp and place sequence instances from the data:

- For every object that users can directly interact with, change in position is checked every 100ms
- If the change in position measured over 100ms window is above a certain threshold, that instance or array index corresponds to the beginning of grasp action
- Once grasp action is detected, the person will move and eventually place the grasped object on a surface, at which point, the delta position change over 100ms window falls below the threshold and settles to 0. The corresponding time instance refers to the place action

Additionally, a couple of noise handling measures are introduced:

- Tiny movements in objects which could either be because of the motion capture noise, or unintended taps, gets classified as a valid grasp-place sequence. To eradicate this, only the grasp-place sequence that lasts longer than 500ms are considered as valid
- Due to occlusion of markers on objects, the system loses tracking momentarily, freezes the object's position in space, and continues thereafter, once tracking is restored. Such patterns get detected as multiple grasp-place actions. To overcome this problem, sequences whose grasp point is less than 300ms of previous place point, are merged as one grasp-place sequence

Once we have the precise indices corresponding to grasping and placing, we fetch relevant information required to create features for affordance reasoning. Individual models are built for each affordance and comparison is made against baseline heuristic approaches.

4 Methods

With all the necessary background information covered, and the data processing part discussed, we present in this chapter, the complete description of our affordance modeling, starting with an abstract design to encode and learn affordances, followed by individual model designs for each affordance using neural networks and heuristic approaches.

4.1 General Affordance Model

We model affordances by building a relationship between action, agent and environment by taking inspiration from subsection 2.1. However, our focus is to model human affordances, with the aim of using the learnt affordances to improve human motion prediction and robot collaboration tasks. The formulation varies in this scenario, considering we learn affordances from the ecological context, where the emphasis is on detecting and reasoning on affordances that are already encoded in adult humans. This varies from the developmental psychology viewpoint, where the focus is on decoding the learning process of perceiving affordances during a child's development.

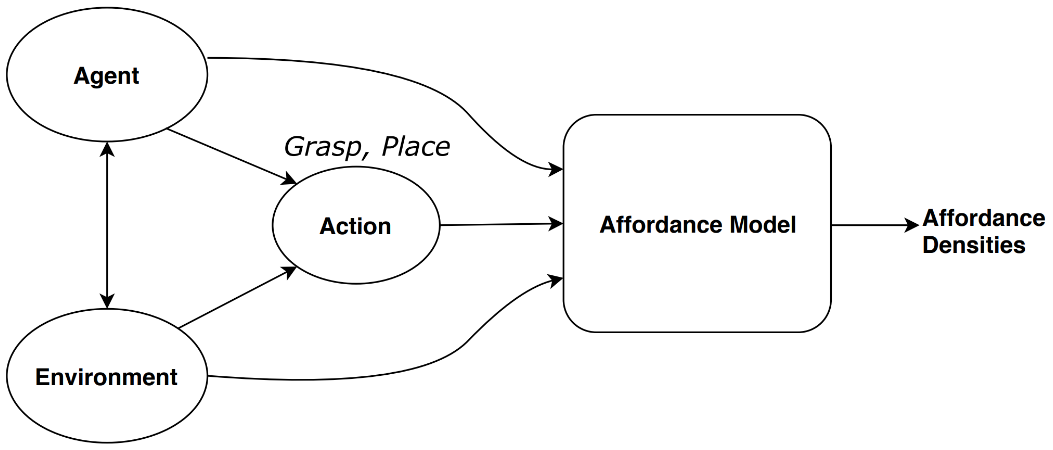


Figure 4.1: General Affordance Model Structure.

Figure 4.1 shows the block diagram of our proposed affordance model. The agent in our case is the skeleton model of human within the motion capture environment. We encode the movement of skeleton in the environment as the agent information. For representing the environment, we do not use any visual information, but instead use only the analytical representation in terms of object position, their unique types, and surface identification on which the objects rest. This encoding also informs the object for which a given affordance is active. From the skeleton data and the environment, we identify the active affordance, which in our case is either graspability or placeability. Neural networks are used to learn the latent space representation of the relationship between the agent and the environment, and using them, it learns the manifolds in 3D space representing affordance densities.

Separate networks of similar structure are used for modeling graspability and placeability affordances. This is because of the difference in the formulation of manifolds representing these two affordances. Grasp density is a property of the agent, in terms of the region around the objects where the hand should be, for initiating grasps. Place density, on the other hand, is a property of the environment in terms of where an object can be placed, which is specific to the surfaces. This density exists in a 2D manifold in 3D space.

In the following sections, we elaborate on the specifics related to modeling of these individual affordances.

4.2 Graspability Affordance

For understanding graspability affordance from a human's perspective, we model it as follows: *Given that the subject wants to grasp an object of a particular type from its current resting surface, predict the likelihood of the right wrist position for successful grasp action.* The model can then be queried for every object in the scene to get the complete dynamic mapping of the grasp affordance from a human's perspective. Figure 4.2 illustrates this, by visualizing the grasp densities of the objects in the scene obtained from one of our models.

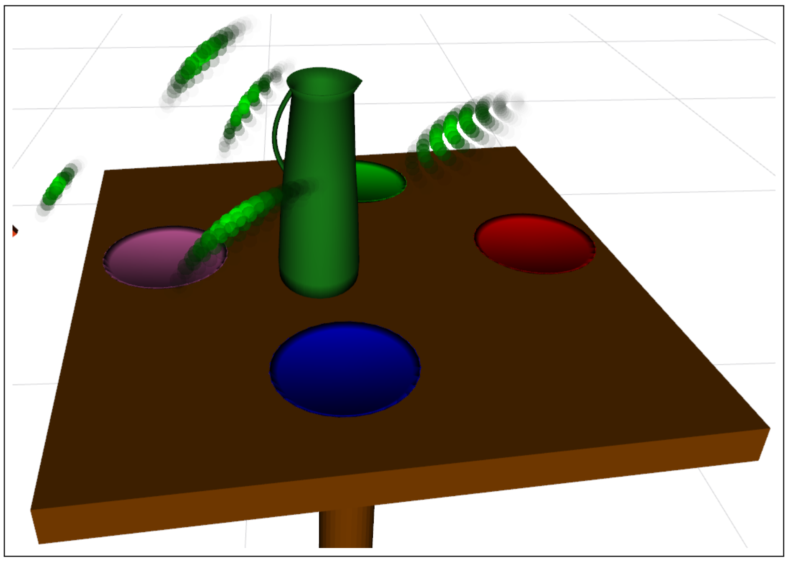


Figure 4.2: Green colored curved regions above the objects indicate grasp densities obtained from our model for all the objects on the table.

4.2.1 Features

The following features are considered for modeling graspability:

- The skeleton state relating to human motion encodes the agent information. To keep the model from not overfitting to a particular scene arrangement, we do not use world frame of reference, but instead consider object reference frame. This defines the inter-relationship between the agent and the environment. Since, the position of skeleton joints change over time, we use Long Short-Term Memory (LSTM) to capture hidden representations. Adopting full state representation of human (a total of 21 joints) using all joints was an option, but considering only the upper portion of the skeleton(total of 9 joints), was sufficient. We use the position of joints and discard the orientation information. Trajectories of these joints of window length 1 s are used, which are each spaced out 50 ms, comprising a total of 20 time steps
- The dynamics of grasping vary among different objects; for example, grasping a small cup is different from grasping a jug. Additionally, grasping action for the same object differs for different support planes. For instance, grasping a cup from the top compartment of the big shelf is distinct from grasping the same cup from the table. To distinguish the dynamics of skeleton, we encode the object type and the corresponding plane it is currently resting on, in a one-hot encoding vector. In our setting, we have 4 object types, and 10 distinct planes. The one-hot encoding vector is therefore of length 14 and represents the environment data
- We allow the model to see grasping possibilities at different time instances before the actual grasp. For the same target grasp point, we train the network with time series data ranging between 0.1 s and 1 s before grasping

Ground truth for graspability network is the property of the agent in terms of the position of right wrist in object frame of reference at grasp point. We take two approaches to model the right wrist dynamics: one, to use the position as a 3D point in Euclidean space, and the other, to model the position by defining a unit vector for direction, and scale for distance. Accordingly, all the features and ground truth are extracted from auto-segmentation described in section 3.2.

4.2.2 Model design

The structure of our base graspability model is shown in figure 4.3. It closely follows the structure of the affordance model shown in figure 4.1. The final layer of our network differs for the two approaches mentioned in previous section.

Solving the standard regression problem of minimizing mean squared loss is ruled out, since we are interested in capturing the inherent uncertainty by modeling the conditional probability distribution. Owing to the computation time, we restrict ourselves to model only aleatoric uncertainty, as capturing epistemic uncertainty with either Monte Carlo Dropouts or ensemble strategy requires multiple forward passes of the network.

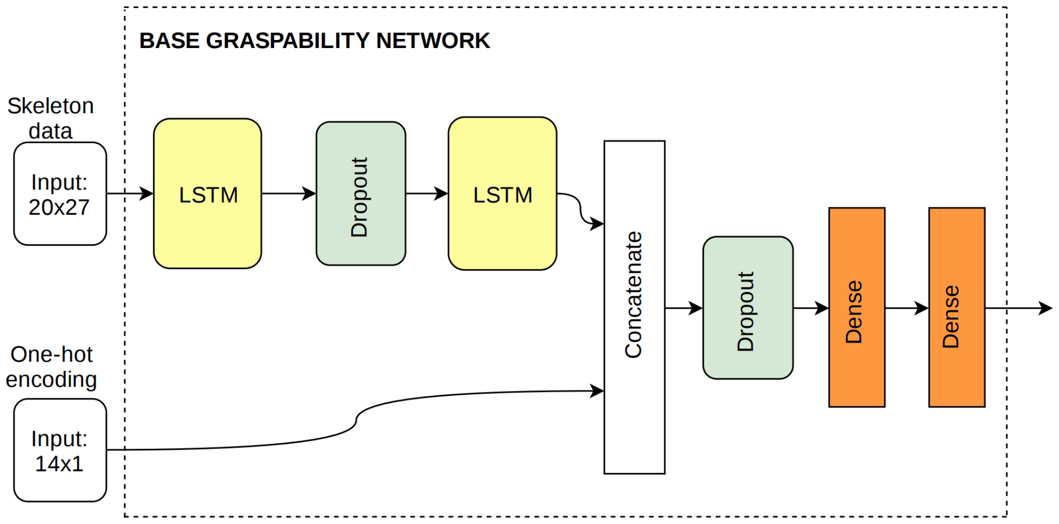


Figure 4.3: Base graspability model architecture; the final layer depends on the type of approach used.

Gaussian model

In the first network type, wrist position is modeled as a 3D position in Euclidean space and the final layer of this network outputs the parameters of Gaussian distribution having diagonal covariance structure. The cost function is NLL, given by equation 4.1, where d is 3 and covariance matrix $C(x_i)$ has a diagonal structure.

$$L_{Gaus}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{d}{2}\ln(2\pi) + \frac{1}{2}\ln(|C(x_i)|) + \frac{1}{2}(y - \mu(x_i))^T C^{-1}(x_i)(y - \mu(x_i))\right) \tag{4.1}$$

The number of output neurons are 6, owing to 3 dimensional mean and variances in each of the 3 dimensions. We then tried the multi-modal approach by using MDN. However, despite training it with different possible values of mixing modes, in every scenario the output predictions always collapsed to one distinct mode. This meant that the hidden representation was uni-modal and using a single Gaussian to represent the distribution was apt.

vMF model

The intention for vMF formulation is to model grasp points as a distribution on a 2D manifold defined on the surface of a sphere. Since unit vectors form directional data, we model it using vMF distribution defined by equation 2.11, which describes a probability distribution on a hypersphere. In this scenario, the dimension is 3. The distance part of the ground truth that describes the scale measure, to retrieve the actual grasp point, is modeled as a standard regression problem using mean squared loss function. The number of output neurons for vMF model are 4; 3 for the mean direction and 1 for the kappa term that defines the spread. Additionally, the output neurons corresponding to the mean direction

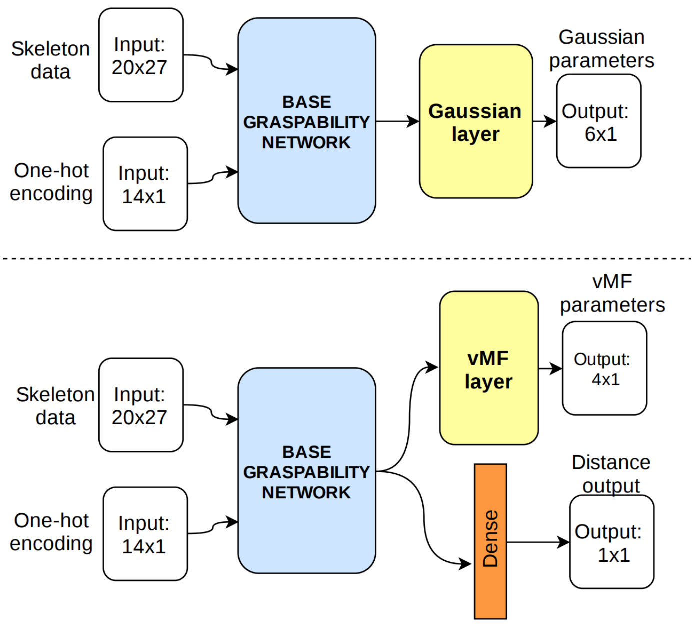


Figure 4.4: Top picture refers to graspability network with Gaussian output, while the bottom one shows vMF model.

should satisfy the unit norm constraint. The network is trained on two loss functions simultaneously, one, NLL of vMF distribution, evaluated at ground truth direction and the other, mean squared loss for distance parameter. Figure 4.4 shows the final layers and output shape of Gaussian and vMF approaches.

4.2.3 Baseline

In order to evaluate our model, we create a baseline method based on distance measure to predict graspability:

- From the training data, we take the ground truth 3D positions of the right wrist and calculate the distance to grasped objects. Since the grasp position varies for different combination of object and planes, a look up table is created for all possible combinations of object types and planes. A total of 40 table entries are possible from the data, and distances are populated in each of the respective groups, based on the object type and the plane it is currently resting on
- Mean distances for each of the groups are then calculated and stored in the lookup table
- During test time, based on the object type and the plane it is placed on, the mean distance is indexed from the lookup table. To get the actual prediction position, we calculate the unit vector along the direction of right wrist, starting from object, and compute the 3D position from this unit vector and the distance

4.3 Placeability Affordance

Placeability affordance is designed in the following manner: *Given the subject currently holds an object of certain type and wants to place it on a particular surface, predict the most likely region on the plane as a probability distribution where the object can be placed.* The model can then be queried for every surface in the scene to get the complete dynamic mapping of the place affordance from a human's perspective. Figure 4.5 illustrates this, by visualizing the place densities for placing a cup on the table and the big shelf.

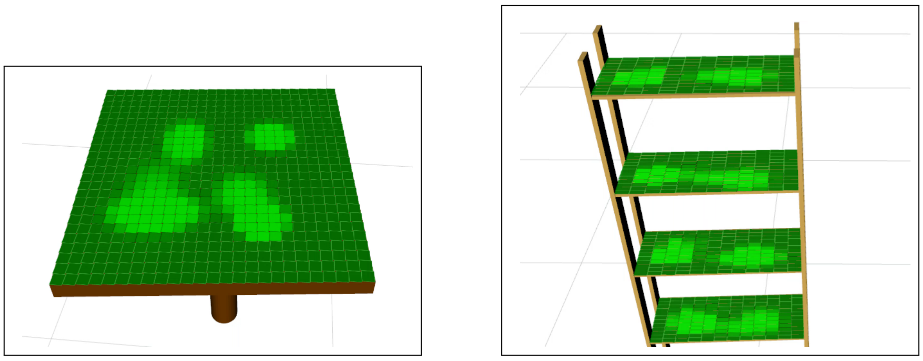


Figure 4.5: Place densities queried from our model for placing a cup on the table and the big shelf. Brighter regions correspond to higher densities.

4.3.1 Features

Placeability affordance is modeled using the features described below:

- The time series skeleton state data is the strongest feature in this case as well. The joints, sliding window and number of time steps remain the same as graspability network. However, the positions are now in plane frame of reference rather than object reference frame. We also include positions of the object in hand for which the placeability affordance is active
- The one-hot encoding input remains identical to graspability network and encodes the environment details
- We allow the model to see placing possibilities at different time instances before the actual place action. For the same target place point, we train the network with time series data ranging between 0.1 s and 3 s before placing
- Ground truth for placeability affordance is the 2D position of the object on the surface, in plane frame of reference

Plane features

Modeling placeable regions on a surface is intricate compared to modeling grasp densities. The model output depicting the place density which is defined on a 2D manifold, cannot be anywhere on the surface of the plane. The following violations are possible when considering valid placeable regions:

C1. Region outside the plane's surface

C2. Region within the objects that are already placed on the surface

Environment encoding for placeability model needs additional features to represent the plane. This is achieved by creating hand crafted raw visual features which are then fed to the part of our analytical model depicting the environment. The properties of the objects in the scene are known in prior, in terms of their geometrical shapes and physical dimensions. Using this information, we create 2D features; a 24×24 image of plane's top view by projecting the objects to 2D space, based on their shape, size and position on the plane. In the actual scene, this 24×24 grid covers $1\,\mathrm{m}^2$ area. We consider the following features:

- Binary occupancy map where the pixels correspond to value 1, if they are in the valid regions and 0, if the pixels are in invalid zones based on conditions C1 and C2
- In order to keep the network consistent to have similar data types, pixels in the valid region of binary occupancy map are converted to 2D position in the plane reference frame. Pixels in the invalid region are mapped to (0,0) value. This gives us 2 feature maps, one each in x and y dimension, and can be considered as features informing on the placeable points on the plane
- From the binary occupancy map, we also create a Signed Distance Field (SDF) map. The sign indicates if a given point lies in the valid or invalid region. For valid regions, the sign is negative and for invalid region, the sign is positive. The value is nothing but the distance of a pixel to its closest boundary

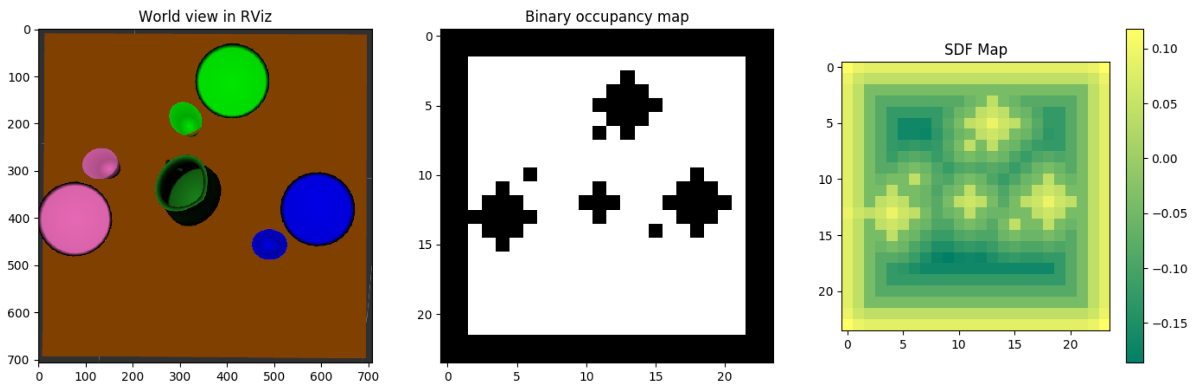


Figure 4.6: An example showing plane features for table occupancy, created from data.

Figure 4.6 shows an example of top view of the table in RViz and the corresponding generated binary occupancy map and SDF. In total, we have 4 feature maps of size 24×24 to represent the plane. CNN network is used to capture hidden environment representation from the feature maps.

4.3.2 Model design

Placeability network is shown in figure 4.7 and closely follows the model structure proposed in figure 4.1. Notice that in this case, there are two branches to capture environment dependencies, namely the one-hot encoding part and the CNN block. Like the graspability model, we capture conditional probability distribution from the placeability network and concentrate on aleatoric uncertainty. Placeability problem is multi-valued; for instance, consider the possibilities of placing a plate on the table when the subject is 2 m away, and the table is empty. Since the table in our scene can seat four people, there are four possible locations where he can place it. Mapping is multi valued for the same configuration and the network should be able to understand this. MDN best fits this scenario for modeling multi-modal distribution, and we design it by using multivariate Gaussian kernel with diagonal covariance. Choosing the number of kernels to represent the data was done based on observing the data. We find that 7 kernels best explained the data.

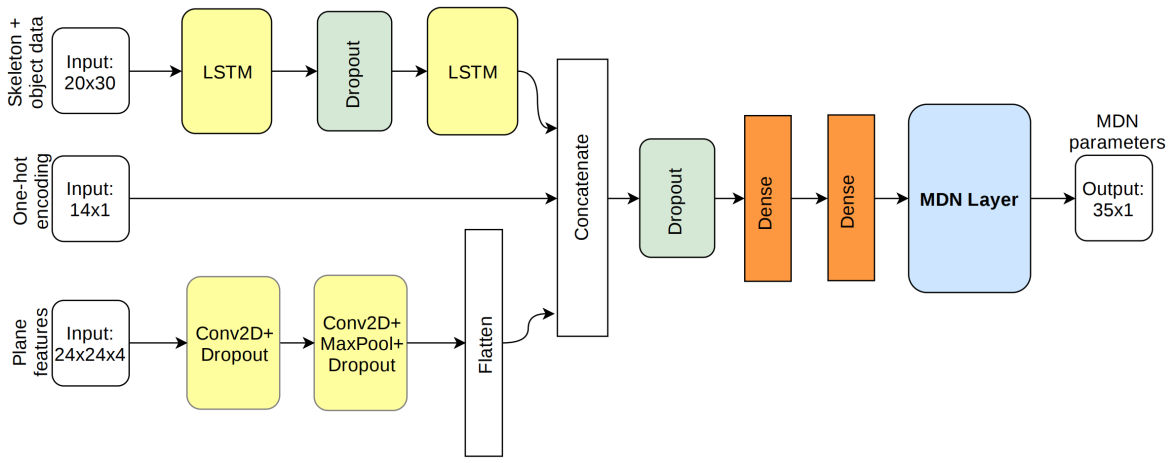


Figure 4.7: Placeability model architecture.

We further improve our placeability model with the intention of making it more robust against violating valid placement condition C2. We design two models in this regard, one with the addition of a constrain function to penalize the predictions in invalid regions and the other, using transfer learning technique wherein, environment features related to plane occupancy are learnt separately.

Constraint approach

We modify the cost function to include a penalty term. If the mean sampled from the output mixture model is in the invalid region, we penalize the network by adding a positive value to the loss, while if the mean is in the valid region, the network is incentivized with a negative value added to the loss term. This is achieved by using the corresponding SDF map for that training sample, as SDF has positive values in invalid regions and negative for valid points. From the predicted mixture model parameters, 2D mean is sampled and transformed into pixel coordinates corresponding to 2D feature maps. Value of the SDF is indexed at that transformed position and added to the loss function, after scaling it by a factor.

Transfer learning approach

In order to learn stronger environmental representations from the 2D feature maps, we train the CNN part separately with a different target and loss function, and then, use it in the main network. To achieve this, we build an autoencoder type of network with input being the 4 feature maps and one-hot encoding vector as mentioned in section 4.3.1. The output is the binary occupancy map of the plane after the object is placed on the plane. The model is trained with the standard mean squared loss. The network architecture is shown in figure 4.8.

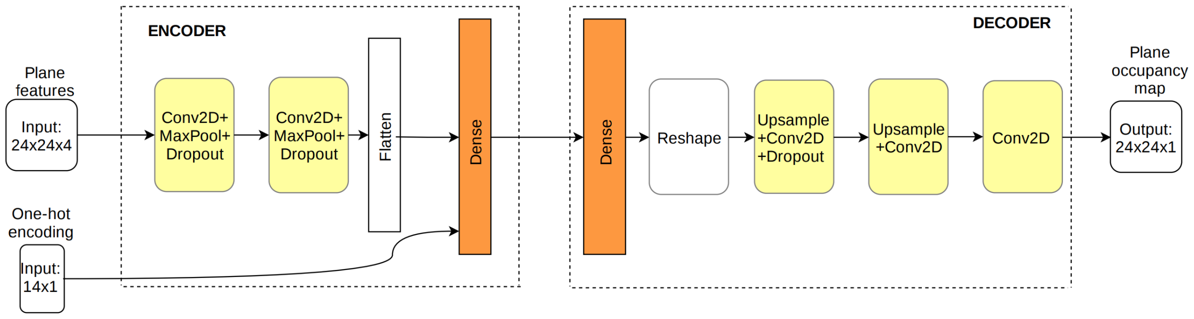


Figure 4.8: Autoencoder network architecture used to learn plane features.

The pre-trained encoder model is connected to the main placeability model by replacing the CNN part. The encoder model weights are made non-trainable when the overall placeability network is trained. The intuition here is that, with the autoencoder, we capture the latent representation that are unique for different combinations of the occupancy map. With the pre-trained encoder network producing distinctive feature representation, the main model should learn to not predict outputs in invalid regions. The model architecture of placeability network with encoder connected is shown in figure 4.9.

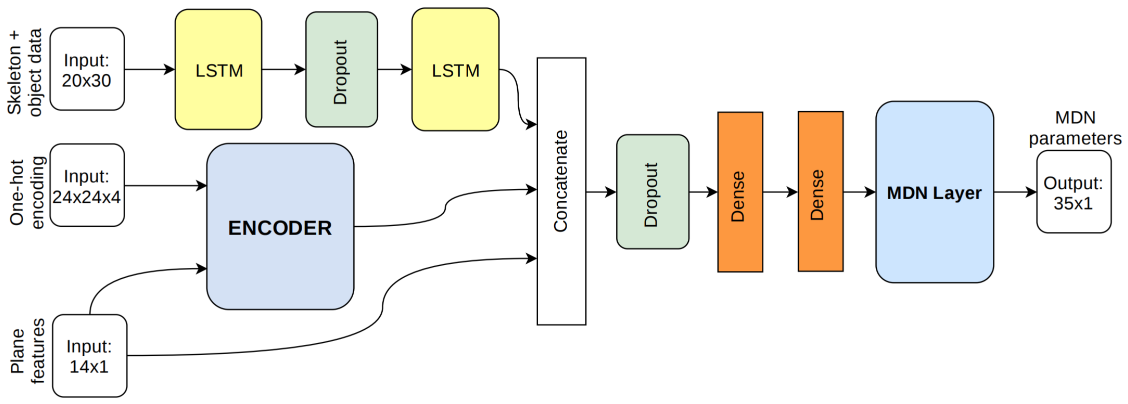


Figure 4.9: Modified placeability network with pre-trained encoder connected.

4.3.3 Baseline

A heuristic approach to solve placeability is based on distance map and SDF. The intuition is that, the subject will always select a region on the surface which is closest to him and that fits the object in hand, without obstructing the already placed objects on the plane. We use the following steps to predict the place position using heuristics:

- A 24×24 distance map is created where the entries are Euclidean distances between valid points on the plane and the skeleton position. Distance for all the invalid points are set to infinity
- Each value in SDF signifies the distance between a point and its nearest border. For valid points, the absolute value tells us how far either the plane boundary or the object boundary is. The positions on the surface of the plane where negative SDF value is greater than the object in hand, forms the candidate place points set
- From the distance map and the candidate points, the point which has the smallest distance is selected as the place position

An example showing SDF and the corresponding distance map created is shown in figure 4.10. The white parts in the distance map indicates the invalid points, which are set to infinity. Notice that the user is approaching the big shelf from top right direction as the distance is smallest in that region.

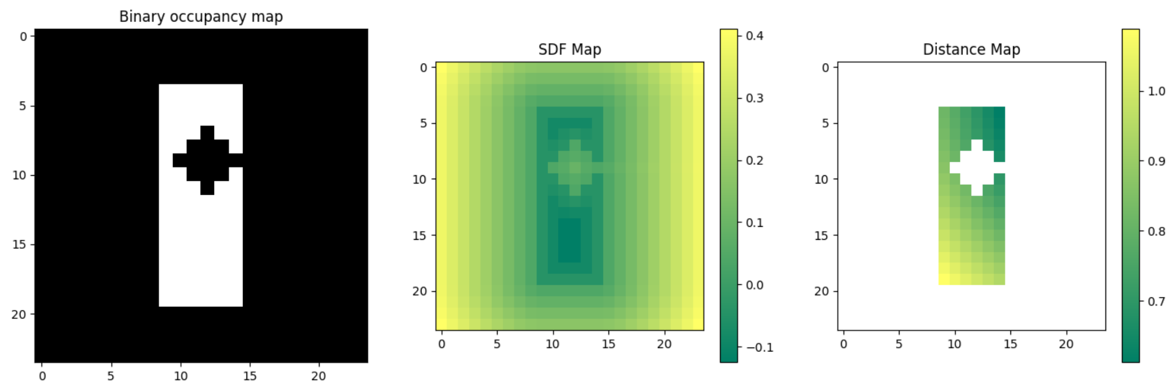


Figure 4.10: Distance map and SDF map features used in baseline for placeability. This image corresponds to a plane on the big shelf.

5 Evaluation

Following the techniques and implementation presented in the previous chapter, results obtained and the metrics used to quantify them are analyzed and presented in this chapter. The affordance sub-system integration with the complete system is discussed and the behavior in live recording is showcased.

A total of 5 users participated in recording session, with each session being approximately 25 minutes long. From the auto-segmentation algorithm for extracting grasp-place sequence, a total of 1551 samples were detected. The extracted sequences were verified to be accurate, by playing back the data in RViz and observing that the grasp-place sequences occurred at the detected indices.

For training the models, we split the data based on the subjects. From the 5 users who participated in the experiment, they were split into train and test sets, such that the ratio between the training and the test set in terms of the number of samples was approximately 7:3. Considering we use sequential data (skeleton position, body position) at multiple time steps, the number of samples for train set and test set increases by a factor of the number of time steps used.

We implemented our models using Keras[Cho+15] functional API, with TensorFlow[AAB+16] as backend. To develop the loss functions used to train our custom models, we used Tensorflow distributions[DLT+17] package.

5.1 Learning Results

Since, the cost function is NLL for all the models and as such, does not have a physical significance, standard mean squared loss is used to compare the error between the ground truth and predicted mean for evaluating network performance.

Graspability

The vMF model for graspability has two loss functions, and the network is trained with both the losses simultaneously. For vMF output parameters, the loss is NLL of vMF distribution, while for distance output parameter, the loss is MSE. In Table 5.1, the value under NLL loss for vMF model is the sum of both losses.

For calculating MSE with the Gaussian model, output parameters of the network that correspond to 3D mean are considered as prediction point, and this is used to compare against the ground truth grasp point. For the vMF model, based on the predicted mean direction and distance, we calculate the 3D position and consider it as the prediction point. Table 5.1 shows the best results obtained from the network. Results obtained from baseline is also listed, computed at 1s before grasping.

			Train set		**Test set**	
Models	**epochs**	**batch size**	**NLL**	**MSE**	**NLL**	**MSE**
Gaussian Model	200	25	−6.9783	0.0025	−4.5167	0.0043
vMF model	200	32	−1.2721	0.0042	0.2487	0.0070
Baseline	–	–	–	0.0188	–	0.0250

Table 5.1: Results obtained for Graspability models.

By observing MSE, we can see that both the neural network models are quite on par in terms of performance, while beating the baseline by a significant margin. These models however digress in the manner of uncertainty estimation. With Gaussian model, we get a spherical covariance structure indicating the confidence interval around the mean position. This interval gives the possible locations in 3D space, where the human wrist should position, in order to grasp the object. In case of the vMF model, uncertainty is defined on a 2D manifold, i.e. the surface of a sphere with its center at object centroid, and radius being the predicted distance. Output vMF parameters inform on the direction and spread of the distribution on this manifold. Since this gives a density on a surface around the object, it reflects on the possible approach angle of the wrist for successful grasping.

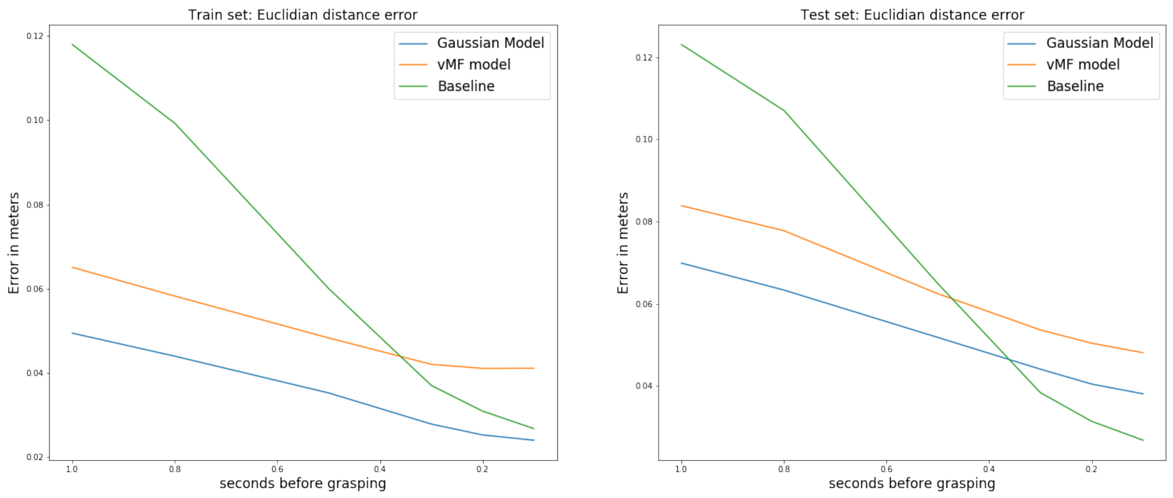


Figure 5.1: Prediction of grasp point over time for the three methods.

Figure 5.1 shows the performance of our deep learning models and baseline as time progresses towards actual grasping. In terms of predicting the mean position, along the time axis, Gaussian model always performs better than the vMF model. Baseline method lags significantly when the subject is far away, but as he approaches the object and starts doing the grasp action by moving the hand, direction of the wrist aligns to the actual grasp point, bringing down the error and eventually, performing better than the deep learning models.

Placeability

In case of placeability affordance, we show the results obtained from different variants of our MDN networks. To sample the mean from MDN output, a mixture component is first chosen by drawing j from categorical distribution with probabilities, defined by the mixing modes. Mean of j^{th} component is the mean of MDN and corresponds to the predicted place point for our calculation. Table 5.2 shows the best results obtained from the networks. Results obtained from our baseline method is also listed. The loss listed for MDN+penalty is the sum of NLL and penalty function.

			Train set		Test set	
Models	**epochs**	**batch size**	**NLL**	**MSE**	**NLL**	**MSE**
MDN+no CNN features	120	128	−3.1769	0.0184	−2.6904	0.0239
MDN+CNN features	150	128	−4.3029	0.0136	−2.6378	0.0213
MDN+CNN+penalty	150	128	−3.7081	0.0151	−1.6122	0.0245
MDN+transfer learning	120	128	−4.1168	0.0115	−2.7793	0.0194
MDN+transfer+penalty	150	128	−4.2717	0.0107	−2.4688	0.0196
Baseline	–	–	–	0.0799	–	0.2415

Table 5.2: Results obtained for Placeability models.

For MDN model with transfer learning approach, we trained the autoencoder network to extract meaningful representations. MSE loss was used for training, and the results are listed in the Table 5.3.

Model	**epochs**	**batch size**	**Train set MSE**	**Test set MSE**
Autoencoder	1000	128	0.0060	0.0127

Table 5.3: Results obtained for Autoencoder network.

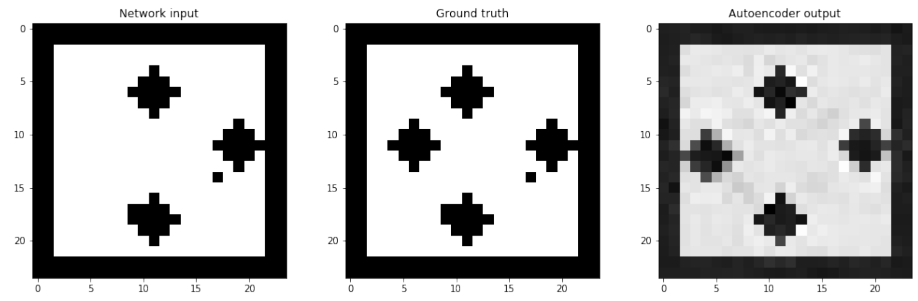


Figure 5.2: Autoencoder network prediction on a test set example.

An example of the input fed to network and the corresponding ground truth is shown in Figure 5.2. Although the resemblance of predicted binary occupancy map is not quite similar to the ground truth in terms of finer details, on a coarser level, the addition of new object on the plane at a similar position as in ground truth, is detected by the model. This means that the encoder part has learnt distinct representation for different plane occupancy configurations in the latent space.

Algorithm 1: Accuracy metric for valid placement

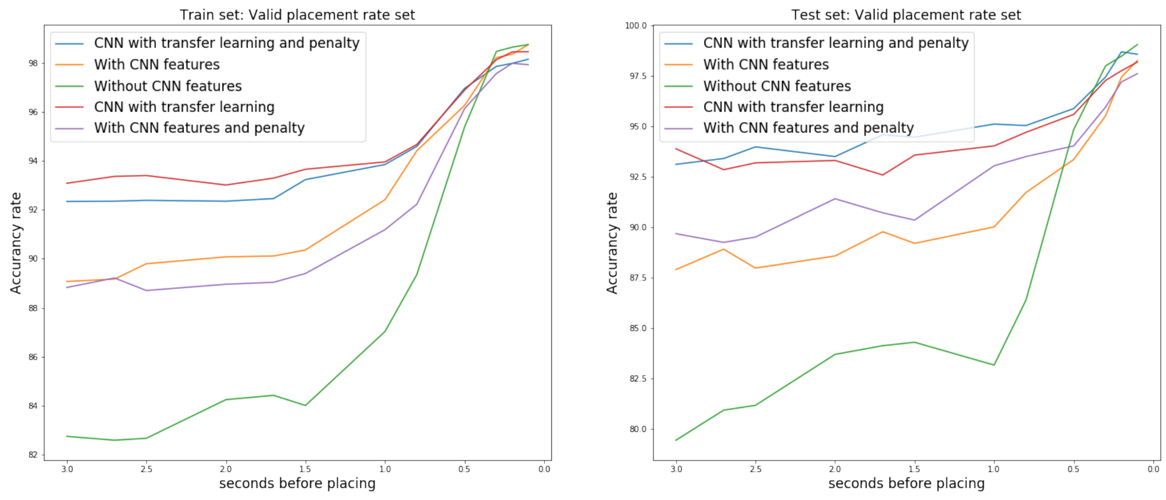
```
Data: Predicted mean array: $pred\_array$, Occupancy map array: $occup\_array$
Result: Valid placement rate: $acc$
$count = 0$;
while Arrays not empty do
    $pred\_px$ = convertToPixel($pred$);
    $contour\_plane, contour\_objs\_list$ = extractContours($occup$);
    if $pred\_px$ inside $contour\_plane \wedge pred\_px$ outside $c, \forall c \in contour\_objs\_list$ then
        $count = count + 1$;
    end
end
$acc = count/$len($pred\_array$)
```

For placeability affordance, invalid region performance cannot be concluded using mean squared error, as it does not reflect on whether the predictions are valid or not. A custom metric is created to decide if placement is in the valid region. For this purpose, during preprocessing phase, along with the normal 24×24 binary occupancy map that is used for our models, we also create a higher resolution map of 240×240 pixel density. Using this map as input to Algorithm 1, we create a measure of the percentage of predictions that satisfy the conditions of validity.

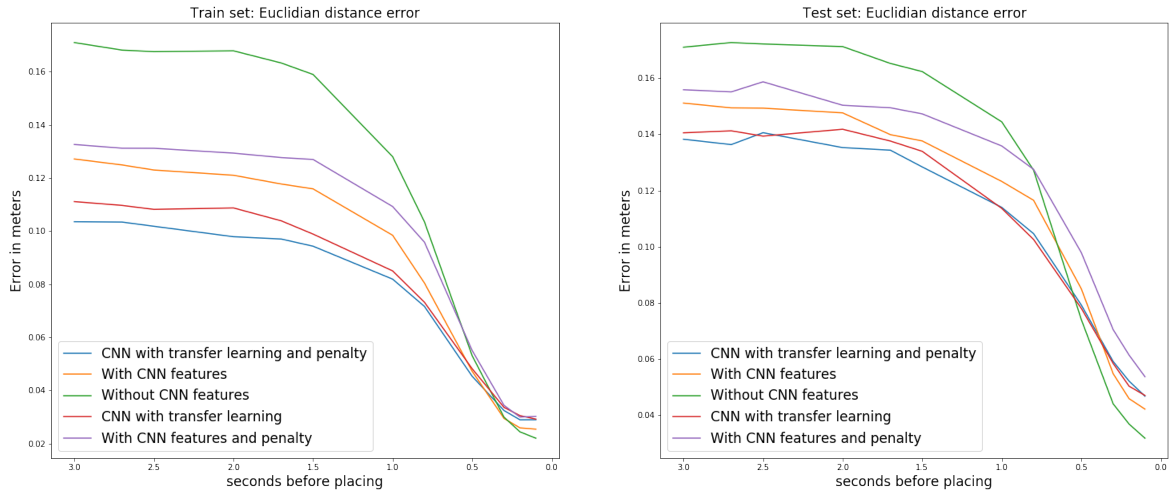
We use this metric to compare different placeability models by observing the performance over time. For the baseline approach, the accuracy from our metric is 100%, since it is purely based on logic and not learnt from data. Figure 5.3 shows the accuracy based on our metric and the distance error respectively, for models with and without CNN features, as well as considering the additional constrain approaches.

Without 2D plane features, the performance of the model is significantly lower compared to approaches with CNN network added. However, as the user approaches the plane and performs placing action to place the object, error falls lower than CNN based networks. This is because of the strong correlation between the data types of skeleton joints used for LSTM, and the target position on the plane. With CNN features added, performance definitely improves when the person is far away from the plane, but there still exists ambiguity in a number of cases, where the predictions are still in the region of invalidity. If we only think of the model in terms of learning CNN features with target being 2D position, there is no direct correlation between the input and output, as they do not exist in the same space. Consider two scenarios, where the object being placed, trajectory taken by the human, and the plane on which object will be placed, remains the same, and the only differentiating factor is the plane occupancy map, in terms of number of objects currently placed; the output from CNN part fails to learn and provide strong feature vector to differentiate these two occupancy maps.

Transfer learning approach improves the result by a margin of 5% consistently, along the time axis. This is because, our Autoencoder model inherently learnt unique latent space representation from the CNN features, since it was trained to produce binary occupancy



(a) Plots of valid placement rate.



(b) Plots of distance error.

Figure 5.3: Prediction performance of placeability over time for all the models.

map, that exists in the same space as input feature maps. When the trained encoder is connected to the main network, its only purpose is to create hidden representations, that should have helped the decoder to reconstruct the target occupancy map after placing action. Note that, when we train the main network, the encoder part is made non-trainable. For the same scenario we mentioned, pre-trained encoder part produces unique features for the two plane occupancy configurations, thereby forcing the model to predict in the free regions. We observe that including penalty function to the cost function, in order to enforce it to predict in the valid regions does not have much impact on the network predictions. It slightly improves the generalization capacity of the model as it performs better than the models without penalty on the test set.

An interesting observation in placeability, is to check the uncertainty in network predictions at different time instances before place action. This can be seen by checking the mixture components predicted by MDN network and the corresponding density. Figure 5.4 visualizes

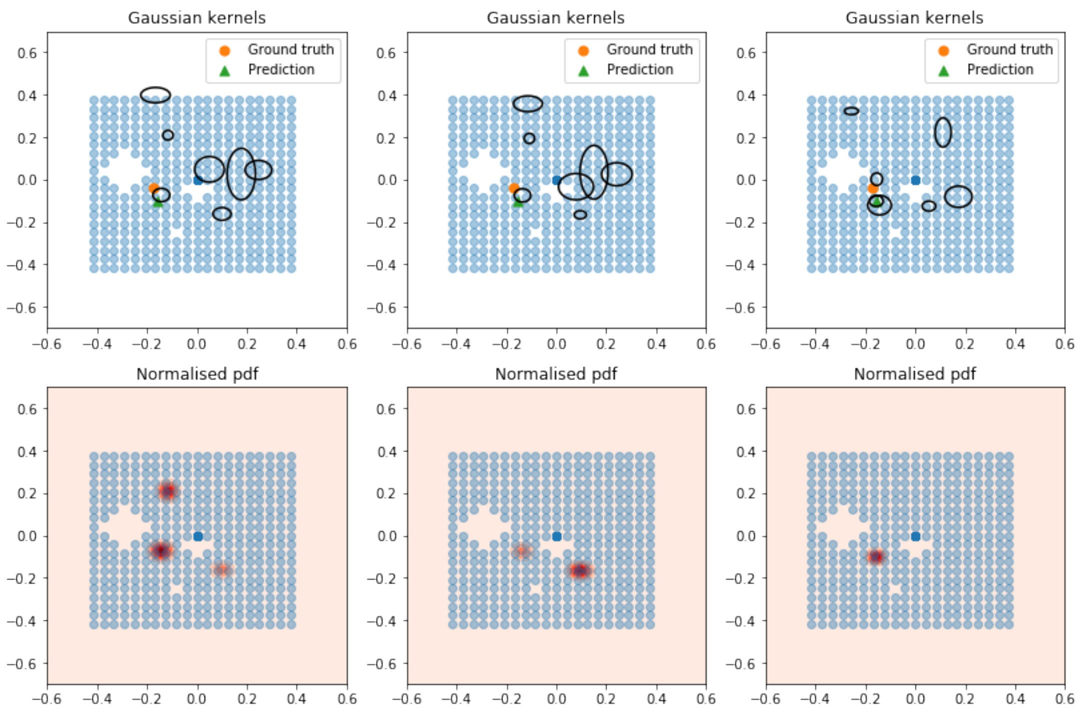


Figure 5.4: MDN prediction at 4s, 1s and 0.5s respectively. Top images show all 7 Gaussian kernels, though most of them have low probability as verified by the density images below.

the same on a test set example for placing a cup on the table at three time instances. The model has clearly understood the spatial aspect of the environment in that, it has identified the potential and most likely placing positions on the table. Also, when the subject is far away from the table, there are multiple possibilities of potential placeable regions and as the subject moves towards the table, that uncertainty reduces and confines to one dense most likely region. It is clear that this behavior would not have been possible to learn with standard uni-modal conditional density.

5.2 Sub-system Integration

A sub-system is designed in ROS that processes information and produces affordance features in real time and is integrated with the overall human motion prediction system. Active affordance in our setup is detected based on the current state of the subject's wrist; if the hand is free, it means that the subject is planning to grasp an object close to his proximity and if there is an item in hand, the subject is planning to place that object on surfaces close to his vicinity. Accordingly, neural network models are queried for objects and planes in proximity for grasp and place affordances, respectively. The system is designed based on Algorithm 2.

```
Algorithm 2: Affordance sub-system
Data: Mocap bodies position, skeleton positions at 20 Hz
Result: Affordance density functions + visualization
Initialize gridpoints_grasp, gridpoints_place;
while new mocap frame observed do
    if buffers not filled then
        FillBuffers($bodies\_buffer, skeleton\_buffer$);
    else
        UpdateBuffers($bodies\_buffer, skeleton\_buffer$);
        if CheckObjInHand() is True then
            // Active affordance is placeability
            $surfaces\_list$ = getPlanesInProximity() ;
            for plane in surface_list do
                createLSTMfeatures($plane, skeleton\_buffer$); // following section 4.3.1
                createCNNfeatures($plane, bodies\_buffer$); // following section 4.3.1
                $mdn\_params$ = predictPlaceability();
                createHeatmaps($mdn\_params$, gridpoints_place);
            end
        else
            // Active affordance is graspability
            $bodies\_list$ = getObjsInProximity() ;
            for obj in bodies_list do
                createLSTMfeatures($obj, skeleton\_buffer$) // following section 4.2.1
                $output\_params$ = predictGraspability();
                createHeatmaps($output\_params$, gridpoints_grasp);
            end
        end
    end
end
```

Using predictions from the models, we visualize the distributions by publishing them as marker topics in RViz. For graspability, we have two models, and visualizing them differs from one another. In case of Gaussian network, from the output parameters, we construct ellipsoids in RViz, with centroid at the mean of distribution and variances in each of the three directions, scaled to 95% confidence interval values, defines the shape and spread of the ellipsoid. For vMF model, the distribution is defined on the surface of a unit sphere. Accordingly, we create heatmaps based on the probability density function created from the model output parameters. A specific number of points on a unit sphere defined with the object centroid as centre are created, and these points are queried for density using the vMF distribution. The points on this sphere are then scaled to the output distance parameter from the model, in order to get the true distance of the hand to the object. The resulting heatmaps are displayed in RViz. Figure 5.5 and Figure 5.6 shows visualization of grasp densities for a test set example using vMF model and Gaussian model respectively. Notice that, for both the approaches, the uncertainty reduces as the subject approaches the object.

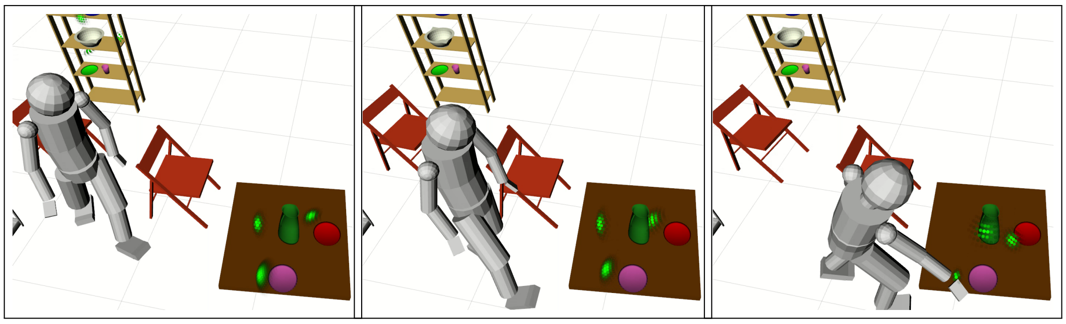


Figure 5.5: Visualization of grasp densities as heatmaps for vMF model at 3 time instances.

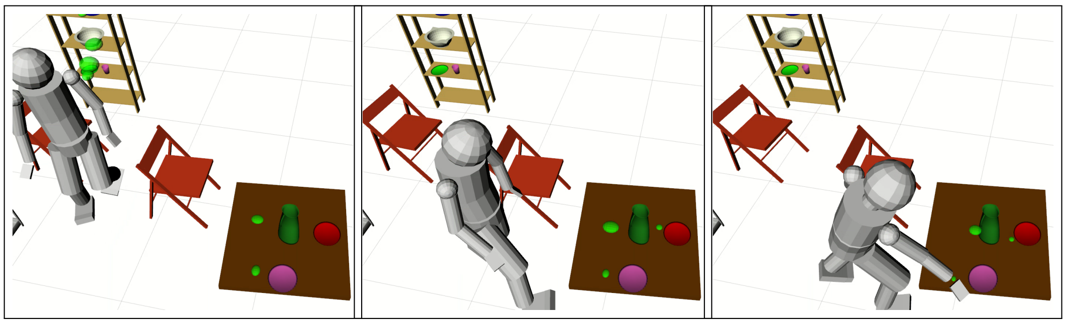


Figure 5.6: Visualization of grasp densities as spherical markers for Gaussian model at 3 time instances.

For placeability visualization, from the output MDN parameters, a mixture model is created. A 2D grid with sufficient number of points covering the surface of the plane, where the network was queried is created. The points are then queried for probability density using the mixture model. The end result is visualized in RViz as markers. We showcase the behavior of our best placeability model under different scenarios. Figure 5.7 shows an example of a subject placing a blue plate on the table and figure 5.8 shows an example of a subject placing a pink cup on the table. In both these scenarios, we can see the multiple possibilities(4) predicted by the network in terms of place densities when the person is far away from the table. As he approaches the table, this uncertainty reduces and concentrates to one dense region, where the object is being placed.

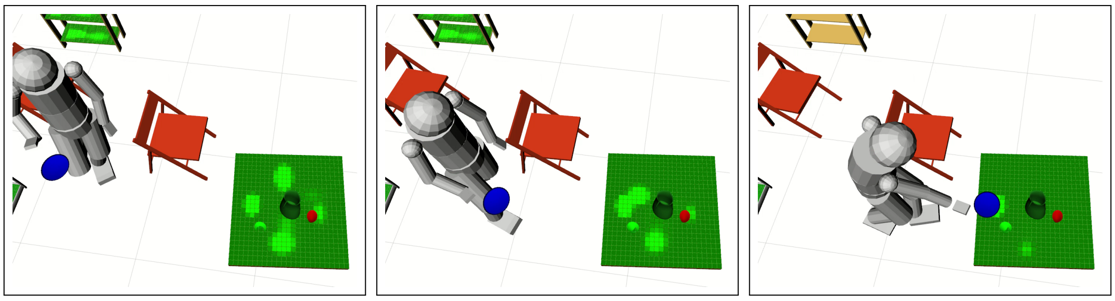


Figure 5.7: Visualization of place densities as heatmaps at 3 time instances: plate on table.

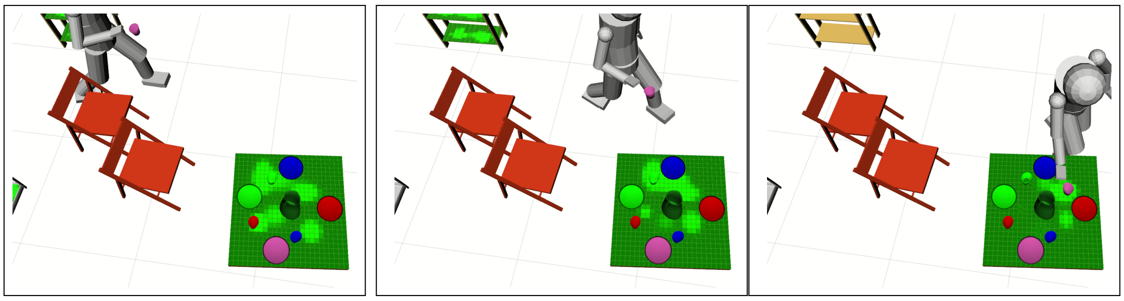


Figure 5.8: Visualization of place densities as heatmaps at 3 time instances: cup on table.

Figure 5.9 shows an example of a subject placing a jug on the table. In this case, we can see a single dense region at the center of the table even when the subject is far away. This is because, in our experiments related to setting the table for a number of persons, the jug was almost always kept at the center of the table. With the help of environment features, our affordance model has clearly picked up the distinction that when it comes to jug and bowls, humans always tend to place it at the center of the table. Figure 5.10 shows an example of a subject placing a pink plate on the big shelf. We can see a different density pattern predicted on the big shelf than that on the table, which is also because of the environment features.

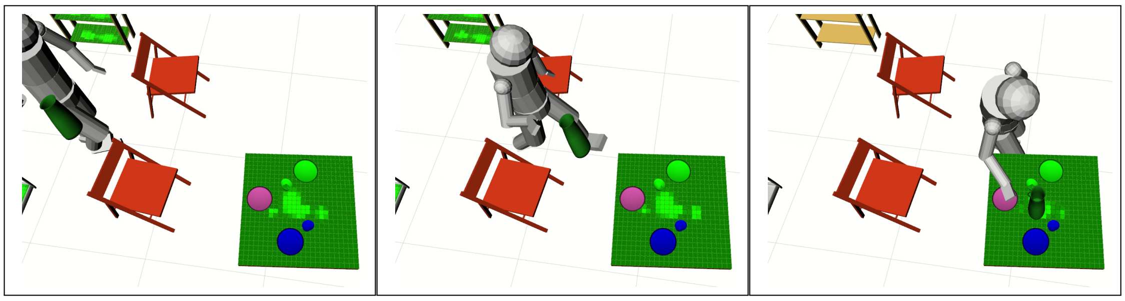


Figure 5.9: Visualization of place densities as heatmaps at 3 time instances: jug on table.

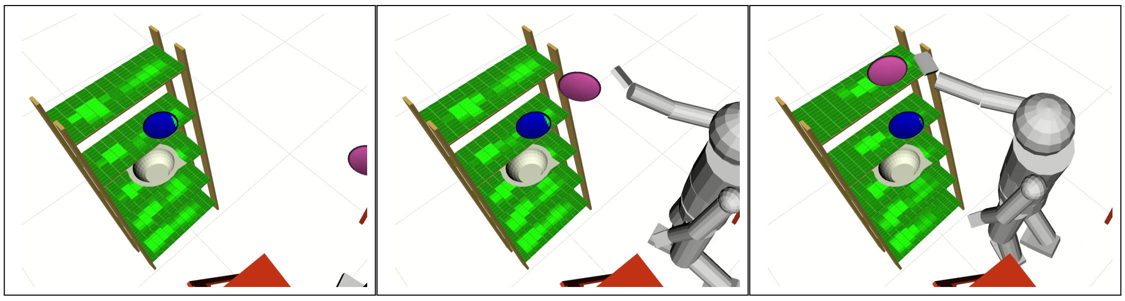


Figure 5.10: Visualization of place densities as heatmaps at 3 time instances: plate on big shelf.

6 Discussion

We presented a system to learn human object affordances using motion capture data. A user study was conducted with human subjects to collect data for this purpose in which, the actors were subjected to two affordances, namely graspability and placeability. The intention of the thesis was to use this data and encode the dynamic action possibilities in the environment, by modeling affordances as density functions.

We modeled the two affordances as conditional probability distributions using deep learning methods by capturing the implicit uncertainty. For grasp affordance with vMF model, the uncertainty encodes the possible approach angles of the human hand for successful grasp action. For place affordance with MDN model, the uncertainty is encoded as possible regions on the surface where the object can be placed. Within our experimental framework, the results achieved were quite good, with the error in predictions in terms of mean of the predicted distribution was minimal, while the ground truth was always enclosed in the uncertainty regions. The affordance prediction models were then packaged as a sub-subsystem, with the purpose being, to produce action possibilities from human's point of view, in real-time, for efficient modeling of human motion prediction.

6.1 Alternatives and Limitations

An alternative approach to model the affordances would have been to use point cloud data sampled from mesh models of objects in the simulated world. For grasp affordance, the potential would be defined on the surface of objects, rather than density around the position of the wrist for grasping. This could have helped to achieve better semantics, in case the models are used directly for robots with the intention of performing imitation tasks. With the current semantics, for graspability, additional learning has to be dealt with to achieve the same.

A limitation of our models is with respect to scalability in terms of objects in the scene. While adding new objects of existing type does not pose a problem, adding a different object would require the model to be re-trained. However, the implementation is robust, in the sense, there would only be changes in configuration files to add new objects, and the rest, including pre-processing and real-time integration, remains the same. For graspability network, our data contained grasps using only the right hand. Modeling both hands would require additional input to distinguish between left or the right hand.

6.2 Future Work

As mentioned in the previous sections, a number of improvements can be made to generalize the models to handle more cases. A natural extension of this work is to integrate it with the long-term human motion prediction system, where the human motion is broken down into hierarchies, consisting of multiple low-level and mid-level tasks. These tasks have to be performed in a defined order or parallelly to achieve a high-level task. The dense potentials of action possibilities in the environment obtained from our affordance model can act as mid-level features for high-level task planning. Another field of research where this work looks promising, is in human robot collaboration. The affordance densities can also act as features informing on the possible human intentions. This in turn, can be integrated into a robot's intelligence framework to anticipate and collaborate safely with humans in the same shared environment.

Bibliography

[AAB+16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.* 2016. arXiv: 1603.04467 [cs.DC] (cit. on p. 45).

[Bis94] C. M. Bishop. "Mixture density networks". Technical Report. Birmingham, 1994. URL: http://publications.aston.ac.uk/id/eprint/373/ (cit. on pp. 27, 28).

[CAT+10] M. Costantini, E. Ambrosini, G. Tieri, C. Sinigaglia, G. Committeri. "Where Does an Object Trigger an Action? An Investigation About Affordance in Space". In: *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale* 207 (Oct. 2010), pp. 95–103. DOI: 10.1007/s00221-010-2435-8 (cit. on p. 18).

[Cho+15] F. Chollet et al. *Keras*. https://keras.io. 2015 (cit. on p. 45).

[CLTF17] C.-Y. Chuang, J. Li, A. Torralba, S. Fidler. *Learning to Act Properly: Predicting and Explaining Affordances from Images.* 2017. arXiv: 1712.07576 [cs.CV] (cit. on pp. 21, 22).

[DBP+09] R. Detry, E. Baseski, M. Popovic, Y. Touati, N. Kruger, O. Kroemer, J. Peters, J. Piater. "Learning object-specific grasp affordance densities". In: *2009 IEEE 8th International Conference on Development and Learning.* June 2009, pp. 1–7. DOI: 10.1109/DEVLRN.2009.5175520 (cit. on p. 20).

[DD09] A. Der Kiureghian, O. Ditlevsen. "Aleatory or Epistemic? Does It Matter?" In: *Structural Safety* 31 (Mar. 2009), pp. 105–112. DOI: 10.1016/j.strusafe.2008.06.020 (cit. on p. 25).

[DJKS16] A. Dehban, L. Jamone, A. Kampff, J. Santos-Victor. "Denoising auto-encoders for learning of objects and tools affordances in continuous space". In: May 2016, pp. 4866–4871. DOI: 10.1109/ICRA.2016.7487691 (cit. on pp. 19, 20).

[DLT+17] J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, R. A. Saurous. *TensorFlow Distributions.* 2017. arXiv: 1711.10604 [cs.LG] (cit. on p. 45).

[Gal16] Y. Gal. "Uncertainty in Deep Learning". In: 2016 (cit. on p. 26).

[GG15] Y. Gal, Z. Ghahramani. *Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning.* 2015. arXiv: 1506.02142 [stat.ML] (cit. on p. 26).

[Gib66] J. Gibson. *The senses considered as perceptual systems.* Houghton Mifflin, 1966. URL: https://books.google.de/books?id=J9ROAAAAMAAJ (cit. on pp. 15, 17).

[Gib86] J. Gibson. *The Ecological Approach to Visual Perception.* Resources for ecological psychology. Lawrence Erlbaum Associates, 1986. ISBN: 9780898599596. URL: https://books.google.de/books?id=DrhCCWmJpWUC (cit. on p. 17).

[GSJB14] A. Gonçalves, G. Saponaro, L. Jamone, A. Bernardino. "Learning visual affordances of objects and tools through autonomous robot exploration". In: *2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC).* May 2014, pp. 128–133. DOI: 10.1109/ICARSC.2014.6849774 (cit. on pp. 19, 20).

[GSS14] I. J. Goodfellow, J. Shlens, C. Szegedy. *Explaining and Harnessing Adversarial Examples.* 2014. arXiv: 1412.6572 [stat.ML] (cit. on p. 26).

[HKT18] M. Hassanin, S. Khan, M. Tahtali. *Visual Affordance and Function Understanding: A Survey.* 2018. arXiv: 1807.06775 [cs.CV] (cit. on p. 21).

[Hum01] G. Humphreys. "Objects, affordances ... action!" In: *The Psychologist* 14 (Aug. 2001), pp. 408–412 (cit. on p. 18).

[JUC+18] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, J. Santos-Victor. "Affordances in Psychology, Neuroscience, and Robotics: A Survey". In: *IEEE Transactions on Cognitive and Developmental Systems* 10.1 (Mar. 2018), pp. 4–25. DOI: 10.1109/TCDS.2016.2594134 (cit. on p. 17).

[Jul17] Julian Straub. *Bayesian Inference with the von-Mises-Fisher Distribution in 3D.* 2017. URL: http://people.csail.mit.edu/jstraub/download/straub2017vonMisesFisherInference.pdf (cit. on p. 29).

[KG17] A. Kendall, Y. Gal. *What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?* 2017. arXiv: 1703.04977 [cs.CV] (cit. on p. 26).

[KGS12] H. S. Koppula, R. Gupta, A. Saxena. *Learning Human Activities and Object Affordances from RGB-D Videos.* 2012. arXiv: 1210.1207 [cs.RO] (cit. on pp. 15, 23).

[KS16] H. S. Koppula, A. Saxena. "Anticipating Human Activities Using Object Affordances for Reactive Robotic Response". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.1 (Jan. 2016), pp. 14–29. DOI: 10.1109/TPAMI.2015.2430335 (cit. on pp. 15, 23).

[LPB16] B. Lakshminarayanan, A. Pritzel, C. Blundell. *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles.* 2016. arXiv: 1612.01474 [stat.ML] (cit. on p. 26).

[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 (Feb. 2008), pp. 15–26. DOI: 10.1109/TRO.2007.914848 (cit. on pp. 19, 20).

[NKCT16] A. Nguyen, D. Kanoulas, D. G. Caldwell, N. G. Tsagarakis. “Detecting object affordances with Convolutional Neural Networks”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2016, pp. 2765–2770. DOI: 10.1109/IROS.2016.7759429 (cit. on pp. 21, 22).

[Nor01] J. Norman. “Ecological Psychology and the Two Visual Systems: Not to Worry!” In: *Ecological Psychology - ECOL PSYCHOL* 13 (Apr. 2001), pp. 135–145. DOI: 10.1207/S15326969ECO1302_6 (cit. on p. 18).

[Nor02] J. Norman. “Two visual systems and two theories of perception: An attempt to reconcile the constructivist and ecological approaches”. In: *The Behavioral and brain sciences* 25 (Mar. 2002), 73–96, discussion 96. DOI: 10.1017/S0140525X0200002X (cit. on p. 18).

[NW94] D. A. Nix, A. S. Weigend. “Estimating the mean and variance of the target probability distribution”. In: *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN’94)*. Vol. 1. June 1994, 55–60 vol.1. DOI: 10.1109/ICNN.1994.374138 (cit. on p. 26).

[PG97] A. Patla, M. Goodale. “Obstacle avoidance during locomotion is unaffected in a patient with visual form agnosia”. In: *Neuroreport* 8 (Jan. 1997), pp. 165–8. DOI: 10.1097/00001756-199612200-00033 (cit. on p. 18).

[RT16] A. Roy, S. Todorovic. “A Multi-scale CNN for Affordance Segmentation in RGB Images”. In: vol. 9908. Oct. 2016, pp. 186–201. ISBN: 978-3-319-46492-3. DOI: 10.1007/978-3-319-46493-0_12 (cit. on pp. 15, 21, 22).

[SAJ10] K. C. Soska, K. E. Adolph, S. P. Johnson. “Systems in development: motor skill acquisition facilitates three-dimensional object completion.” In: *Developmental psychology* 46 1 (2010), pp. 129–38 (cit. on p. 18).

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html (cit. on p. 26).

[Sra16] S. Sra. *Directional Statistics in Machine Learning: a Brief Review*. 2016. arXiv: 1605.00316 [stat.ML] (cit. on p. 29).

[VLBM08] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol. “Extracting and Composing Robust Features with Denoising Autoencoders”. In: *Proceedings of the 25th International Conference on Machine Learning*. ICML ’08. Helsinki, Finland: ACM, 2008, pp. 1096–1103. ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390294. URL: http://doi.acm.org/10.1145/1390156.1390294 (cit. on p. 20).

[War84] W. Warren. “Perceiving Affordances: Visual Guidance of Stair Climbing”. In: *Journal of experimental psychology. Human perception and performance* 10 (Nov. 1984), pp. 683–703. DOI: 10.1037/0096-1523.10.5.683 (cit. on p. 17).

All links were last followed on October 23 2019.

Declaration