Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Gaze Based Intent Prediction for Human Robot Collaboration Using Neural Networks

Simon Tobias Bihlmaier

Course of Study:	Informatik
Examiner:	Prof. Dr. rer. nat. Marc Touissant
Supervisor:	Philipp Kratzer, M.Sc., Dr. Jim Mainprice
Commenced:	April 25, 2019
Completed:	October 25, 2019

Abstract

Predicting the intentions of humans is very useful for human robot collaboration, since it can enable robots to proactively adapt to future activities of the human. This thesis compares intent prediction performance of different machine learning models based on neural networks that use different ways to encode input data. A dataset with a human picking up objects and placing objects was recorded for this work. It includes three dimensional positional data from motion capture and eye gaze data. Four different representations of the eye gaze, together with object and human skeleton positional data were evaluated and compared by their classification performance for intent prediction in a pick and place scenario. The resulting system can predict which object the human intends to pick up next and where to place it afterwards. This thesis shows the importance of including eye gaze in pick and place intent prediction.

Kurzfassung

Die Vorhersage der Absicht eines Menschen ist sehr nützlich für Mensch-Roboter-Kollaborationen, da sie es dem Roboter ermöglicht sich proaktiv an zukünftige Aktionen der Menschen anzupassen. Diese Thesis vergleicht die Qualität von Absichtsprognosen verschiedener Modelle des Maschinellen Lernens die auf neuronalen Netzwerken basieren. Diese erhielten dabei auf unterschiedliche Weisen kodierte Eingangsdaten. Für diese Arbeit wurde ein Datensatz erzeugt, bei dem Menschen Objekte aufheben und ablegen. Darin sind dreidimensionale Positionsdaten von einem Motion Capture System und Blickrichtungsdaten enthalten. Es wurden vier verschiedene Repräsentationen der Blickrichtung zusammen mit Positionsdaten der Objekte und des menschlichen Körpers ausgewertet und über ihre Klassifikationsqualität bei der Vorhersage in einem Szenario, in dem Menschen Objekte aufheben und ablegen verglichen. Das resultierende System kann prognostizieren, welches Objekt der Mensch als nächstes aufheben und wo er es danach ablegen will. Diese Thesis zeigt, wie wichtig es ist, die Blickrichtung bei der Vorhersage in einem solchen Szenario miteinzubeziehen.

Contents

List of Figures

List of Tables

1 Introduction

Today, there is a rapidly rising number of robots that are working in shared environments with humans. This leads to an increased need for robots to be able to understand, which goals the humans are pursuing. Intent prediction allows robots to preemptively adapt their tasks and actions around those of the humans. Not only does this increase safety, but could also increase acceptance of the robots by the humans.
To improve intent prediction performance of robots, one recent trend is to take eye gaze direction into account. As humans plan their next goals and actions, they tend to look at objects and places, which are involved in their current action. There already exist tetherless goggles which can track the head, eyes and hands and have wireless network capabilities. This would make an intent prediction solution, using only eye gaze much more practical in day-to-day life.
Previous works on intent and activity anticipation have focused on using support vector machines as their learning method of choice. With the rise of neural networks in recent times, there is a movement towards using them for these tasks.
The goal of this thesis is to explore what set of features combined with which neural network structure works best for intent prediction of a human. The intent is predicted in a pick and place scenario, with the intent being which object the human will pick up or where the human wants to place a held object.
For this purpose a dataset was recorded, that has a human move through a scene with a table and shelves, movable objects and obstacles. The human picks up and places down objects. The data contains the positions and orientations of the joints of the human and the objects. It also contains the direction of the eye gaze of the human.
Different neural network models were evaluated and videos were created, visualizing the predicted intent of the human.
The results show that the gaze is a strong indicator for pick and place intent prediction, improving model performance for both picking and placing and providing good results as a baseline. Models for placing intent prediction were able to outperform a baseline predicting the place, that the human is looking at.

This thesis starts with Chapter 2, containing related work concerning intent prediction, using eye gaze in human robot interaction and neural networks. Chapter 3 describes the setup that was used for data capturing and how the data was captured. Following Chapter 4 explains the contents of the dataset, as well as the processing steps to get the training data from the dataset. The way that intent prediction is done in this thesis, including the network structure, is the content of Chapter 5. In Chapter 6 the performance of the neural network models is evaluated for different inputs. In the last chapter, the results of the thesis are discussed and suggestions are made for possible future improvements and applications of intent prediction. The appendix contains details of the additions to the existing motion capture and eye gaze tracking setup.

2 Background

Intent prediction plays an important role in human robot interaction. In a human-robot team, where the human and the robot collaborate, in most cases the human assigns a goal for the team. For the robot to be able to successfully collaborate with the human and assist in achieving the goal of the team, the robot needs to know the intent of the human and therefore has to estimate it [BWB08]. Figure 2.1 shows the place that intent estimation takes in an architecture for collaboration of a cognitive robot. Sensors are used to observe the environment, perception and learning convert the sensor data to knowledge about the state of the environment and the actors. This knowledge can then be used to do intent prediction for each of the actors and form a joint intention from the individual intentions. In action planning, a sequence of actions is determined to fulfill the joint intent. The actions are then executed by the actors.

This chapter presents previous works in intent prediction and activity anticipation, as well as works on eye gaze behavior of humans. The second part of this chapter gives some background on the neural network structures and learning methods that were used.

2.1 Related Work

The works of Koppula et al. and Hoffmann et al. center around activity anticipation [HKTM18; KGS13; KS16]. They are similar in that they generate trajectories of possible future activities based on the state of the scene in the present and the past . These trajectories are then rated and the action of the highest rated trajectory is the anticipated one. This is in contrast to this thesis, where the neural network will do anticipation directly on the data up to prediction time, without calculating trajectories for activity recognition. The learning methods and data differ:

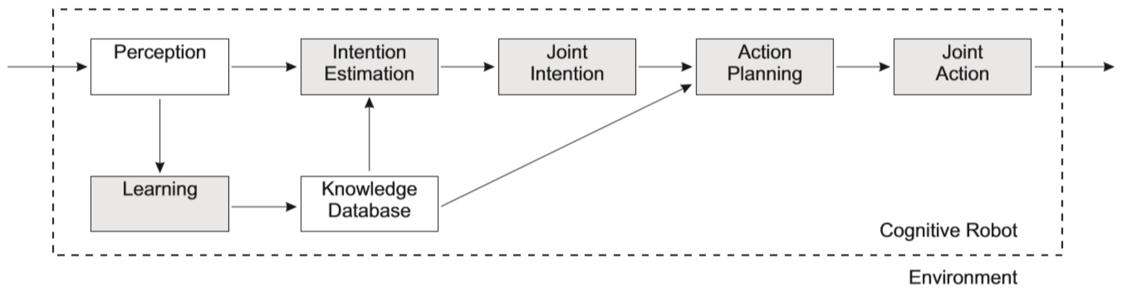


Figure 2.1: Intent Estimation in a Cognitive Robot [BWB08]

term	**explanation**
object affordances	properties of an object that determine the set of (sub-)activities which it can be involved in
activity	behavior of a human in a certain time frame to reach a goal - in this thesis: moving to a location and picking up an object or placing it somewhere
sub-activities	part of a more involved activity that requires performing multiple sub-activities that have their own sub-goals
intent	goal that a human wants to reach during a certain time frame (by executing an activity)
prediction	statement about the future
activity recognition	given data of the human performing some activity, determining which activity it is
gaze ray	ray in the direction that the human is looking, originating from the head of the human
skeleton	position and orientation of rigid bodies, corresponding to different parts of the human body
scene	the environment of the experiment (positions of table, shelves, etc.)
scenario	the scene and the tasks for the experiment
episode	time interval, that ends with picking up an object or putting it down and (ideally) starts, when the intent for the activity forms

Table 2.1: Human Robot Interaction terms used in this thesis and their explanations

Koppula et al. use RGB-D videos from the CAD-120 dataset and anticipatory temporal conditional random field (ATCRF) as learning method. Hoffmann et al. use motion tracking and eye gaze tracking data and compare Support Vector Machine (SVM) and LSTM as learning methods.

Recently, there is a rising interest in utilizing eye gaze information for human robot interaction, especially when predicting the intent of humans [AS; DRT+18; HKTM18; HM16]. Huang and Mutlu conducted an experiment, where customers looked at pictures of shop items [HM16]. Using data from eye tracking their system predicts which shop item the customer wants to be served by the robot. For the learning algorithm, high level features like the duration of looking at an object or how often an object was looked at. A support vector machine was used as learning algorithm. While the human choosing and object to pick up in the scene and the participants choosing shop items are similar, less processed data and lower level features are used to train a neural network.

Admoni and Srinivasa describe how eye gaze behavior can be included in a shared autonomy system to improve its assistive capabilities [AS]. They use the eye gaze to enhance the goal prediction of their system. Instead of motion capturing using multiple cameras and reflective markers, they reconstruct the three dimensional positions based on fiducial markers.

Duarte et al. did an experiment for action anticipation using nonverbal cues including gaze [DRT+18]. An actor sits next to two neighbors at a table. The actor wears eye tracking and motion capture gear. This is similar to the setup used in this thesis. The actor can either give the ball to one of his neighbors or place the ball on one of the three spots marked on the table. The video of the actor performing one of these tasks was then shown to other participants, up to certain points in time. Showing the saccadic eye movement to the goal, already resulted in the participants being 50% accurate with their predictions. One of the conclusions is that eye gaze provides the key information that helps humans identify actions of other humans correctly. One last interesting result is that humans are far more precise in predicting place-actions than give-actions in this experiment.

Generally, people look at objects before before manipulating them. The eye gaze usually has already moved to the object that will be manipulated, before the movement of the hand has started. The eye gaze often shifts to the object that the human plans to manipulate next before completing the current action [AS17]. The behavior of shifting the gaze to the next object can also be observed in the data recorded for this thesis.

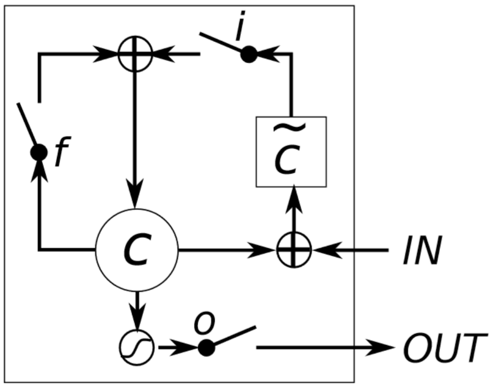
2.2 Neural Networks

Previous works on anticipation have focused on SVM as learning algorithm [HM16].
In recent times, more and more works are using neural networks for activity anticipation and recognition [HKTM18; JZSS15; TY17].
Jain et al. propose structural recurrent neural networks for modeling spatio-temporal relationships between objects and humans [JZSS15]. Their evaluation showed improvement over state-of-the-art in problems like modeling human object interactions, human motion modeling and driver maneuver anticipation.
Truong and Yoshitaka expanded upon the work of Jain et al. by taking the impact of the human on the state of objects and object-object relationships into consideration [TY17]. They show good performance in their evaluation on anticipation tasks.

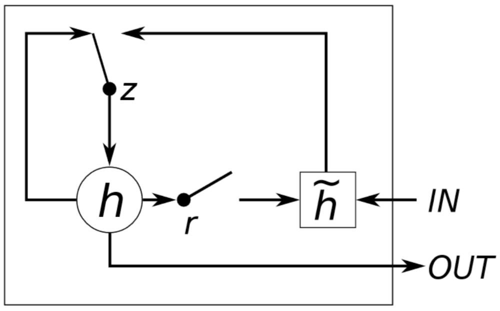
2.2.1 Recurrent Neural Network Cell Types

One type of neural network is recurrent neural networks (RNN). RNN are suitable for learning on time series data which makes them a good choice for intent prediction. Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) are choices for recurrent units, which deal with the vanishing gradient problem, that traditional recurrent neural networks suffer from.

LSTM was proposed by Hochreiter and Schmidhuber to provide an alternative to traditional recurrent neural networks that does not suffer from decaying error back flow [HS97]. The LSTM cell shown in Figure 2.2a consists of 3 gates: input gate i, forget gate f, output gate o. The forget gate takes the input and the recurrent output from the previous cell and decides based on those which parts of the memory state c should be reset and how much should be forgotten. The input gate decides based on the same inputs which parts of the cell input should be added to the memory state.



(a) LSTM cell



(b) GRU cell

Figure 2.2: LSTM and GRU cell comparison [CGCB14]

The output gate determines based on the same inputs which parts of the activated memory state should be the output of the cell. In this thesis the Keras implementation of LSTM is used [Cho15].

Cho et al. proposed gated recurrent units for learning phrase representations with a encoder-decoder RNN [CMG+14]. Chung et al. evaluated LSTM units and GRU on polyphonic music modeling and speech signal modeling tasks [CGCB14]. They found both units to be comparable for those tasks. Figure 2.2b depicts a GRU cell. It consists of 2 gates, update gate z and reset gate r. The update gate takes the last memory state concatenated with the current input to determine the mixture between current and past information that will be passed on to the next cell. The reset gate takes the same input and determines how much and which parts of the memory $\tilde{h}$ from the last cell should be forgotten before mixing. In the last step, activated and added to current input, past state is mixed with current memory $\tilde{h}$ to get the memory state h. Memory state h is the output of the cell. As GRUs are lacking the output gate, they have less weights than LSTM units. This thesis uses the default Keras GRU implementation [Cho15].

2.2.2 Early Stopping and Parameter Tuning

Prechelt tested different methods for deciding, when to do early stopping during neural network training, using a validation set [Pre98]. The method for early stopping in this thesis is similar to the UP method, which gave good results for the probability of finding a good solution. The UP method stops training, if for some fixed number of epochs, generalization error has not decreased.

Greff et al. [GSK+17] did an empirical study on parameter search for LSTM-networks. Different versions of LSTM cell structures were evaluated. They found output and forget gate to be the most critical gates. They optimized hyperparameters like hidden state size and the learning rate, concluding that they are independent. They make suggestions on how to efficiently adjust the hyperparameters.

2.2.3 Dropout

Dropout is a regularization method for neural networks that was proposed by Srivastava et al. [SHK+14]. They showed that dropout improves performance of neural networks on different supervised learning tasks. Dropout helps neural networks overfit less by assigning each node of the network in every training stage a probability, by which the node and its connections are dropped from the network during training . Nodes that are dropped out are removed from the network which means that all weights coming from this node are temporarily set to 0 and are not affected by training. Training time per stage is reduced, because not all of of the nodes are active.

For recurrent layers, dropout was shown to fail. Gal and Ghahramani propose a dropout technique called variational RNN [GG16]. Variational RNN use the same dropout mask at every time step for inputs, outputs and recurrent layers. Applying this dropout technique for GRU and LSTM models they were able to outperform existing techniques in language modeling and sentiment analysis tasks. The variational dropout implementation of Keras was used in this thesis [Cho15].

3 Capture Setup

The system that was used for recording the dataset for this thesis consists of many different programs that are distributed across different computers. Figure 3.1 shows the distribution of the programs on the different computers. The Windows machine receives the video of the motion capture cameras from the OptiTrack[1].

The motion capture setup from OptiTrack consists of 12 cameras that are specialized for motion capturing. They track reflective markers in the motion capture space. All objects in the scene and the suit that the human is wearing have these markers attached to them. Figure 3.2a shows the motion capture scene with a participant wearing the motion capture suit. The software motive running on the Windows machine is used to determine the positions of the markers in three dimensional space. Using this information, it derives the positions and orientations of the objects that are tracked.

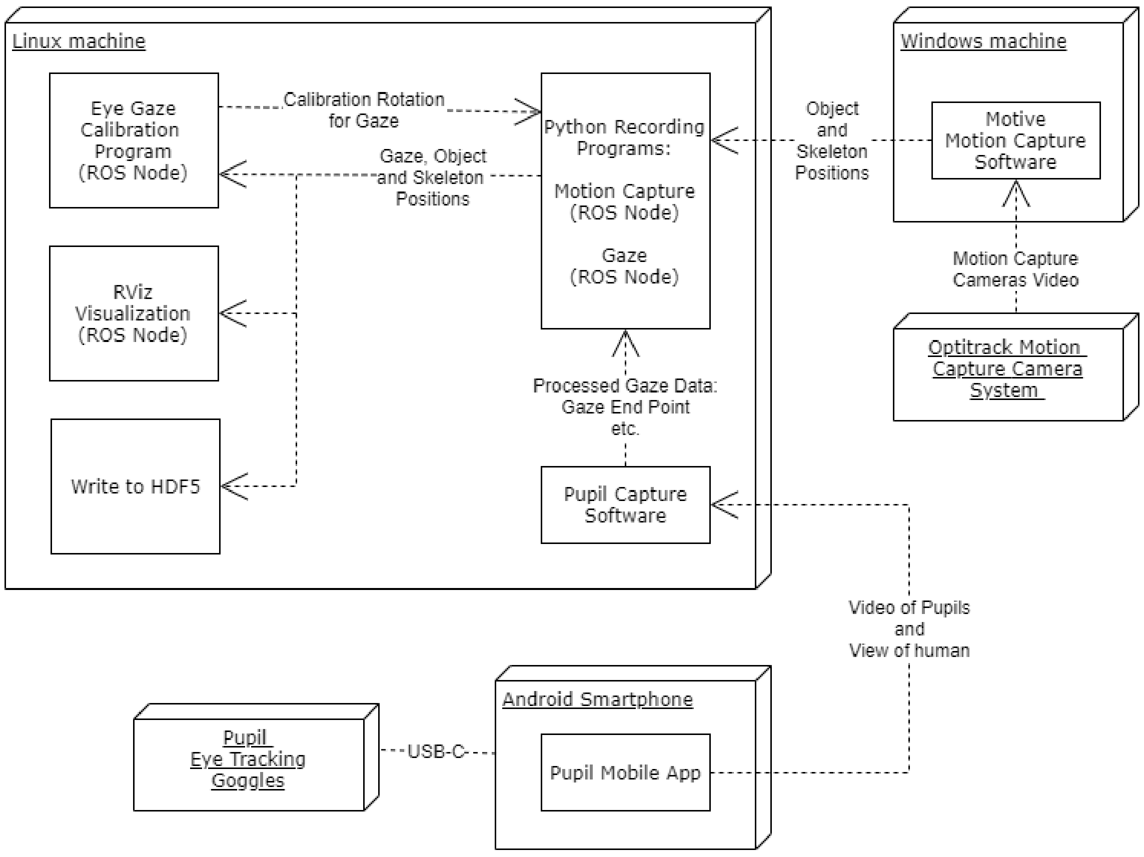
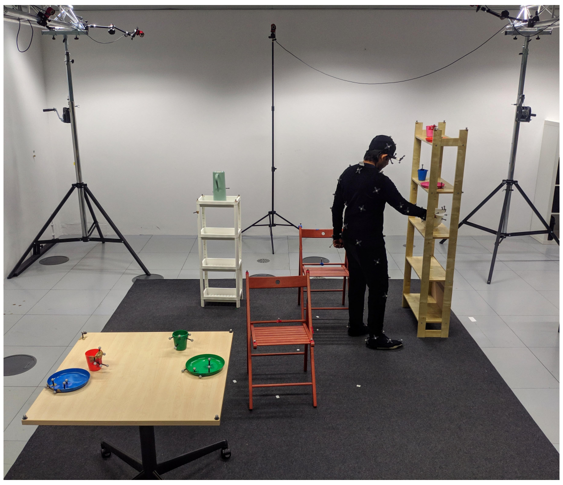
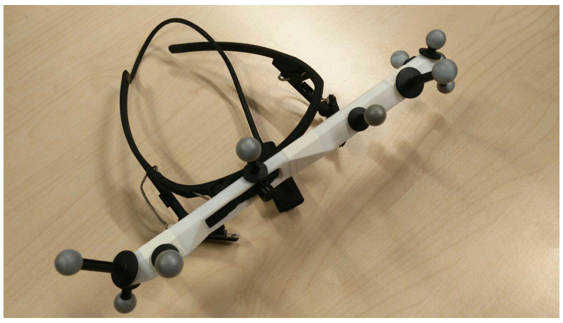

Linux machine
Eye Gaze Calibration Program (ROS Node)
Calibration Rotation for Gaze
Python Recording Programs: Motion Capture (ROS Node) Gaze (ROS Node)
Gaze, Object and Skeleton Positions
RViz Visualization (ROS Node)
Write to HDF5
Processed Gaze Data: Gaze End Point etc.
Pupil Capture Software
Windows machine
Motive Motion Capture Software
Object and Skeleton Positions
Motion Capture Cameras Video
Optitrack Motion Capture Camera System
Video of Pupils and View of human
Android Smartphone
Pupil Mobile App
Pupil Eye Tracking Goggles
USB-C

Figure 3.1: Diagram of the technical solution for motion capture with eye tracking

[1]https://www.optitrack.com/



(a) Motion capture scene



(b) Eye tracking goggles with removable 3D printed attachment for motion capture markers

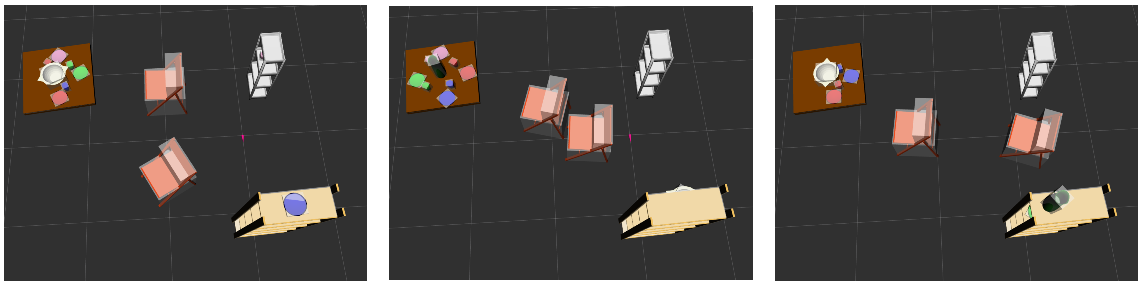
Figure 3.2: Goggles with removable attachment for motion capturing

These positions are then sent to the Linux machine.
Eye tracking goggles from PupilLabs[2] were used to capture the gaze. One of the contributions of this thesis to the setup, was to design and 3D print a removable attachment for the goggles where the motion capture markers for tracking the position of the goggles can be placed on. The goggles with the attachment clipped onto the world camera can be seen in Figure 3.2b. More details about the attachment can be found in Section A.1. Another contribution of this thesis to the setup was to find a way to make the eye tracking goggles independent of a wired connection to the Linux machine. The final solution was to use the Pupil Mobile Android app which sends the video of the eyes of the human and the world camera to the Pupil Capture Software running on the Linux machine over Wi-Fi. More details on this can be found in Section A.2.2. On the Linux machine, python programs for recording the gaze data and the motion capture data are running. These programs are ROS-nodes, restructuring and broadcasting the corresponding data. The live data is visualized using RViz. Simultaneously, the data is written to HDF5 files. The final contribution to the recording setup is a program that computes a rotation to align the coordinate system of the goggles with the coordinate system of the world, as well as modifications to the Python Recording Programs, programs for displaying the recordings and the Write to HDF5 component. This makes it possible to do a calibration procedure using the program before each recording. This calculated calibration is then automatically saved in the HDF5 file and the calibrated version of the gaze is displayed in RViz. More details concerning the calibration program can be found in Section A.2.1.

Obstacles. To prevent participants always walking on the same paths to their targets, two chairs were placed in the scene. The participants had to walk around the chair obstacles to reach their target. The positions of the chairs changed three times throughout the recording for one participant. These three positions were the same for all participants.

Limitations of the setup. With adjusting the motion capture cameras to cover the scene, especially in spaces where many objects can be present at the same time and attaching as many markers as possible to the objects, motion capturing worked very well overall for tracking all the objects in

[2]https://pupil-labs.com/



(a) Chair configuration 1 **(b)** Chair configuration 2 **(c)** Chair configuration 3

Figure 3.3: Different configurations of the chairs

the scene and the skeleton. Still, sometimes objects were temporarily not tracked properly through occlusions of markers. This could lead to objects turning upside down, or the feet of the participants not having the correct orientation when walking between chairs.

3.1 Task system

For recording the dataset, a set of tasks was defined that the participants of the recording session should perform in a random order. The recording system saves the tasks during data capturing, which makes it possible to know which task the participant was performing at every time step in the recording. It also ensures that the data does not reflect the preferences of the participants too much, for example when choosing which object should be picked next.

Criteria. The tasks should be easy to understand. The participants should not have to think much after they get a task assigned, such that the tasks do not have to be explained or repeated while recording, which will lead to delays in the data and the participants looking at the conductors of the experiment. To realize this, tasks should represent realistic higher level goals.
Every task should lead to many actions. This is important because it leaves some room for the participants of the experiment to decide which action they would like to do first. The participants would also start looking at other persons in the back of the experiment setup when they were waiting for the next task. The frequency of this behavior is smaller, when there are more actions for each task, because there is less idle time.

List of Tasks. In table 3.1 all tasks that were used during recording are listed. Each task had an occurrence probability attached to it. The tasks would be shown to the conductor of the experiment with their corresponding probability. The conductor of the experiment would then decide if the task is possible in the current state of the scene.
Figure 3.4 shows one of the interpretations of the task: "set the table for 4 persons" by the participants. The information that the participants got for this task was to place a cup and a plate for each person, and either jug or bowl on the table. None of the objects are positioned in the red edge zones. Because most of the participants interpreted this task the same way, there are few examples of objects being placed in the corner sections of the table.

Probability	**Task**
0.08	Set the table for 1 person
0.08	Set the table for 2 persons
0.08	Set the table for 3 persons
0.08	Set the table for 4 persons
0.03	Clear table
0.16	Put the jug and the bowl on small shelf
0.16	Put all cups on small shelf
0.03	Put blue and pink objects on big shelf
0.03	Put blue and red objects on big shelf
0.03	Put blue and green objects on big shelf
0.03	Put pink and red objects on big shelf
0.03	Put pink and green objects on big shelf
0.03	Put red and green objects on big shelf
0.03	Put all cups on big shelf
0.03	Put bowl and jug on big shelf
0.03	Put all cups on big shelf
0.03	Put all plates on big shelf

Table 3.1: Table containing the instructions that were used, paired with the probability of the task being assigned to the participant

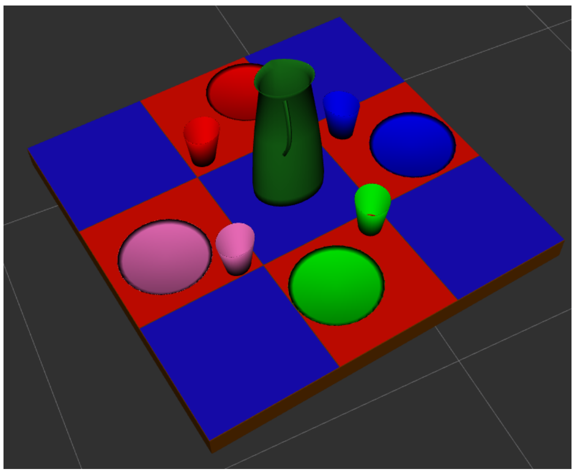


Figure 3.4: Result of the “set the table for 4 persons” task

3.2 Capture Workflow

Each of the participants got a short introduction on how the capture setup works, what the tasks are and what they mean. The motion capture suit was put on and the goggles were calibrated (See A.2.1). Then a short recording was done, to get the participants used to the capture setup and the tasks. After that the final recording session was started. Sometimes the program used for tracking the skeleton and objects in the scene would fail during recording. In this case, the recording was stopped and a new motion capture skeleton model was created. Then the recording was resumed. Because of that, the recording data for some participants was split up into multiple recordings. In the evaluation part of this thesis, the merged recording data for each participant is used.

4 Dataset and Data Processing

In total, about 200 minutes of data were recorded. Capturing was done with six participants. Five participants were male, one was female. All participants were students.

4.1 Dataset Contents

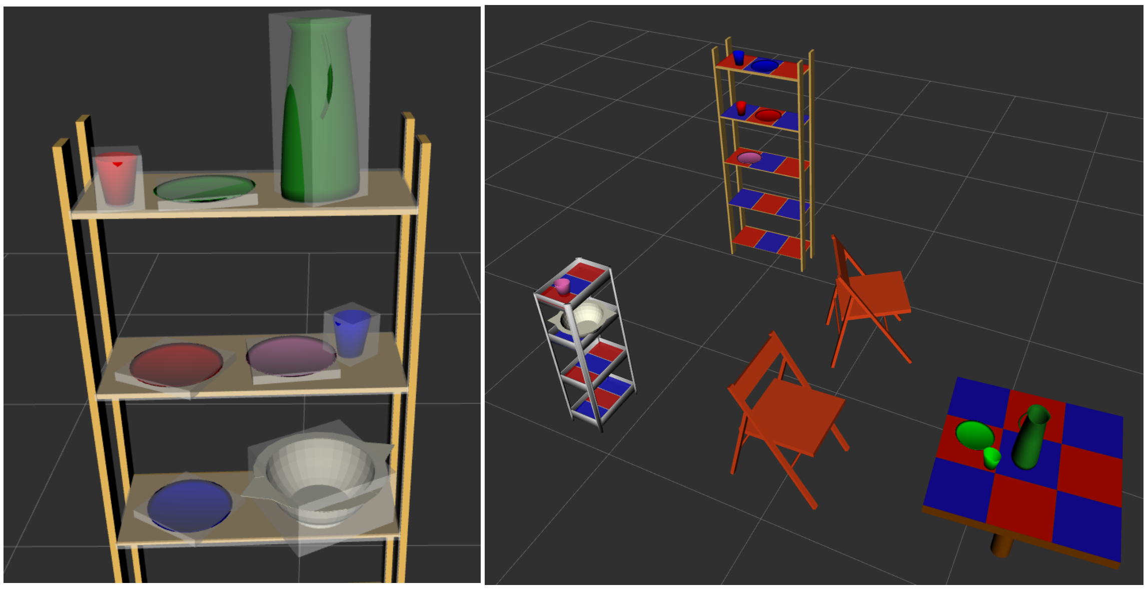
Following information is available for all recordings:

- Positions and orientations of the objects, including the Eye Tracking goggles
- Positions and orientations of the limbs of the skeleton
- Start point and distant end point of the gaze
- Calibration rotation for gaze end point
- Sizes of the bounding boxes of the objects and their position and orientation relative to the corresponding object
- Tasks and the timestamps at which they were posed to the participant

The objects can either be movable or non-movable. The non-movable objects were a table, a small shelf and a big shelf. The chairs are also non-movable, but change positions to each one of the three configurations throughout the recording (See Figure 3.3). The non-movable objects are also obstacles, especially the chairs, since they are located in the middle of the scene. The movable objects are a bowl, a jug, also 4 plates and 4 cups in colors red, green, blue and pink.

4.2 Features

When using neural networks for machine learning often raw data is used as input to the network. By fitting the network to the training data, it should learn to compute the relevant features. Still, it may be useful to test different representations of the same data as input to the network and evaluate its performance on those different inputs. A relevant example for this thesis would be computing the distance of a point to a ray. In the recorded data, the eye gaze ray has a start point and a point indicating the direction of the ray. The objects are represented by a point. Choosing one of the different representations of the eye gaze discussed in Section 4.2.2 may simplify this task for the network.



(a) Bounding boxes of the movable objects

(b) Place Partitions: Red and Blue Surfaces

Figure 4.1: Scene Details

The features can be split into gaze based and skeleton based features. Gaze based features use the start and end point of the gaze as a basis, while the skeleton based features use the positions of the rigid bodies corresponding to the limbs of the skeleton. A second way to categorize the features is position based or distance based features. Position based features are based on points, while distance based features use the distance between points. All position based features use relative positions. As reference point the position of the torso rigid body was used. This means that all positions from the features are relative to the position of the centroid of the torso of the human. The gaze based features used in this thesis are presented in section 4.2.2, the skeleton based ones in section 4.2.1.

One additional feature that is not available straight from the recording are the place partitions. The table and shelf board surfaces are divided into rectangles, shown in Figure 4.1b. The rectangles are marked with a red and blue chessboard pattern, where each colored rectangle is one of the partitions. The table is divided into 9 place partitions, in a three by three grid. Surfaces of the shelf boards are divided into 3 place partitions each. The place partitions are used for place classification.

4.2.1 Skeleton based features

The skeleton based features are the following:

1. Positions of the rigid bodies making up the skeleton of the human
2. Distances of the right hand position to the positions of all the objects
3. Distances of the right hand position to the centroids of the place partitions, seen in Figure 4.1b

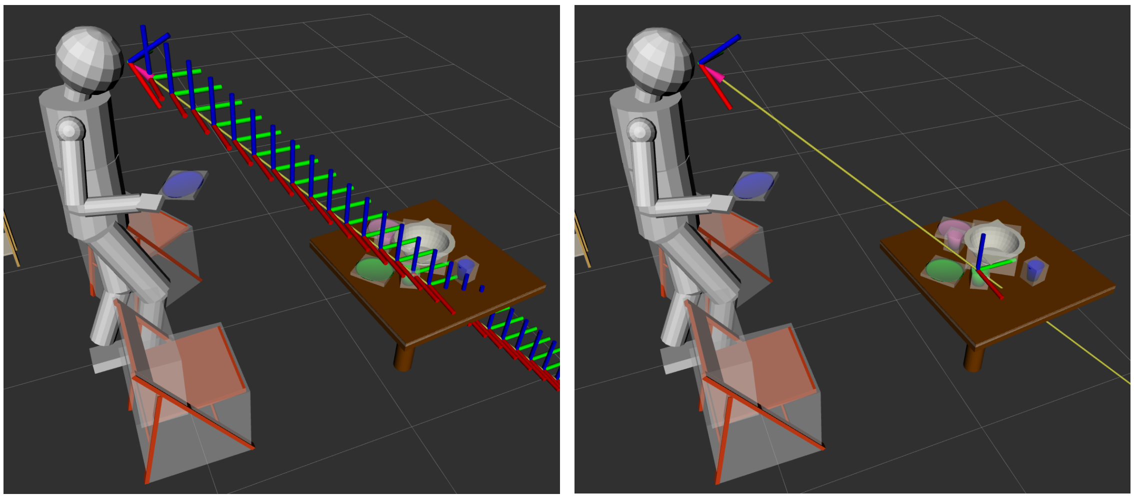
The skeleton based features, over time give the information on where the body parts currently are relative to the torso. This information can be used to learn patterns of movement, for example moving the arm to form a reaching movement to some object.

4.2.2 Gaze based features

Information of the eye gaze can be encoded in several different ways:

1. Gaze start point and second point on gaze ray indicating its direction
2. Points, placed along the gaze ray (Figure 4.2a)
3. Gaze ray intersection point with the bounding boxes of the objects (Figure 4.2b)
4. Distances of the objects from the gaze ray
5. Distances of the centroids of the place partitions, seen in Figure 4.1b

The information received from the recording system concerning the gaze is the gaze start point and a point on the gaze ray indicating its direction. The positions of these points can be directly used as an input to the network. Gaze start and end point only give the network information about the direction of the gaze ray.
Using the gaze ray, the intersection point with the bounding volumes of the objects in the scene can be calculated. Only the first intersection point from the ray origin is used. An example of the gaze intersection point can be seen in Figure 4.2b. In addition to the direction of the gaze ray, the gaze intersection point also gives information on where exactly the object is that the person is looking at. One flaw of using the intersection point is that when no object is hit, some default value has to be the output of the calculation. Here, the result will be an all-zeros vector. That leads to the second flaw of the intersection point which lies in the unstable nature of eye gaze tracking. When the gaze is not recognized correctly and the gaze direction is inaccurate, the gaze can miss the object and the default value is returned, giving no information at all except that no object was hit.
Another possibility is to place uniformly spaced points on the gaze ray, originating from the gaze start point and going through the gaze end point which will be referred to as gaze trace points. Figure 4.2a shows the points as RViz transforms. The 25 points are spaced 0.1 meters apart. Not only do the gaze trace points indicate the direction of the gaze ray, but also give a notion of distance. If the human is close to the object, the points close to the head will be nearest to the objects that the human is looking at. If the object that the human is looking at is far away, the points further along the ray will be in the proximity of the object.
The gaze ray can also be used to calculate ray-point distances to the centroids of the objects in the scene. Given the gaze start and end point and the positions of the objects, the neural network would have all the necessary information to compute this feature. Still, it may be beneficial to give this information directly to the network, as it is easy to calculate and not known, if the network is able to find this representation. A benefit of this representation is that the network only needs to return the negated input vector in order to classify the object nearest to the gaze ray. The drawback of this encoding is that there is no information on how far the human is away from the object.
The same properties apply to the distances from the centroids of the place partitions, when placing an object. The distance to the centroid decreases, as the human is looking at the partition. The



(a) Points placed along gaze ray **(b)** Gaze intersection point with scene

Figure 4.2: Visualization of gaze encoding methods

distance D of a point P to the gaze ray, given by start point S and end point P is computed as follows:

$$\text{given point } P \text{ , ray start point } S \text{, ray end point } E \text{ :}$$
$$M = E - S$$
$$t = \frac{M * (P - S)}{M * M}$$
$$D = \begin{cases} |P - S| & t <= 0 \\ |P - (S + t * M)| & t > 0 \end{cases}$$

A more high-level way to encode gaze is to directly state which object is nearest to the gaze ray as one-hot-encoding or the gaze intersection point. This way a score can be accumulated over time for each object, depending on how long and how often the object is looked at. These high level features are not explored in this thesis.

5 Intent Prediction

5.1 Intent and Ground Truth

The intent of the human is the next activity that the human is planning to do. For our scenario, the intent of the human could either be to pick up an object from one of the surfaces in the scene or to put a held object on a surface at a certain position. Based on the proximity of an object to the hand of the human and the movement of the object, it can easily be determined, if the object is currently held. If an object is held, the next intent will be to place the object somewhere, in the other case it will be to pick up an object. So the timespan in which no object is held, has the ground truth to pick the object that is held at the end. For placing, the position where an object will be placed is discretized with the place partitions (See Figure 4.1b). The timespan where an object is held has the ground truth to place the object on the place partition that the object is standing on.
The intent can change during the timespan of executing an activity, but this is hard to detect. For example if the participant noticed that there is no space at the position where it was planned to place the object, then the intent changes to place the object somewhere else. This case was ignored for determining the ground truths.

5.2 Timing of Training Data

As already discussed in the previous section, it is hard to determine when the intent of the human forms, if the intent has changed or if the human is just waiting and having no intent of doing an action. This is why a time limit of 2.5 seconds before picking or placing happens was introduced for the episodes. The short time frame is less likely to include waiting time or intent changes and more likely to only include the time, when the acitivity attached to the intent ground truth is being executed. Figure 5.1 shows, how these episodes of length 2.5 seconds are partitioned into training instances of length 0.87 seconds. The training instances are offset by 0.33 seconds. From the 1 second time interval, 14 frames from motion capture and eye gaze are taken in a uniform time interval.

With this timing of the episodes, there are probably still intent changes at the end of the episodes. The corresponding observations can be found in Section 6.3.

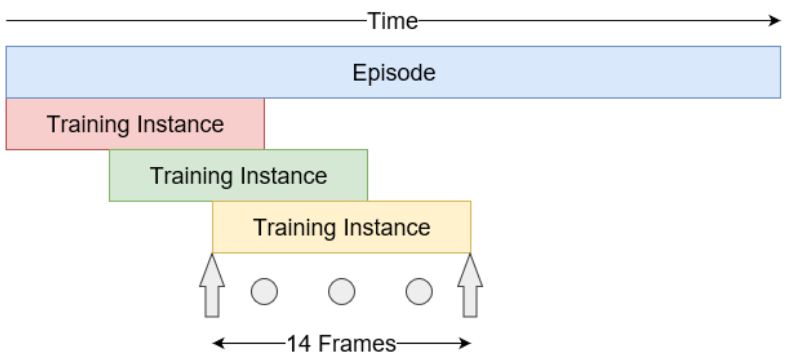


Figure 5.1: Episode and Train Instances

5.3 Network Architecture

To deal with the given time series data, the network architecture seen in Figure 5.2 was used. It consists of up to three stacked recurrent layers and a fully-connected (dense) output layer. The hidden state size and stack size of the recurrent layers can be adjusted to achieve the best result possible. If recurrent layers are stacked, the first layer receives the inputs to the model as its input. Following layers in the stack receive the hidden state of the previous recurrent layer for each time step as input. The fully connected output layer is softmax activated and receives the hidden state of the last recurrent layer in its last time step as input. Its number of nodes corresponds to the number of classes.

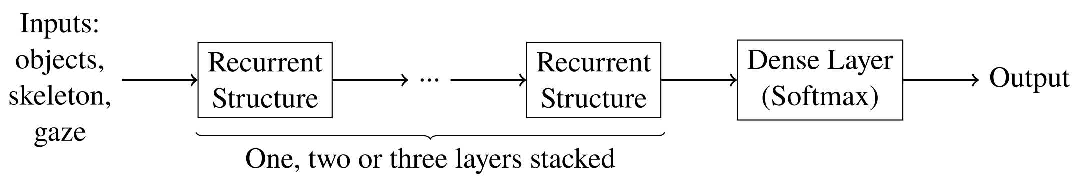


Figure 5.2: Simple recurrent network architecture

5.4 Prediction outcomes

The softmax output of the neural network is a vector of the length of the number of classes. The values of this vector can be interpreted as predicted probabilities of the different classes. Based on these scores, the category with the highest score can be determined. One can also look at how close the scores of the most probable classes are or rank the classes by their probability. In some applications it may not be important to know which object will be picked up or where exactly an object will be placed, but to a have selection of objects which may be picked up or a selection of places where something may be placed. In this case, all classes can be returned that have a score above a certain threshold.

6 Evaluation

In this chapter, the performance of the models on different inputs is compared. Picking and placing intent prediction are evaluated separately. The data that was recorded for one participant was used as validation set. All other recordings were used to train the models on. Validation set and test set are used synonymously. The final results are validated with leave-one-out cross-validation for the data from each of the six participants. This will also provide information about potential differences between the participants and show that the models can be applied to different persons. During leave-one-out cross-validation, data from one of the participants is used as validation set and the rest of the data is used to train the models on. This is repeated so that data from every participant is the validation set once.

In Keras, neural networks are trained in epochs. An epoch corresponds to a complete pass over the training data. All networks were trained with Early Stopping and cross entropy loss as metric. Early stopping patience was set to 10 episodes. That means training is stopped if there is no decrease from the lowest cross entropy loss achieved on the validation set for 10 episodes. Through parameter search on a representative model, a learning rate of 0.0001 and a batch size of 5 were chosen.

6.1 Metrics

Accuracy: Accuracy is the number of instances, which were classified correctly, divided by the number of all instances. This gives some indication on how well the model performs on the validation data overall. However, classes that have little representation in the validation data, are weighted very lightly. A Model performing especially well or especially bad compared to bigger classes on these small classes will not be reflected in the accuracy score. For single label classification, the accuracy is the same as the micro F_1-score.

F_1-score:
Precision p is the fraction of positive results that are correct (true positives tp) with the total number of positive results $tp + fp$ (false positives fp):

$$p = \frac{tp}{tp + fp}$$

It answers the question which percentage of results that were classified to be in a class, are actually in that class.
Recall r is the fraction of positive results that are correct with the total number of correct results $tp + fn$ (false negatives fn):

$$r = \frac{tp}{tp + fn}$$

It answers the question which percentage of correct results were given.
The F_1-score is the harmonic mean of precision p and recall r. It is calculated as follows:

$$F_1 = 2 * \frac{p * r}{p + r}$$

There are also other F-scores which either put more emphasis on recall or precision.
Precision and recall can be computed for each class separately or for all classes. In the first case, recall and precision are computed with per-class tp, fp, fn values. The resulting class-F_1 scores computed from class precision p_c and recall r_c can be averaged, resulting in the macro F_1-score:

$$\frac{\sum_{c \in C} F_1(p_c, r_c)}{|C|}$$

This score is not influenced by how big each of the classes is, the F_1-score for each class is weighted the same in the macro F_1-score. The micro F_1-score on the other hand takes the sizes of the classes into consideration, by computing precision and recall on the total number of tp, fp, tn, fn over all classes, and from those calculating the micro F_1-score.

6.2 Baselines

Four baselines were used to compare their performance to the performance of the models. Two baselines for picking and two baselines for placing:

- Pick Baselines:
  - object gaze ray distance
  - object hand distance
- Place Baselines:
  - place partition gaze ray distance
  - place partition hand distance

The baselines use the distance from the gaze ray and the hand to the positions corresponding to the intent prediction classes. For picking, the distance object gaze ray feature was used as the first baseline. The object with the smallest distance is the object that is predicted to be picked next. The second baseline for picking uses the distance object hand feature in the same way as the first baseline. Similarly, the two baselines for placing use the distance features corresponding to the place partitions, rather than the objects. The first baseline uses the distance place partition gaze ray feature and predicts the place partition with the smallest distance to be the one that will be placed to. The second baseline does the same, using the distances to the centers of the place partitions.

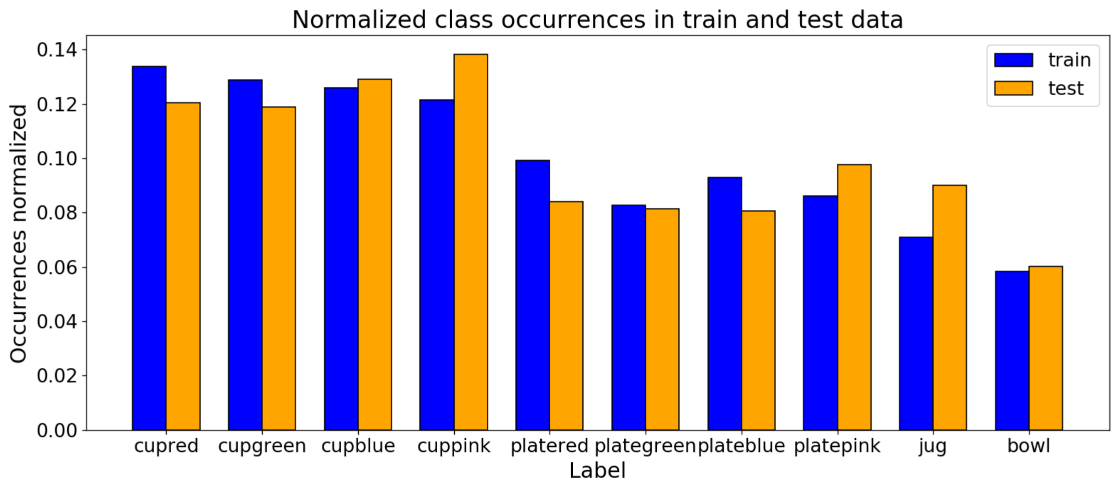


Figure 6.1: Occurrences of classes in train and test set, normalized by the corresponding total number of instances, on pick episodes.

	highest accuracy	lowest accuracy	average accuracy	standard deviation
no gaze	0.405	0.182	0.324	0.052
start and end point	0.422	0.139	0.291	0.099
intersection point	0.425	0.256	0.361	0.044
trace points	0.451	0.324	0.400	0.034
distance	0.597	0.397	0.519	0.058

Table 6.1: Model accuracy results for different gaze encodings, best, worst, average and standard deviation of model accuracy over all parameters

6.3 Evaluation of Picking Results

For picking, splitting off the recordings from one person, resulted in 5804 instances for training and 1179 instances for validation.

In Figure 6.1, the classes and their corresponding normalized number of occurrences can be seen. The normalized number of occurrences is displayed for both train and test set. The normalized numbers of occurrences are evenly distributed between train and test set. The Figure shows that cups are picked more frequently than the other objects.

First the performance of the different gaze encoding methods was evaluated. For this, the positions of the skeleton parts, the objects and one of the four gaze features were used as input. The same set of inputs was also used without gaze feature. For the different inputs, models with varying hidden state size, recurrent network stack size, recurrent network cell type and variational dropout on the recurrent layers, were used. The hidden state sizes that were tested are 50, 100 and 200. Recurrent stack sizes between 1 and 3 were tried. LSTM and GRU were used as recurrent structures. The networks were trained with and without variational dropout of 0.2 on the output and recurrent

connections of the recurrent layers. This resulted in 36 different combinations for each type of gaze encoding and in total 180 combinations.
In Table 6.1, the best and worst results on these different combinations can be seen, as well as the average and standard deviation for each of the models over all parameters.
With only skeleton as input, the network with the best accuracy was performing worse than the other models. With a 1.7% gap to the start and end point input.
Representing the gaze in its original form, with start and end point, the performance of the best network was the worst, compared to the other gaze encoding methods. The best accuracy that could be achieved training over all combinations of parameters was 42.1% with an average accuracy of 29.1% and a standard deviation 9.9% of the accuracy. This means that training was not very stable, and training outcomes were almost random, with the lowest training result for accuracy being 13.9% which is very close to chance. In the case of picking chance would be 0.1.
Surprisingly, the best result for the gaze intersection point is only 0.3% better than the original method. The lowest accuracy is much higher than for the original method, improving from 13.9% to 25.6%. The average accuracy is also much higher and the standard deviation is more then halved, suggesting that models were able to learn better from this gaze encoding method.
The highest accuracy of the gaze trace points improves further on the accuracy of using the gaze intersection point, with an increase of 2.6%. Both lowest accuracy and average accuracy increase as well, with the standard deviation decreasing.
Having the distance from the gaze ray to the intersected objects in the gaze ray improved the performance a big deal, improving another 14.6% from using gaze trace points. While lowest accuracy and average accuracy improve upon those of gaze trace points, standard deviation is only lower than when using gaze start and end point, being the second highest.

To get an understanding of how the models perform over time on the different sets of inputs, the models were evaluated for time steps leading up to the action. The results for each of the time steps were then averaged over all episodes in the test set. The performance of the best models for each input set can be seen in Figure 6.2. The trajectories of the models for gaze input start end, intersection, gaze trace and no gaze are very similar and stagnate at 1.5 seconds in their accuracy of about 42%. The trajectory of the model with gaze distances as input on the other hand starts to rise more rapidly than the other models at the 1.5 second mark and holds an accuracy of about 60% until the end.

Following observations where made about the choice of parameters. Over all gaze encoding methods, the GRU model always achieved the highest accuracy, with exception of gaze trace points, where the LSTM achieved the highest result. The models with the highest accuracy on each input set always used dropout, except for the input set using the gaze distances.

One interesting observation is that when using variational dropout in combination with position based features, the accuracy is almost always improved. When using the gaze distances, it only improved the accuracy when using a LSTM model and decreased the performance of the GRU model.

To further improve the accuracy of the models, different gaze encoding methods were combined in one model. Because the networks were able to learn much better from the distance based gaze feature, the skeleton based feature of hand distance to the objects is added to the set of inputs. One model was only trained on the distance-based features.
This resulted in the following sets of inputs:

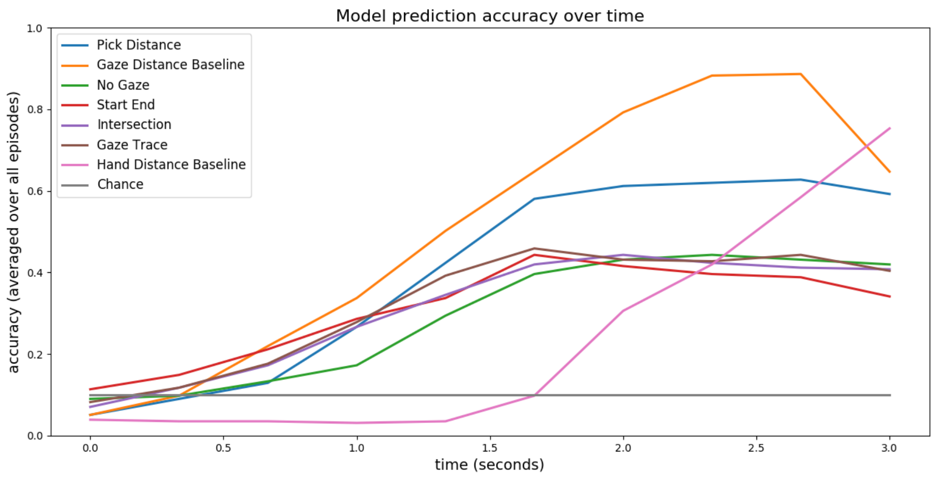


Figure 6.2: Pick prediction accuracy for the best models to each set of inputs over time. Averaged over all episodes of the test set, leading up to the actions

1. Combined Model: skeleton, objects, gaze trace points, gaze object distances, hand object distances
2. Only Distance Gaze Hand: gaze object distances, hand object distances
3. Only Distance Gaze: gaze object distances
4. Only Distance Hand: hand object distances

Because of the findings from parameter search, GRU with variational dropout is used for the Combined Model. The Only Distance Models use GRU without variational dropout. The number of hidden units is 200 and the stack size is 1.

The results for these models can be seen in Table 6.2. The Combination model did not achieve better results by using gaze distances and gaze trace points over using only distances in combination with the skeleton and the object positions. Only Distance Gaze and Only Distance Gaze Hand model performed better than the model using the gaze distances from parameter search which was using the skeleton positions and object positions as well. Performance improved from 60% to 63%. Including the distances from the hand does not show a clear improvement, as both models achieved the same accuracy of 63%. The model using only the distance to the hand performs better than the hand distance baseline with an accuracy of 43% compared to 33% of the baseline. Still, the models are not able to outperform the gaze baseline. The hand distance baseline has a relatively low accuracy, because the performance is evaluated over data from the whole episode. This means there are instances in the data, where in their corresponding timespan, the human is not near the object yet.
Figure 6.3 shows the prediction accuracy over time of a selection of models. The time goes from left to right, with the object being picked at the 3 second mark. The accuracy of the models and the baseline is steadily rising, up until 2.6 seconds. Here, the models stagnate and the gaze baseline is dropping. The hand baseline accuracy keeps increasing until the end. The gaze baseline accuracy

Models	Com-bined Model	Only Distance Gaze	Only Distance Hand	Only Distance Gaze Hand	Gaze Distance Baseline	Hand Distance Baseline
accuracy	0.56	0.63	0.43	0.63	0.78	0.33
macro F_1-score	0.55	0.62	0.41	0.63	0.78	0.33
Per class F_1-scores						
cupred	0.58	0.65	0.50	0.55	0.74	0.38
plateblue	0.58	0.60	0.39	0.63	0.74	0.25
jug	0.55	0.58	0.34	0.60	0.74	0.35
plategreen	0.47	0.59	0.24	0.59	0.77	0.32
platered	0.53	0.53	0.45	0.60	0.82	0.35
cupgreen	0.62	0.67	0.48	0.67	0.76	0.28
cupblue	0.61	0.68	0.53	0.71	0.79	0.37
cuppink	0.53	0.63	0.47	0.59	0.75	0.28
platepink	0.45	0.58	0.34	0.61	0.82	0.28
bowl	0.59	0.67	0.38	0.72	0.78	0.38

Table 6.2: Accuracy, macro F_1-score and Per-Class F_1-scores for Pick Classification

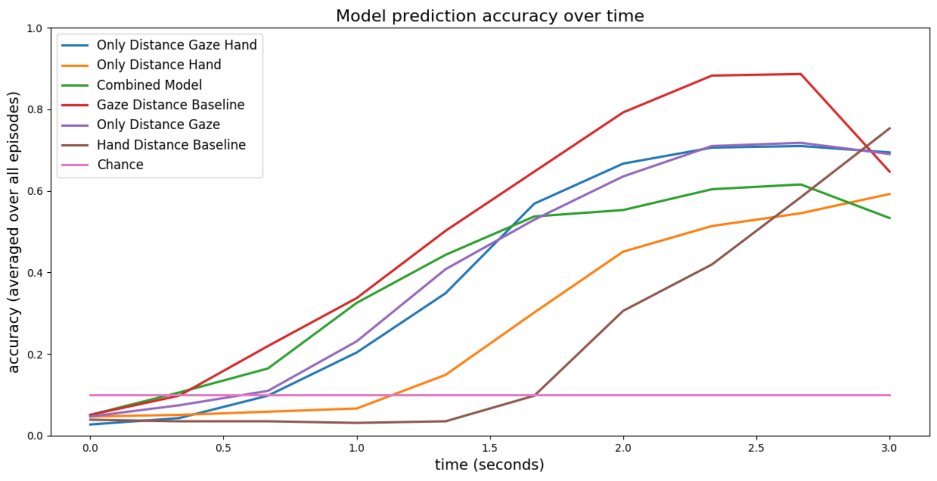


Figure 6.3: Pick prediction accuracy of final models over time, averaged over all episodes of the test set, leading up to the actions

	p_1	p_2	p_3	p_4	p_5	p_6
Combined Model	0.529	0.607	0.581	0.546	0.487	0.464
Only Distances Gaze Hand	0.664	0.662	0.644	0.677	0.604	0.587
Gaze Distance Baseline	0.777	0.813	0.775	0.890	0.729	0.809
Hand Distance Baseline	0.540	0.555	0.329	0.444	0.430	0.426

Table 6.3: Leave-one-out cross-validation of models over the recordings of each participant p_x

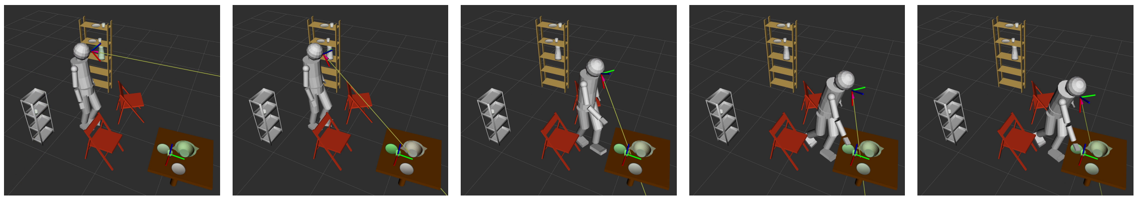


Figure 6.4: Frames from a video visualizing the model prediction value for each object

outperforms the models at all time steps, especially from 1.5 seconds to 2.7 seconds, where the area between the curves of the models and the baseline is especially big.

Table 6.3 displays the results of leave-one-out validation over the recordings for each participant p_x, where the data from p_x was used as validation set. p_3 is the participant, whose recording was used as validation set. The performance of the gaze baseline was varying a lot between the different recordings, with a difference of 16.1% between p_4 and p_5. This could be caused by differences in gaze behavior between different persons or the quality of gaze calibration that was achieved in their recording.

Figure 6.4 shows a series of frames from a video visualizing the values of the output vector of the network for each of the classes. Each of the classes corresponds to an object. The score is displayed through the color of this object. The higher the prediction score, the higher the intensity of its green color. The object with the highest score has a RViz transform displayed at its centroid. In the first frame of the video, the human is standing and looking in the direction of the table. The model gives low scores for the objects on the table. The cup, the bowl and the plate left of the bowl appear in a light green color, as the model is not sure which of the objects on the table will be picked. With the human approaching the table in the follwing frames, the model gives the cup higher scores, its green color is getting more intense. In the last frame the human can be seen picking up the cup.

The video gave some insights, into how the accuracy values of the models and the baseline come about. The Gaze Distance Baseline always predicts the object closest to the gaze ray as the human and the gaze move. The machine learning models show similar behavior to the Gaze Distance Baseline in some cases but are overall more consistent in their decision, not immediately jumping from the old predicted object to a new object when the new object is closer to the gaze ray. This is why for some picking episodes the model can be seen to be more accurate in its prediction than the Gaze Distance Baseline in the case where the baseline model jumps between objects that are close together. For most episodes, if the human starts looking at the object that he wants to pick up, he keeps looking at the object until he picks it up. This means that unless there appear inaccuracies in

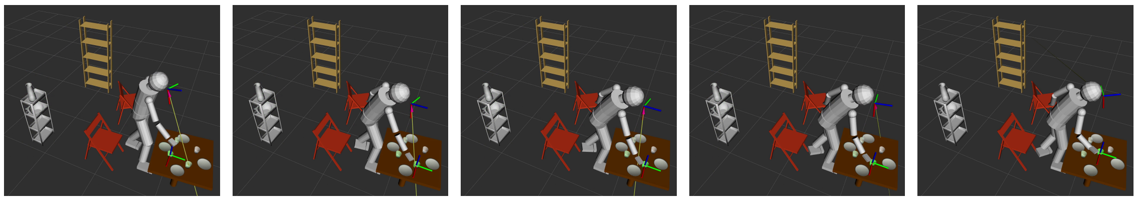


Figure 6.5: Frames from the video showing the gaze shifting to the shelf while picking from the table

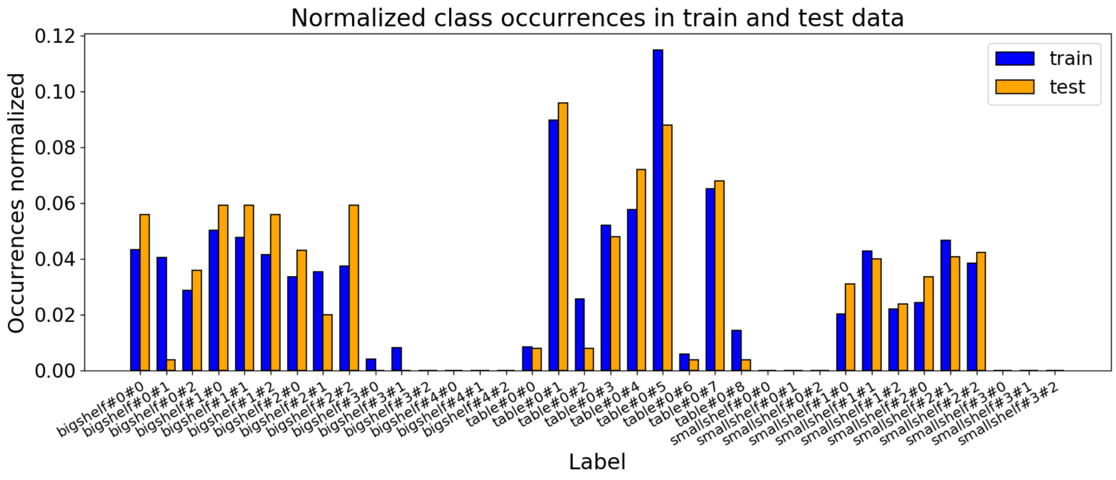


Figure 6.6: Occurrences of classes in train and test set, normalized by the corresponding total number of instances, on place episodes.

the gaze tracking, the baseline is always predicting the correct object. Because of that it so hard for the models to beat the baseline as relying on past time step data and positional data of the human and the objects will not result in an advantage of the model if the human keeps looking at the object.

From this video, one interesting observation for some of the episodes was the human shifting the gaze in the direction of his next goal at the end of the episode, when after picking, the intent changes to placing. The human is able to quickly learn, where the table and shelves are, as they do not change their position during the recording. Then, while reaching for the object, the human already shifts his gaze towards the table or shelves where he wants to place the object. Video frames from one such episode can be seen in Figure 6.5. While grasping the object, the human is already turning and shifting the gaze towards the shelf. This behavior can also be observed in the accuracy of the gaze baseline. The gaze baseline is always predicting the object that the human is looking at to be picked next. The baseline has a strong dropoff in average accuracy over the last 0.25 seconds of the episodes. These observations suggest an intent change in the last quarter second of some of the pick episodes to a placing intent while the picking action of the previous intent is still going on.

	highest accuracy	lowest accuracy	average accuracy	standard deviation
no gaze	0.414	0.209	0.352	0.054
start and end point	0.514	0.336	0.446	0.042
intersection point	0.455	0.267	0.389	0.055
trace points	0.598	0.388	0.541	0.049
distance	0.535	0.330	0.452	0.054

Table 6.4: Model accuracy results for different gaze encodings: best, worst, average and standard deviation of model accuracy over all parameters

6.4 Evaluation of Placing Results

Splitting the recordings from one participant resulted in 5866 instances for the training set and 1251 instances for the validation set. Class names like bigshelf#0#2 in Figure 6.1 mean that something will be placed on the big shelf, on shelf board 0, at place partition 2 of that shelf. The Figure shows that the normalized number of occurences for the more common classes are in most cases even between the training and validation data. But there are some instances where the training set has a much higher normalized occurence value for a class than the test set. This is the case for for bigshelf#0#1 and table#0#5, where in the training data, objects are placed much more frequently on one part of a shelf board or the table, than is the case in the test data. This can be caused by preferences of the participants, because the tasks do not specify where on the table or the shelves something should be placed. Some of the places, like the shelf levels that are closest to the ground, were not used at all by the participants. That is why there are no occurrences for the classes corresponding to these places, for example placebigshelf#4#0. The corners of the table - table#0#0, table#0#2, table#0#7, table#0#9 - were also barely used by the participants.

The evaluation of the performance for models on different gaze encoding methods from Section 6.3 for picking was repeated for placing. The parameter search included different gaze encodings, recurrent structures, dropout, stack size and number of units.
The highest, lowest and average accuracy as well as standard deviation of the accuracy over all parameters are shown in Table 6.4. The results are separated by gaze encoding methods.
With no information about the gaze available, the models performed significantly worse than with gaze information available, with a 4.1% performance decrease compared to the gaze encoding method with the lowest maximum accuracy, the gaze intersection point.
Surprisingly, encoding the gaze with the gaze intersection point resulted in a lower performance, than with the original gaze encoding, using gaze start and end point, with a big difference of 5.9% of the highest accuracy, respectively. Lowest accuracy and average accuracy were also much lower for the gaze intersection point and the standard deviation higher compared to gaze start and end point.
Having the start and end point in the set of inputs, resulted unexpectedly good performances, with a highest accuracy of 51.4%. The accuracy was not far off the one of the model using distances, with a difference of 2.1%. In fact, the average accuracy for start end point was higher than for gaze distances.
The second best gaze encoding method for placing were the distances, achieving an accuracy of 53.5%. The best representation for placing turned out to be the gaze trace points, with a highest accuracy of almost 60%. An improvement of 6.5% over using distances, as well as having a much

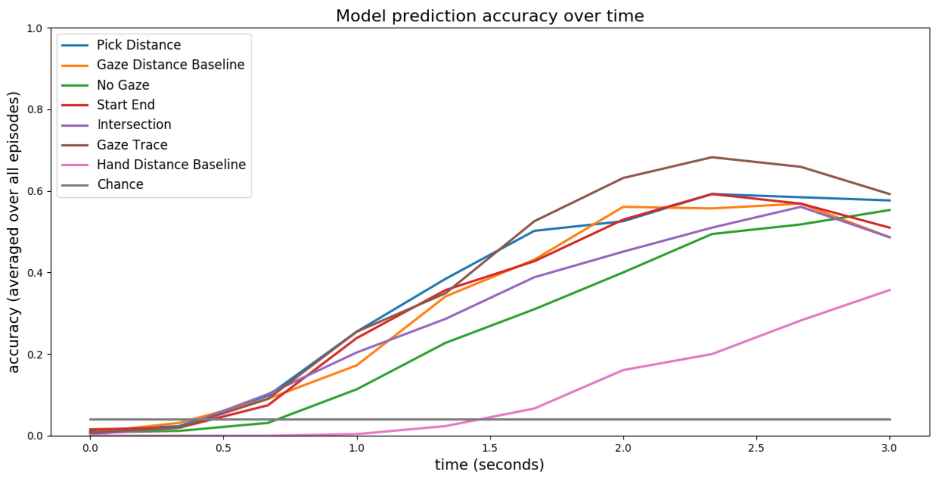


Figure 6.7: Place prediction accuracy for the best models to each set of inputs over time. Averaged over all episodes of the validation set, leading up to the actions.

higher average accuracy. Gaze trace points improved model performance by 8.9% over distances. This suggests the networks were able to learn much more from the gaze trace points than the distances in case of placing.

Figure 6.7 shows the performance of the model on the different inputs over time. 2.7 seconds before the end of the episode, the prediction accuracy of all models with gaze becomes better than chance. The model without gaze information has a similar plot to the hand baseline, but it is steeper and reaches a higher accuracy. The intersection point model shows the same behavior as seen in Table 6.4. Its accuracy stays between the model without gaze and the one that uses gaze start and end point. Gaze baseline model, gaze start end model and gaze distance model have similar curves. They are for the most part lower than the model Gaze Trace model. The gaze trace model is better than the gaze baseline and most of the other models over the complete interval.

1. Combined Model: skeleton, objects, place part positions, gaze trace points, gaze place distances, hand place distances, gaze object distances, hand object distances
2. Only Distance Gaze Hand: gaze object distances, hand object distances
3. Only Distance Gaze: gaze object distances
4. Only Distance Hand: hand object distances

With the results from parameter search, GRU with 200 units and a stack size of 1 was chosen as network architecture. For the combined model, dropout was used. For the distance based models, no dropout was used.

The results for these models can be seen in Table 6.5. For the results in the table, all classes that do not occur in the validation set were excluded from the evaluation. The accuracy of the combined model (60%) about the same as for the model using only the gaze trace points as gaze feature. Both input sets gaze distances with and without hand distances with 58% and 55%, performed

Models	Gaze Distance Baseline	Hand Distance Baseline	Only Distance Gaze Hand	Only Distance Gaze	Only Distance Hand	Com-bined Model
accuracy	0.51	0.16	0.58	0.55	0.40	0.60
macro F_1-score	0.41	0.08	0.45	0.42	0.30	0.46
Per class F_1-scores						
bigshelf#0#0	0.64	0.0	0.62	0.60	0.40	0.67
bigshelf#0#1	0.0	0.0	0.0	0.0	0.0	0.0
bigshelf#0#2	0.18	0.18	0.37	0.59	0.0	0.58
bigshelf#1#0	0.73	0.02	0.60	0.58	0.41	0.61
bigshelf#1#1	0.48	0.07	0.21	0.27	0.23	0.37
bigshelf#1#2	0.55	0.34	0.34	0.5	0.21	0.46
bigshelf#2#0	0.79	0.0	0.5	0.52	0.30	0.71
bigshelf#2#1	0.25	0.05	0.31	0.41	0.18	0.27
bigshelf#2#2	0.63	0.59	0.63	0.71	0.45	0.79
table#0#0	0.27	0.0	0.0	0.0	0.0	0.0
table#0#1	0.71	0.0	0.74	0.72	0.49	0.73
table#0#2	0.12	0.02	0.0	0.0	0.0	0.0
table#0#3	0.66	0.0	0.65	0.60	0.40	0.70
table#0#4	0.51	0.07	0.65	0.44	0.75	0.56
table#0#5	0.69	0.24	0.68	0.67	0.44	0.65
table#0#6	0.18	0.0	0.0	0.0	0.0	0.0
table#0#7	0.73	0.10	0.75	0.73	0.37	0.71
table#0#8	0.25	0.05	0.0	0.0	0.0	0.0
smallshelf#1#0	0.35	0.0	0.67	0.54	0.32	0.36
smallshelf#1#1	0.27	0.0	0.82	0.56	0.76	0.73
smallshelf#1#2	0.16	0.0	0.65	0.65	0.45	0.60
smallshelf#2#0	0.56	0.0	0.38	0.1	0.10	0.50
smallshelf#2#1	0.08	0.06	0.60	0.49	0.51	0.57
smallshelf#2#2	0.14	0.22	0.64	0.44	0.41	0.57

Table 6.5: Accuracy, macro F_1-score and Per-Class F_1-scores for Place Classification

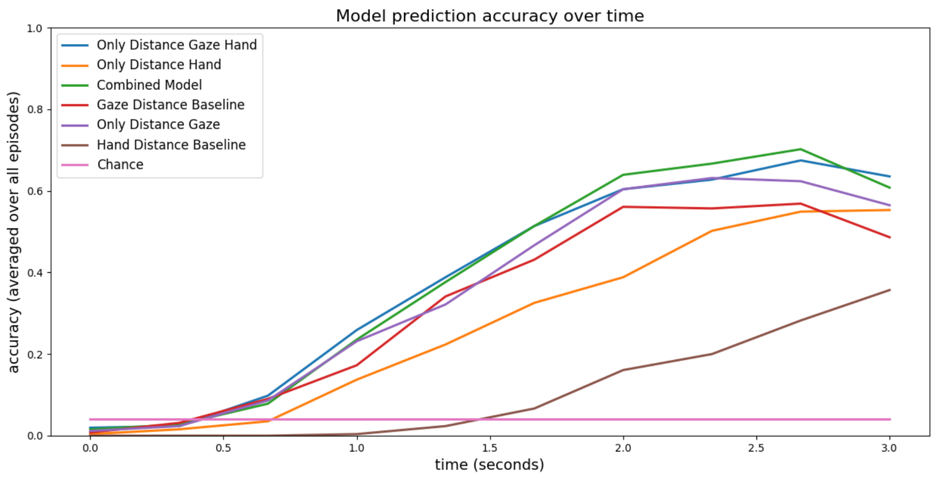


Figure 6.8: Place prediction accuracy of final models over time, averaged over all episodes of the test set, leading up to the actions.

	p_1	p_2	p_3	p_4	p_5	p_6
Combination Model	0.586	0.645	0.609	0.561	0.526	0.602
Only Distance Gaze Hand	0.570	0.619	0.607	0.549	0.478	0.571
Gaze Distance Baseline	0.558	0.604	0.514	0.684	0.476	0.573
Hand Distance Baseline	0.168	0.200	0.161	0.145	0.217	0.186

Table 6.6: Leave-one-out cross-validation of placing prediction accuracy over the recordings of each participant p_x

better than their counterpart from input parameter search with 54%, having the skeleton and object positions in addition to the distances. The macro F_1-score suffered heavily from the places with low representation in validation and training data in case of the corners of the table. Low representation the the validation set in the case of bigshelf#0#1. The models in this case always had a F_1-score of 0. The task for placing something on the table was "set the table". Because of that, the place affordance in this case was to place plates and cups in the middle and on the edges of the table but not in the corners of the table (See Figure 3.4). This probably biased the network towards not predicting objects to be placed in the corners of the table.
The Combined Model has the best performance of the models and baselines. It outperformed the Gaze Distance Baseline by 9% in accuracy and 5% in the macro F_1-score. The Combined Model comparably well compared to the Gaze Distance Baseline on predictions for the small shelf, where the Gaze Distance Baseline mostly had low scores.

The prediction accuracy over time graph for placing can be seen in Figure 6.8. Both Combined Model and Only Distance Gaze Hand model outperform over the whole 3 second time frame that is displayed, with the performance of the models in the first 0.6 seconds being very similar. 1.6 seconds before the end of the episode, the model prediction accuracy of the Combined Model on the validation set is on average already higher than 40%.

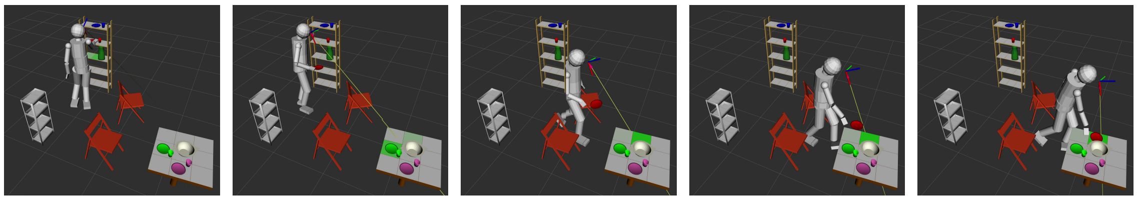


Figure 6.9: Frames from a video visualizing the model prediction value for each place partition

Table 6.6 shows the place results for leave-one-out cross-validation. The data from p_x acts as validation set while the rest is used to train the model. Like for picking, the two participants with the highest disparity of the accuracy of the baseline on their data are p_4 and p_5. The difference in accuracy is 20.8%. Also, column p_4 is the only column, where the Gaze Distance Baseline outperforms Combination Model and Only Distance Gaze Hand Model significantly. This is especially striking, because the difference of 13.5% is very big in this one case. This suggests, that the gaze behavior for the participant was different or that the quality of the eye tracking was especially good for this participant. As the routine for gaze calibration was the same for all participants, the first option is more likely. As participants were wearing eye tracking googles, this could have in influenced them to focus stronger on were they are looking, leading to them directly looking at the object that they want to pick or at the place that they want to place something to. For all columns, except p_4, the Combination Model is outperforming the Gaze Distance Baseline. Excluding p_4, it outperforms the baseline in accuracy on average by 4.86%.

To visualize the the predictions of the model for each episode, a video was recorded displaying the prediction score of each place partition in every time step. The intensity of the green value of the place partition indicates how high its prediction score is. Figure 6.9 shows some frames of this video for a single episode, with the predictions from Combined Model. The person is placing the red plate on the top side of the table. In the first frame, the person is picking the plate and the model predicts places on the big shelf to put the object on. This makes sense, because the human is close to the shelf and the gaze is pointed in the direction of the shelf. In the second frame, the human has turned and is looking in the direction of the table. The network starts predicting the plate to be placed on the left side of the table. As the human is approaching the table in the third frame, the network corrects its prediction to place the plate on the top side of the table. This prediction turns out to be correct, as can be seen in the last frame, where the human is placing the plate on the top side of the table.

7 Discussion

The results from the evaluation show that the usefulness of the gaze varies with the task. For picking, the high accuracy of the baseline shows, that the human is almost always looking at the object, before picking it up. For placing, the accuracy of the baseline was much lower, indicating the human was not always looking at the exact place on the table before placing something there. When placing something on a surface, there is no clear target position that the human can lock the gaze on. For picking this is the case, as the human can lock his gaze to the target object. This could be a possible explanation for the difference in accuracy of the gaze baseline between pick and place.
Some of the models for placing are able to achieve better average accuracy over all episodes averaged than the gaze baseline.
The gaze encoding methods that showed the best results were the distances from the gaze ray and points placed along the gaze ray. Depending on the task, the gaze encoding method that is best may vary. For picking, using distances as gaze encoding was clearly the better choice. For placing, using points along the gaze ray was a better choice than using distances.
The performance of the models using gaze distances only is comparable or even better in some cases than the performance of the networks having the skeleton positions or hand distances as inputs. These results suggest that for achieving good performance on pick and place intent prediction, skeleton based features are not required, if eye gaze and object positions are available. The performance of the baselines and some models over time showed the typical behavior of humans shifting their gaze to the object corresponding to the next task, while still executing the previous task. This may be the cause that leads to a decrease in intent prediction performance at the end of the episode for some models and especially the baseline.

7.1 Outlook/Future Work

There are still many possibilities that can be explored in order to improve intent prediction. One of them would be, to use the video from the perspective of the human that the eye tracking goggles record. The solution that was chosen for enabling untethered movement through the scene (description in Section A.2.2) did not allow to record video from the eye tracking goggles. That is why the dataset recoded for this thesis does not contain video of from the perspective of the human of the scene. With a solution allowing video recording, the video could be used for intent prediction. The output from the eye tracking goggles, when recording video, contains the video images and information, where the human is looking at in the image for every frame. With the gaze being as strong of an indicator for the intent of the human, this could already be enough to achieve good intent prediction results in a pick and place scenario. Having the goggles provide all of the input would improve the practicality of the system greatly by not requiring tracking of the limbs of the human.
Another possibility to improve intent prediction is by using affordance recognition to identify pickable and placeable objects. This could allow the model to adapt to new environments, where the affordances of the objects are unknown. Affordance recognition could also be used to find spaces in the scene that afford placing. For example spaces on a table that are not already occupied. This could be used to find and narrow down possibilities for placing intent prediction.
Using the predicted goal of the human from intent prediction itself, the robot can avoid conflicting with that goal. For example, when the human wants to pick an object up, the robot could avoid blocking the access to that object by holding or taking the object.
Intent prediction could be used in robotics applications alongside other methods from human robot interaction, like motion prediction and path planning, enabling robots to understand and predict actions of humans more precisely. The intent can serve as the goal for the motion of the human. In the case of path planning, the robot could avoid the path, that the human will take, or position itself along the path of the human to hand an object to the human, that will be required to fulfill the goal. Doing motion prediction, more precise information about the movement of the human would be available, for example to realize the robot handing over an object to a human.

A Appendix

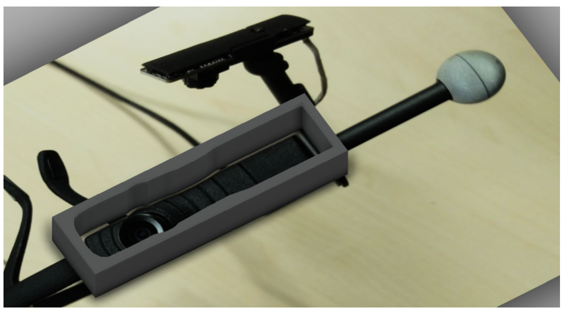
A.1 3D-Printed Attachment for Eye Tracking Goggles

After looking at some recorded data from motion capture, it became clear the markers attached to the goggles were easily occluded when the actor was turning his back to the majority of the motion capturing cameras. Also the motion capture markers were not spaced very far apart in the previous solution, which was attaching the markers directly to the world camera. The solution was to 3D-print an attachment for the world camera of the eye tracking goggles. The markers can be fastened to the attachment. With this solution, some markers are visible form behind the head. The markers are also spaced apart further and are not as much located on a plane as well. The first version of the attachment can be seen it Figure A.1a. The final version that was used to for recording the dataset can be seen in Figure A.1b.

A.2 Eye Gaze Tracking Setup

A.2.1 Goggles Orientation Calibration Procedure

The goggles are tracked by a set of motion capture markers, like the other objects in the motion capture scene. This means that they have some orientation attached to them, that is chosen by the motion capture software. Because the direction of the gaze ray is based on the orientation of the goggles, for the gaze ray to align with the gaze of a person in the motion capture scene it is necessary to find a rotation that accomplishes this. To solve this problem, a python program was written that integrates into the existing skeleton-publisher software. The program captures at some chosen timepoints: a point A in the scene and the gaze end point B, both in the coordinate system


(a) First version with picture reference in CAD

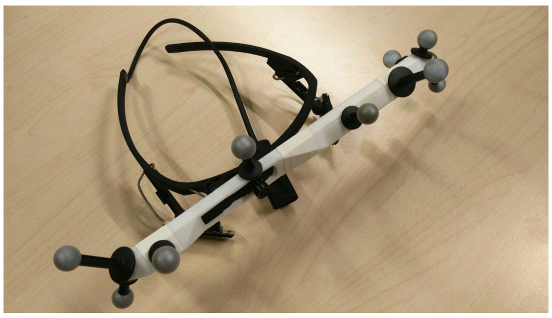

(b) Goggles with final version of attachment

Figure A.1: Two versions of the removable attachment clip for markers to the goggles

of the goggles. It also captures the position of the goggles in the coordinate system of the motion capture scene which is point C. While doing this, the person wearing the eye tracking goggles is looking at point A and moving around in the scene. Now there is a list of points A, B and C. On these points Wahba's Problem can be solved using Singular Value Decomposition. This results in a rotation which can be applied to the orientation of the goggles. Now the gaze ray is aligned with the gaze of the person in the data that was used.

The calculated rotation was integrated with the existing software, such that a calibration procedure can be easily started before the capturing. When capturing, the calibration is then automatically loaded, such that the orientation of the gaze ray is correctly displayed in RViz. When viewing some already captured data, the same applies. When calculating features, the gaze calibration rotation is also automatically applied to the gaze end point, which is used for computation of the gaze features.

A.2.2 Connection of eye tracker to recording computer

The setup had some limitations concerning freedom of movement of the subjects. The goggles that were used to capture the gaze point have a USB-cable attached to them. This cable could get tangled up in scene objects and the legs of the participants and lead to unnatural and restricted movement. Different solutions to this problem were considered and tested. One solution was to attach the goggles to a microcontroller running pupil and sending the eye gaze information over the network. This was tested on the Raspberry Pi 3B running Ubuntu MATE. Looking at low framerates of the live video and the need to buy a more powerful microcontroller, it was decided to go with the Pupil Mobile app. The app runs on Android smartphones and streams world and eye videos to Pupil Capture running on a Desktop Computer in the same network. The goggles connect to the smartphone via USB Type-C to USB Type-C cable. Different smartphones and settings were tested for capturing. Unfortunately the world video footage that was recorded with Pupil Mobile, was always corrupted.

Bibliography

[AS] H. Admoni, S. Srinivasa. “Predicting User Intent Through Eye Gaze for Shared Autonomy”. In: *The 2016 AAAI Fall Symposium Series: Shared Autonomy in Research and Practice Technical Report FS-16-05* (), p. 6 (cit. on p. 14).

[AS17] H. Admoni, B. Scassellati. “Social Eye Gaze in Human-robot Interaction: A Review”. In: *J. Hum.-Robot Interact.* 6.1 (May 2017), pp. 25–63. ISSN: 2163-0364. DOI: 10.5898/JHRI.6.1.Admoni. URL: https://doi.org/10.5898/JHRI.6.1.Admoni (cit. on p. 15).

[BWB08] A. Bauer, D. Wollherr, M. Buss. “Human–Robot Collaboration: A Survey”. In: *International Journal of Humanoid Robotics* 05.1 (Mar. 1, 2008), pp. 47–66. ISSN: 0219-8436. DOI: 10.1142/S0219843608001303. URL: https://doi.org/10.1142/S0219843608001303 (cit. on p. 13).

[CGCB14] J. Chung, C. Gulcehre, K. Cho, Y. Bengio. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *arXiv:1412.3555 [cs]* (Dec. 11, 2014). arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555 (cit. on p. 16).

[Cho15] F. Chollet et al. *Keras*. 2015. URL: https://keras.io/ (cit. on pp. 16, 17).

[CMG+14] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. In: *arXiv:1406.1078 [cs, stat]* (June 3, 2014). arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078 (cit. on p. 16).

[DRT+18] N. F. Duarte, M. Raković, J. Tasevski, M. I. Coco, A. Billard, J. Santos-Victor. “Action Anticipation: Reading the Intentions of Humans and Robots”. In: *IEEE Robotics and Automation Letters* 3.4 (Oct. 2018), pp. 4132–4139. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2861569 (cit. on pp. 14, 15).

[GG16] Y. Gal, Z. Ghahramani. “A Theoretically Grounded Application of Dropout in Recurrent Neural Networks”. In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett. Curran Associates, Inc., 2016, pp. 1019–1027. URL: http://papers.nips.cc/paper/6241-a-theoretically-grounded-application-of-dropout-in-recurrent-neural-networks.pdf (cit. on p. 17).

[GSK+17] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, J. Schmidhuber. “LSTM: A Search Space Odyssey”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (Oct. 2017), pp. 2222–2232. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2582924 (cit. on p. 16).

[HKTM18] J. Hoffmann, P. Kratzer, M. Toussaint, J. Mainprice. “Towards Activity Recognition and Anticipation using Motion Data and Gaze”. In: *Workshop on Robotic Co-workers 4.0: Human Safety and Comfort in Human-Robot Interactive Social Environments 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, Madrid, Spain* (2018), p. 5 (cit. on pp. 13–15).

[HM16] C.-M. Huang, B. Mutlu. “Anticipatory Robot Control for Efficient Human-Robot Collaboration”. In: *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*. HRI ’16. event-place: Christchurch, New Zealand. Piscataway, NJ, USA: IEEE Press, 2016, pp. 83–90. ISBN: 978-1-4673-8370-7. URL: http://dl.acm.org/citation.cfm?id=2906831.2906846 (cit. on pp. 14, 15).

[HS97] S. Hochreiter, J. Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 15).

[JZSS15] A. Jain, A. R. Zamir, S. Savarese, A. Saxena. “Structural-RNN: Deep Learning on Spatio-Temporal Graphs”. In: *arXiv:1511.05298 [cs]* (Nov. 17, 2015). arXiv: 1511.05298. URL: http://arxiv.org/abs/1511.05298 (cit. on p. 15).

[KGS13] H. S. Koppula, R. Gupta, A. Saxena. “Learning human activities and object affordances from rgb-d videos”. In: *The International Journal of Robotics Research* 32.8 (2013), pp. 951–970 (cit. on p. 13).

[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. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2015.2430335 (cit. on p. 13).

[Pre98] L. Prechelt. “Early Stopping - But When?” In: *Neural Networks: Tricks of the Trade*. Ed. by G. B. Orr, K.-R. Müller. Red. by G. Goos, J. Hartmanis, J. van Leeuwen. Vol. 1524. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 55–69. ISBN: 978-3-540-65311-0 978-3-540-49430-0. DOI: 10.1007/3-540-49430-8_3. URL: http://link.springer.com/10.1007/3-540-49430-8_3 (cit. on p. 16).

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958 (cit. on p. 17).

[TY17] A. M. Truong, A. Yoshitaka. “Structured LSTM for human-object interaction detection and anticipation”. In: *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). Aug. 2017, pp. 1–6. DOI: 10.1109/AVSS.2017.8078543 (cit. on p. 15).

All links were last followed on October 23, 2019.

Declaration