CHALMERS
UNIVERSITY OF TECHNOLOGY

Intended Human Arm Movement Direction Prediction using Eye Tracking

Downloaded from: https://research.chalmers.se, 2023-10-28 14:03 UTC

Citation for the original published paper (version of record):
Pettersson, J., Falkman, P. (2023). Intended Human Arm Movement Direction Prediction using Eye Tracking. International Journal of Computer Integrated Manufacturing, In Press.
http://dx.doi.org/10.1080/0951192X.2023.2229288

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology.
It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004.
research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

International Journal of Computer Integrated Manufacturing

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tcim20

Intended Human Arm Movement Direction Prediction using Eye Tracking

Julius Pettersson & Petter Falkman

To cite this article: Julius Pettersson & Petter Falkman (2023): Intended Human Arm Movement Direction Prediction using Eye Tracking, International Journal of Computer Integrated Manufacturing, DOI: 10.1080/0951192X.2023.2229288

To link to this article: https://doi.org/10.1080/0951192X.2023.2229288

© 2023 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

Published online: 29 Jun 2023.

Submit your article to this journal

Article views: 61

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=tcim20

INTERNATIONAL JOURNAL OF COMPUTER INTEGRATED MANUFACTURING
https://doi.org/10.1080/0951192X.2023.2229288

Taylor & Francis
Taylor & Francis Group

OPEN ACCESS Check for updates

Intended Human Arm Movement Direction Prediction using Eye Tracking

Julius Pettersson and Petter Falkman

Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden

ABSTRACT

Collaborative robots are becoming increasingly popular in industries, providing flexibility and increased productivity for complex tasks. However, the robots are still not interactive enough since they cannot yet interpret humans and adapt to their behaviour, mainly due to limited sensory input. Prediction of human movement intentions could be one way to improve these robots. This paper presents a system that uses a recurrent neural network to predict the intended human arm movement direction, solely based on eye gaze, utilizing the notion of uncertainty to determine whether to trust a prediction or not. The network was trained with eye tracking data gathered using a virtual reality environment. The presented deep learning solution makes predictions on continuously incoming data and reaches an accuracy of 70.7%, for predictions with high certainty, and correctly classifies 67.89% of the movements at least once. The movements are, in 99% of the cases, correctly predicted the first time, before the hand reaches the target and more than 24% ahead of time in 75% of the cases. This means that a robot could receive warnings regarding in which direction an operator is likely to move and adjust its behaviour accordingly.

ARTICLE HISTORY
Received 8 April 2022
Accepted 28 May 2023

KEYWORDS
Virtual reality (VR); Recurrent neural networks (RNN); Human-robot collaboration (HRC); human intention prediction; eye tracking

1. Introduction

Collaborative robots are becoming increasingly popular in industries (Makrini et al. 2017). The advantages of having humans and robots in the same workspace interacting with each other are many, such as improved flexibility (Krüger, Lien, and Verl 2009) and increased productivity for complex tasks (Krüger, Lien, and Verl 2009). However, the robots are not interactive enough since they cannot interpret humans and adapt to their swift changes in behaviour in a way that another human would. The main reason is that the collaborative robots today are limited in their sensory input, which makes the human responsible for staying out of the way.

Human intention prediction can be achieved using camera images and probabilistic state machines (Awais and Henrich 2010) with the goal of determining between explicit and implicit intent. It can also be achieved using 3D-vision, speech recognition, and wearable sensors with the objective of predicting intention in hand-over tasks (Wang et al. 2022). It was proposed by Mainprice and Berenson (2013) to use a Gaussian Mixture Model and data from a Kinect camera to predict human motion, reporting about 80% classification accuracy, on eight movement classes, after 60% of the trajectory has been observed. Other ways are to monitor eye gaze to predict an upcoming decision (Huang and Mutlu 2016) for robot control or analyze bioelectric signals, such as electromyography, to predict human motion (Luzheng et al. 2019). In the paper by Haji Fathaliyan, Wang et al. (2018) it is shown that eye gaze can be used to recognize actions related to pouring and mixing a powder-based drink. Shi, Copot, and Vanlanduit (2021) presents a way of using Earth Mover's Distance to calculate the similarity score between the hypothetical gazes at objects and the actual gazes to determine if the human visual intention is on the object or not. It was shown by Ravichandar, Harish, and Dani (2016) that is possible to use a Kinect camera to capture eye gaze and arm movements, and use that to predict the goal location of a reaching motion, reporting a success rate of above 80% after 40% of the trajectory has been observed. The work by Gomez Cubero and Rehm

CONTACT Julius Pettersson pjulius@chalmers.se Department of Electrical Engineering, Chalmers University of Technology, Gothenburg 412 96, Sweden

According to the Swedish Ethics Review Act (2003:460) this research is exempt.

© 2023 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

(2021) shows that it is possible to use an LSTM-based neural network, together with a wearable eye tracker, to predict intention regarding which object is about to be picked out of three objects in a virtual reality environment. They achieve an accuracy between 70% and 80% for test sequences that are 3–14 s long using the gaze projected on the surface where the objects are placed.

Other fields that have been rapidly expanding and could make collaborative robots smarter through an understanding of the operators behaviour and intentions are; virtual reality, eye tracking, gathering and management of large datasets, and artificial intelligence.

Eye-tracking (ET) is an objective, painless, and noninvasive (Gould et al. 2001) way to gather more insight into how a person is reasoning from measurements and analysis of where the person is directing their gaze (Karatekin 2007). It is possible to gain insight into the alternatives a person is considering or what strategy is used while performing a task, based on what a person is looking at. There are three types of interesting eye movements when observing visual attention: fixation, saccades, and smooth pursuits (Duchowski 2017). ET has, for example, been used in an industrial context with gaze as machine control input (Jungwirth et al. 2018), to evaluate new ways to facilitate human – robot communication (Tang, Webb, and Thrower 2019), analyze the navigational intent in humans and how they interact with autonomous forklifts (Chadalavada et al. 2020), and investigate pedestrians' understanding of an autonomous vehicle's intention to stop at a simulated road crossing (Hochman, Parmet, and Oron-Gilad 2020).

Virtual Reality (VR) can be described as a technology through which visual, audible, and haptic stimuli is able to give the user a real-world experience in a virtual environment (Dahl et al. 2017). Benefits such as being able to provide more relevant content and present it in a suitable context (Rizzo et al. 2004) are reasons to promote the use of VR. It can, for example, be used when making prototypes (Abidi et al. 2016), to train operators in assembly (Al-Ahmari et al. 2016), and improve remote maintenance (Aschenbrenner et al. 2016).

VR makes it possible to have an all-in-one system for the gathering of movement and interaction data where the developer has full control over the data and has the ability to add or remove visual and audible distractions. VR also removes the risk of injuries when the user interacts with industrial equipment in the VR environment (VRE).

The use of modern technologies such as ET and VR, therefore, make it possible to collect larger amounts of data, with higher accuracy, and at a higher pace than before (Pettersson et al. 2018). These large volumes of data, created at high speed, and with great variety (Andrew et al. 2012), is referred to as Big Data. One way to process this data is through the use of an area of artificial intelligence called deep machine learning (Samek, Wiegand, and Müller 2017). Big data and artificial intelligence has been shown to be important tools to improve industrial manufacturing (Morariu et al. 2018; Nagorny et al. 2017; Wang et al. 2018).

Combining these areas to increase the intelligence of the collaborative robots can be broken down into the following three stages:

S1: *Movement Direction Classification*

Deep machine learning requires large amount of data to train the neural networks. The first step is therefore to create a virtual, measurable environment that is capable of gathering all the necessary data. The environment has to limit distractions and ambiguous stages to ensure that it is possible to evaluate any results and draw conclusions using domain knowledge. The end goal of the first stage is to be able to classify the movement direction of the test participant upon completion of the test. A solution to this stage has been provided by Pettersson and Falkman (2020).

S2: *Fixed Movement Direction Prediction*

The goal of the second stage is to be able to predict the human movement direction angle in the horizontal plane, ahead of time, based solely on a set of

historical gaze data. This was achieved by Pettersson and Falkman (2021) at 62.50% accuracy for a *fixed* timestep ahead of ~500 ms.

S3: *Dynamic Movement Intention Prediction*

Finally, the third stage is to be able to predict the intended movement direction of a test participant ahead of time in a *dynamic* online fashion. This means that the system should be able to handle a continuous stream of data along with being able to make predictions on movement sequences of different length. In this stage it is important to incorporate uncertainty estimation regarding the network's performance in order to later be able to safely utilize the intended functionality in collaborative manufacturing. The implementation of a real-world application could later be achieved using the same ET technology mounted to the safety glasses that the operator already wears.

The contribution of this paper is a system that reaches one important milestone in the work towards a solution to the third stage (**S3**). It is achieved through the development of a recurrent neural network that is able to continuously predict the intended human arm movement direction, solely based on eye gaze, ahead of the arm movement being completed. The developed neural network provides estimates of its own uncertainty, which is used to determine whether a prediction should be trusted or not. The goal of predicting the direction that a human is about to move/is moving their hand was separated into the following two objectives:

- **Primary** - the main goal is to determine the discrete horizontal direction corresponding to the box that was clicked,
- **Secondary** - the secondary objective is to distinguish between whether the movement occurred on the upper or lower level of boxes.

This work is based on adjustments and improvements of the VR test case presented in (Pettersson and Falkman 2021), and the collection of new movement data that was used to train the artificial neural network.

2. Preliminaries

This section provides brief descriptions of convolutional neural networks, recurrent neural networks, and dropout as a Bayesian approximation, that are used to perform the human intention prediction.

Convolutional neural networks (CNNs) are one type of artificial neural networks (ANNs) that are more robust to shift, scale, and distortion invariance (LeCun et al. 1998) than fully connected (FC) networks, and are therefore better at detecting spatial and temporal features. It is achieved by convolving or sub-sampling the input to the layer with local receptive fields (LeCun et al. 1998) (filters) of a given size [n x m]. Each filter has n number of trainable weights + a trainable bias and these are shared (LeCun et al. 1998) for all outputs of the filter.

Recurrent neural networks (RNNs) are a subgroup of ANNs that are used to process sequences of data (Goodfellow, Bengio, and Courville 2016). An RNN shares its weights across several timesteps (Goodfellow, Bengio, and Courville 2016) whereas a FC network would have separate weights for each part of a sequence. In an RNN, the current step is not only a function of its input but also depends on all the output states previous in time (Goodfellow, Bengio, and Courville 2016). Traditional RNNs tend to suffer from problems with exploding or vanishing error gradients (Goodfellow, Bengio, and Courville 2016; Hochreiter and Schmidhuber 1997) that prohibits proper learning over longer time instances. Long Short-Term Memory (LSTM) cells (Hochreiter and Schmidhuber 1997) are designed to solve this problem using a constant error flow (Hochreiter and Schmidhuber 1997) the network, together with three gates that open and close in order to access it (Hochreiter and Schmidhuber 1997). The input gate determines when the internal state of the LSTM cell is affected by the input to the cell, the forget gate handles when the cell's internal memory resets, and the output gate controls when the current state of the cell influences the error flow (Hochreiter and Schmidhuber 1997). An LSTM network may contain multiple cells and the network learns to control each individual gate (Hochreiter and Schmidhuber 1997) and cell.

Dropout is a deep machine learning method that is used to reduce overfitting (Hinton et al. 2012) by randomly ignoring, with probability p, each neuron

in a network layer every time a training case is presented to the network. The dropout method can be used to approximate Bayesian inference (Gal and Ghahramani 2016). It is achieved by enabling dropout at all times, not only during the training of the network, which means that the network will also randomly omit some neurons when making predictions, causing variation. The mean prediction and the model uncertainty can, according to Gal and Ghahramani (2016), be obtained by making N number of predictions on the same data and they suggest that $N \in [10, 1000]$ should give reasonable results. This way of using dropout provides a way to reason about model uncertainty that is easy to implement and less computationally expensive than alternative methods (Gal and Ghahramani 2016). They suggest using a probability $p \in [0.1, 0.5]$ for dropping a neuron.

3. Experimental setup

This section will present the development of the VR environment, the test execution, the gathered data, and the way the data was processed, including selection of features and labels to train the ANN with.

3.1. Development of the VR test environment

The VR environment (VRE) will be visualized using the HMD and the two hand-held controllers that are part of the '*Tobii Eye Tracking VR Devkit*' (Tobii 2020), which is an ET solution based on the HTC-Vive. The system is capable of tracking the position and orientation of the HMD and the hand held controllers. The eye gaze is tracked with *Binocular dark pupil tracking* at a frequency of 120 Hz. This type of eye-tracking is achieved by illuminating the eyes, off-axis compared to the cameras that are used to capture images of the reflected light as it bounces off the retina and exits the eye, causing the pupil to appear darker than the rest of the eye. The images are used to calculate a gaze direction vector based on the positional relationship between the cornea and the pupil. The ET can be performed in the entire 110° field of view of the HTC-Vive HMD (Tobii 2020), with an accuracy of ~0.5° and a delay of ~10 ms from the illumination of the eye until the data is available in the SDK. The eye tracker is individually calibrated to each test participant using a 5-point calibration strategy available in the SDK. The calibration is based on that the user is instructed, visually and audibly, to focus her/his gaze on 5 pre-defined points in the VRE and that gaze data is used in the SDK to calculate a 3D-model of the eye.

The 3D components in the project are modelled in the software Blender and implemented in a VRE using Unity, a game creation engine. Unity supports VR through Steam VR SDK and custom written scripts in C# that makes it possible to implement all the desired functionality from the SDK as well as the intended test logic.

The VRE designed to collect the data consists of four stages: language selection where the test participant selects whether the written instructions in the VRE should be given in Swedish or English, ET calibration, an information form where the participant enters age, gender, and whether they are right handed or not, and the last stage is the test itself. The test stage, Figure 1, is an alteration of the test in (Pettersson and Falkman 2021).

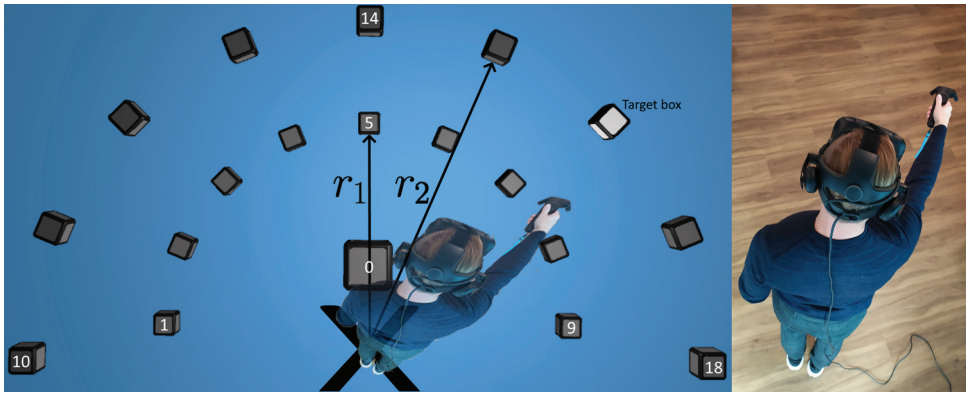

Figure 1. A top-down view of the block placements, the radii r_1, r_2, a lit target box and the test participant reaching for the target. Note that the human is not actually visible in the VRE.

The test starts with an even distribution of nine cubes each, at two different heights and radii, which allows for a wide range of movements in different directions. One additional box, #0, is positioned 30 cm in front of the participant at height h_0 and acts as a neutral position close to the body. The height levels (h_0, h_1, h_2), measured from the floor, and radii (r_1, r_2) are individually adjusted to each test person based upon a calibration procedure using the two controllers. The participant is instructed to raise their hands forward in three steps and click the touchpads at these locations to collect the different controller positions. The first step is to stand still with the head pointing forwards and the arms resting along the sides of the body, the second step is to raise the forearms to a horizontal level, pointing forwards, while keeping the elbows fixed against the sides of the body, and the third one is to fully extend the arms and raise them to a horizontal level, pointing forwards. The heights, h_1 for the inner cubes and h_2 for the outer ones, are calculated as the average distance to the floor from the controllers for the second and third position whereas $h_0 = h_1 - 0.1m$. The radii, r_1 for the inner semicircle and r_2 for the outer one, are defined as the average distance along the z-axis between the controllers and the HMD, as seen in Figure 1 where the black *X* corresponds to the HMD position. Note that the human is not actually visible in the VRE and that the participant's only point of reference to their own body is the controller.

Each test starts with a warm-up sequence of movements in random directions in order to make the participant accustomed to the VRE. The warm-up is followed by a sequence of 76 movements using the right hand and 76 movements using the left hand. The sequence is randomized for everyone in a way that ensures balanced data and that all combinations are used.

The cubes are lit up one at a time, marked as the target box in Figure 1, and the task is to reach for the box that is lit and touch it while simultaneously pressing the touchpad on the controller to make the cube disappear. After a cube has disappeared, and a delay of 0.2 s, the next cube in the pre-defined sequence is lit. The delay is used as a way to force a slower pace throughout the test and data is collected during this time. The alterations of the test are:

- The previous test was considered too long and strenuous by the participants and was, therefore, halved in length, i.e. fewer number of movements in total.
- The test sequence was randomized as suggested in future improvements.
- Data is now collected during the short delays between the cubes appearing, this is necessary in order to be able to use and evaluate the developed network in a continuous manner that imitates a real-world system where perfect segments of data rarely are available.
- The sweeping motions were excluded due to the fact that the eye-hand movement connection is very different from the search and click behaviour that the random cubes induce.

The test is launched when the test participant presses the start button in the environment. Data is then collected, in the same manner as in (Pettersson and Falkman 2020, 2021), during the time between two pressed cubes and saved as one data sample. The data that is collected from each test participant, each test, and at each timestamp shown in Table 1 are: the eye gaze direction vector for each eye (EyeDirection), the coordinate in the virtual room where the gaze hits (EyeHitpoint), which object is gazed upon (EyeHitObject) as well as the size and position of the pupils (PupilDiameter, Pupilposition). The head specific data that is collected are the position (HeadPosition) and rotation (HeadRotation), and the same data is also obtained from the two controllers that are held one in each hand (ControllerPosition, ControllerRotation). The general information about the user includes an anonymous participant ID, age, gender, language used, whether the person is right handed or not, as well as the date and time when the data was gathered.

3.2. Description of test execution

The data was collected in the VR-area of a laboratory at Chalmers University of Technology in Gothenburg. All test participants were given the same instructions regarding putting on the headset, calibrating the eye-tracker, entering the required information, perform-

ing the height and reach calibration, starting the test, and the test specific instructions. The full set of instructions are as follows:

Calibration instructions

(1) Put on the headset and adjust it such that the displays are centered in front of the eyes.
(2) Receive a controller in each hand. The controllers are used to navigate the menus (using the laser pointer), touch the cubes during the test, and acknowledge all actions using the click function of the touchpad.
(3) Now it is time to:
  (a) Choose the desired language, either Swedish or English by clicking the corresponding flag using the laser pointer.
  (b) Calibrate the eye-tracker:
    (i) Stand still with your head pointing forwards.
    (ii) Press 'Calibrate' using the laser pointer.
    (iii) Focus your gaze on the red dots that appear on the screen until they disappear, without moving your head.
    (iv) Press 'Done' when the calibration is complete.
  (c) Enter the required information using the laser pointer.
  (d) Calibrate the height and reach parameters:
    (i) Stand still with the head pointing forwards and the arms resting along the side of the body, then click the touchpad on the right controller.
    (ii) Raise the right forearm to a horizontal level, pointing forwards, while keeping the elbow fixed against the side of the body then click the touchpad on the right controller.
    (iii) Extend the right arm fully and raise it to a horizontal level, pointing forwards, then click the touchpad on the right controller.
    (iv) Repeat steps 3.d.i-3.d.iii for the left arm.
    (v) Press 'Done' when the calibration is complete or 'Re-calibrate' to start over if something went wrong.

Test instructions

(1) Press the 'Start'-button on the screen by reaching towards it and touching it.
(2) (Reach for the cube that is lit up and touch it, press the touchpad while doing so to acknowledge the completion of the movement.
(3) Wait for the next cube to light up.
(4) Repeat steps 2 & 3 until the cubes stop emitting light.
(5) Press the 'Done'-button and remove the headset.

3.3. *The obtained dataset*

The dataset consists of 3192 data points obtained from 21 participants, collected at Chalmers University of Technology. The data consists of a majority of younger adults with a higher level of education. The gender distribution of the collected data is 24% female, 76% male and 0% other. The variations in age ranged from the youngest

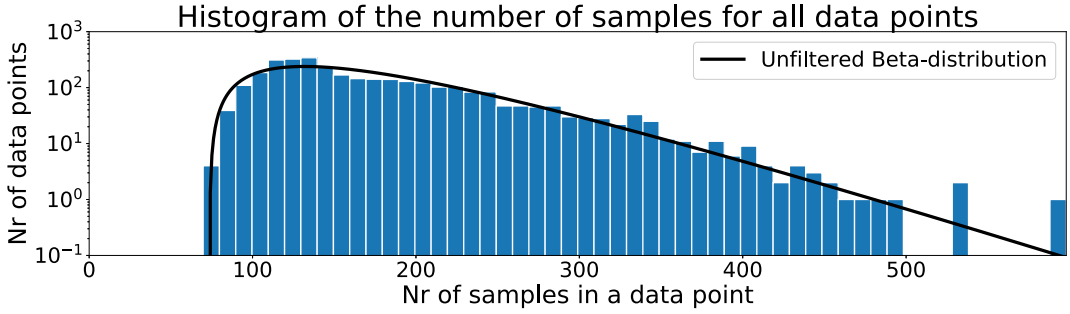

Figure 2. Histogram of the number of samples for all data points.

Table 1. Table of data parameters collected during the test.

Type	Parameter
Participant	ID
Participant	Age
Participant	Gender
Participant	Date & time
Participant	LanguageIsEnglish
Participant	IsRightHanded
Test specific	BoxClicked
Test specific	Timestamp
ET	EyeDirection [x, y, z], (left, right)
ET	EyeHitpoint [x, y, z], (left, right)
ET	EyeHitObject (left, right)
ET	PupilPosition (left, right)
ET	PupilDiameter (left, right)
HMD	HeadPosition [x, y, z]
HMD	HeadRotation [x, y, z]
Controllers	ControllerPosition [x, y, z], (left, right)
Controllers	ControllerRotation [x, y, z], (left, right)

participant being 23 and the oldest 30 years old with an average age of 26.

3.4. Filtering of the data

The same procedure for filtering as in (Pettersson and Falkman 2020, 2021) was applied in this paper and starts with the data from the tests being loaded into the computer memory from previous storage in files on the harddrive. The warm-up sequences, described in Section 3.1, were discarded before the visual inspection of the histogram in Figure 2 that shows the length of all data points, i.e. how many data samples that were collected from the time a box was lit until it was destroyed. It can be seen that the dataset contains a few outliers and the shape of the data in the histogram can be approximated using a Beta distribution, indicated by the black solid line in the figure. A threshold was set to the mean (μ) plus three standard deviations (σ) of this distribution, as suggested by Pettersson and Falkman (2020), with the values from Table 2, such that a maximum of $\mu + 3\sigma = 175 + 3 \cdot 66 = 373$ data samples was allowed for the data point to be used in the classification.

The next step removed all samples, within each data point, that contained NaN values and replaced these with the gaze vector from the previous sample. NaN values appear when the ET fails to read the eye properly, the most common cause, according to Pettersson and Falkman (2020), being due to the participant blinking.

3.5. Selection of features and labels

The features, shown in Table 3, that were used as input to the network are the combined eye gaze direction vector (x, y, z), obtained as an average of the separate gaze vectors from each eye, the *y*- and *z*-coordinates of the HeadPosition, and the pupil diameter, averaged between left and right eye. The *x*-coordinate of the HeadPosition was removed as it corresponds to the participant's height, which is constant during the entire duration of the test due to the fact that they remain standing and does, therefore, not provide much information to the network since the boxes are individually calibrated to the participant's height and reach. The HeadRotations were removed since the focus point of the gaze is more interesting and because of the fact that the head is often rotated in conjunction with the eyes, therefore, providing limited information to the network. The reason that information such as EyeHitPoint and EyeHitObject are not used is because they require specific knowledge of all objects in the environment, something that is possible to know in a VRE but would limit the possibility to implement the system in a real-world scenario.

The boxes from 10 to 18 were re-labeled as 1–9 and coupled with a boolean, IsUpperLevel, that is set to one for these and zero for all others in order to fit the primary and secondary classification objectives described in Section 1.

The ID, age, and gender were used to manage the dataset as well as to provide some general information, these were however not used to train the network.

3.6. Preprocessing of the data

The selected features mentioned in the previous section, Table 3, were feature-wise normalized between $[-1, 1]$ in order to make sure that different value of magnitude between features does not bias the

Table 2. Distribution information for unfiltered and filtered data.

Type	Mean	Median	σ	Min	Max	N
Unfiltered	175	152	66	74	598	3192
β-filtered	171	151	62	74	371	3141

Table 3. Description of data used in the classification.

Type	Feature
Input	AvgRLEyeDirection $[x, y, z]$
Input	AvgRLPupilDiameter
Input	HeadPosition $[y, z]$
Label	BoxClicked $(0-9)$
Label	IsUpperLevel $(0, 1)$

network to emphasize the importance of one feature over another.

The training data was augmented by appending N_T nr of additional copies of the training data, where the ordering of the different movements in each new copy was shuffled to provide a new set of transitions between movements. The number of copies to use is investigated in Section 5. No noise was added to the data augmentation based on the fact that noise tends to worsen the learning of an LSTM network (Greff et al. 2016).

After augmentation, the data was transformed using a sliding window of size $w = 350$ and step $N_s = 70$, where every window in turn was split into $N_w = 14$ nr of subwindows of size $w_s = 25$ samples.

The window size, $w = 350$, i.e. the number of historical gaze data samples used as network input, was set to two times the mean length from Table 2 in order to have a window that most likely spans across two different movements and thereby captures at least one transition between movements. The subwindow size, was set to the same window size as in (Pettersson and Falkman 2021) and it was chosen based on the fact that the window should be able to capture entire saccadic eye movements that last, 10–100 ms (Duchowski 2017).

The amount of data and its quality is also dependent on the step size, N_s, that is used when sweeping over the training data. This means the number of timesteps that are skipped before the next time

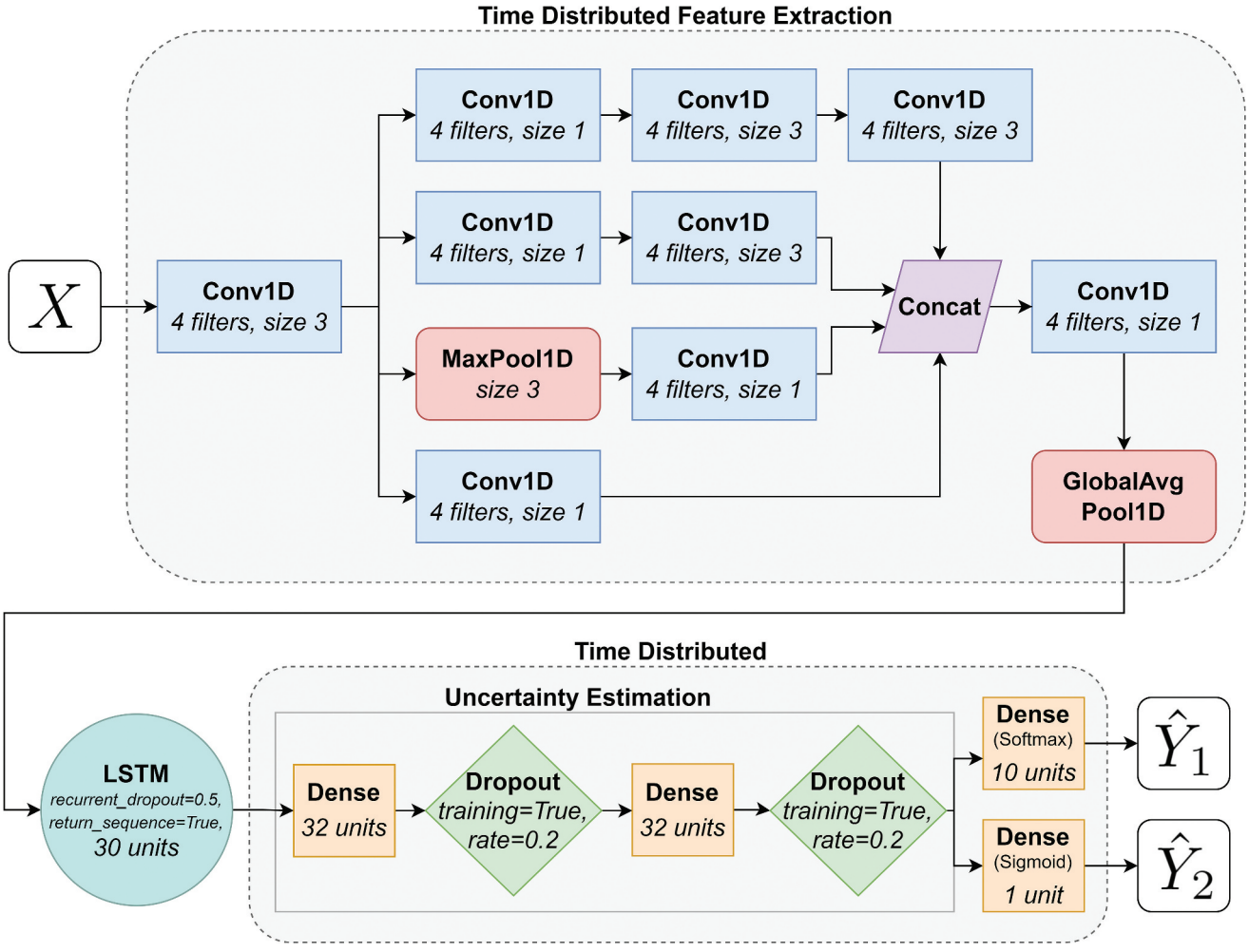

Figure 3. An overview of the RNN architecture.

window is picked. The value for N_s should be large in order to provide the network with as unique information as possible, however, in order to not accidentally skip any movements it should not exceed the shortest movement, i.e. 73 samples. It was, therefore, set to the largest even multiple of $w = 350$ that satisfies this criteria, namely $N_s = 70$.

Every subwindow is coupled with a label corresponding to the correct box/vertical position of the last sample of the subwindow. This means that each window, $w = 350$, is divided into $N_w = 14$ subwindows that each get their own label and that the network makes $N_w = 14$ separate predictions, for each $w = 350$, regarding the class that the subwindow belongs to.

The proportions of the data splits are; 45% of the data for training, 5% for validation, and the remaining 50% was used for testing and evaluation of the network.

4. Neural network design

The recurrent neural network used in this paper, Figure 3, was built as follows: the network takes the input X and feeds it to a TimeDistributed and slightly modified version of the feature extraction that was used in (Pettersson and Falkman 2020). The difference being the addition of an LSTM layer, adapting (adding time distribution) the network to work with the LSTM, and the AvgPooling1D followed by a flattening was replaced with a single GlobalAvgPooling1D that achieved a similar outcome, extracting the most important features and reducing its dimensions, in a single layer. TimeDistributed refers to the fact that the parts of the network encircled by the dotted grey lines are applied to each timestep of the recurrent network.

The layout of the feature extraction uses the Conv1D-layers, blue rectangles in the figure, as the base for extracting information from the data and the design itself was inspired by the inception modules from Inception-v3 (Szegedy et al. 2016). The Conv1D-layers analyze the time-dependency between a few nearby data samples across all features. The idea behind the inception modules, according to Pettersson and Falkman (2020), was to calculate convolutional features with several filter sizes in parallel while at the same time reducing the number of parameters and complexity of the network compared to using a fully connected (FC) neural network.

The network also contains pooling layers (red rounded rectangles) that reduce the spatial size of the data fed to them, only keeping the most significant parts. This further reduces the number of parameters in the network (Pettersson and Falkman 2020). The purple parallelogram manages the dimensions of the network connections through concatenation along the last dimension of the layers connected to it. The TimeDistributed outputs from the global pooling layer are fed to an LSTM layer (teal circle, *return_sequence=True* means that all timesteps are used as output and *recurrent_dropout = 0.5* refers to dropout being applied recurrently with probability 0.5) that manages the time aspect of the network and thereby gives it its recurrent properties.

The last part of the network are the TimeDistributed uncertainty estimation (UE) (Gal and Ghahramani 2016) implemented as follows: two dense layers (yellow squares, corresponding to FC layers) each followed by a dropout layer (green diamonds, *training=True* means that it is used also when making predictions). The UE is followed by two TimeDistributed final dense layers, one with as many neurons as there are output classes (10) that gives the output $\hat{Y}_1$ and one with a single neuron that gives the binary output for $\hat{Y}_2$. The outputs $\hat{Y}_1$ and $\hat{Y}_2$ are obtained once for each timestep.

All layers of the network use the tanh activation function apart from the final dense layers that use a softmax activation to enable multi-class classification ($\hat{Y}_1$) and sigmoid activation to enable binary classification ($\hat{Y}_2$). The network was trained using the adam optimizer (Kingma and Jimmy 2014), sparse categorical crossentropy as the loss function for $\hat{Y}_1$, and binary crossentropy as the loss function for $\hat{Y}_2$. The training was performed until the validation loss stopped decreasing, terminating using early stopping. The final network configuration contains 20 layers with 8550 trainable parameters in total.

4.1. Evaluation procedure

This section provides a description of how predictions are made with UE (Gal and Ghahramani 2016), what metrics that were used to evaluate the network and how to interpret them, and a brief description of how to simulate making predictions on a continuous flow of data.

The difference when making a prediction with UE is that several predictions are made on the same data in order to obtain a mean value and a standard

deviation of the prediction. The pseudo code for this is shown in Algorithm 1 (Pettersson and Falkman 2020).

Once the means and standard deviations have been obtained from the network, these can be used to determine the network's confidence, high mean and low standard deviation, to make an accurate prediction. The implementation is shown in Algorithm 2 (Pettersson and Falkman 2020), where a prediction is accepted if the mean minus two standard deviations is larger than a chosen limit. One way of choosing this limit will be covered in Section 5.

Algorithm 1 Pseudo code for predicting with UE.

Input: X, nrOfPredictions
Output: $\hat{Y}$, $\hat{Y}_{STD}$
predictions = []
for $i = 0$ to nrOfPredictions **do**
  predictions[i] = model.predict(X)
end for
$\hat{Y}$, $\hat{Y}_{STD}$ = mean(predictions), std(predictions)
return $\hat{Y}$, $\hat{Y}_{STD}$

Algorithm 2 Pseudo code that accepts or discards a prediction.

Input: $\hat{Y}$, $\hat{Y}_{STD}$, lowerLimit
Output: $\hat{Y}$
if $\hat{Y} - 2 * \hat{Y}_{STD} >$ lowerLimit **then**
  Accept $\hat{Y}$ as the prediction for this sample.
else
  Discard $\hat{Y}$, the network is not confident enough.
end if

The performance of the networks was evaluated using the following custom metrics;

- A_P = Accuracy of predictions that are above UE threshold,
- A_M = Accuracy of how many movements are correctly identified at least once,
- M_T = Mean fraction of time left until the completion of the task,
- A_{VP} = Vertical accuracy, evaluated whenever there is a box prediction.

These were considered more suitable to use to evaluate the network on how well it is able to utilize its notion of UE in order to predict the intended movement direction, compared to a standard accuracy metric that does not capture the aspect of UE at all.

The evaluations have been performed in a way that imitates the continuous flow of data in a real world system. This is done by making a prediction for every timestep of the test set, starting with all zeros as the input and then shifting the input data by one at a time in order to 'obtain' new information. The last of the N_w predictions (the last subwindow) at each timestep is the one that is evaluated since that subwindow contains the most recent data.

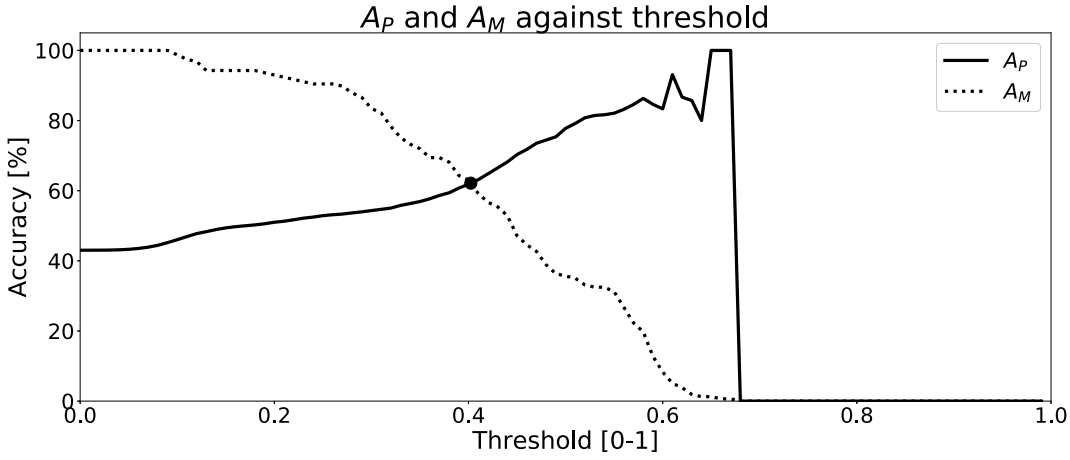

Figure 4. A figure showing the selection of the optimal threshold for the first network configuration.

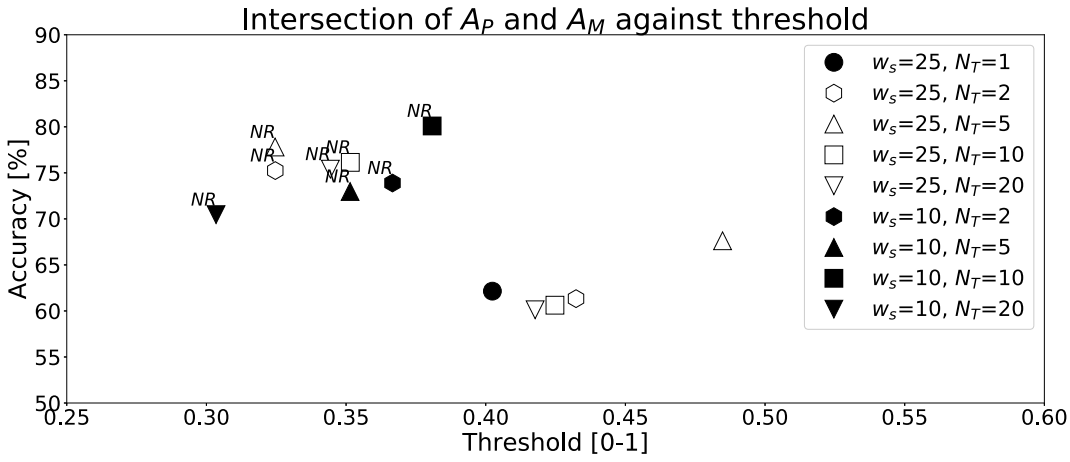

Figure 5. A figure showing the comparison of different network configurations, using the intersections of the metrics A_P and A_M. *NR* stands for not using randomized copies of training data.

5. Results

This section will provide the prediction results, on the validation set and the test set. The first network configuration that was evaluated, on the validation set, was $w = 350$ and $N_T = 1$, the results can be seen in Figure 4. The validation set contains 27,323 timesteps in total and the performance was evaluated using $n = 25$ for the UE. The intersection point between the two metrics, A_P and A_M, can be seen as the networks optimal performance since the accuracy for the predictions are high while at the same time, as many movements are covered as possible. This is, therefore, the point that specifies the threshold, Th_L, that is used to further evaluate the network. Due to the fact that

Table 4. Table showing a performance comparison between the first network configuration and the best model, evaluated on the test set.

First and best model – Test Set						
w_s	N_T	Th_L	A_P	A_M	M_T	A_{VP}
25	1	0.40	54.18%	50.64%	32.12%	70.38%
10	10	0.38	70.70%	67.89%	37.20%	81.29%

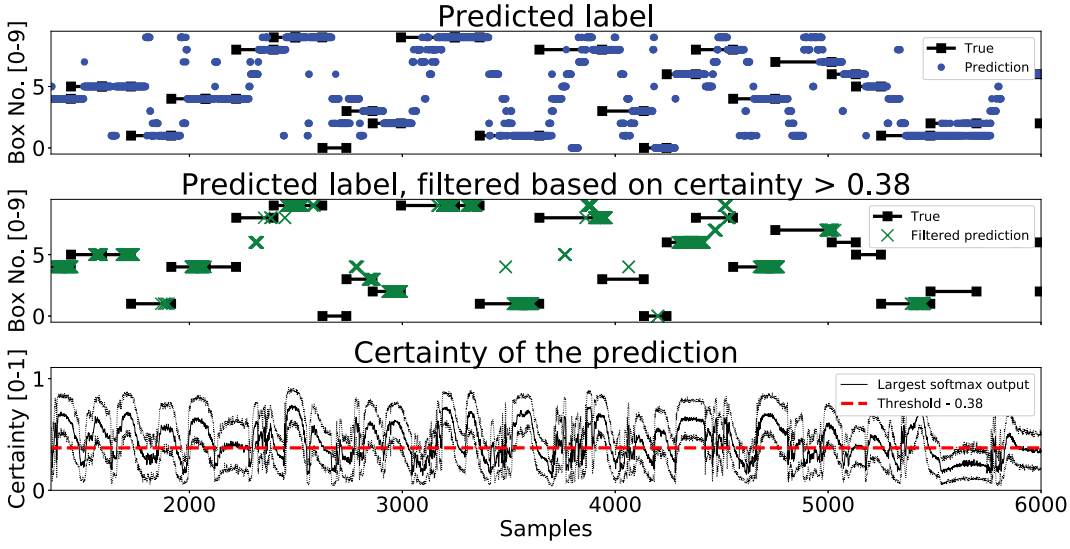

Figure 6. A figure that shows a prediction segment from the network configuration, $w_s = 10$ and $N_T = 10$ with *NR*, obtained on the test set.

predictions are filtered out based on an increasing threshold, once it reaches above the highest certainty of the network, all predictions will be filtered out i.e. no predictions are made, and the accuracy therefore goes to zero, as seen in Figure 4.

Different configurations were evaluated in order to determine the effect of changing the subwindow size w_s, the number of training copies N_T, and whether the order of the movements in the augmented data should be randomized or not (*NR*=No Randomization). The comparison between these is based on the intersection points, described above, shown in Figure 5 where the black dot corresponds to the original configuration. It is clearly seen that changing N_T does not improve the results unless *NR* is used as well. The best performing combination is $w_s = 10$ and $N_T = 10$ with *NR*. Different values for w, [175, 700, 1400], were also tested but these showed no significant improvement.

The original configuration, $w = 350$ and $N_T = 1$, and the best performing one from Figure 5, $w_s = 10$ and $N_T = 10$ with *NR*, were further evaluated on the test set. The test set consists of 250,660 timesteps in total and the performance was evaluated using $n = 25$ for the UE. The resulting comparison, using each networks optimal threshold obtained on the validation set, can be seen in Table 4. This table also shows the performance measures obtained using M_T and A_{VP}, the best performing model achieves an average M_T of 37.2% ahead of the action being completed. The timing of the predictions will be further evaluated at the end of this section. The vertical predictions were evaluated every time a new prediction was accepted as a contribution to A_P, the result for the best model is $A_{VP} = 81.29\%$.

Figure 6 shows a segment of prediction results from, $w_s = 10$ and $N_T = 10$ with *NR*, obtained on the test. The upper graph displays the predicted and true labels without taking the UE into account. The graph in the middle shows the predictions that were above the specified threshold and the graph at the bottom shows the fluctuation in certainty regarding the most probable output class.

A comparison of filtered predictions using different thresholds, $0.75 \cdot Th_L$, Th_L, and $1.25 \cdot Th_L$, for the same segment of predictions from, $w_s = 10$ and $N_T = 10$ with *NR*, is shown in Figure 7. The top plot shows that more movements are covered when the threshold is lowered, however, more of the predictions are incorrect. The middle one is the same as in Figure 6 and the plot at the bottom shows that increasing the threshold would result in higher prediction accuracy at the cost of missing more of the movements completely.

Figure 8 shows an aggregation of all hand movements from the test set created in order to further evaluate the performance of the network with regard to time. The upper graph shows the normalized distance, d_i, that the controller travelled from the moment the previous box was clicked until the next one, calculated at each sample i for each movement as:

$$d_i = 1 - \frac{|\mathbf{p}_{end} - \mathbf{p}_i|}{|\mathbf{p}_{end} - \mathbf{p}_{start}|} \tag{1}$$

where $\mathbf{p}_{start}$ is the coordinate $(x, y, z)^T$ of the controller for the first sample of the movement and $\mathbf{p}_{end}$ is the last one. The normalized distance was then plotted with an alpha of $0.03 \in [0, 1]$ and normalized time in order to show the characteristics of all movements on the same scale. The lower graph shows the velocity, v_i, towards the target, $\mathbf{p}_{end}$, at each sample i for each movement, calculated as:

$$v_i = f_s \cdot (\mathbf{p}_i - \mathbf{p}_{i-1})^T \cdot \frac{\mathbf{p}_{end} - \mathbf{p}_i}{|\mathbf{p}_{end} - \mathbf{p}_i|} \tag{2}$$

where f_s is the sample frequency of the eye tracker. The velocity towards the target was then plotted with the same alpha and the same normalized time as described above. The results from the velocity calculations sometimes, due to positional tracking errors, result in unreasonable values. The velocity v_i was therefore removed if it exceeded 2.5 m/s.

From the figure it is clear that the data is noisy and with some variation, however, a few trends are clearly emerging as well. The figure shows that there is little to no movement in the beginning of each time series followed by a segment with varying amount of movement, both towards and away from the target, up until about the halfway point. Around the halfway point the combination of the distance and the velocity graph shows a stationary segment followed by a new segment of movement that slows down towards the end. However, during the second movement segment almost all movements have positive velocity towards the target the entire time.

The normalized distance data from Figure 8 was used in Figure 9 to further investigate the time ahead of movement completion (TAMC) data that was

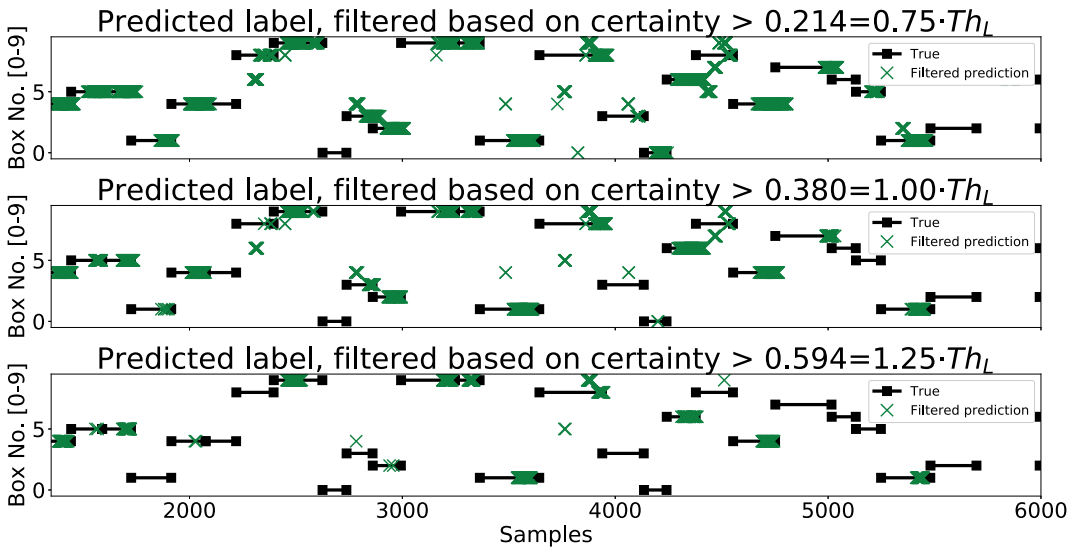

Figure 7. A figure that shows a comparison of filtered predictions using $0.75 \cdot Th_L$, Th_L, and $1.25 \cdot Th_L$ for the same segment of predictions from, $w_s = 10$ and $N_T = 10$ with *NR*.

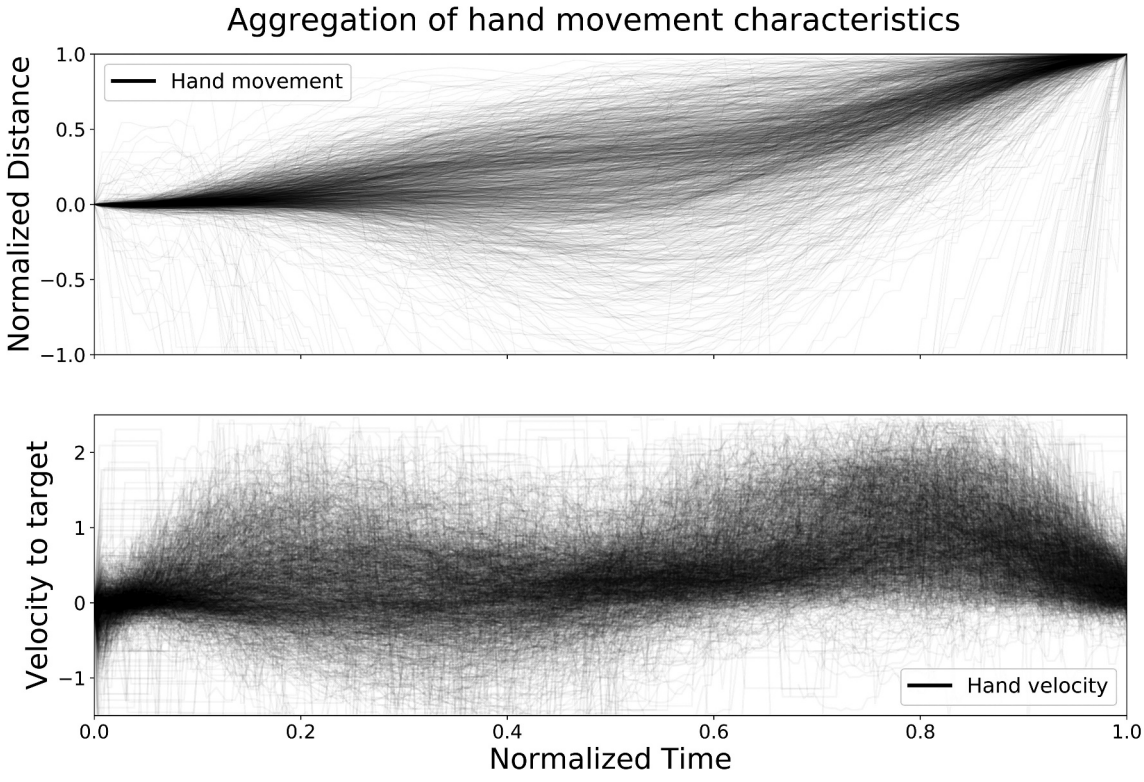

Figure 8. The figure shows an aggregation of all hand movements from the test set, with respect to distance left to target and velocity towards target, plotted with normalized time.

summarized in the evaluation metric M_T. The figure shows the first correct prediction from each of the movements that were correctly classified at least once together with the mean value M_T. The 5th, 25th, 50th, 75th, 95th, and 99th percentiles were added in order to give a more nuanced description of the TAMC and the normalized time values for these are summarized in Table 5. The first 5% of the correct predictions are

most likely 'lucky-shots', i.e. a result of the network sticking to the prediction from the previous movement, which due to the randomness of the test was the same target twice in a row. The reason being that the test person does not know the next target for the first 0.2 s, i.e. first 5.4% to 27.0% for the max/min duration of the movements, due to the delay that was inserted between each task in the VRE, as described in Section 3.1. The movements are, in 99% of the cases, correctly predicted the first time, before the hand reaches the target and more than 24% ahead of time in 75% of the cases, which corresponds to about 302 ms for the median movement duration. The histograms, top and right, show the distributions of correct predictions with regard to the normalized time and normalized distance respectively.

Finally, the histogram of the TAMC-distribution from the top of Figure 9 was re-scaled to the same range as the velocity, i.e. max 2.5 (for visual purposes) and plotted together with the velocity graph from Figure 8 with normalized time. It can be seen that the shape of the histogram is very similar to the second movement segment of the velocity graph, however, shifted a bit to the left. This indicates that the intended movement direction often is predicted at the early timesteps of the final movement towards the target.

6. Discussion

The goal of this paper was to provide a system for intended human arm movement prediction and the two classification objectives, presented in Section 1, were **Primary** - *determine the discrete horizontal direction corresponding to the box that was clicked* and **Secondary** - *distinguish between whether the movement occurred on the upper or lower level of boxes*. The best network reached an accuracy of 70.70% for the primary objective, correctly classifies 67.89% of the movements at least once, and an accuracy of 81.29% for the secondary objective. These results might seem far from 100%, however, it is important to remember that human behaviour is complex and difficult to capture. It is, therefore, perhaps impossible to reach 100% and maybe not a requirement for intention prediction to provide value. Considering that the system predicts upcoming movement directions, before the completion of these events, solely based on eye

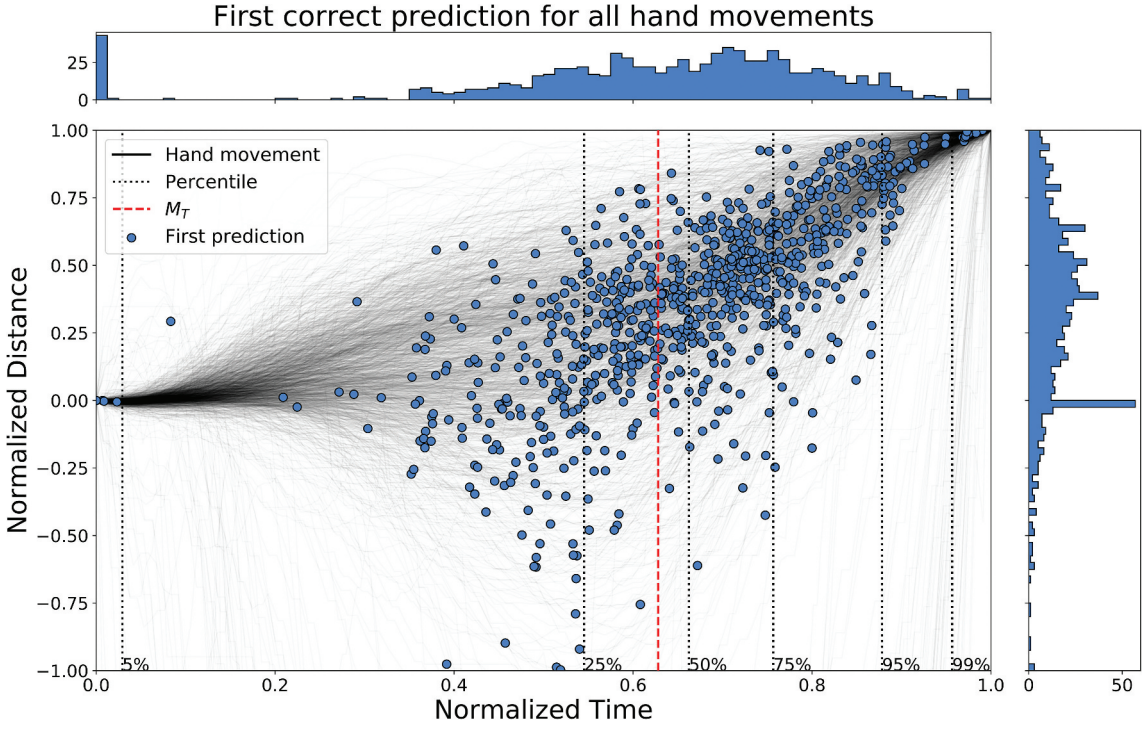

Figure 9. A figure that shows the first correct prediction for all hand movements from the test set, plotted with normalized time and distance to target.

Table 5. A table that summarizes the normalized time values for the percentiles shown in Figure 9.

Percentile	5	25	50	75	95	99
Normalized time [0, 1]	0.03	0.55	0.66	0.76	0.88	0.96

gaze and without knowing the directions specifically only the number of directions (10) it becomes easier to see the benefits of the system. A robot could receive warnings, ahead of time, about in which direction an operator is likely to move and adjust its behaviour either by a quick halt or more preventative measures, depending on its current position. The parts where the network is uncertain may also prove useful, as it could potentially be used to inform the robot to maintain a larger distance between the operator and itself since the human is behaving in a way that is difficult to predict.

The results presented in this paper are comparable, in terms of accuracy, to the work by Gomez Cubero and Rehm (2021) and achieved using a much smaller LSTM-network. The task presented here has 10 movement directions instead of three possible choices and the duration of a movement sequence is much lower, less than 4 s. Compared to the work by Chaandar Ravichandar, Kumar, and Dani (2016), it is clear that their combined approach of tracking both gaze and arm movements results in higher accuracy than a gaze-only solution. However, the small LSTM-network proposed in this paper should be significantly faster at inference time, which is crucial for an online application.

The main advantage of using eye gaze compared to a hand tracking based systems is that the gaze can give insights into the decision leading up to a movement, thereby gaining a head start compared to a hand tracking system that, by definition, has to wait for the movement to start before it can be detected. A hand tracking system may, however, be more robust since it makes predictions based on the actual movement, which suggests that a combination of the two approaches may work complementary.

The movements are, in 99% of the cases, correctly predicted the first time, before the hand reaches the target and more than 24% ahead of time in 75% of the cases, which corresponds to about 302 ms for the median movement duration. It can be concluded from Figure 9 that the metric, M_T, does not give an accurate description of the TAMC since it does not even cover the 50th percentile. The percentiles are, therefore, a more suitable way of reasoning around the TAMC and the behaviour of the intention prediction system.

The velocity graph and the plateau of the distance to target, Figure 8, indicates that the actual movement towards the target, on average, most likely

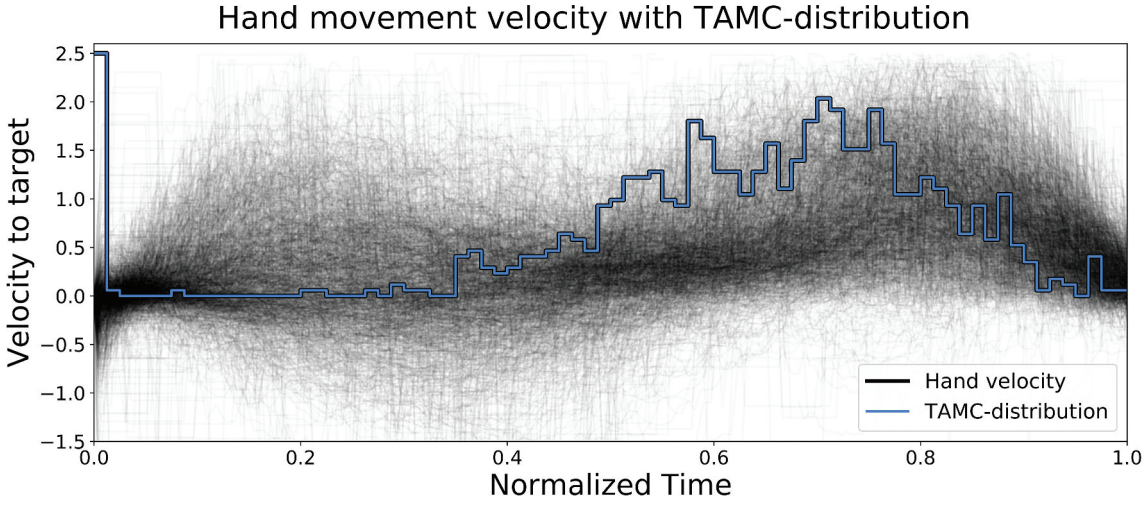

Figure 10. A figure that shows the normalized prediction time ahead of movement completion (TAMC) distribution plotted with normalized time and velocity to target for all hand movements from the test set.

starts after half of the time has elapsed. The intuition being that why would the test person slow down and stop halfway to the target if the target has been identified. The movements prior to that are likely movements related to either a retraction from the previous target to a more neutral position or related to the search for the next target. This argument is strengthened by the fact that the test person does not know the next target for the first 0.2 s, i.e. first 5.4% to 27.0% for the max/min duration of the movements, due to the delay that was inserted between each task in the VRE. Combining this reasoning with the fact that the TAMC-distribution from Figure 10 leads the velocity graph for the second movement segment, one could argue that the intention prediction results are probably better, with respect to TAMC, than what is shown by observing the entire movement duration as in Figure 8.

The use of uncertainty as a way to decide whether to trust a prediction or not, clearly improves the performance of the network for the primary objective. The filtered behaviour is displayed in the middle graph of Figure 6 and it shows that the network sometimes struggles to predict the correct box. However, when that happens, the predictions are mostly neighbouring boxes and these predictions could be used as well to give indications regarding, e.g. which half of the environment that the human intends to interact with. There are quite sharp drops in certainty when the target changes. This high uncertainty carries information too, regarding that the human is behaving in a way that is difficult to predict. The way the threshold is calculated clearly affects the results, for example, increasing the threshold would result in a higher prediction accuracy but would result in more movements being missed entirely. Lowering the threshold would have the opposite effect, as shown in Figure 7. However, there may exist more refined ways of selecting and utilizing the UE threshold, for example, a dynamic threshold that changes based on the certainty or variation of the last few predictions would likely give a different outcome.

The results presented in this paper are based on making predictions on every new sample of incoming data. This is potentially a waste of resources since the amount of new information in a single sample is probably not enough to affect the prediction results. Therefore, making predictions every n:th sample may give better results and reduce the total amount of computation required. Selecting the appropriate n depends on the application, since a too large n for a certain application would make predictions too late, perhaps even after the end of a movement. A suitable n for the presented application may be in the range of $n \in [1, 10]$, since that would correspond to making predictions at least 10 times per second, which should be fast enough to maintain most of the TAMC.

In addition to the previously mentioned factors that affect the prediction performance, both with respect to accuracy and TAMC, there may also exist other configurations of the current neural network or other network architectures all together, which could give better results.

Somewhat surprisingly, Figure 5 shows that *not* randomizing (*NR*) the copies of training data performs the best. The reasoning behind the randomization was to use the same data to create new transitions between movements. It is, however, possible that this creates too large discontinuities in the data, compared to the test sequence that consists of continuous data, thereby decreasing performance. The fact that the use of data augmentation improves the results could be seen as an indicator that the developed network could benefit from training on a larger dataset.

The eye tracker used in this paper collects data at 120 Hz, which means that it should capture most eye movements, including the faster saccades that typically range from 10 to 100 ms in duration. A faster tracker, up to maybe 200 Hz, could give some additional information regarding the fastest eye movements. However, it is not certain that it will improve the results of the intention prediction as it will also give more data samples that are similar to each other for the slower movements, i.e. there will be a trade-off between new information and overflow of data, and this problem is likely becoming more prominent for even faster tracking systems. A slower tracker would deal with the issue of too much data, however, due to the nature of the eye and the rapid movements, saccades, the lower limit to capture the majority of eye movements is probably around or slightly above $1/10$ ms, i.e. about 120 Hz.

The gaze-based intention prediction results presented in this paper show promise. However, there are clearly several steps to go before it is possible to use such a system in a real-world application together with industrial robots. The experiment has been

structured around using ET-hardware that already exists today in the form of safety goggles. This means that most of the prediction stage should be possible to implement in the real world also. However, the evaluation of the performance of such a system would require other ways than the controllers used today to keep track of the users hands and whether or not they have reached their target location.

7. Conclusions and future work

This paper has presented a system that uses a recurrent neural network to predict the intended human arm movement direction, solely based on eye gaze, utilizing the notion of uncertainty to determine whether a prediction should be trusted or not. The developed deep learning solution makes predictions on continuously incoming data and reaches an accuracy of 70.7%, for predictions with high certainty, and correctly classifies 67.89% of the movements at least once. The movements are, in 99% of the cases, correctly predicted the first time, before the hand reaches the target and more than 24% ahead of time in 75% of the cases, which corresponds to about 302 ms for the median movement duration. These results show that gaze-based arm movement intention prediction is a promising step towards enabling efficient human-robot collaboration.

Potential immediate next steps to build upon the results of this paper are: investigate alternative ways to utilize the uncertainty and/or to set the threshold that determines whether to trust a prediction or not, explore the impact of making predictions less often, i.e. at every nth timestep $(n>1)$,and finally there may exist other neural network architectures that could improve the current results.

To further evaluate the performance of the presented system for human intention prediction, the implementation of a VRE with more complex tasks would be of interest. For example, an assembly station where the operator collaborates with a virtual robot, such a system could be evaluated using an external control system that receives the predicted intentions and makes the robot adapt accordingly. A natural extension to this, if the results are successful, would be to implement the same system in a real-world version of that application. This would include using safety glasses with built-in ET instead of a VR-headset and the evaluation of using such a system in a real-time environment.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Abidi, M., A. Al-Ahmari, A. El-Tamimi, S. Darwish, and A. Ahmad. 2016. "Development and Evaluation of the Virtual Prototype of the First Saudi Arabian-Designed Car." *Computers* 5 (4): 26. https://doi.org/10.3390/computers5040026.

Al-Ahmari, A. M., M. H. Abidi, A. Ahmad, and S. Darmoul. 2016. "Development of a Virtual Manufacturing Assembly Simulation System." *Advances in Mechanical Engineering* 8 (3): 168781401663982. https://doi.org/10.1177/1687814016639824.

Andrew, M., E. Brynjolfsson, T. H. Davenport, D. J. Patil, and D. Barton. 2012. "Big Data: The Management Revolution." *Harvard Business Review* 90 (10): 60–68.

Aschenbrenner, D., N. Maltry, J. Kimmel, M. Albert, J. Scharnagl, and K. Schilling. 2016. "Artab-Using Virtual and Augmented Reality Methods for an Improved Situation Awareness for Telemaintenance." *IFAC-Papersonline* 49 (30): 204–209. https://doi.org/10.1016/j.ifacol.2016.11.168.

Awais, M., and D. Henrich. 2010. "Human-Robot Collaboration by Intention Recognition Using Probabilistic State Machines." In *19th International Workshop on Robotics in Alpe-Adria-Danube Region (RAAD 2010)*, Budapest, Hungary, 75–80. IEEE.

Chadalavada, R. T., H. Andreasson, M. Schindler, R. Palm, and A. J. Lilienthal. 2020. "Bi-Directional Navigation Intent Communication Using Spatial Augmented Reality and Eye-Tracking Glasses for Improved Safety in Human–Robot Interaction." *Robotics and Computer-Integrated Manufacturing* 61:61. https://doi.org/10.1016/j.rcim.2019.101830.

Dahl, M., A. Albo, J. Eriksson, J. Pettersson, and P. Falkman. 2017. "Virtual Reality Commissioning in Production Systems Preparation." In *22nd IEEE International Conference on Emerging Technologies And Factory Automation (ETFA)*, Limassol, Cyprus, 1–7. IEEE.

Duchowski, A. T. 2017. *Eye Tracking Methodology: Theory and Practice*. Springer International Publishing. https://doi.org/10.1007/978-3-319-57883-5.

Gal, Y., and Z. Ghahramani. 2016. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep

Learning." In *International Conference on Machine Learning*, New York, USA, 1050–1059.

Gomez Cubero, C., and M. Rehm. 2021. "Intention Recognition in Human Robot Interaction Based on Eye Tracking." In *IFIP Conference on Human-Computer Interaction*, 428–437. Springer.

Goodfellow, I., Y. Bengio, and A. Courville. 2016. *Deep Learning*. Cambridge, MA: MIT Press. http://www.deeplearningbook.org.

Gould, T. D., T. M. Bastain, M. E. Israel, D. W. Hommer, and F. Xavier Castellanos. 2001. "Altered Performance on an Ocular Fixation Task in Attention-Deficit/hyperactivity Disorder." *Biological Psychiatry* 50 (8): 633–635. https://doi.org/10.1016/S0006-32230101095-2.

Greff, K., R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. 2016. "LSTM: A Search Space Odyssey." *IEEE Transactions on Neural Networks and Learning Systems* 28 (10): 2222–2232. https://doi.org/10.1109/TNNLS.2016.2582924 .

Hinton, G. E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. 2012. "Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors." *arXiv preprint arXiv:1207.0580*.

Hochman, M., Y. Parmet, and T. Oron-Gilad. 2020. "Pedestrians' Understanding of a Fully Autonomous Vehicle's Intent to Stop: A Learning Effect Over Time." *Frontiers in Psychology* 11:11. https://doi.org/10.3389/fpsyg.2020.585280 .

Hochreiter, S., and J. Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Huang, C.-M., and B. Mutlu. 2016. "Anticipatory Robot Control for Efficient Human-Robot Collaboration." In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Christchurch, New Zealand, 83–90. IEEE.

Jungwirth, F., M. Murauer, M. Haslgrübler, and A. Ferscha. 2018. "Eyes are Different Than Hands: An Analysis of Gaze as Input Modality for Industrial Man-Machine Interactions." In *Proceedings of the 11th Pervasive Technologies Related to Assistive Environments Conference*, Corfu, Greece, 303–310. ACM.

Karatekin, C. 2007. "Eye Tracking Studies of Normative and Atypical Development." *Developmental Review* 27 (3): 283–348. https://doi.org/10.1016/j.dr.2007.06.006.

Kingma, D. P., and B. Jimmy 2014. "Adam: A Method for Stochastic Optimization." *arXiv preprint arXiv:1412.6980*.

Krüger, J., T. K. Lien, and A. Verl. 2009. "Cooperation of Human and Machines in Assembly Lines." *CIRP Annals* 58 (2): 628–646. https://doi.org/10.1016/j.cirp.2009.09.009.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86 (11): 2278–2324. https://doi.org/10.1109/5.726791.

Luzheng, B., A. G. Feleke, and C. Guan. 2019. "A Review on EMG-Based Motor Intention Prediction of Continuous Human Upper Limb Motion for Human-Robot Collaboration." *Biomedical Signal Processing and Control* 51:113–127. https://doi.org/10.1016/j.bspc.2019.02.011.

Mainprice, J., and D. Berenson. 2013. "Human-Robot Collaborative Manipulation Planning Using Early Prediction of Human Motion." In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 299–306. IEEE.

Makrini, E., K. M. Ilias, D. Lefeber, and B. Vanderborght. 2017. "Design of a Collaborative Architecture for Human-Robot Assembly Tasks." In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, 1624–1629. IEEE.

Morariu, O., C. Morariu, T. Borangiu, and S. Răileanu. 2018. "Manufacturing Systems at Scale with Big Data Streaming and Online Machine Learning." In *Service Orientation in Holonic and Multi-Agent Manufacturing*, 253–264. Springer International Publishing. https://doi.org/10.1007/978-3-319-73751-5_19 .

Nagorny, K., P. Lima-Monteiro, J. Barata, and A. Walter Colombo. 2017. "Big Data Analysis in Smart Manufacturing: A Review." *International Journal of Communications, Network and System Sciences* 10 (3): 31–58. https://doi.org/10.4236/ijcns.2017.103003.

Pettersson, J., A. Albo, J. Eriksson, P. Larsson, K. W. Falkman, and P. Falkman. 2018. "Cognitive Ability Evaluation Using Virtual Reality and Eye Tracking." In *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, Ottawa, ON, Canada, 1–6. IEEE.

Pettersson, J., and P. Falkman. 2020. "Human Movement Direction Classification Using Virtual Reality and Eye Tracking." In *30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM)*, Procedia Manufacturing, Athens, Greece.

Pettersson, J., and P. Falkman. 2021. "Human Movement Direction Prediction Using Virtual Reality and Eye Tracking." In *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, Valencia, Spain, IEEE.

Ravichandar, C., A. K. Harish, and A. Dani. 2016. "Bayesian Human Intention Inference Through Multiple Model Filtering with Gaze-Based Priors." In *2016 19th International Conference on Information Fusion (FUSION)*, Heidelberg, Germany, 2296–2302. IEEE.

Rizzo, A. A., M. Schultheis, K. A. Kerns, and C. Mateer. 2004. "Analysis of Assets for Virtual Reality Applications in Neuropsychology." *Neuropsychological Rehabilitation* 14 (1–2): 207–239. https://doi.org/10.1080/09602010343000183.

Samek, W., T. Wiegand, and K.-R. Müller. 2017. "Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models." *arXiv preprint arXiv:1708.08296*.

Shi, L., C. Copot, and S. Vanlanduit. 2021. "Gazeemd: Detecting Visual Intention in Gaze-Based Human-Robot Interaction." *Robotics* 10 (2): 68. https://doi.org/10.3390/robotics10020068.

Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. "Rethinking the Inception Architecture for Computer Vision." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.

Tang, G., P. Webb, and J. Thrower. 2019. "The Development and Evaluation of Robot Light Skin: A Novel Robot Signalling System to Improve Communication in Industrial Human-Robot Collaboration." *Robotics and Computer-Integrated Manufacturing* 56:85–94. https://doi.org/10.1016/j.rcim.2018.08.005.

Tobii, A. B. 2020. *Tobii Pro VR Integration – Based on HTC Vive Development Kit Description*. V.1.7 - En-Us Ed. Tobii AB. Accessed: February 13, 2020: https://www.tobiipro.com/siteassets/tobii-pro/product-descriptions/tobii-pro-vr-integration-product-description.pdf/?v=1.7 .

Wang, W., R. Li, Y. Chen, Y. Sun, and Y. Jia. 2022. "Predicting Human Intentions in Human–Robot Hand-Over Tasks Through Multimodal Learning." *IEEE Transactions on Automation Science and Engineering* 19 (3): 2339–2353. https://doi.org/10.1109/TASE.2021.3074873.

Wang, J., M. Yulin, L. Zhang, R. X. Gao, and W. Dazhong. 2018. "Deep Learning for Smart Manufacturing: Methods and Applications." *Journal of Manufacturing Systems* 48:144–156. https://doi.org/10.1016/j.jmsy.2018.01.003.