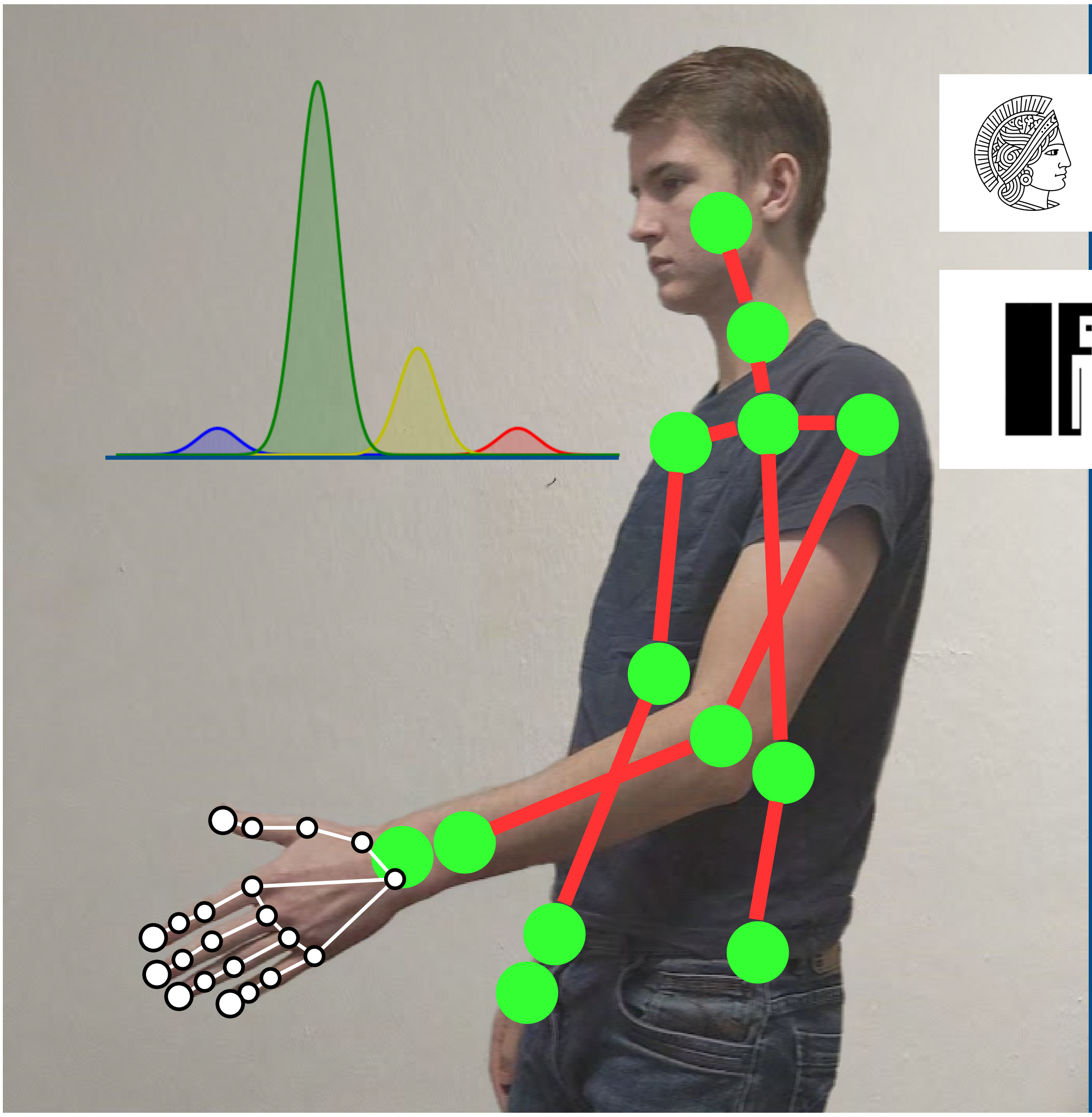
Visual Hierarchical Recognition And Segmentation Of Interactions

Visuelle hierarchische Interaktionserkennung und -segmentierung
Bachelor thesis by Erik Prescher
Date of submission: December 12, 2022

1. Review: Prof. Dr. Jan Peters
2. Review: Prof. Dr. Dr. Ruth Stock-Homburg
Supervision: Vignesh Prasad M.Sc
Darmstadt



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit versichere ich, Erik Prescher, die vorliegende Bachelorarbeit gemäß § 22 Abs. 7 APB der TU Darmstadt ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 12. Dezember 2022

E. Prescher

Abstract

Robots are currently being integrated more and more into everyday life, which also increasingly involves interactions between humans and robots. Part of this interaction, especially with humanoid robots, is social habits such as handshaking or waving. However, these interactions are sometimes very complex and involve many different tasks, such as recognizing the current task and determining the appropriate reaction. Therefore the current greeting interaction must first be recognized and then the adaptively created response motion needs to be created based on the human's motions. Predicting this can be very difficult, which is why it makes sense to divide the action into several segments, thus decreasing the overall complexity. For example, shaking hands can be divided into the segments "reaching", "shaking" and "retrieving". For each of these segments, it is then possible to use separate models that are adapted to the specifics of each segment. It is important that the method also works in a real-time environment, meaning it may only have low latencies.

This work provides a way to detect and segment interactions using skeletal trajectories and RGB video data. The method follows a hierarchical approach, in which at first the action and then the current segment is determined with the help of this. In the next step this method can be combined with motion-generating methods and thus serve as a framework for interaction recognition, segmentation, and execution.

Zusammenfassung

Roboter werden derzeit mehr und mehr in den Alltag integriert, wobei es auch immer häufiger zu Interaktionen zwischen Menschen und Robotern kommt. Ein Teil dieser Interaktionen, insbesondere mit humanoiden Robotern, sind soziale Gewohnheiten wie Händeschütteln oder Winken. Allerdings sind diese Interaktionen mitunter sehr komplex und beinhalten viele verschiedene Aufgaben, wie z.B. das Erkennen der aktuellen Interaktion und die Reaktion auf den aktuellen Zustand derselben. Daher muss erst die aktuelle Begrüßungsinteraktion erkannt werden, und anschließend die adaptiv erstellte Antwortbewegung auf die Bewegungen des Menschen angepasst werden. Dies kann sehr schwierig sein, so dass es sinnvoll ist, die Ausführung in mehrere Segmente aufzuteilen und so die Komplexität eines Segments in Bezug auf die gesamte Ausführung zu verringern. Zum Beispiel kann das Händeschütteln in die Segmente „Greifen“, „Schütteln“ und „Zurückziehen“ unterteilt werden. Für jedes dieser Segmente ist es nun möglich, separate Modelle zu verwenden, die an die Besonderheiten des jeweiligen Segments angepasst sind. Wichtig ist, dass die Methode auch in einer Echtzeitumgebung funktioniert und dementsprechend nur geringe Latenzen aufweisen darf.

Diese Arbeit bietet eine Möglichkeit zur Erkennung und Segmentierung von Interaktionen anhand von Skelett-Trajektorien und RGB-Videodaten. Die Methode verfolgt einen hierarchischen Ansatz, bei dem zuerst die Aktion und dann das aktuelle Segment mit Hilfe dieser zuvor bestimmten Aktion bestimmt wird. Zukünftig kann diese Methode mit bewegungsgenerierenden Methoden kombiniert werden und so als Framework für die Erkennung, Segmentierung und Ausführung von Interaktionen dienen.

Contents

1. Introduction **2**

1.1. Motivation 2

1.2. Contribution 4

1.3. Outline 4

2. Foundations And Related Work **5**

2.1. Foundation 5

2.2. Related Work 11

3. Classification and Segmentation of Interactions **14**

3.1. Problem Statement 14

3.2. Proposed Approach 14

3.3. Action Recognition 16

3.4. Action Segmentation 18

4. Experiments And Evaluation **21**

4.1. Datasets 21

4.2. Implementation Details 23

4.3. Experiments On NuisiRGB 27

4.4. Segment Recognition 36

4.5. Realtime Performance 38

5. Discussion **41**

5.1. Conclusion 41

5.2. Outlook 42

A. Live Demonstrations **47**

B. Simulation With Robot **52**

C. List of Abbrevations 53

List of Figures

2.1. Comparison of the Independent Fusion Model and Correlated Fusion Model 7
2.2. Difference of the edges between spatial graph (a) and temporal graph (b). 8
2.3. Hand Landmarks 9

3.1. Hierarchical Data Flow 16
3.2. Segment Recognition With Action As Prior Knowlegde 19

4.1. Setting of NuiSI-RGB 23
4.2. Features of Nuisi-RGB Dataset 23
4.3. Instantaneous and Prolonged Segment Sequences 30
4.4. The Evaluation Accuracy of the Baseline ST-GCN. 30
4.5. The Evaluation Accuracy of the Action ST-GCN. 31
4.6. The Evaluation Accuracy of the Action Point History Classifier 31
4.7. The Evaluation Accuracy of the Segment ST-GCN during the training. . . . 32
4.8. Confusion Matrix of the Baseline ST-GCN. 33
4.9. Confusion Matrix for Recognition of all Actions with ST-GCN 34
4.10. Confusion Matrix for Recognition of all Actions with Hand Pose 35
4.11. Confusion Matrix for Recognition of All Actions with Bayesian Net Fusion . 36
4.12. Confusion Matrix of the ST-GCN Baseline for Segmentation. 37
4.13. Confusion Matrix of the Segment ST-GCN. 38
4.14. Image Series of Live Handshake 39

A.1. Image Series of Live Highfive 47

A.2. Image Series of Live Rocket 48
A.3. Image Series of Live Wave 49
A.4. Image Series of Live Clapfist 50
A.5. Image Series of Live Fistbump 51

B.1. Image Robot Simulation 52

1. Introduction

Due to the increased use of robots in everyday life [20, 31, 15], robots must act correctly when working with humans. Hence new advances are being made in human-robot interaction (HRI). HRI refers to any interaction between a robot and a human, which can be divided into four different areas [34].

1. The human as a supervisor, where the human controls a robot in the execution of a specific task.
2. Remote control of a robot.
3. The human as a passenger driven by a robot, as in an autonomous car.
4. Social interaction between a human and a robot.

This work is categorized in the fourth category, where the robot is capable of correctly interpreting the actions of a human. This involves robots learning social behavior, such as shaking hands or waving and then adapting to the gestures of the human. To accomplish this goal the robot has to perform two basic tasks. Second, it has to interpret the movements of humans and has to anticipate these. Second, it must generate and execute its motions on the basis of the seen and the anticipated movements [6].

1.1. Motivation

Robots have evolved from taking over repetitive tasks and are now able to generalize tasks and respond correctly to different conditions. In addition, robots are not only used in factories and closed-off areas but they are also used in public places like restaurants and hotels [12, 15]. Currently, automatic vacuuming [39, 8] or mowing robots [32] are also very common in Europe. However, these robots usually have no sensors to measure the activity of people and thus keep away from them, but they can only detect objects in a short range. Thus, the interaction between robots and humans is very one-sided. The situation is different with service robots, which can be used in restaurants. This is mostly used in Japan or China and even there in only a few restaurants [21]. In such conditions, the robot must be able to interact with people because of the increased number of humans and the desired interaction with customers. The trend that robots are increasingly integrated into everyday life makes research into how robots and humans interact more important. Not only to make the interaction as natural as possible for humans but also to ensure the highest possible level of safety. To ensure this, the robot must be able to interpret what the surrounding persons will do next to be able to react accordingly.

Information is constantly exchanged during human interactions. This kind of communication is mostly non-verbal through body posture, touch or eye-contact [25] and is usually sufficient to perform various social interactions. These may include handshakes or fistbumps, which have increased significantly through the Covid-19 pandemic However, in order to perform one of the greeting interactions, one must recognize which interaction the counterpart initiates in order to respond with the right movement. Its a common experience when one makes a mistake by performing a fistbump oneself and the opposite person tries to shake one's fist. While between two people it is only a possibly unpleasant situation, between a person and a robot it can lead to injuries [40]. This illustrates the importance of being able to reliably recognize the social interaction of the human counterpart.

The robot must not only recognize the action performed by the other actor, it also has to react in the correct way. Therefore the robot has to consider many different aspects. First of all, it has to react differently depending on the action. This is why the classification of the action is important. Further difficulties arise during the reaction of the robot. As an Example the interaction *handshake* can be devided into the following:

1. Move its hand to the hand of the other person.
2. Anticipate where its hand must be when it meets the other persons hand.
3. The robot has to grab the hand and adjust the pressure of its hand very sensitively.
4. When the robot has successfully grasped the person's hand, it must now perform the familiar sinusoidal shaking. In this shaking motion the difficulty lies in coordinating the movement simultaneously with the respective other.
5. After an indefinite number of repetitions, the robot must release the grip and return to the starting position.

It is immediately obvious that each of these steps is very difficult in itself and brings problems and obstacles. For this reason, it is useful not to consider the action as a whole, but to divide the action into individual segments and then create different networks for each of the segments, each of which solves one of the problems. This segmentation reduces the complexity of the problem, which may enable a network to solve the problem that was previously too large after all.

1.2. Contribution

This work offers a hierarchical approach that uses both RGB videos and skeleton trajectories. Different technologies are combined in a hierarchical setting to first detect the action during a demonstration and then use this prediction for segmentation. The presented approach is fully supervised and can detect, as well as segment live demonstrations. Several supervised and unsupervised approaches have already been presented, which can recognize the current action and segment. One of them is presented in MILD [28] which this thesis is meant to be a bridge to. This approach could then be connected to a framework that combines hierarchical recognition, segmentation, and motion generation for human-robot interaction.

1.3. Outline

The second chapter covers the basics, which are necessary to understand the content of this work. Also, in the second part of this chapter, other works are described and discussed to get a better understanding of what has already been done and what the current problems are. Chapter 3 describes the approach of this thesis and the experiments which were performed to evaluate the approach. Not only the used libraries and methods but also our own dataset is explained in detail. Finally an outlook is given on the next possible step and how this work can be incorporated into other projects and can be conjoined to a framework of motion generation for human-robot interaction.

2. Foundations And Related Work

This chapter introduces the foundations, which are important for the understanding of this thesis, and explains the different components which have been used in the thesis. Afterwards, a variety of other works are discussed, which deal with a similar topic and are therefore also relevant for this thesis and add a further background knowledge.

2.1. Foundation

2.1.1. Bayesian Classifier Fusion

Classification is a big tasks of Machine Learning. First classes must be designed according to the desired use. Then inputs must be classified into one of the previously defined classes. To improve this classification and at the same time make it more robust, there are several possibilities like ensembles and random forest [30, 24, 3]. Using different models and/or architectures and combine the results of all networks in the evaluation achieves to increase the robustness. However, the individual models may be biased or the selection of the training data may have led to a separation and thus too little variance within the training data. To counteract this behavior and also to include other additional variables, so-called Bayesian models can be used [37].

Independent Opinion Pool

Assuming that the classifiers are conditionally independent, it is possible to apply the basic Bayes' rule. To apply those, Bayes' rule with the uninformed prior t_i can be transformed to the sought $P(t_i|o_i^1, ..., o_i^K)$ to:

$$P(t_i|o_i^1, ..., o_i^K) \propto \prod_{k=1}^{K} P(t_i|o_i^k),$$

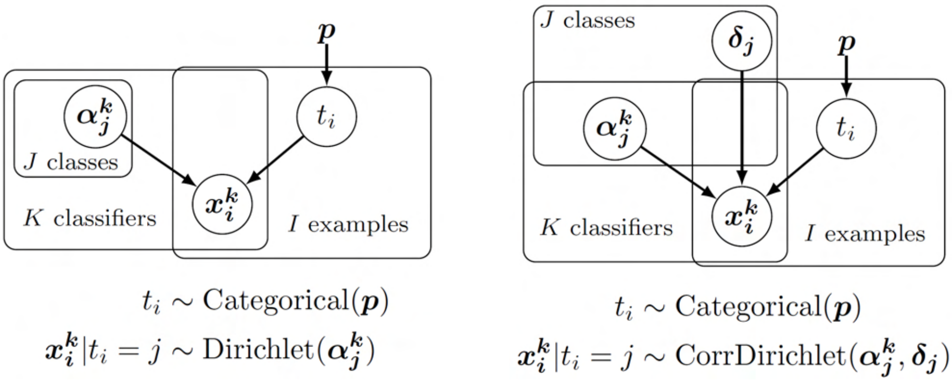
This distribution has to be normalized to a sum of 1 and is also known as Independent Opinion Pool (IOP) [2], which is Bayes optimal given base classifiers with conditionally independent attributes and an uninformed prior.

Independent Fusion Model

While the IOP Bayes is optimal under the above condition of independence, properties of the different classifiers cannot be considered, such as uncertainty, bias, and variance. To model these properties, the *Independent Fusion Model* (IFM) has been designed. The IFM models the output properties of each classifier with a fixed categorical output distribution. The distribution $P(x_i^k|t_i)$ observed from multiple training samples of the classifier from x_i^k over the true label t_i. Those distributions are then modeled to a Dirichlet distribution. The model is shown graphically in Figure 2.1.

Correlated Fusion Model

The problem of the Independent Fusion Model is that most classifiers trained on the same classes are very highly correlated [13]. For this reason, the IFM is extended to a Correlated Fusion Model (CFM). The different classifiers' outputs correlation can now be modeled with the Correlated Fusion Model.



p
α_j^k
t_i
J classes
x_i^k
K classifiers
I examples
$t_i \sim \text{Categorical}(p)$
$x_i^k | t_i = j \sim \text{Dirichlet}(\alpha_j^k)$
J classes
δ_j
p
α_j^k
t_i
K classifiers
x_i^k
I examples
$t_i \sim \text{Categorical}(p)$
$x_i^k | t_i = j \sim \text{CorrDirichlet}(\alpha_j^k, \delta_j)$

(a)

(b)

Figure 2.1.: Comparison of the Independent Fusion Model (a) and the Correlated Fusion Model (b). At the CFM one can see, that the classes are now also dependent from the pairwise correlation δ_j of the classifiers.[1]

2.1.2. Spatial Temporal Graph Convolutional Neural Network

The work of Yan et al [43] deals with skeleton-based action recognition using Spatial Temporal Graph Convolutional Networks. Spatial Graph Convolution and Spatial Temporal Modeling are used for prediction.

Spatial Graph Convolution

However, before considering skeleton trajectories over time, the skeleton graph must first be analyzed over a frame. Therefore a graph CNN model is applied to a single frame. For the frame, the edges of the skeleton are taken. For the edges $E_{skeleton}(\tau) = \{J_{ti}J_{tj} | t = \tau, (i,j) \in H\}$, where H stands for the naturally connected human joints and τ for the current time step. The naturally connected human joints are those given from the anatomy of humans.

[1] Images extracted from *S. Trick and C. Rothkopf, "Bayesian classifier fusion with an explicit model of correlation," in Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (G. Camps-Valls, F. J. R. Ruiz, and I. Valera, eds.), Proceedings of Machine Learning Research, PMLR, 2022*

Spatial Temporal Modeling

To model the temporal connections as a graph, not the human-connected joints within a frame, but the same joints over several frames are modeled. The number of frames is defined as window size Γ. Thus temporal edges are defined as $E_{temp}(\tau) = \{J_{ti}J_{t'i}|t' = t+1, \lfloor t-\tau \rfloor < \Gamma\}$. The difference between the spatial graph and the temporal graph is shown in Figure 2.2.

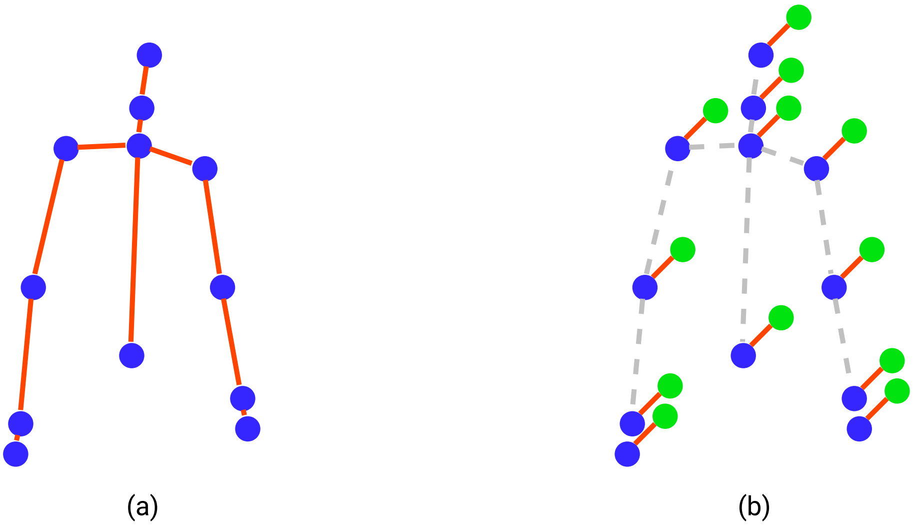


Figure 2.2.: Difference of the edges between spatial graph (a) and temporal graph (b).

2.1.3. Hand Gesture Recognition

Complementary to Action Recognition, the gesture of the hand is also considered and is then connected using the prediction of the ST-GCN. This allows us to improve the recognition of the actual action segment hierarchically. Of course, the hand gesture has to be recognized first, for which the "hand-gesture-recognition-using-mediapipe"[2] is used. As the name says, this uses Mediapipe [44] to get the skeleton trajectory of the hand. Mediapipe-Hands is able to recognize 21 hand landmarks only from RGB videos or photos.

[2] Code available at `https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe`

Palm Detector The first operation of Mediapipe is to start identifying all hands visible in an image. Therefore the Palm Detector is used. Hands exist in many different sizes and the network should be able to detect and landmark both smaller and larger hands of children and adults. For this purpose, the palm is searched to estimate the bounding box around it. This is done because the palm and fist are much easier to recognize than a hand with the fingers.

Hand Landmark Model After the 21 hand landmarks are searched for within the recognized bounding boxes. The landmarks consist of x, y, and a relative depth, estimated from the neural network. In addition, a certainty is calculated as whether a hand is actually in this area, and also predicted whether it is a left or a right hand. In Figure 2.3 the joints and edges are shown, which can be recognized by the Hand Landmark Model. These landmarks can now be used to recognize different gestures. In this work, a pre-trained model of "hand-gesture-recognition-using-mediapipe" is used.

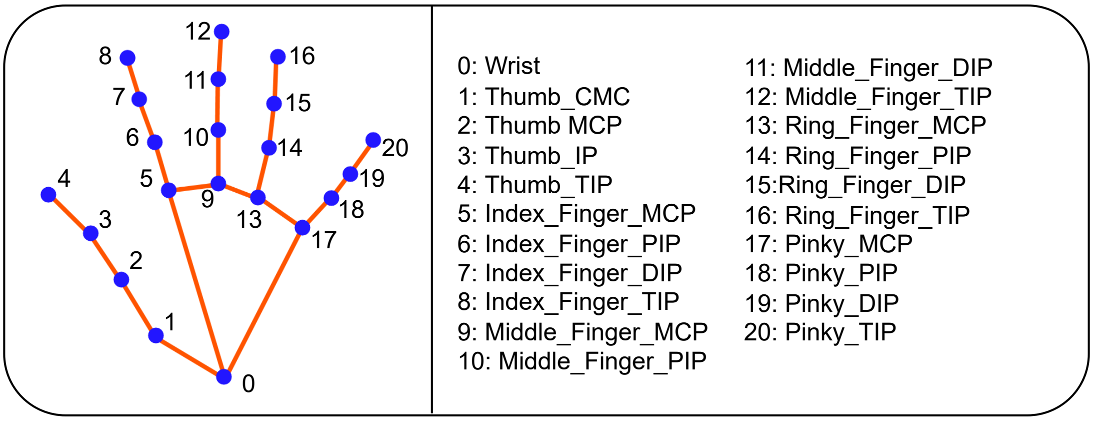


Figure 2.3.: Hand Landmarks

Hand Gesture Recognition The results of the hand landmark model are analyzed by the "hand-gesture-recognition-using-mediapipe" by Kazuhito Takahashi which uses the Mediapipe Hands network described above to recognize the different gestures and hand positions. The library is also able to create its datasets and easily label them, which is not used in this work. The pre-trained model by the library is used, which can recognize the following hand poses Open, Close, and Pointing. Open and Close refers to whether the hand is more open or more closed. Pointing means that all fingers are folded except for the index finger, which remains extended. When pointing, the fingertip is highlighted and the movement is interpreted over several frames. Here, "Clockwise Turning" and "Anti-Clockwise Turning" are recognized as examples.

2.1.4. Hidden Semi-Markov Model

A Hidden Semi-Markov Model is a separate form of a Hidden Markov Model. A HMM has the property that the probability of transitions to other states depends only on the current state. This is only partially the case with the HSMM.

Hidden Markov Model

There are two discrete-time random processes $\{A_t\}$ and $\{B_t\}$ with $t \in \mathbb{N}$. Here $\{A_t\}$ is not observable but hidden, this is where the name "Hidden Markov Model" comes from. Now we want to draw conclusions about the course of the random process we do not know with the help of the random process we do know. For this one uses the Markov properties.

1. The current value of the first process depends exclusively on its last value.
2. The current value of the second process depends only on the value of the first process.

Here, a Hidden Markov Model is described with a 5-tuple $\lambda = (S, V, A, B, \pi)$

- $S = \{s_1, ..., s_N\}$ - The set of all states of the random variable A_t.
- $V = \{v_1, ..., v_M\}$ - The alphabet of observation (emissions) of B_t
- $P \in \mathbb{R}^{N \times N}$ - The transition matrix between the states and gives the respective probabilities to change from one state to the other one
- $B \in \mathbb{R}^{N \times M}$ - The observation matrix and gives the probability to make observation v_m in state s_n
- $\pi \in \mathbb{R}^{N}$ -The initial distribution, so π_n describes with what probability s_n is the initial state

In order to use already labeled data for this learning method, the initial assignment π_n can be set specifically. For this purpose, the emission probabilities for each state s_n are characterized via a normal distribution with mean μ_n and the covariance $\sum_n$ as $\mathcal{N}(b_t; \mu_n, \sum_n)$. For a deeper insight into training HMMs and HSMMs, the following works are recommended [5, 26].

Hidden Semi-Markov Models

Now HSMMs have the same structure as HMMs, except that the unobservable random process does not satisfy the Markov properties listed above. In the HSMM, the probability that the hidden state changes vary depending on how long one has been in it since the state occurred. This also allows the modelling of random dependencies in the length of e.g. action segments.

2.2. Related Work

This section gives an overview over different approaches proposed. The section is categorized between action recognition and action segmentation. At the end in Section 2.2.3 a work is presented where the results of this work might be used to improve their results

2.2.1. Action Recognition

There are already many different approaches to classifying the different interactions. One of the most widely used is the 3D skeleton-base action recognition [29]. Here, the skeleton trajectory is extracted from the image or with the help of specialized cameras with depth sensors, instead of learning from the RGB video. Therefore handcrafted features were used, which consider the relative 3D rotations and translations between specific joints. However, the methodology of handcrafted features has only been successful on single specimen datasets [41]. For this reason and the fact of success of Deep Learning algorithms, more and more Convolutional Neural Networks (CNN) [7], Graph Convolutional Networks (GCN) [43], and also Recurrent Neural Networks (RNN) [17] have been used. These are then applied to the extracted skeleton trajectory to predict the action. There are two contexts for videos of interactions. One is the spatial context, which examines the joints in a frame. The other is the temporal context, which looks at the changes between frames. For the former, GCNs can be used, for which a graph must first be created from the joints of the skeleton. On the one hand, these can be created automatically or handcrafted according to the anatomy-derived human joints. For the latter, the joints that are connected to a bone are much more strongly related. this seems more logical, but some approaches also connect the symmetric joints hierarchically [16]. In ST-GCN [43] the temporal context is established with the help of Temporal Convolutional Networks (TCN). However, RNN-based Long Short Term Memory (LSTM) [11] can also be used to exploit the temporal context, which is used in multiple action recognition works [19, 38]. For a deeper insight a detailed overview can be found in [45, 27, 10, 42].

2.2.2. Action Segmentation

In action segmentation, there are many supervised [1]. semi-supervised [36] and unsupervised [14, 23] approaches. In supervised methods, such as Semi-supervised "*representation learning from surgical videos*" [1], a sequence-to-sequence transformer is used to divide the long input videos into smaller segments. This is done to counteract the typical over-segmentation in frame-by-frame segmentation.

With semi-supervised methods, there are multiple ways to give an unsupervised method the advancement of information based on a labeled subset of data. For example, in [35] the contrast learning part is making sure the unsupervised feature extraction also correlates the actual labels with the subset of labeled data. Another method is for example the initialization of an HSMM with properties of the already labeled data. A concrete example can be seen in Section 4.2.4.

Unsupervised methods use similarities of repeated executions. This is also used in transition state clustering (TSC) and can identify start and end points (transition states) within a demonstration. A hierarchical approach was taken. First, possible transition states are selected using a hierarchical Dirichlet process-Gaussian mixture model. The Dirichlet process is used because a demonstration can have a varying number of transition states. After a selection of intersection points is found using the different features, TSC is used to select the points that are transition states. An overview in a more datailed manner can be found in [42, 22]

2.2.3. Application Of Action Recognition and Segmentation

Physical Interactions for Social Robots

In [9], the generation of action responses of the robot with the help of Recurrent Wasserstein Autoencoder (RWAE) and Bayesian Interaction Primitives (BIP) [6] are proposed. They did not only try to train whole actions but also divided the actions into different segments to be able to train them in a more modular way. For this purpose, a classifier was built to recognize the different actions and segments. This prediction was then given to the decoder as an additional input to increase the accuracy. The different actions were recognized at an accuracy of around 80%. The segment had an accuracy of just around 60 percent and in some cases even just down to around 20% for some segment classes. The simple classifier approach of her has the drawback that different segments get confused with each other. This thesis aims to improve this to be then able to improve movement generation algorithms, which are using action segmentation.

3. Classification and Segmentation of Interactions

In this chapter, the problem is formalized in Section 3.1. In the next section, the proposed approach for solving the mentioned problem is described. In this section 3.2, the hierarchical structure of the proposed approach is described first. Subsequently, the individual modules are described and explained in the following subsections.

3.1. Problem Statement

The goal of this work is to detect different social actions. For this purpose, not only the skeleton trajectories of a person shall be used, but also the RGB video data. Specifically, only the demonstration of one person has to be sufficient. This constraint is especially to be beneficial for HRI Interaction motion generating works, which are also using classification and segmentation. Such a use case is introduced in Section 2.2.3. Additionally, not only the current action but also the current segments *Stand*, *Reach*, *Action*, and *Retrive* should be recognized. Furthermore, this method should work in a live environment, and therefore be fast and have a low delay from demonstration to prediction. Since the goal is to predict live data the method must work on noisy data of a KinectV2[1].

3.2. Proposed Approach

As we have seen in the work introduced in [9] the interaction segmentation has a rather bad accuracy in recognizing the segments, especially with *reaching*, *raise* and *retrieve*. This

[1] The Kinect V2 is a camera device with an inbuild skeleton detection model. Additional to the RGB Video an infrared sensor the distance can be obtained and the Joints coordinates can be calculated in the 3D-Space.

low performance is, even more, the case in multitask learning, but also in non-multitask learning these segmentations sometimes have an accuracy of only 50%. It can be deduced that it is difficult not only to recognize the actual action but also to segment the complete action at the same time. To improve this accuracy, a hierarchical approach was chosen.

In Figure 3.1 the complete flow of the data and the different networks can be seen. Here one can see again which data of the dataset is used for which network and how it is processed. First, the data captured from the KinectV2 and the hand landmarks of Mediapipe are preprocessed, which was described in Section 2.1.3. This data is then given to the first module.

Action Recognition In the first Module are the Action ST-GCN and the Point History Classifier. The Action ST-GCN is given the preprocessed skeleton trajectory and the information on whether the Hand of the actor is opened or closed. The Point History Classifier is given the hand trajectory. Both get a window of a predefined size. The two models predict the action and output its certainty. This output is then given to the second module.

Fusion In layer 2 the output of the models gets fused with the *Bayesian Classifier Fusion*. For the fusion, the certainty for each action and model is weighted and gives then a more robust output of the predicted action. The predicted action is then given to the third and final module and as an output of the complete Network.

Segment Recognition In the last module the predicted action and the features extracted from the the Action ST-GCN are then used to predict the current segment we are in. These predicted segments are then post-processed with the help of an HSMM. The final output is now a tuple of the current action and the current segment the action is in.

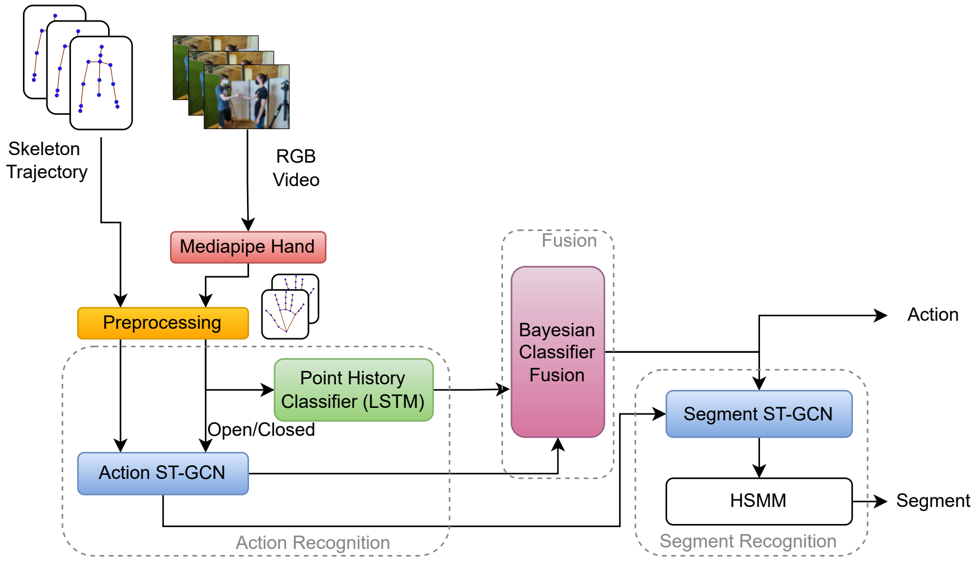


Figure 3.1.: Hierarchical Data Flow

3.3. Action Recognition

In this section, the first module of the model is proposed, which is capable of recognizing the different actions. Two different technologies are used and then get ensembled with the help of the *Baysian Network Fusion*.

3.3.1. Action Recognition With Skeleton Trajectory

The ST-GCN described in Section 2.1.2 was used to predict the action of the demonstration. It takes the different Joints gathered by the KinectV2 as input and gives the prediction certainty over the different actions as output.

Preprocessing

For the pre-processing of the skeleton trajectory mainly 2 things were done. Firstly the normalization of the rotation and the translation and then the filtering of the used joints. The translation was done so that the neck is in the origin. With this translation, it was then possible to rotate over around the neck. Then the skeleton was rotated so that the shoulders and the hips are on the Y-axis and the person is looking in the direction of positive X.

$$\begin{aligned}\vec{E}_{shoulders} =& J_{lshoulder} - J_{rshoulder}\\ \vec{Y}_{helper} =& J_{waist} - J_{rshoulder}\\ \vec{V}_{view-direction} =& \vec{E}_{shoulders} \times \vec{Y}_{helper}\\ \vec{V}_{spine} =& \vec{E}_{shoulders} \times \vec{V}_{view-direction}\end{aligned}$$

For this calculation, we need to get a vector in which direction the person is looking and the direction of the spine. To get $\vec{V}_{view-direction}$ we calculate the norm of the plane which is spanned by the shoulders and waist. With the $\vec{V}_{view-direction}$ and the $\vec{E}_{shoulders}$ we are now able to calculate the direction vector of the spine. For the translation, just the direction vector to the neck is negated.

$$Axis_x = \frac{\vec{E}_{shoulders}}{\vec{E}_{||shoulders||}},\ Axis_y = \frac{\vec{V}_{view-direction}}{||\vec{V}_{view-direction}||},\ Axis_z = \frac{\vec{V}_{spine}}{||\vec{V}_{spine}||}$$

$$rot = \begin{pmatrix} Axis_{x,x} & Axis_{x,y} & Axis_{x,z} \\ Axis_{y,x} & Axis_{y,y} & Axis_{y,z} \\ Axis_{z,x} & Axis_{z,y} & Axis_{z,z} \end{pmatrix}, trans = \begin{pmatrix} J_{neck,x} \\ J_{neck,y} \\ J_{neck,z} \end{pmatrix}$$

Since the recognition should work in real time as much as possible, the action must be divided into individual equally sized, overlapping windows. To be able to use the last T frames, a buffer of predefined window size, T is used. The buffer takes the last T frames and gives them to the different neural networks.

3.3.2. Action Recognition With Hand Pose

In order to improve the prediction and make it more robust, not only the prediction from the ST-GCN of the Action Recognition is used, but also the hand pose. This hand pose makes it possible to better distinguish between similar movements, provided that the hand poses are different. For example *Clap* and *Fistbump*, which have a high motion similarity. However, with *Fistbump* the hand is closed for the time of the action, but with the *Clap* action the hand remains permanently open. To gather the hand skeleton the video is given in *Mediapipe Hands*. The output is not only the hand skeleton but also whether the recognized hands are the left or the right hands of the actors. Since in the Nuisi-RGB dataset no left-hand actions were recorded, only the right-hand poses are used and filtered accordingly. Furthermore, it is ensured that only the right hand of the currently focused person is used.

Preprocessing

For the hand pose, the RGB-Video and the skeleton trajectory are matched, so that both the 3D joints and the hand pose have the same number of data points for the respective demonstrations. For the normalization it did not matter where the hand was located, initially, a window always started with the coordinates $(0, 0)$. Afterward, the relative distance to the initial position was always considered dependent to the size of the complete image.

$$\textit{hand-joint-pos}(x_i, y_i) = \begin{cases} (0,0) & \text{if } i = 0 \\ \frac{(x_i,y_i)-(x_0,y_0)}{(width,height)} & \text{otherwise} \end{cases}$$

For example, if one actor started with the hand in the center, the maximum values were 0.5 and -0.5.

3.4. Action Segmentation

The proposed method is also capable of predicting the current segment of the demonstration with prior knowledge of the predicted action. Therefore the prediction output of the action recognition is given to the segment classifier. In addition, the feature layers are reused from the action recognition to get the extracted features. This structure is shown in Figure 3.2. In this method, only the feed-forward layers are retrained to predict the current segment. With that, the feature extraction layers of the Action ST-GCN are partly reused. Additionally, it is reasonable to exploit different properties of the demonstrations like the length of the different segments and the fixed segment sequences. To achieve this a Hidden Semi Markov Model is used. The predictions of the Segment ST-GCN are getting buffered to be then able to post-process those predictions.

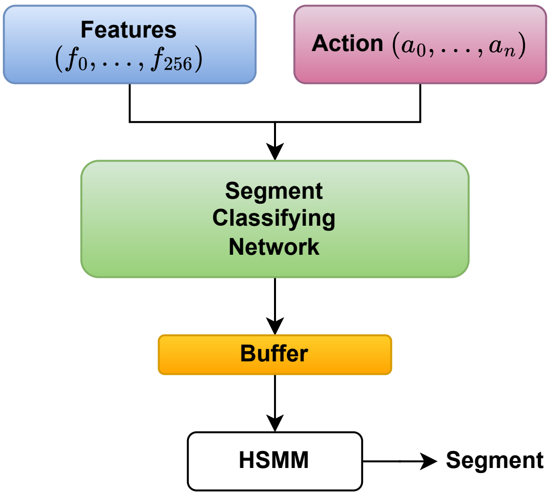


Figure 3.2.: Segment Recognition With Action As Prior Knowlegde

Algorithm 1: Algorithm For Live Inference

```
$B \leftarrow$ buffer($window\_size$);
while $capturing$ do
    $video, skeleton \leftarrow$ kinect_capture;
    $hand, pose, skeleton \leftarrow$ preprocess($video, skeleton$);
    $B$ push $hand, pose, skeleton$;

    if $B$ is filled then
        $A_{STGCN}, features \leftarrow$ ActionStGcn($B_{skeleton,pose}$);          /* Action */
        $A_{LSTM} \leftarrow$ ActionLSTM($B_{video}$);
        $A_{fused} \leftarrow$ BayesionFusion($A_{STGCN}, A_{LSTM}$);

        $S_{STGCN} \leftarrow$ SegmentStGcn($A_{fused}, features$);             /* Segment */
        $S_{HSMM} \leftarrow$ HSMM($S_{STGCN}$);

        Output($A_{fused}, S_{HSMM}$);
```

4. Experiments And Evaluation

4.1. Datasets

In the course of this work, different datasets were available, each of which has its advantages and disadvantages. NTU-RGB+D [33, 18], Shakefive2[1], and Bütepage et al. [4] were considered, but in the end, it was decided to use our dataset, which is more suitable for the application purpose and therefore comes closest to the real-world data. In the following, the difficulties listed led to this decision, and hence the settings were designed in a way to minimize the influences which were causing those problems.

4.1.1. Problems And Desired Properties

The various datasets already available have various problems which are not fixable for this thesis. Here these problems are listed and the desired properties are explained.

Distance

All different datasets had different distances between the camera and hands, which were as a result of it not always recognizable. However, these are needed to improve the recognition even further, with the proposed method. This was especially a problem with NTU-RGB+D. For this reason, a distance would be better here, which would also correspond to another person who is interacting with the other person. Accordingly, a camera position is best, that is as close as possible to the other person, but also so far away that you have the hands with the shooting angle of the camera at any time and any action in the picture.

[1] Shakefive2 is publicly available from `http://www2.projects.science.uu.nl/shakefive/`

Occlusion

With Shakefive there were always occlusions shortly after the interaction because the two actors walked past each other after the action. This caused the skeleton trajectories to randomly fly into the origin. This occlusion led to the skeleton trajectory not being recognized and displayed correctly. This resulted in poor recognition of the action class, especially during the action. It is important to avoid this occlusion as far as possible in a dataset, at best also the occlusion of the hand, for example when shaking hands.

Angle Of View

It was noticeable that the angles of view were mostly to the side of the interactors. The long-term goal of this bachelor thesis is to enable robots to recognize and interpret the movements of an opposite person and in the future to react accordingly. Hence a viewing angle in front of or slightly beside the actuators is more appropriate. The correct viewing angle would also allow the hand to be seen throughout the execution of the action and avoid occlusion as much as possible. To apply such a viewing angle for both actors, one camera for each actor might be needed.

4.1.2. Own Data Set - NuiSI-RGB

Due to the problems mentioned above, which make the other data sets difficult to use for our application area, it was decided to create our own data set. This data set should be more applicable to our application domain and should be as close as possible to the later real-world applications. Also, as mentioned above, one camera was used for each actor. The sensors are directed diagonally in front of the person and thus have a frontal view of the actor. The complete setup of the setting is shown in Figure 4.1. With the help of the frontal oblique view, the occlusion of body parts is to be prevented. Above all, however, the oblique angle of view ensures that the actor's hand can be seen at all times so that the action is always fully visible. In addition, it should be noted, the camera films the person slightly from the right, and thus even with a handshake, the hand is visible at all times. This way of setting up solves the *Occlusion* problem.

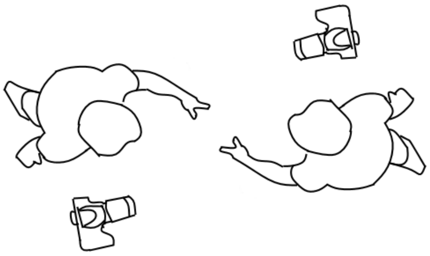


Figure 4.1.: Setting of NuiSI-RGB

Actions	6
Samples	120
FPS	15
Modalities	RGB, Depth, IR, 3D-Joints
Sensors	KinectV2
Subjects	2

Figure 4.2.: Features of Nuisi-RGB Dataset

4.2. Implementation Details

In this section, the implementation and handling of the data is described and gives an deeper insight of this work.

4.2.1. Spatial Temporal Graph Convolutional Networks

Here, the use of the framework from Yan et al. [43] is described, as well as how to prepare the data for the the built in Data Loader.

Data

For the ST-GCN, the 3D skeleton trajectories of the NuiSI dataset was used. The data were then preprocessed as in the section above. The proprocessed data is then put into the form $\{N, C, T, V, M\}$ so that the dataloader of the framework is able to process those.

N	Number of Samples
C	Number of Dimensions
T	Length of Input (Window Size)
V	Number of Joints
M	Number of Actors

Training

For training, the pre-trained model was only partially loaded, since it was not possible to apply all layers due to the different number of joins and data points. Afterward, backpropagation was applied to all layers. For this, Cross Entropy Loss is used as the loss function:

$$l(x,y) = L = \{l_1, l_2, ..., l_N\}^T, l_n = -w_{y_N} log \frac{exp(x_{n,y_n})}{\sum_{c=1}^{C} exp(x_{n,c})}$$

With the help of the pretrained model, it was possible to train the various models with 100 epochs each. In most cases, however, the maximum accuracy was already reached after 30 to 40 epochs.

4.2.2. Point History Classifier

To classify the actions and segments of the actions, the *Point History Classifier*[2] was used as a basis and slightly modified. Originally two different classifiers were introduced. One is the *Keypoint Classifier*, which takes all joints of the hand of a frame as input and then predicts the pose. The other is the *Point History Classifier*, which interprets the movement of the fingertip of the index finger and uses the last 16 frames. In order to be able to use the complete hand and the movement for the classification, the *Point History Classifier* was changed in such a way that it no longer only uses the x and y coordinate of the fingertip per frame, but also like the *Keypoint Classifier* all joints of the hand can be used.

[2] The code is available at `https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe`

Data

For the hand pose classifier, only RGB video data is needed. Using Mediapipe, the 25 joints of the hand were created as in Section 2.1.3. Then the data was preprocessed as described above and normalized into the sliding window form. This data had to be converted into a csv file with the following form for each data point or row

$$(label_{nr}, posX_{J_1,F_1}, posY_{J_1,F_1}, ..., posY_{J_N,F_1}, posX_{J_1,F_2}, ..., posY_{J_N,F_M})$$

with

N	Number of Samples
M	Frames Window Size
J	Joint
F	Frame

Training

For the training, the simple neural network was used which had an additional *LSTM* layer. The model was written in Keras and only needed an adjustment of the layer sizes due to the changed number of joins. The training itself was executed on a maximum of 1000 epochs, whereby the training could be terminated automatically after 500 to 900 epochs by different stopping criteria. The loss used was the *sparse categorical cross-entropy loss* which is calculated with

$$SCE = -\sum_{neuron=1}^{classes} y_{true_{neuron}} * ln(y_{pred_{neuron}})$$

The training runs are further explained in Section 4.3.2, where also the evaluations during the training are visualized.

4.2.3. Bayesian Network Fusion

Now the results from the Action ST-GCN and the Point History Classifier had to be merged. For this, as described before, the Bayesian Network Fusion of Trick et. al. [37] was used. This uses the output of the different classifiers to create a correlated fusion model. This can then be used to fuse the different outputs to make the overall output better.

Data

As training data, the test data was run on both the Action ST-GCN and the Point History Classifier (with LSTM). The respective output vectors were then converted into vectors of probabilities for the different classes using a softmax. The probability p for the respective class j for an output vector $o = \{o_0, o_1, ..., o_{|J|-1}\}$ is defined as

$$p(x)_j = \frac{e^{x_j}}{\sum_k e^{x_k}}$$

The input to the Bayesian Fusion Net has now been transformed into the following shape {N, K, J}.

N	Number of Samples
K	Number of Classifiers
J	Number of Classes

4.2.4. HSMM

The HSMM was trained using the predictions of the Segment ST-GCN. These corresponded to the segment classes and were now given as an entire demonstration as input. That means the input had a shape of $\{D, F_d, S\}$ with

D	number of demonstrations
F_d	number of frames of each demonstration,
S	number of segments.

To initialize this unsupervised method, the average and the covariance were calculated for each segment class. This can then be used to train the HSMM with the training data.

4.3. Experiments On NuisiRGB

This section describes how to perform the experiments on the data set we created. The different labels are explained in more detail and the labeling of the segments is described in more detail. This defines more precisely when a segment begins and when a segment ends. Afterward, the different training runs and their results are presented. In addition, further experimental setups and their execution are presented.

4.3.1. Labeling

In the Tables, the different action classes and the labeling of the segments are described.

Action Classes

In this table each action is described individually to define the different actions exactly. Thereby you can already get an idea of how exactly to divide these actions into different segments and why the different actions all consist of similar segments and can be subdivided accordingly.

Action	Description
Fistbump	A typical fistbump is where one first closes the fingers into a fist. Then both people bring the fist in front of the body, where the two fists touch for a short time. After that, the hand is taken back to its original position.
Clapfist	A different version of the fistbump. Right before the fistbump gets performed a clap is done. With the clap, the movement of the handshake is initially carried out, only that here the hand is not gripped but only strongly clapped against each other. Then, in a fluid motion, the fist bump is executed without returning to the intimate position.
Handshake	A typical handshake is where both persons grasp each other's hands in front of their bodies. They then move the joined hands in an oscillating up and down motion. This motion gets repeated for some time until the actors let go of each other's hands and get back to the initial position.
Highfive	A classical highfive where the actors raise their hands in front of them at the level of their heads. There, they clap the two hands together with some momentum. Then the hands are returned to the initial position.
Wave	The actors raise their hands next to their heads with the palm facing the other person. Then an oscillating left and right motion is performed for some time. After that, the actors lower their hands and return to the initial position.
Rocket	An extended version of the fistbump. The actors align their fists in front of their bodies, with one of the fists over the other one. Then they both raise their fists to the level of their heads where they stop and return to the initial position. This motion depicts a starting rocket that then explodes.

Segment Classes

The complete demonstration of the interactions was now divided into 4 classes. The default where every action is beginning with is *Standing*. *Standing* describes the waiting of one of the actors, to begin with, one of the actions by starting *Reaching* with its hand. Also after an action ends the actor is going back into the *Standing* state. The first actual part of the action is *Reaching*, where the actor moves his hand to that of the interaction partner. Then the *action* segment starts, as long as the action type is prolonged. After that, the *Retrieving* segment begins, till the person reaches their initial position.

Segment	Description
Standing	The initial segment describes the default state where no action is performed. The right hand is placed against the body and is rather still. The segment begins as soon as the hand starts moving to go to the region where the action is performed. The segment ends as soon as the hand is placed again against the body and stops moving.
Reaching	With the reaching segment, the interaction of the demonstration starts and describes the time interval between standing and reaching the action region or touching the other person's hand. As soon as the Hand starts moving to the action region the reaching segment begins. The segment ends as soon the hands are touching or when the perpetual motion begins like in the wave action.
Action	After the reaching segment, an action segment can be entered. This is the case when performing an *prolonged* action. The action segment starts the moment the action region is reached and ends as soon as the action region is left and the hand motion directs back to the initial position.
Retrieving	As soon as the action region is left and the hand starts moving back to the initial position the Retrieving segment starts. It ends the moment the initial position is reached.

Prolonged And Instantaneous Actions

As mentioned before a differentiation between instantaneous and prolonged action was made. This has an effect on the way of labeling the actions. The prolonged actions are clapfist, handshake, rocket, and wave since each of these actions was having a prolonged interaction with the other person. Instead, highfive and fistbump are just having one instant point connection between the reaching and the retrieving segment. This is where the segment sequences differ from each other, which can be seen in Figure 4.3.

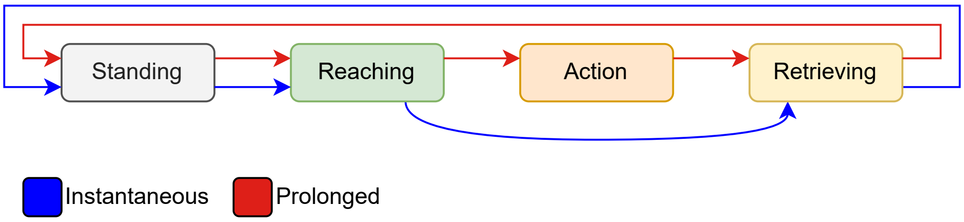


Figure 4.3.: This Graph shows the difference between the Instantaneous and Prolonged Segment Sequences.

4.3.2. Training

This subsection describes the final training runs of the different networks and shows it progress of the accuracy of the evaluation dataset. The accuracy is the mean F1-score over all six labels. After this section, those results will be evaluated and classified from various points of view.

Baseline Action Recognition

For the baseline, the basic ST-GCN got trained on the NuisiRGB skeleton trajectories. The network was trained for over 250 Epochs but converged to just over 90 percent after 15 Epochs since the pretrained model was just finetuned. The evaluation accuracy is shown in Figure 4.4. It is immediately noticeable that the basic ST-GCN already works well out of the box, and also does not overfit on the rather few data points of just under 5000.

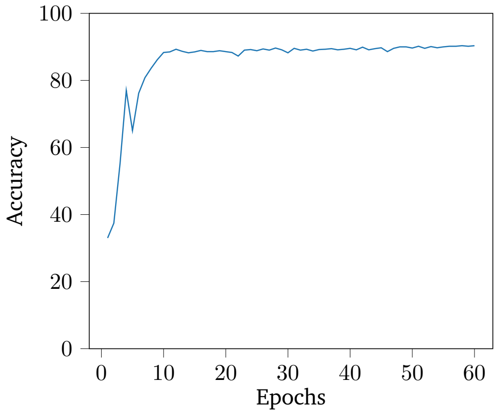


Figure 4.4.: The Evaluation Accuracy of the Baseline ST-GCN.

Action ST-GCN

The Action ST-GCN got trained with the additional data on the certainty of whether the hand is opened or closed. This led to an increased accuracy of 2% with a total of 92.4% which shows that the additional information gives a decent value to the network. The network was trained for 500 epochs but was also converging pretty fast due to the pre-trained features. The evaluation accuracy during the training is shown in Figure 4.5.

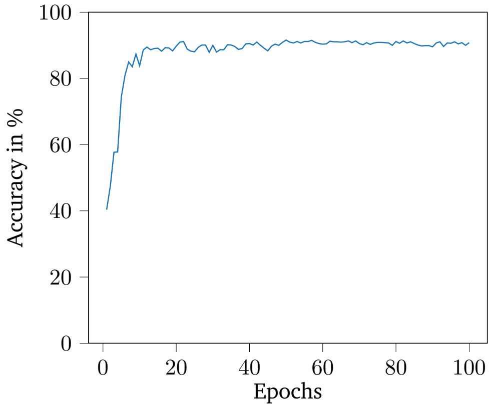


Figure 4.5.: The Evaluation Accuracy of the Action ST-GCN.

Action Point History Classifier

The Action Point History Classifier using the LSTM was trained with the hand skeleton data. The model weights got randomly initialized and were trained for 500 Epochs. In Figure 4.6 the graph of the evaluation accuracy during the training is shown. The training converged to the maximum accuracy after around 350 Epochs with an accuracy of 92%.

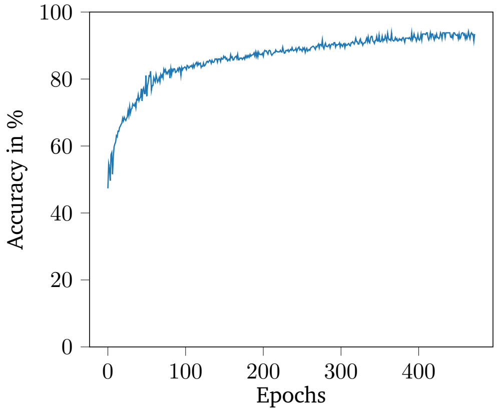


Figure 4.6.: The Evaluation Accuracy of the Action Point History Classifier during the Training.

Segment ST-GCN

In Figure 4.7 can be see that the accuracy is already very good after the first epoch. This shows that the extracted features and the given action prediction work well. The accuracy ends afterward at an accuracy of 86%.

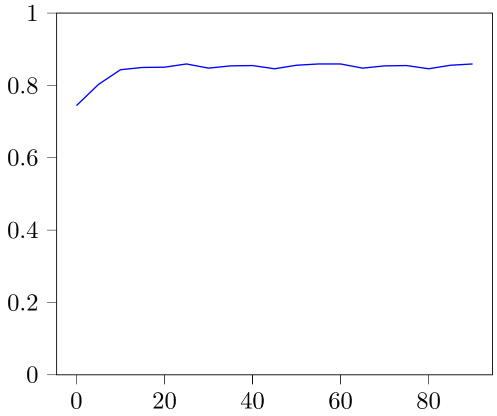
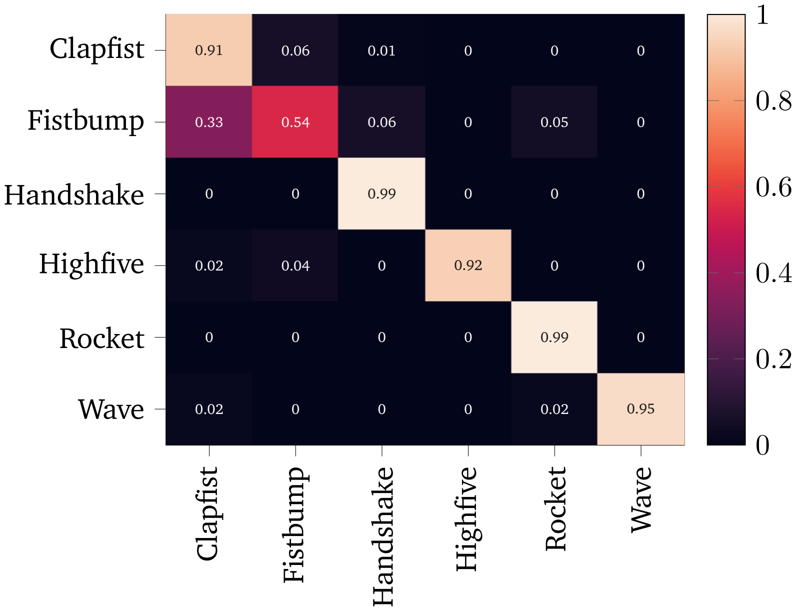


Figure 4.7.: The Evaluation Accuracy of the Segment ST-GCN during the training.

4.3.3. Evaluation

In this chapter, the results of Action Recognition, the Segment Recognition are presented. First, the baseline of action recognition and segmentation are looked at. Then confusion matrices of the different proposed networks are shown. Within this, a distinction is made between Skeleton Recognition and Hand Pose Recognition and then the effects of ensemblement using the Bayesian Net are explained. Afterward those results are compared to the baseline.

Baseline



	Clapfist	Fistbump	Handshake	Highfive	Rocket	Wave
Clapfist	0.91	0.06	0.01	0	0	0
Fistbump	0.33	0.54	0.06	0	0.05	0
Handshake	0	0	0.99	0	0	0
Highfive	0.02	0.04	0	0.92	0	0
Rocket	0	0	0	0	0.99	0
Wave	0.02	0	0	0	0.02	0.95

Figure 4.8.: Confusion Matrix of the Baseline ST-GCN.

You can see in Figure 4.8 that ST-GCN already works well in action recognition, but Fistbump is very strongly confused with Clapfist. This needs to be improved with the help of the ensemble and the hand pose. In total an accuracy of just over 90% is reached.

Action ST-GCN

Action Recognition is the first level of the hierarchical segmentation and is predicting which of the actions is currently performed.

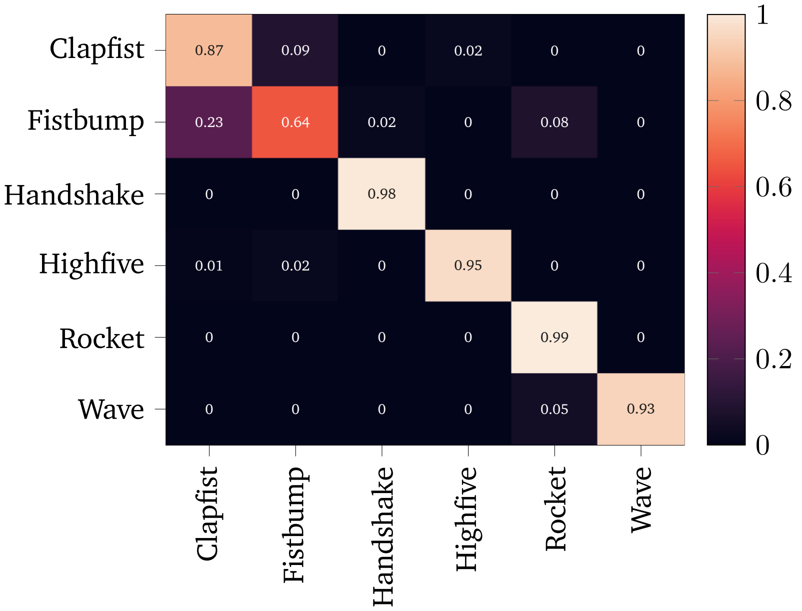


Figure 4.9.: Confusion Matrix for Recognition of all Actions with ST-GCN

Figure 4.9 shows the confusion matrix for Action Recognition with Skeleton Trajectory. You can see that all of them are recognized very well, except for the action *wave*. The action *fistbump* is interchanged with *clapfist*, which is quite intuitive that this would happen, cause of the similar fistbumps at the interactions. This assumption is also confirmed by a deeper investigation since the confusions mostly occur around the touching part, where the fistbump is performed. The overall accuracy of the Action ST-GCN is 92.14% and therefore performs 2% better than the plain ST-GCN without pose estimation input.

Action Point History Classifier

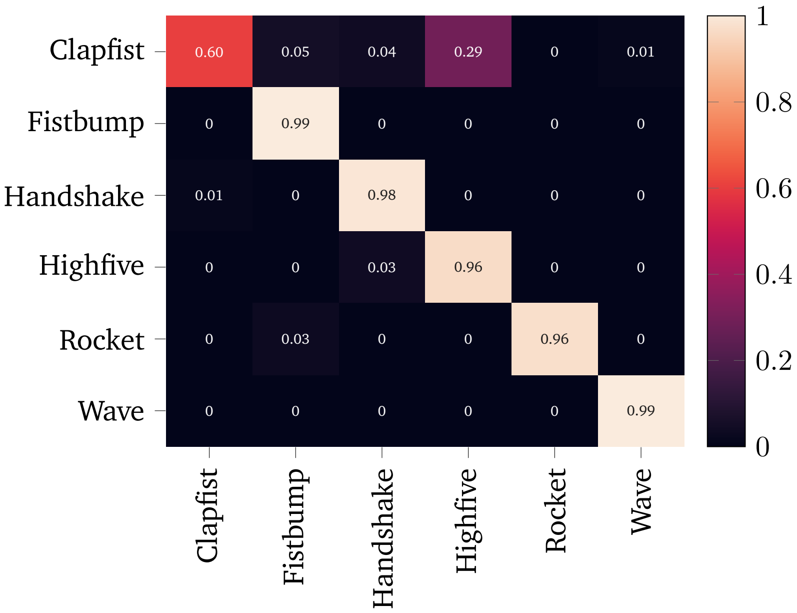


Figure 4.10.: Confusion Matrix for Recognition of all Actions with Hand Pose

Figure 4.10 shows the confusion matrix for Action Recognition with Hand Pose. Using Hand Pose Recognition, all classes except *clapfist* are recognized very well. Here it is especially hard to explain why *highfive* is often predicted instead of *clapfist*. Nevertheless, the detection of clapfist is inaccurate in relation to the other action classes. At first, this is not a problem due to the ensemble using the Bayesian Net, since this weakness can be compensated well. But the clapfist has also a relatively low accuracy on the Action ST-GCN with an accuracy of 87%. Here the performance of the Bayesian Fusion will be very interesting. Overall, hand pose recognition has an accuracy of 93% over all actions.

Fusion With Bayesian Net

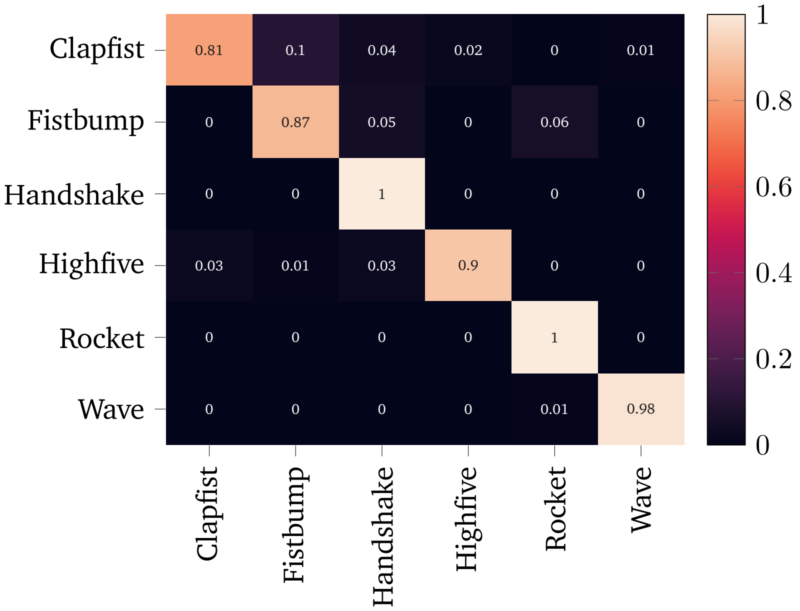


Figure 4.11.: Confusion Matrix for Recognition of All Actions with Bayesian Net Fusion

Figure 4.11 shows how the ensemble is improved by Bayesian Net. The accuracy was improved from 92% of the Action ST-GCN and 93% from the Point History Classifier to an accuracy of 95%. Therefore being able to improve the Baseline Action ST-GCN from just over 90% to over 95%. This shows that the proposed approach has been able to increase the action recognition accuracy.

4.4. Segment Recognition

This subsection presents the recognition of the current segment, i.e. *stand*, *reach*, *action*, and *retrieve*. As described above, first the segment is predicted with the Segment ST-GCN and afterward post-processed with the HSMM.

ST-GCN Baseline

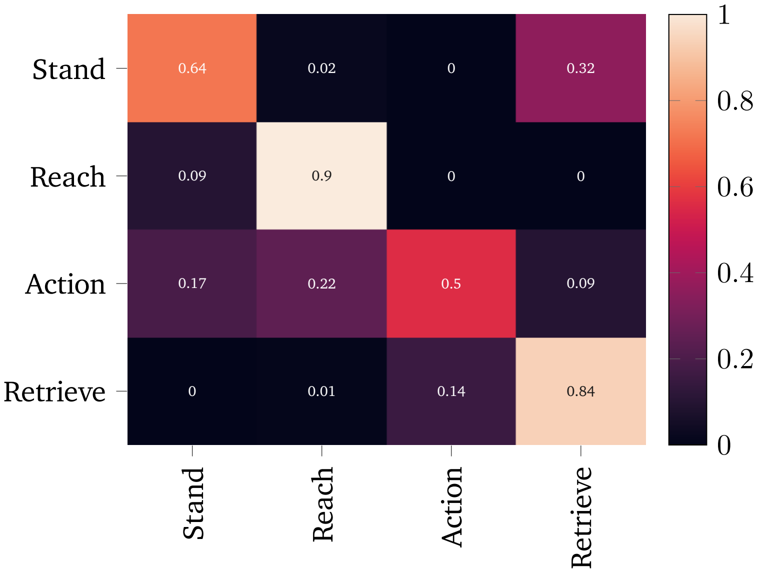


Figure 4.12.: Confusion Matrix of the ST-GCN Baseline for Segmentation.

Figure 4.12 shows the confusion matrix of the ST-GCN Baseline trained on the segment classes. It shows that *Reach* and *Retrieve* are badly recognized with the plain ST-GCN. The overall accuracy over the classes is 72%.

Segment ST-GCN with HSMM

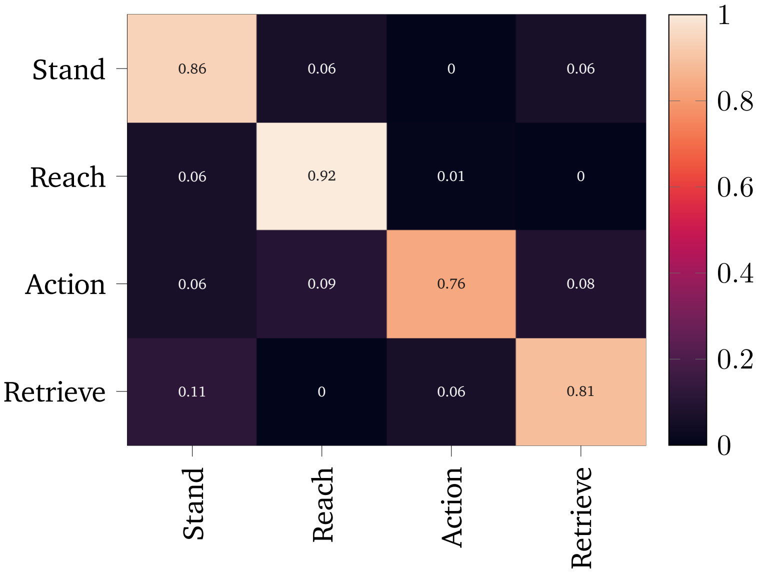


Figure 4.13.: Confusion Matrix of the Segment ST-GCN.

Figure 4.13 shows how well the individual segments are recognized and with which they are confused. It can be seen that, except for the *action* segment, only segments adjacent to each other in the sequence are confused with each other. This shows that, as expected, the transitions are the most difficult to recognize and that errors occur there accordingly. It shows that the action given additional to the input and also the HSMM is giving an advantage to the plain ST-GCN shown in Figure 4.12.

4.5. Realtime Performance

In addition to testing on the NuisiRGB dataset, experiments were also conducted in a live environment. For this, a person was asked to perform the movement of various interactions as a dry demonstration, as if a second person was there. The demonstrations were repeated five to seven times in one run. The demonstration was recorded with KinectV2, which was also used to obtain the NuisiRGB dataset. Afterward, the best recognition of each action is presented in the following sections and the appendix. These tests are showing if and which actions can be well recognized and/or segmented. In the following, the Handshake live demonstration is shown in detail. Afterward, the performance of other interactions are discussed. The series of frames from more actions can be found in the appendix.

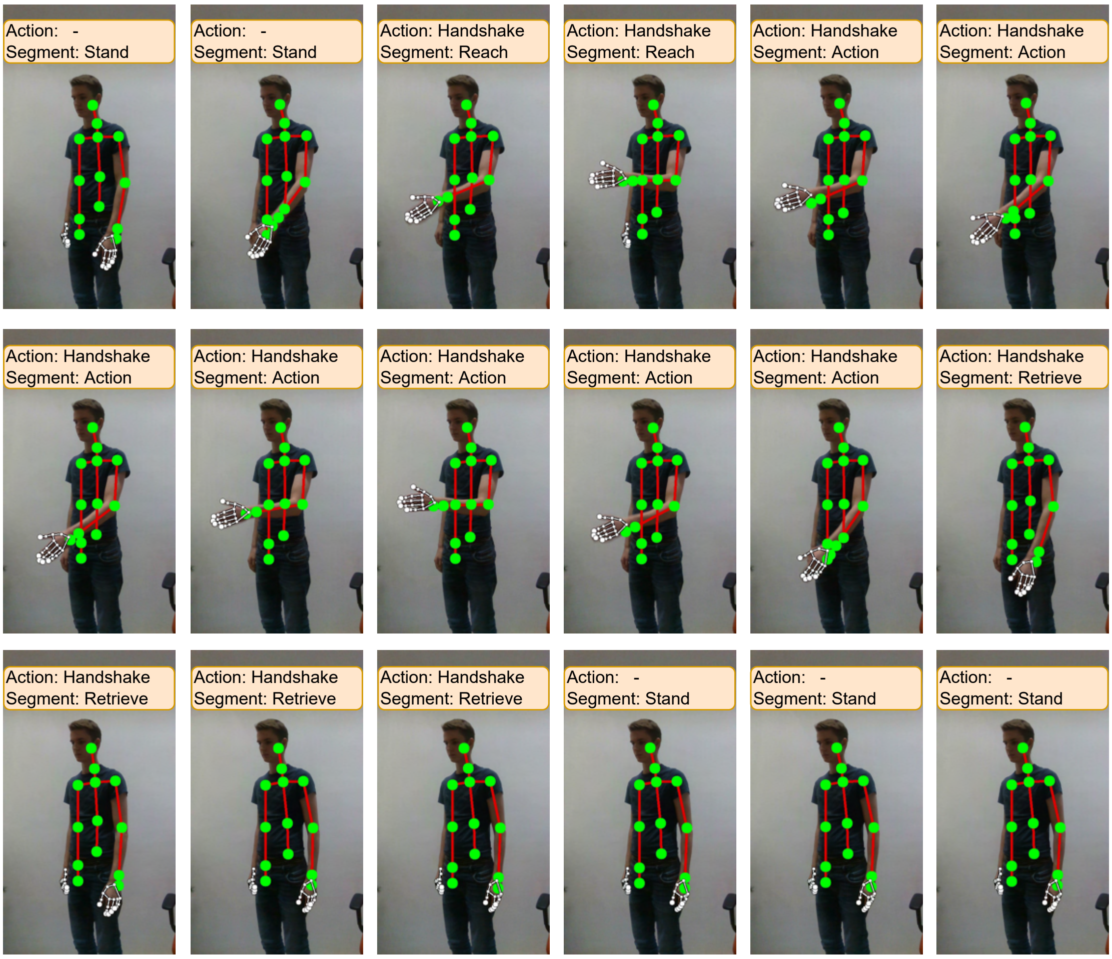


Figure 4.14.: The image series of a live demonstration of a handshake, which was recognized and segmented using the presented method. The prediction is written in the orange block and corresponds to the live prediction.

Action	Action Recognition and Segmentation Performance
Handshake	As seen in Figure 4.14 the action got perfectly recognized over the whole demonstration. The handshake was recognized very well in each demonstration in the test. In addition, the shaking movement was also tried for different lengths of time and the recognition also worked without problems. The segmentation also showed no problems for shaking segments of different lengths and has very good accuracy in the live environment.
Wave	As seen in Figure A.3 the Wave action is also recognized very well with only one misclassification at the beginning and the end of the retrieving. The segmentation is also working well and is just having a flaw at the end when not changing fast enough to the stand segment. Both recognition and segmentation were working pretty solid on all of the test demonstrations.
Clapfist	Figure A.4 shows that clapfist is also recognized well and is only slightly confused with a fistbump. This was also the case in most executions of the test, with a few exceptions. In the segmentation, the action segment between clap and fistbump was usually recognized a little too late, but the remaining segments were predicted very well.
Fistbump	Figure A.5 shows that fistbump is often confused with clapfist, which was at least partly to be expected since the second part of clapfist is similar to that of fistbump. Accordingly, it can be seen that clapfist is predicted preferentially with the fistbump movement. The segmentation worked very well in all demonstrations with the limitation that the touch action is predicted for one to two frames.
Rocket	Figure A.2 shows that the rocket is poorly recognized. The Segmentation instead was still really good and robust even though the Action wasn't well predicted.
Highfive	Highfive was also not recognized at all in the live environment and was always recognized as a clapfist. But the segmentation worked well also with the instantaneous setting and the action segment was mostly jumped as shown in Figure A.1.

5. Discussion

In the last chapter, the results of this thesis are summarized and presented. In Section 5.2, topics are addressed which were not part of this works focus. Also, an outlook on future work is given, which can be supported by this thesis.

5.1. Conclusion

In this thesis, a specific approach was presented to recognize and segment social human interactions. For this approach, both skeletal trajectories and RGB video data were used as input. The interaction detection and segmentation were performed hierarchically by first predicting the action and then using this prediction to perform the segmentation. Action recognition was performed using the skeletal trajectories of the body and the hand-skeletal data extracted from the RGB video data, each with two different models. Accordingly, the differences in hand pose between the different interactions were exploited. These were then merged using a Bayesian fusion network to increase accuracy. For the segmentation, the features of the interaction recognition were reused and a complete demonstration was divided into a total of 4 segments. These predicted segments were then injected into an HSMM to exploit the properties of the order and length of the segments. Thus, the ST-GCN could be improved and optimized for the presented task.
However, it also turned out that only some actions are mostly recognized correctly in a live recognition with data in different settings. The Wave, Handshake, and Clapfist are recognized well, Highfive and Fistbump are somewhat lost and Rocket is only recognized in the upward movement. The segmentation, on the other hand, was very successful, reliably being able to determine the current segment of all actions, even in live generated data. Only in the instantaneous actions, the action segment is recognized from time to time. Accordingly, it can be seen that the method presented also works in live environments, with the segmentation, in particular, standing out.

5.2. Outlook

The segmentations of all actions, especially with the HSMM, also worked very well during the live inference and only the transition points were slightly shifted, mostly to the back. The recognition of the different actions worked well for 4 of the 6 actions but Rocket as well as Highfive could not be recognized well. It was not found why especially rocket was not recognized well during the live inference, since the Rocket action has a very distinct motion during the action segment. Accordingly, the investigation was not one of the main focuses and thus remains as a topic for further research in a future work For each action-segment tuple with predefined robot movement actions, an actual interaction with a robot was started. Thereby, recorded live actions of a person are segmented by the presented method and the action is predicted. Depending on which action is predicted, a fixed sequence of movements will be performed. The first form of this can be seen in B.1. Unfortunately, the time was not sufficient to completely implement it. Afterward, it would have been possible to transfer this simulation to the Pepper robot relatively easily. This would allow us to compare the interaction of the segmented hardcoded movement with completely hardcoded movements of the robot and to perform user studies. This user study could then be used to determine whether this form of interaction feels more natural and safe for different people.

I leave the connection of this work with a motion generating work, like MILD [28], open for future work. This would create a specific implementation to not only predict the current action and segment but also directly generate the correct response motion of the robot using this information.

References

[1] Nadine Behrmann et al. “Unified Fully and Timestamp Supervised Temporal Action Segmentation via Sequence to Sequence Translation”. In: (2022).

[2] James O. Berger. “Statistical Decision Theory and Bayesian Analysis.” In: (1985).

[3] Avrim Blum. “Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain”. In: *Machine Learning* 26.1 (1997), pp. 5–23.

[4] Judith Bütepage et al. “Imitating by Generating: Deep Generative Models for Imitation of Interactive Tasks”. In: *Frontiers in Robotics and AI* 7 (2020).

[5] Sylvain Calinon. “A tutorial on task-parameterized movement learning and retrieval”. In: *Intelligent service robotics* 9.1 (2016), pp. 1–29.

[6] Joseph Campbell and Heni Ben Amor. “Bayesian Interaction Primitives: A SLAM Approach to Human-Robot Interaction”. In: (2017).

[7] Guilhem Chéron, Ivan Laptev, and Cordelia Schmid. “P-cnn: Pose-based cnn features for action recognition”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 3218–3226.

[8] J. Fink et al. “Living with a Vacuum Cleaning Robot.” In: (2013).

[9] Martina Gassen. “Learning a library of Physical Interactions for Social Robots”. In: (2021).

[10] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. “Going deeper into action recognition: A survey”. In: *Image and vision computing* 60 (2017), pp. 4–21.

[11] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[12] Stanislav Ivanov, Craig Webster, and Katerina Berezina. “Robotics in Tourism and Hospitality”. In: *Handbook of e-Tourism*. Ed. by Zheng Xiang et al. Cham: Springer International Publishing, 2020, pp. 1–27. ISBN: 978-3-030-05324-6.

[13] Robert A. Jacobs. “Methods for combining experts’ probability assessments.” In: (1995), pp. 867–888.

[14] S Krishnan et al. “Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. The International Journal of Robotics Research.” In: (2017).

[15] Min-Kyu Kwak, JeungSun Lee, and Seong-Soo Cha. “Senior Consumer Motivations and Perceived Value of Robot Service Restaurants in Korea”. In: (2021). ISSN: 2071-1050.

[16] Jungho Lee et al. “Hierarchically Decomposed Graph Convolutional Networks for Skeleton-Based Action Recognition”. In: (2022).

[17] Guy Lev et al. “Rnn fisher vectors for action recognition and image annotation”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 833–850.

[18] Jun Liu et al. “Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.10 (2019), pp. 2684–2701.

[19] Jun Liu et al. “Spatio-temporal lstm with trust gates for 3d human action recognition”. In: *European conference on computer vision*. Springer. 2016, pp. 816–833.

[20] Heather C. Lum. “Chapter 7 - The role of consumer robots in our everyday lives”. In: *Living with Robots*. Academic Press, 2020, pp. 141–152.

[21] Neelima Mishra, Dinesh Goyal, and Ashish Dutt Sharma. “Automation in restaurants: ordering to robots in restaurant via smart ordering system”. In: *Int J Technol Manage* 4 (2018).

[22] Yujian Mo et al. “Review the state-of-the-art technologies of semantic segmentation based on deep learning”. In: *Neurocomputing* 493 (2022), pp. 626–646.

[23] Adithyavairavan Murali et al. “TSC-DL: Unsupervised trajectory segmentation of multi-modal surgical demonstrations with Deep Learning”. In: (2016).

[24] Mahesh Pal. “Random forest classifier for remote sensing classification”. In: *International journal of remote sensing* 26.1 (2005), pp. 217–222.

[25] Deepika Phutela. “The Importance of Non-Verbal Communication”. In: (2016).

[26] Emmanuel Pignat and Sylvain Calinon. “Learning adaptive dressing assistance from human demonstration”. In: *Robotics and Autonomous Systems* 93 (2017), pp. 61–75.

[27] Ronald Poppe. “A survey on vision-based human action recognition”. In: *Image and vision computing* 28.6 (2010), pp. 976–990.

[28] Vignesh Prasad et al. *MILD: Multimodal Interactive Latent Dynamics for Learning Human-Robot Interaction*. 2022.

[29] Bin Ren et al. “A survey on 3d skeleton-based action recognition using learning method”. In: *arXiv preprint arXiv:2002.05907* (2020).

[30] Dymitr Ruta and Bogdan Gabrys. “An overview of classifier fusion methods”. In: *Computing and Information systems* 7.1 (2000), pp. 1–10.

[31] S. Šabanović. “Robots in Society, Society in Robots”. In: (2010).

[32] Haydar Sahin and Levent Guvenc. “Household robotics: autonomous devices for vacuuming and lawn mowing [Applications of control]”. In: *IEEE Control Systems Magazine* 27.2 (2007), pp. 20–96.

[33] Amir Shahroudy et al. “Ntu rgb+ d: A large scale dataset for 3d human activity analysis”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1010–1019.

[34] Thomas B. Sheridan. “Human–Robot Interaction: Status and Challenges”. In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* (2016).

[35] Dipika Singhania, Rahul Rahaman, and Angela Yao. “Iterative contrast-classify for semi-supervised temporal action segmentation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 2. 2022, pp. 2262–2270.

[36] Ajay Kumar Tanwani et al. “Motion2vec: Semi-supervised representation learning from surgical videos”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 2174–2181.

[37] Susanne Trick and Constantin Rothkopf. “Bayesian Classifier Fusion with an Explicit Model of Correlation”. In: *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Ed. by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera. Proceedings of Machine Learning Research. PMLR, 2022.

[38] Amin Ullah et al. “Action recognition in video sequences using deep bi-directional LSTM with CNN features”. In: *IEEE access* 6 (2017), pp. 1155–1166.

[39] Iwan Ulrich, Francesco Mondada, and J.-D. Nicoud. “Autonomous vacuum cleaner”. In: *Robotics and Autonomous Systems* (1997). Intelligent Robotic Systems SIRS’95. ISSN: 0921-8890.

[40] Milos Vasic and Aude Billard. “Safety issues in human-robot interactions”. In: *2013 ieee international conference on robotics and automation*. IEEE. 2013, pp. 197–204.

[41] Lei Wang, Du Q Huynh, and Piotr Koniusz. “A comparative review of recent kinect-based action recognition algorithms”. In: *IEEE Transactions on Image Processing* 29 (2019), pp. 15–28.

[42] Daniel Weinland, Remi Ronfard, and Edmond Boyer. “A survey of vision-based methods for action representation, segmentation and recognition”. In: *Computer vision and image understanding* 115.2 (2011), pp. 224–241.

[43] Sijie Yan, Yuanjun Xiong, and Dahua Lin. “Spatial temporal graph convolutional networks for skeleton-based action recognition”. In: *Thirty-second AAAI conference on artificial intelligence*. 2018.

[44] Fan Zhang et al. “Mediapipe hands: On-device real-time hand tracking”. In: *arXiv preprint arXiv:2006.10214* (2020).

[45] Jing Zhang et al. “RGB-D-based action recognition datasets: A survey”. In: *Pattern Recognition* 60 (2016), pp. 86–105.

A. Live Demonstrations

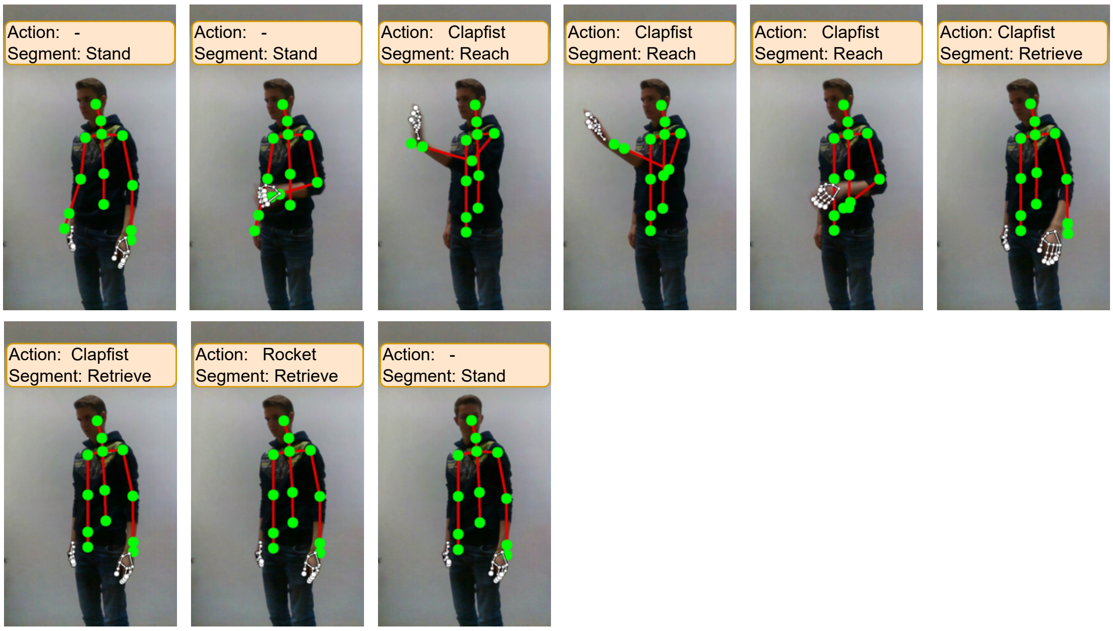


Figure A.1.: The image series of a live highfive demonstration, which was recognized and segmented using the presented method.

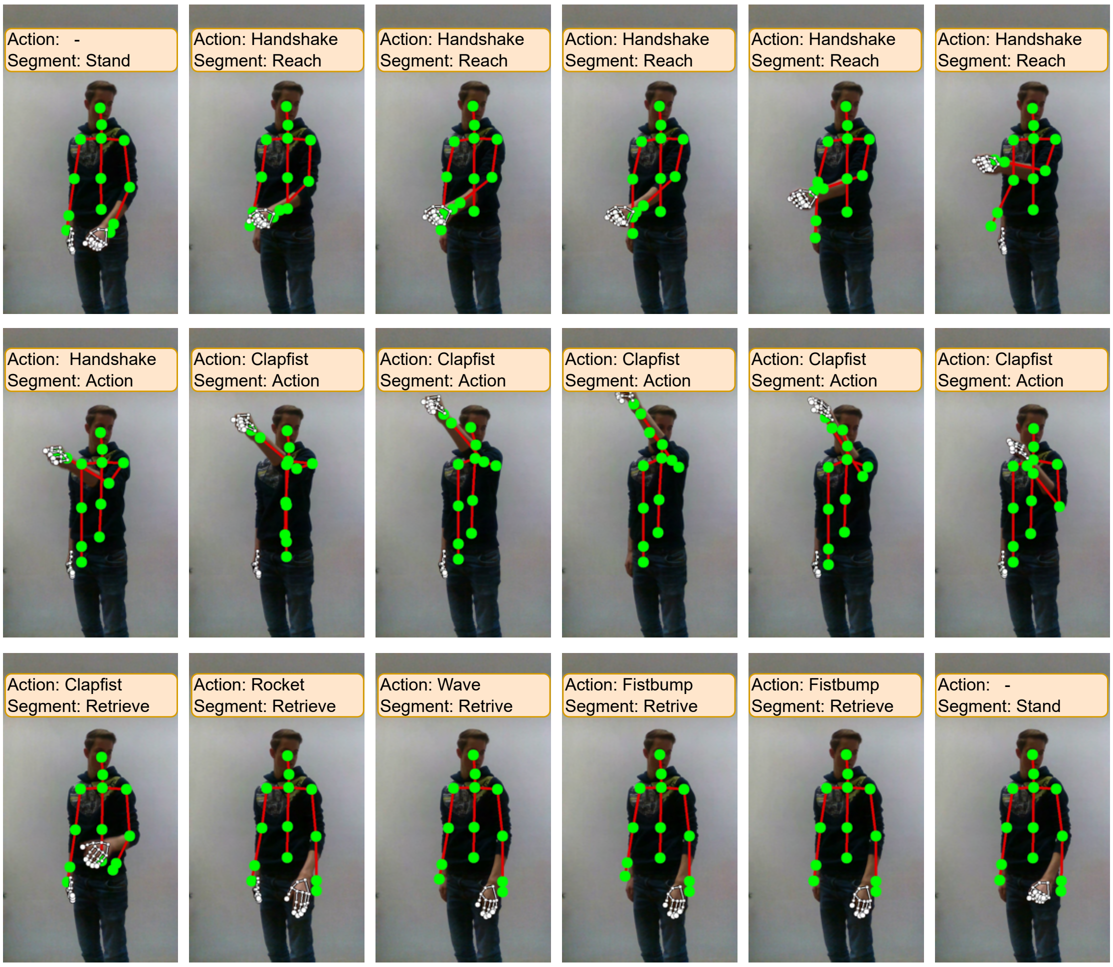


Figure A.2.: The image series of a live highfive demonstration, which was recognized and segmented using the presented method.

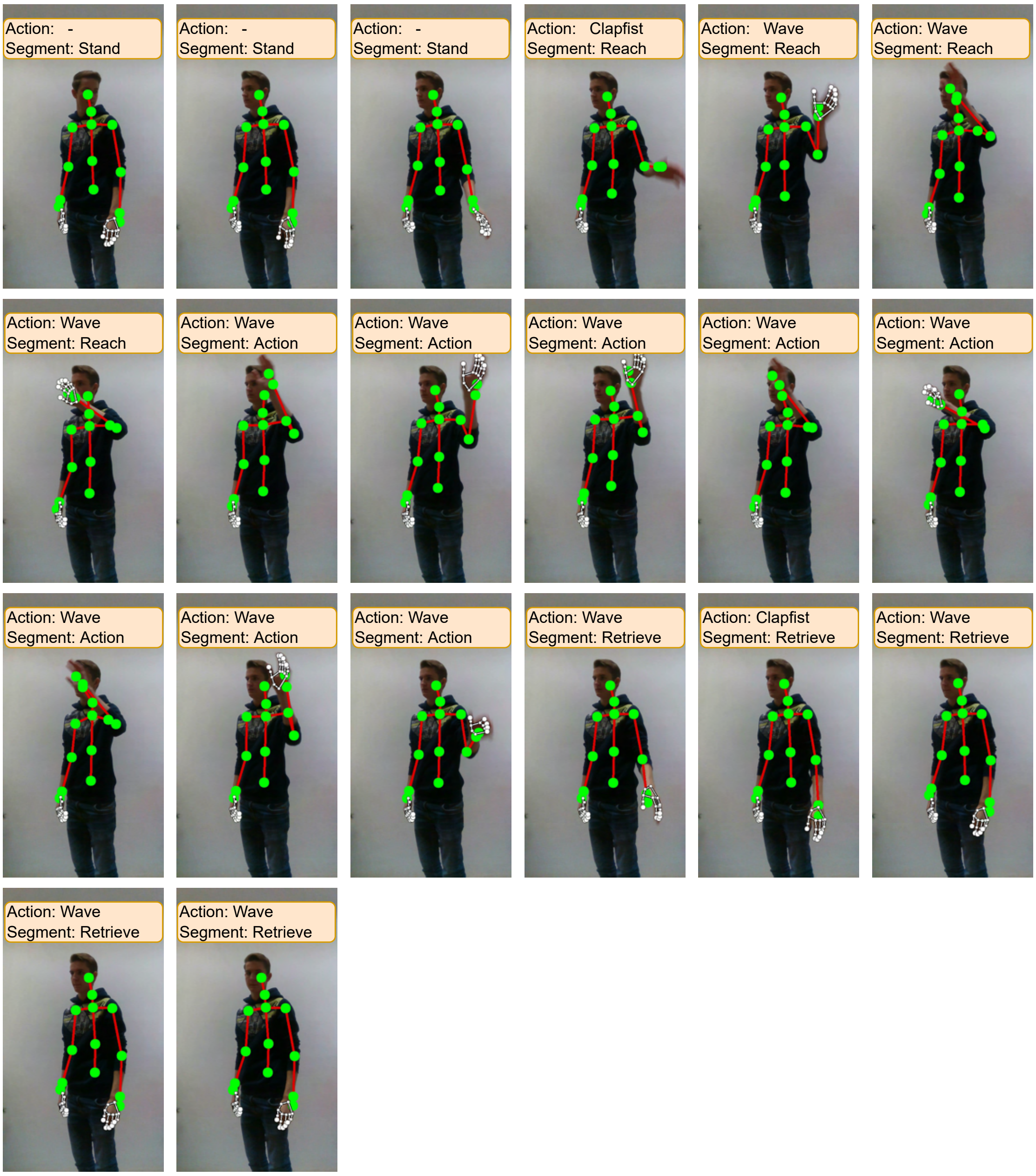


Figure A.3.: The image series of a live wave demonstration, which was recognized and segmented using the presented method.

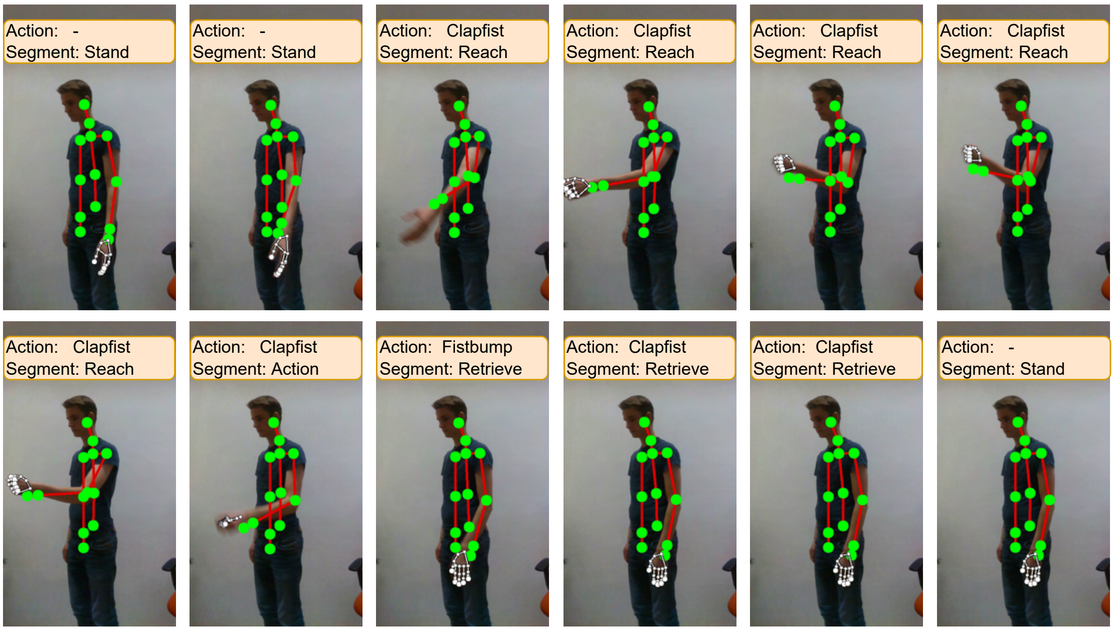


Figure A.4.: The image series of a live clapfist demonstration, which was recognized and segmented using the presented method.

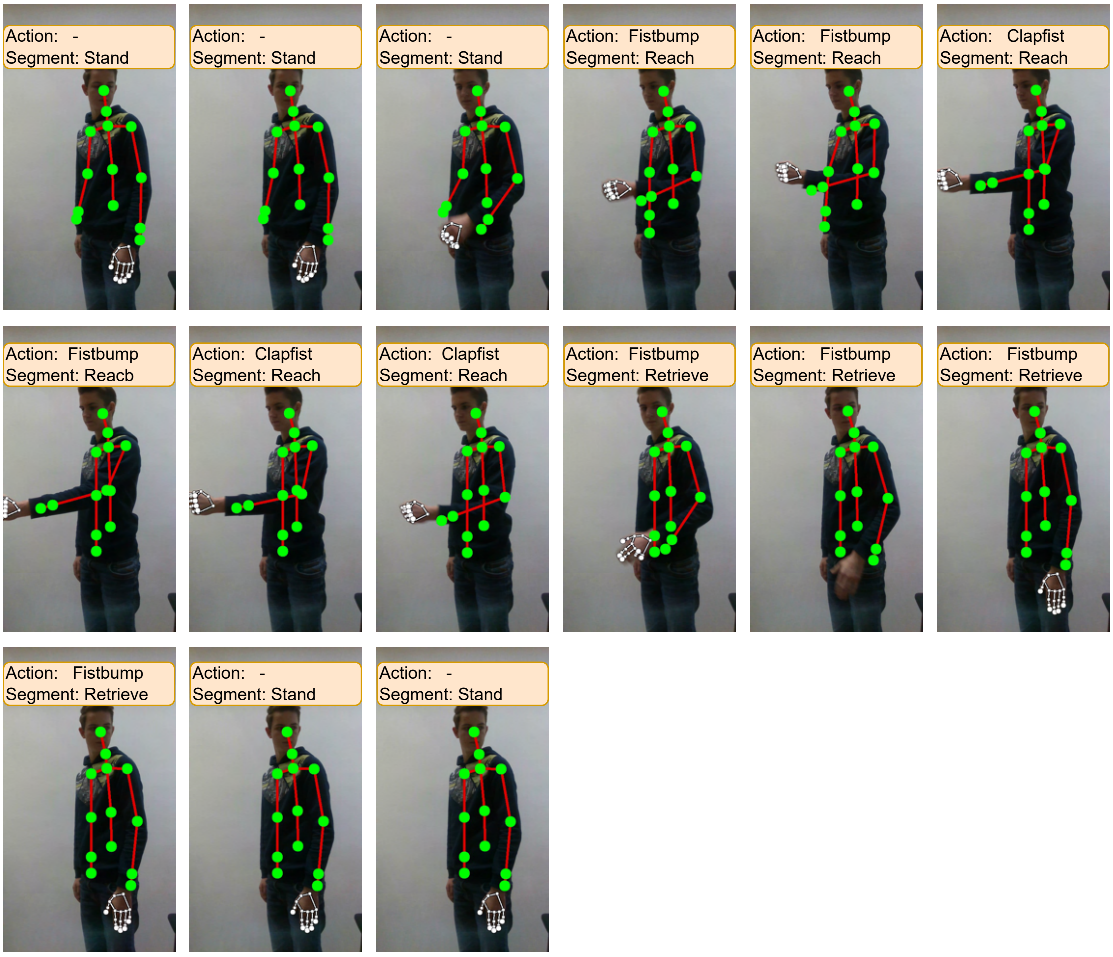


Figure A.5.: The image series of a live fistbump demonstration, which was recognized and segmented using the presented method.

B. Simulation With Robot

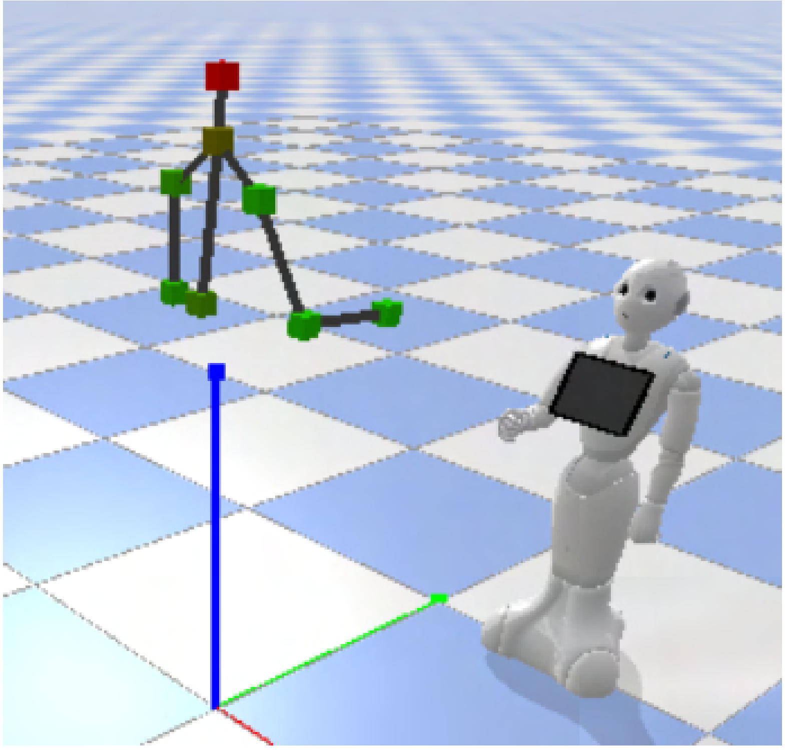


Figure B.1.: Image of the the simulation with qibullet to perform actions with the Pepper robot.

C. List of Abbrevations

HRI Human Robot Interaction

HMM Hidden Markov Model

HSMM Hidden semi-Markov Model

IOP Independent Opinion Pool

IFM Independent Fusion Model

CFM Correlated Fusion Model

CNN Convolutional Neural Network

GCN Graph Convolutional Network

RNN Recurrent Neural Network

ST-GCN Spatial Temporal- Graph Convolutional Network

RWAE Recurrent Wasserstein Autoencode

BIP Baysian Interaction Primitives