# Real-time Person Orientation Estimation and Tracking using Colored Point Clouds*

Tim Wengefeld*, Benjamin Lewandowski, Daniel Seichter, Lennard Pfennig, Steffen Müller and Horst-Michael Gross

*Ilmenau University of Technology, Germany*

ARTICLE INFO

*Keywords*:
person orientation estimation
person perception
mobile robotics
real time

ABSTRACT

Robustly estimating the orientations of people is a crucial precondition for a wide range of applications. Especially for autonomous systems operating in populated environments, the orientation of a person can give valuable information to increase their acceptance. Given people's orientations, mobile systems can apply navigation strategies which take people's proxemics into account or approach them in a human like manner to perform human robot interaction (HRI) tasks. In this paper, we present an approach for person orientation estimation based on computationally efficient features extracted from colored point clouds, formerly used for a two-class person attribute classification. The classification approach has been extended to the continuous domain while treating the problem of orientation estimation in real time. Furthermore, we present an approach for tracking estimated orientations over time using a Bayesian filter. We will show that tracking can increase the accuracy of orientations by up to 3.69° on a dataset recorded with a mobile robot. Best results on this highly challenging dataset are achieved with a regression approach for orientation estimation in combination with tracking. The mean angular error of just 16.49° proofs the applicability in real-world scenarios.

## 1. INTRODUCTION

The current orientation of persons (see Fig. 1) in the surroundings of a robot is a useful attribute for various HRI tasks. In the field of socially aware robot navigation, mainly two core functionalities require orientation information. First, for approaching a person correctly, the orientation of the user provides useful information for positioning in order to allow an unconstrained interaction (Fig. 2a). Second, in proxemic theory, the orientation is used to model a natural personal space around a person, which the robot should not enter during the navigation process (Fig. 2b). However, in such high-level applications, the orientation is typically considered to be given, either through motion capture data [1] or other external sensor systems [2], which cannot be used in real-world scenarios. In many application areas, the sensor setup is limited to the mobile robot platform. Some approaches use closed source 3D skeleton estimators like the OpenNI- or Kinect2-SDK in order to derive orientation information. These in turn have been used for approaching [3, 4] or personal space [5] applications. This might work in an experimental setup, but several limitations restrict the fields of application. For example, the constrained detection space of usually 1.5m − 4.5m limits the navigation planning and interaction horizon. Other specifics, like a prior on the orientation of users with respect to the sensor's coordinates as seen in the Kinect2-SDK, can lead to failures when approaching a person from behind.

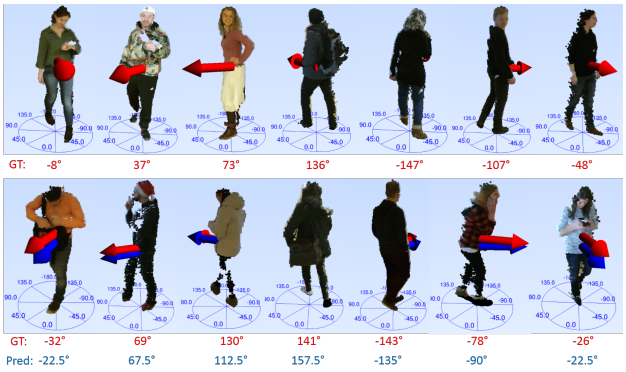
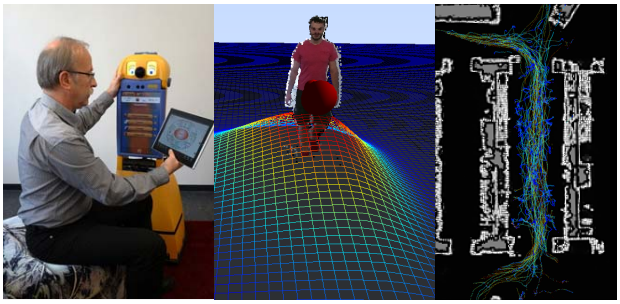



**Figure 1:** Various samples from our train-set (top) and test-set (bottom). The ground truths (red) and predictions (blue) are indicated with arrows.

A reliable orientation estimation is also essential for the prediction of human movement trajectories. In the domain of person tracking, the orientation information is typically extracted from the estimated velocity and, thus, from the past motion trajectory [6]. Unfortunately, this approach fails when rapid direction changes occur, which are likely to happen in narrow space environments like in our clinical [7] or supermarket [8] scenarios (Fig. 2c). In these scenarios, the environment mostly consists of long aisles. At aisle crossings or corners, a person sometimes has no other option than changing the heading direction which causes track failures in most cases since such a motion model still assumes the direction from the last few time steps.

Person orientation estimation itself was undergoing extensive studies. However, for the best of our knowledge, all robotic applications using these information fall back to more or less restricted methods not generally applicable in a

*This work has received funding from the German Federal Ministry of Education and Research (BMBF) to the project 3D-PersA2 (grant agreement no. 03ZZ0460), the project ROTATOR (grant agreement no. 03ZZ0437D) and the project MORPHIA (grant agreement no. 16SV8426)
*Corresponding author
E-mailaddress: tim.wengefeld@tu-ilmenau.de (T. Wengefeld)
ORCID(s):



(a) (b) (c)

**Figure 2:** (a) Robot approaching a person. (b) Modulated personal space cost function with orientation information. (c) Grid map and recorded trajectories of persons in an aisle of our supermarket environment. Extracting orientations only from past movement trajectories leads to misjudgements when people perform lateral movements.

real-world scenario. Therefore, we consider the robust and fast estimation of a person's orientation still as a challenging field for research.

This paper is an extended version of work published in [9]. Additional contributions are:

1. To proof the applicability of the orientation estimation approach in a real-world scenario, we perform additional experiments on a dataset recorded with a mobile robot.
2. We apply an additional tracking of orientations over time using discrete probability distributions.
3. We show that tracking enables us to get estimations at arbitrary time steps and also improves the accuracy by fusing information in a temporal manner.

## 2. RELATED WORKS

Person orientation estimation, as well as the similar problem of estimating the head orientation of a person, is usually treated by one of two estimation strategies. The first strategy handles the estimation problem as a multi-class classification by discretizing the continuous prediction space into several orientation classes. In contrast to that, more recent works perform a regression by predicting the continuous angle directly. Additionally, some approaches apply tracking mechanisms to integrate orientation estimations over time.

### 2.1. Orientation estimation

In [10], a combined detection and orientation estimation approach is presented using histogram of oriented gradient (HOG) features and a decision tree of support vector machines to detect eight upper body orientation classes and one background class. In [11], different combinations of well known RGB feature descriptors like HOG, local binary pattern (LBP) and aggregated channel features (ACF) are used for an eight-class orientation estimation. [12] extends the HOG feature space with the magnitude of gradients from depth images, which increases the feature weights at the boundary of the human body silhouette. In this way, they achieve better performance on data with complex background. All these traditional machine learning approaches have in common, that they divide the prediction space into relatively coarse classes. This comes with the drawback that in addition to misclassifications (reported accuracies range between 40% to 80%), a discretization error is introduced, which is about 11.25° considering a balanced test-set and eight orientation classes. Similar to other domains, recent advances in deep learning have improved the accuracy of orientation estimation approaches significantly. One of the first deep learning approaches for orientation estimation [13] uses a deep convolutional neural network (CNN) with cropped and resized person appearances from RGB-images as input and a softmax layer with eight neurons as output, which gives the confidence for each of the eight trained classes. They report an mean angular error (MAE) of 10.6°. However, their evaluation method neglects the mentioned discretization error, so the MAE must be higher in reality. In [14], a regression approach is presented estimating the head orientation in RGB-images with a deep CNN using two output neurons trained on the sine and cosine part of the prediction angle. They reported a MAE of 20.8° on a real-world dataset recorded in a town center. A similar approach presented in [15] predicts the full body orientation using cropped and resized greyscale images of people as input and two neurons as output. The CNN was trained with a large synthetic training set, and the evaluation was done on a spinning wheel measuring the ground truth with a non changing simple background. Astonishingly good results of 6.9° MAE are reported on this relatively simple test-set. However, lack of publicly available code and just a qualitative evaluation in a lab environment raises doubts about its applicability in the real world. Another way to estimate a person's orientation is to retrieve this information from estimated skeletons. In the past few years, this field of research has produced outstanding results. [16] as probably the most famous representative, commonly known as OpenPose, predicts 2D skeletons in RGB-images. However, from these 2D information the person's 3D orientation cannot be retrieved directly. Other recent approaches are able to estimate 3D skeletons from RGB [17] and RGB-D [18] images. We compare our approach to them in the experimental section. However, when it comes to the application, not every problem can be treated with deep learning due to the need of graphics cards that conflict with the restricted resources on mobile platforms. Even though there are power-saving graphics cards available, like the NVIDIA Jetson series, complex robotic systems, such as [19–21], which also need person re-identification or scene understanding, benefit from computationally less expensive alternatives for specific tasks.

### 2.2. Orientation tracking

Tracking a person's position, movement and estimated orientation was already applied in video surveillance. [22] proposed a coupled orientation tracking from an eight-class classifier, using HOG features in combination with motion

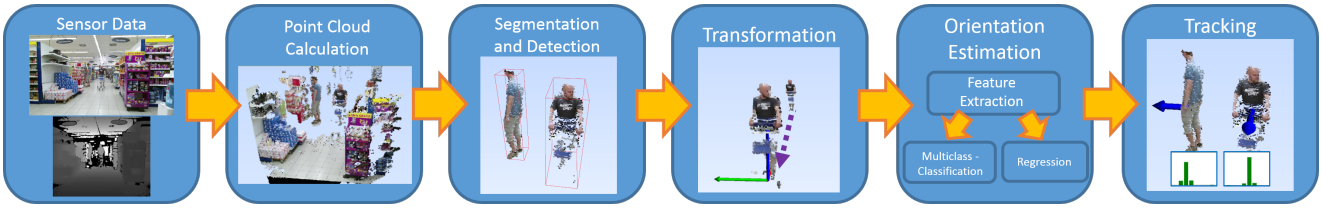



**Figure 3:** Sytem overview of the proposed approach in a supermarket scenery. Color and depth images are processed into a colored point cloud. A segmentation [26] and a subsequent point-cloud-based person detector [27] deliver person point clusters, that are transformed into a reference coordinate system afterwards. For the orientation estimation features are calculated for each cluster separately which are then fed either to a multi-class or to a regression approach to estimate the orientation of the person. Finally, the estimated orientations are tracked over several frames to compensate outliers and, thus, to stabilize the estimate.

information. The persons' positions and orientations are tracked with a particle filter (PF), but only qualitative results were presented for the tracking task. [23] use an unscented Kalman filter (UKF) and orientation estimations from an eight-class random forest (RF) classifier using HOG and LBP features. They achieve an eight-class accuracy of 0.70 on a dataset recorded with a mobile platform. However, they have not compared tracking to pure detection results. [24] propose a system for lower body orientation tracking in a top-down camera setup. The orientation information is retrieved from motion information and group dynamics of persons. For the tracking task, they use a mixture of Gaussian model. They achieve a MAE of approximately 22° for the lower body orientation. For robotic applications, [25] propose a 7D Kalman filter to track the position and orientation of persons in a multi-sensor setup but never evaluated the orientation tracking. All these papers have in common that evaluation of the orientation tracking task is strongly underrepresented in comparison to other contributions. In the experimental section, we show how the temporal tracking of orientations can improve performance compared to the to the framewise estimation of orientations on a dataset recorded on a mobile platform.

## 3. ROBOTIC APPLICATION

The proposed approach for orientation estimation is based on the binary person attribute estimation (e.g. gender) presented in [28], which uses clusters of colored 3D points as input. There, person point clusters are generated within a static sensor setup using a background model. In order to use this approach on a mobile platform, where background models are not feasible, we integrated it in a processing pipeline using the robotic middleware framework MIRA [29] (see Fig. 3). Synchronized color and depth images from a Kinect2 sensor are transformed into a colored point cloud. Afterwards, the segmentation method from [26] is applied to generate candidate point clusters which possibly represent persons. For cluster validation, we use the person detector presented in [27], which currently delivers the best detection results for this data representation. The advantage of this and other point-cloud based person detectors [26, 30] is that they share the required segmentation step with our approach and, therefore, no computational overhead is generated when used in combination. However, arbitrary person detectors combined with a projection into the point cloud coordinate system could be used to validate that a point cluster originates from a person as well. Another advantage of using clusters from point clouds as data representation is the independence from different backgrounds. Hence, the trained classifier can be applied in each environment where a segmentation yields valid samples. For detection accuracies in a supermarket scenario, we refer to [27]. After the detection step, the person point clusters are transformed into a local coordinate system aligned at the cluster's center of gravity while the orientation with respect to the camera remains unchanged. This normalized point cloud representation of each detected person is passed to the orientation estimation module, described in Sec. 4. Finally, the estimated orientation is assigned to a person hypothesis of our person tracker presented in [31]. The tracker fuses the orientation estimation of the current timestep with information from the past. The whole tracking procedure for orientations is described in Sec. 5. The corresponding experiments can be found in Sec. 6.5.

## 4. ORIENTATION ESTIMATION

As mentioned before, the proposed method for orientation estimation is a modification of the human attribute classification approach from [28] which classifies binary attributes like gender or long/short trousers. In this former work, a typical Adaboost approach has been used where a small amount of simple, yet fast to calculate features are extracted from a large amount of different regions of a colored person point cloud in order to train the classifier. More precisely, they computed statistic, geometric and color features, like the number of points, linearity and mean color, from overlapping subregions of the person point cluster. This results in a very large feature vector. However, using the Adaboost training algorithm, it is possible to retain real-time capability in the application phase, since just the most distinctive features have to be calculated. The main idea of Adaboost is, that the designer of the algorithm does not have to put much effort into data preprocessing, feature design, feature weighting, and feature selection, since Adaboost aims to find the most distinctive features by itself during training.

### 4.1. Adaboost fundamentals

Given a feature vector $v$, an Adaboost classifier $F_T(v)$ [32], also called strong learner, can be seen as a linear combination of $T$ weak learners $f_t(v)$ while $t \in T$. It can be described in the notation of Eq. 1:

$$F_T(v) = \sum_{t=1}^{T} \alpha_t f_t(v) \tag{1}$$

Each weak learner $f_t(v)$ outputs a class vote $\in (-1, 1)$, that is multiplied by a weighting factor $\alpha_t$ which was determined during training. In the application phase, the sign of the prediction of $F_T(v)$ determines the predicted class label and the absolute value the confidence. Training an Adaboost classifier is an iterative process by consecutively adding weak learners to the strong learner. During training, each sample $v_i \in V$ of the training set $V$ with size $N$ and $i \in N$ is associated with a label $k_i$ and a sample weight factor $w_i$. Initially every $w_i$ is set to $1/N$. The first step of every training round is to train a new weak learner $f_t(v)$ on a sub training set determined by $w_i$, to fit the class labels $k_i$. Then, the summed weighted error $\epsilon_t$ for misclassified samples is calculated:

$$\epsilon_t = \frac{\sum_{f_t(v_i) \neq k_i} w_i}{\sum_{i=1}^{N} w_i} \tag{2}$$

From $\epsilon_t$, the weak learners weighting factor $\alpha_t$ can be calculated:

$$\alpha_t = \frac{1}{2}\, ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{3}$$

Finally, the new weak learner is added to the ensemble:

$$F_t(v) = F_{t-1}(v) + \alpha_t f_t(v) \tag{4}$$

For the next round of training, the sample weights are adapted to focus the next training rounds on currently misclassified samples:

$$w_i = w_i e^{-k_i \alpha_t f_t(v_i)} \tag{5}$$

This process is repeated until a certain amount of weak learners is reached. In general, any kind of classifier can be used as weak learner for the Adaboost algorithm. However, in most practical applications binary decision trees are used. Advantages of decision trees are their fast training and their low execution times in the application phase. A binary decision tree consists of nodes and leafs. Each node has two connected children, which are either other nodes or leafs. The idea is, that each node takes a single feature from the feature vector $v$ and decides if the sample is passed to one of the two children using a threshold. If the child is another node, this procedure is repeated. If the child is a leaf, then the value that is associated to that leaf is returned. Decision trees are trained iteratively. Starting at the root node, the training algorithm aims to find an element in the feature vectors of all training samples and a threshold that splits the dataset best regarding the homogeneity of the class labels. This procedure is repeated until a maximum depth is reached or further splits do not improve the homogeneity any more. If no further splits are made, the leafs return value corresponds to the mean label value of all training samples, that fall into it. As measure for the class homogeneity we use the Gini impurity. However, the original Adaboost algorithm is just able to solve binary classification problems. In the following, we will show how it can be applied to the problem of orientation estimation.

### 4.2. Orientation estimation by Multi-class classification

A typical approach to overcome this issue is to train several binary classifiers, i.e. one for each orientation class, and make a one-vs.-all decision based on the maximum classification confidence of each classifier. Each sample $v_i$ in the training set is associated with a continuous orientation label $y_i \in [-180, +180)$. To train several binary classifiers, these labels have to be discretized and converted to a binary classification problem. Therefore, we define a set of intervals, that are subdivisions of the original label space. For an eight-class subdivision $\{I^{0^\circ}, I^{45^\circ}, ..., I^{-45^\circ}\}$ for example. The interval $I^\phi$ is used to generate the training label vector $k^\phi$ for one specific classifier $F_T^\phi(x)$ that is trained on exactly one prediction space subdivision. For a classifier $F_T^{45^\circ}(v)$, that shall be trained to the discretized angle of 45°, the corresponding interval is $I^{45} = [22.5, 67, 5)$. The corresponding label vector $k^\phi$ is calculated as:

$$k_i^\phi = \begin{cases} +1 & \text{if } y_i \in I^\phi \\ -1 & otherwise \end{cases} \tag{6}$$

In the application phase, the sample $v$ is classified by each trained classifier $\{F_T^{0^\circ}, F_T^{45^\circ}, ..., F_T^{-45^\circ}\}$. The final prediction is determined as the mean of the interval of the classifier with the highest confidence.

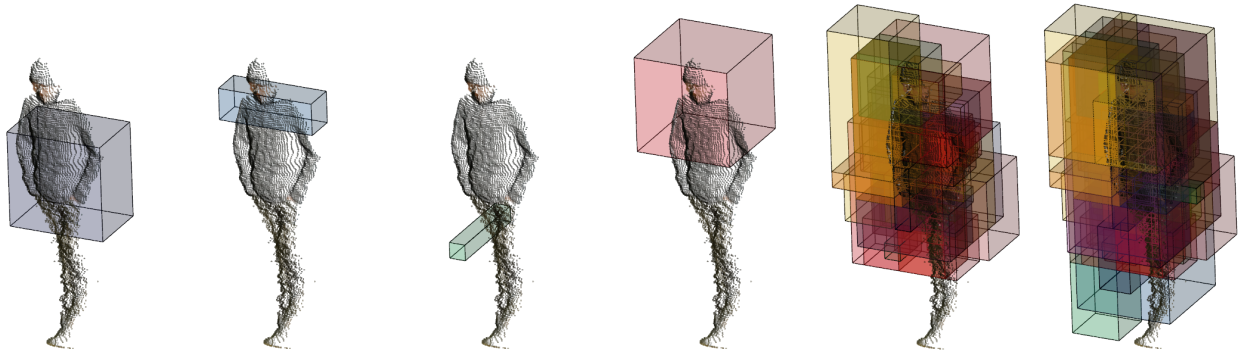
### 4.3. Orientation estimation by Regression

Another strategy is to treat the orientation estimation as a regression problem. The Adaboost algorithm was generalized to regression problems in [33] calling it gradient boosting. The concept is pretty similar to the original Adaboost by consecutively training weak classifiers, also typically decision trees, which solve the problem by finding the optimal features. However, instead of training new weak learners that predict the correct class label, weak learners in gradient boosting are trained to predict the so-called pseudo-residuals $r$, i.e. the negative gradient of a loss function $L$ with respect to the current model. First, the model is initialized with a constant values, e.g. the mean of $y$ for an absolute error loss function:

$$F_0(v) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma) \tag{7}$$

Then, the pseudo residuals $r_i$ are calculated:

$$r_i = \frac{\partial L(y_i, F_{t-1}(v))}{\partial F_{t-1}(v)} \tag{8}$$



(a) (b) (c) (d) (e) (f)

**Figure 4:** Voxel subregions that were selected by the Adaboost training algorithm for a classification model trained on a 0° upper body orientation. The four most descriptive features are the sphericity feature from region (a), the flatness feature from region (b), number of points from region (c) and the sphericity feature from region (d). An overlay of the subregions for the 50 and 100 most descriptive features are depicted in (e) and (f). It can be seen that the classifier has specialized on the upper body and leg region when predicting orientations.

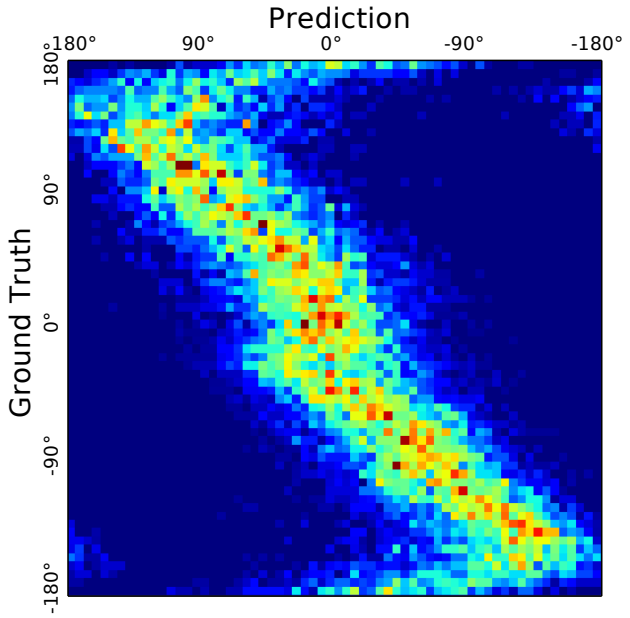



**Figure 5:** Example of a confusion matrix with 64 bins on our orientation test-set with a single regression model. Colors indicate the prediction accuracy per bin. It can be seen, that the model fails to deliver correct predictions at the transition point of the prediction space.

After that, the a weak learner $f_t(v)$ is trained to predict $r$ and added to the current model:

$$F_t(v) = F_{t-1}(v) + \gamma f_t(v) \tag{9}$$

$\gamma$ is a regularization parameter reducing the risk of over fitting. For the the decision tree weak learners, the prediction procedure remains the same. However, in contrast to classification, leafs now return arbitrary real-valued predictions. In the application phase, these weak classifiers can be evaluated independently giving this machine learning approach the same fast prediction capabilities. However, treating the orientation estimation as a regression problem comes along with the problem of periodicity of angles. This leads to the effect, that two samples close to each other in reality would have a high distance for our model if they are on different sides of the transition point of our prediction space, e.g. $-179°$ and $+179°$. Thus, orientations around the transition point cannot be estimated by the model (see Fig. 5). In general, such issues can be handled by transforming the prediction label to a higher dimensional space. However, gradient boosting classifiers are natively not capable to provide multi-dimensional outputs. In [14] the periodicity of the orientation angle is treated by a CNN with a two dimensional output for the sine and cosine part of the prediction angle. In our approach, we have adapted this method and trained individual regression models for the sine and cosine component of the angle.

### 4.4. Training details

**Features:** Since the main focus of this work is the estimation of orientations, we base the feature calculation for colored point clouds on the extensive evaluation from [28]. Therefore, we use exactly the same set of features that performed best in their evaluation. To be specific, we calculate 19 features in 14,023 voxel subregions following [28]. For each subregion, we calculated the following geometric features: number of points inside the voxel, point density, sphericity, flatness, linearity, standard deviation, kurtosis, average deviation from median, normalized planarity residual, height, depth, width, and the width-height aspect ratio. From the points' colors inside a subregion, the component wise mean and standard deviation of the RGB color channels are calculated. This geometric and color features together result in a vector with 266.437 elements, that is used

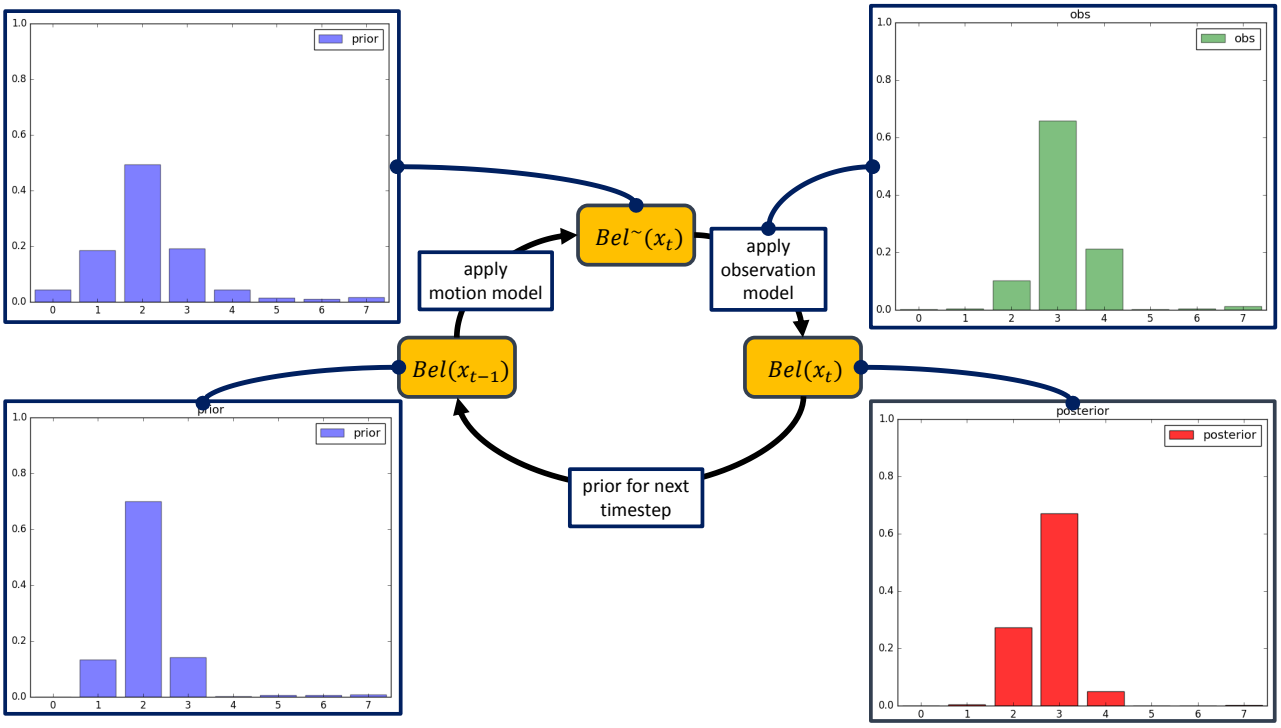



**Figure 6:** Tracking cycle for an orientation probability distribution discretized into eight bins. The belief from the last timestep ($Bel(x_{t-1})$) is predicted to the current time using a motion model. This belief $Bel^{\sim}(x_t)$ is then updated with a new observation from the current timestep in order to get the posterior distribution ($Bel(x_t)$).

to train our approach. During inference, just the most descriptive features have to be calculated as shown in Fig. 4.

**Multi-class classification:** One issue that arises, when the former continuous prediction space is divided into discrete bins, is that samples which are close to each other in the feature space may fall into different classes. Furthermore, if we take noise in our ground truth labels into consideration, which is very likely when training on data that is not artificially generated, one classifier may be trained with similar samples that have different (positive/ negative) labels. This is an issue often ignored in most state-of-the-art publications. To the best of our knowledge, the only approach that treats this issue was presented in [34]. There, the SVM training algorithm was extended with a cost relaxation parameter that weights the errors of adjacent classes lower than fatal misclassifications. Using this extension, they achieve an accuracy improvement for the eight class classification of up to 3.23%. However, the fundamental idea of the Adaboost algorithm is to find importance weights for specific samples by itself during training. Hence, this method cannot be adapted without changing the principle of Adaboost. Therefore, we simply tackled this issue by just ignoring samples adjacent to the positive class in the training step of a classifier for a specific direction. In the experimental section, we will show that we can achieve a notable precision boost with this simple yet efficient training strategy. Furthermore, this allows us to choose more fine granulated subdivisions of our prediction space than the de facto standard of eight classes to reduce the MAE, while a normal one-vs.-all training just increases the MAE for a finer granulated prediction space division.

## 5. ORIENTATION TRACKING

A robot that considers the orientation of people for its social navigation behavior has advantages when there are estimations at any point in time. Waiting for the next prediction might slow down the interaction process and can lead to unintended behavior. To overcome such issues, orientations can be tracked over time. In our previous work [31], we presented a modular tracking approach using different kinds of sensor data and detection cues. We made use of the key components of a Bayes filter, the prediction and the update step of a probability distribution called belief. This concept is able to track different attributes of persons, like their 3D position or posture. In the work presented here, we extend our tracking framework in order to enable a tracking of the estimated orientation state $x_t$ of a person. In particular, we use a $L_1$ normalized discrete probability distribution in order to allow a multi-modal belief representation. Fig. 6 shows the tracking cycle for orientations. The belief from the last time step is predicted to the current time by applying a motion model. Since persons in our scenario can quickly change their orientation, the motion model causes the orientation distribution to converge towards an uniform distribution. From the theory of continuous-time Markov chains, this can be done by a transition rate matrix $\boldsymbol{M}$, where the transition rates from one state to all other states are given and equally distributed in our case. For a certain time interval $\Delta t$, the predicted belief

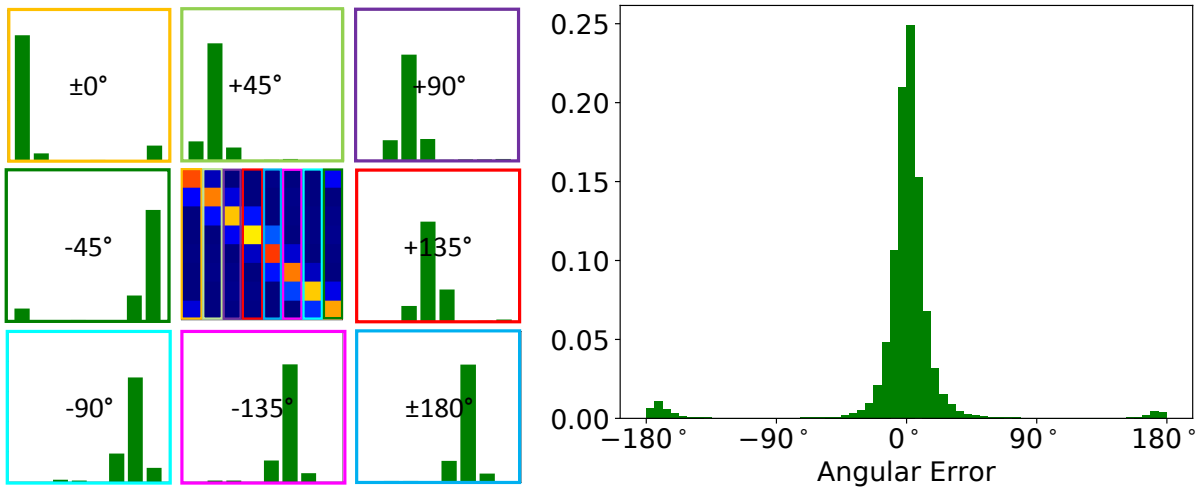



**Figure 7:** Left: confusion matrix from a eight class classifier in the middle and retrieved probability distributions for all classes. Right: normalized histogram over prediction errors from a regression approach from which the probability distribution of the estimate is retrieved.

$Bel^{\sim}(x_t)$ results from eq. 10 by using the matrix exponential of the transition matrix $M$:

$$Bel^{\sim}(x_t) = Bel(x_{t-1})e^{\Delta t M} \tag{10}$$

For the update step, we use the element wise (Hadamard) product of $Bel^{\sim}(x_t)$ and the new observation $obs_t$ followed by a $L_1$ normalization:

$$Bel(x_t|obs_t) \propto Bel^{\sim}(x_t)P(obs_t|x_t) \tag{11}$$

For a multi-class orientation classifier, the probability distribution for an observation can be retrieved from a test-set by calculating the confusion matrix followed by an row-wise normalization (see Fig. 7 left). The columns of the matrix correspond to the probability of how likely each ground truth class is estimated by the predictor. For a regression approach, the probability distribution for an observation can be retrieved by calculating a normalized error histogram over a test-set (see Fig. 7 right). In the application phase, the histogram is shifted to the current estimate of the predictor to get a probability distribution for the estimated angle. The initial orientation belief of a hypothesis is a uniform distribution over all bins. When a new estimation from the proposed orientation estimation pipeline (see Sec. 3) arrives at the tracker, it is associated to a person hypothesis, using the spatial distance. The orientation for each person is then tracked in a separate tracking module, as described above. To retrieve the orientation angle $\phi$ from the tracked distributions, we calculate the weighted sum of the sine and cosine parts of the distribution (see eq. 12). There, $\phi_i$ is the corresponding angle of the $i$-th bin of the probability distribution and $Bel(x_t)_i$ value of it.

$$\phi = atan2\left(\sum_i^n Bel(x_t)_i \sin(\phi_i), \sum_i^n Bel(x_t)_i \cos(\phi_i)\right) \tag{12}$$

In the experimental section 6.5 we will show, that the tracking of orientations gives a significant boost of precision on a real-world dataset with difficult person appearances.

## 6. EXPERIMENTS

Over the years, several evaluation metrics have been developed. Most classification approaches use the accuracy evaluation metric. However, since we are just interested in a precise estimation of the real valued orientation in the application, we will use the mean angular error (MAE) evaluation metric independently from class division for classification or regression. The MAE represents the mean absolute difference between ground truths and predicted angles while taking the circularity of the prediction space into account. Computation times are averaged results per sample over our balanced test-set on an Intel Core i7-4790K using 4 cores.

### 6.1. Datasets

In the literature, a considerable amount of datasets for training and benchmarking orientation estimation of persons are publicly available with RGB [35] and RGB-D [36] data. However, to the best of our knowledge, none of them fulfills our requirements of synchronized depth and RGB data streams to generate point clouds in combination with highly precise ground truth labels suitable for both classification and regression. Therefore, we decided to record our own datasets.

#### *6.1.1. NICR RGB-D Orientation Dataset*

In order to train our approach, we used the dataset from [37]. There, we generated ground truth data using a highly precise external ARTTRACK tracking system [38], which tracks markers using four infrared (IR) cameras with a positional precision of 0.4mm ± 0.06mm [39]. In order to reduce the influence of the markers in the actual point cloud data, the IR markers were placed on a thin pole about 0.5m above the heads (see Fig. 8). The pole itself was fixated under the clothes at the back of the subject to keep the orientation label unaffected from the head/view direction. By means of that, the label represents the orientation of the person's

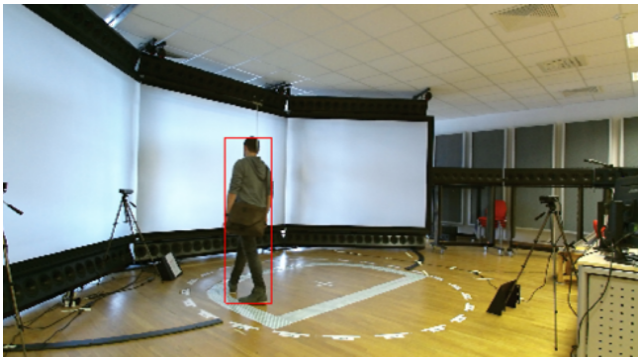
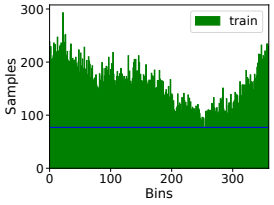


**Figure 8:** Environment in which the NICR RGB-D orientation dataset was recorded.




| dataset | # samples |
|---|---|
| train-full | 57,717 |
| train-balanced | 27,720 |
| test-full | 50,788 |
| test-balanced | 21,600 |

**Figure 9:** Left: histogram of the orientation angles in our recorded training set. The blue line indicates the number of samples drawn for balancing. Right: statistics of the datasets.

thorax. To generate point cloud samples of person appearances, we recorded data using five Microsoft Kinect2 sensors placed in a half circle around the recording area such that the sensors' active boosters did not interfere with each other. Different activities of daily life, like walking around, using cellphones, or conversations were captured. To generate samples for training and for evaluation, we used a background model. Afterwards, we applied the person detection pipeline described in Sec. 3 to filter noise and to use the same preprocessing as in the application phase. During the sessions, 37 persons were recorded in a range of 1.5m to 5m that we divided into groups of 21 persons for training and 16 for the test-set. To give a valid evaluation of the system's generalization capabilities, no person is included in both sets. We also tried to keep the variance of persons' attributes in the test-set high with respect to gender, height, and clothing (see Fig. 1). In total 57,717 samples for the training set and 50,788 samples for test-set were recorded. Unfortunately, since we placed the sensors in a half circle, some of the recorded ground truth angles are overrepresented. Therefore, we balanced the datasets by sampling from 360 angle bins randomly without using a sample twice. The resulting datasets contain 27,720 training samples and 21,600 test samples (see Fig. 9). In the section 6.2 and 6.3, we discuss the results on the balanced test-set only. Results for the full test-set are given in the Tab. 1 and 2 for the sake of completeness as well but are not discussed.

#### *6.1.2. NIKR Tracking Dataset*

To show how the proposed system performs in a real-world application, we evaluated our approach on the publicly available dataset from [31] (see Fig. 10). There, 3D poses, person IDs and orientations were labeled manually at specific key frames and interpolated in between. This dataset includes five scenes with an overall time of 11min 35s, where the robot guides one person through our university building. Synchronized data from the RGB- and depth camera of the Kinect2 sensor were captured at a frame rate of 5Hz. Up to five additional persons appear in the surroundings and cross the path between the robot and the guided person. Note that no person in this dataset is present in the set we used for training. One person is included in the set we used for testing and the remaining individuals are generally not included in the NICR RGB-D orientation dataset. In Sec. 6.5, we discuss how our approach for orientation estimation performs on this real-world dataset and show how the performance can be improved by a temporal tracking of the orientations.

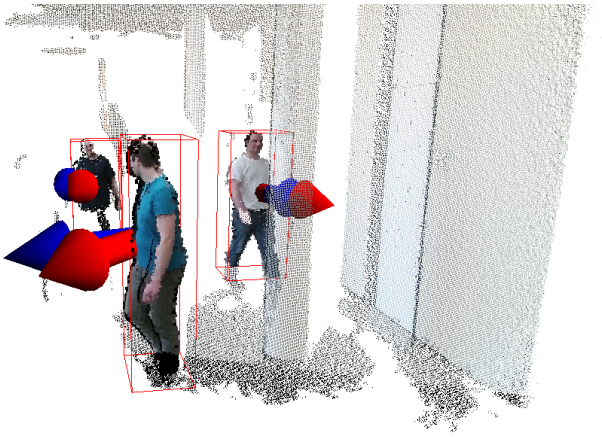


**Figure 10:** Scene from the NIKR tracking dataset. Classified clusters are marked with a red bounding box. The orientation ground truths (red) and predictions (blue) are indicated with arrows.

### 6.2. Classification results

One of the first issues we faced during the performance evaluation of our approach was the computational effort of the Adaboost algorithm during training. Given a large dataset with several parameter configurations and an increasing amount of prediction classes, the training can easily exceed several weeks using the OpenCV [40] machine learning back end. Therefore, the training was not possible on the full balanced training set and, thus, it had to be reduced to 8,000 samples for parameter evaluation, which also needs our maximum amount of 32GB RAM. In order to find the best training parameters, we conducted two series of experiments. The first one is intended to prove the advantage of our training strategy and to find the best number of subdivision for the continuous prediction space, i.e. the number of classes. In the second series, the accuracy of the best performing method should be increased by variating the training parameters, i.e. number and tree depth of the weak learners. For the complete parameter configuration, we refer to the imple-

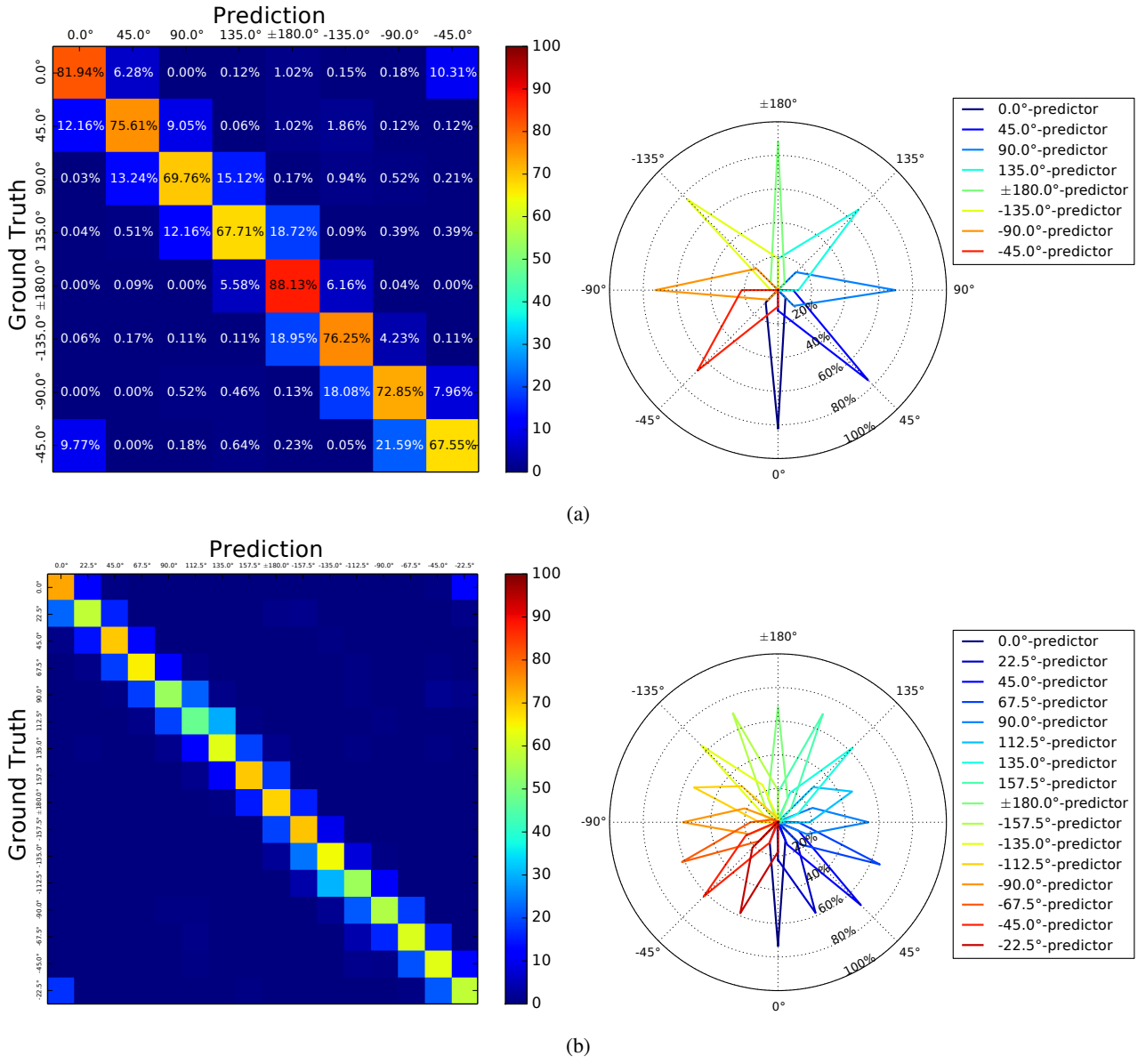



**Figure 11:** Results for proposed orientation classifiers on the balanced test-set: (a) eight-class CV-SAC-WL100-D1-CC8 (MAE: 17.84°) classifier and (b) 16-class CV-SAC-WL500-D2-CC16 (MAE: 12.21°) classifier. The confusion matrices are shown on the left. They indicate how likely each binned ground truth label is predicted to each orientation class. These results are further transformed into a radial coordinate system shown on the right to get a more intuitive representation. In this plot, the radial dimension encodes the prediction probability while the circular dimension encodes the ground truth label.

mentation[1]. The results are shown in Tab. 1. As expected, the SAC (skip adjacent classes) training strategy performs better than the standard approach using all samples. Therefore, we used this strategy in the following experiments. The first classifier trained with eight classes performed surprisingly well with an MAE of 17.84°[2] with an average execution time of only 6.39ms per sample. The results of selected classifiers are further visualized in Fig. 11. There, it becomes obvious (especially in the polar plots) that frontal or backward appearances can be estimated more precisely than sideviews. This can be explained with a larger surface of the persons in such appearances and more descriptive features resulting from them. However, none of the ground truth classes is estimated extremely worse than others and mispredictions are mostly adjacent to the real ground truth rather than pointing into complete different directions. By increasing the number of prediction classes to 16 the MAE drops to 15.40° with an increased execution time of 14.39ms per cluster. However, setting the number of classes to 32 performs worse than 16 but better than eight classes. Therefore, we assume 16 classes to be the optimal subdivision of our prediction space. In our second experimental series, we

[1] https://github.com/TimWengefeld/pointcloud_person_orientation_estimation

[2] The best achievable accuracy for an eight-class classifier due to discretization is an MAE of 11.25° given a well balanced test-set.

| exp. series | identifier | test MAE bal. | test MAE full | avg. time feat. | avg. time class. | #train samples |
|---|---|---|---|---|---|---|
| 1 | CVAB-UAC-WL100-D1-CC8 | 20.92° | 20.57° | 7.89ms | **0.03ms** | 8,000 |
| | CVAB-SAC-WL100-D1-CC8 | 17.84° | 17.51° | **6.36ms** | **0.03ms** | 8,000 |
| | CVAB-UAC-WL100-D1-CC16 | 22.45° | 21.93° | 13.78ms | 0.09ms | 8,000 |
| | CVAB-SAC-WL100-D1-CC16 | 15.40° | 14.78° | 14.30ms | 0.09ms | 8,000 |
| | CVAB-UAC-WL100-D1-CC32 | 24.66° | 23.49° | 29.39ms | 0.22ms | 8,000 |
| | CVAB-SAC-WL100-D1-CC32 | 21.18° | 19.68° | 29.12ms | 0.23ms | 8,000 |
| 2 | CVAB-SAC-WL100-D2-CC16 | 14.31° | 13.76° | 28.73ms | 0.24ms | 8,000 |
| | CVAB-SAC-WL100-D3-CC16 | 26.38° | 25.86° | 24.06ms | 0.15ms | 8,000 |
| | CVAB-SAC-WL500-D2-CC16 | **12.21**° | **11.80**° | 78.96ms | 0.96ms | 8,000 |

**Table 1**
Results of our classification approach with different training parameters in context of mean angular error (MAE) and computation time for feature calculation and classification. The identifier encodes the parameters the multi-class classifier was trained with in the form (machine learning back end [Open**CV** **A**da**B**oost] - training strategy (**U**se/**S**kip **A**djacent **C**lasses) - # weak learners per classifier - tree depth - # prediction classes). Best results regarding accuracy and computation time are highlighted in bold.

| exp. series | identifier | test MAE balanced | test MAE full | time feat | time class | #train samples |
|---|---|---|---|---|---|---|
| 1 | CVGBT-WL800-D1 | 17.68° | 17.27° | **5.36ms** | 0.15ms | 8,000 |
| | CVGBT-WL800-D2 | 15.17° | 14.76° | 7.80ms | **0.13ms** | 8,000 |
| 2 | XGB-WL800-D2 | 12.55° | 12.24° | * | 0.60ms | 27,720 |
| | XGB-WL3200-D3 | **11.52**° | **11.21**° | * | 0.70ms | 27,720 |

**Table 2**
Results of our regression approach with different training parameters in context of mean angular error (MAE) and computation time for feature calculation and regression. The identifier encodes the training parameters of the classifier in the form (machine learning back end [Open**CV** **G**radient **B**oosted **T**rees / **XGB**oost] - # weak learners per classifier - # tree depth). (*) Results for XGBoost are currently just retrieved from the python interface with pre-calculated features. Best results regarding accuracy and computation time are highlighted in bold.

performed a parameter grid search over the number of weak learners (ranging from 100 to 500) and their maximum tree depth (ranging from 1 to 3). Exemplary results can also be found in Tab. 1 (for all results we refer to our github repository). The best combination achieves 12.21° MAE for our classification approach using 16 classes, 500 weak learners, and a weak learner's maximum tree depth of two. However, increasing the number of classes or model complexity comes with the drawback of a higher computation time of up to 79.92ms per sample. Thus, the optimal model has to be chosen for each application separately. It depends on whether accuracy or low latency is more important. As we are more interested in real-time predictions, about 15ms is the maximum execution time to be tolerable for our application, since typically more than one person appears in a scene. In the following, we will refer to the CVAB-SAC-WL100-D1-CC16 classifier with 15.40° MAE and 14.39ms execution time as ours-classification-fast and the CVAB-SAC-WL500-D2-CC16 classifier with 12.21° MAE and 79.92ms execution time as ours-classification-precise.

### 6.3. Regression results

By using the same OpenCV machine learning back end for gradient boosting for regression, the same limitations considering the training set size hold true. In the first experimental series (see Tab. 2), we also performed a parameter grid search over the number of weak learners (ranging from 100 to 800) and their maximum tree depth (ranging from 1 to 3). The best OpenCV regressors (CVGBT-WL800-D2 in the following referred as ours-regression-cv) achieved a MAE of 15.17° with a computation time of 7.80ms, while a classification approach with similar time consumption performs 2.67° worse. However, we did not achieve the accuracy of the best multi-class classifier. However, since the regression approach is faster in general, the training of more complex models seems to be a valid option for future work experiments. Our second experimental series shows the reachable performance of the regression approach when using the complete training data available. Therefore, we changed to the more recent machine learning back end for gradient boosting XGBoost [41], which provides better parallelization support and a more efficient memory management. With the same training parameters, like the best OpenCV re-

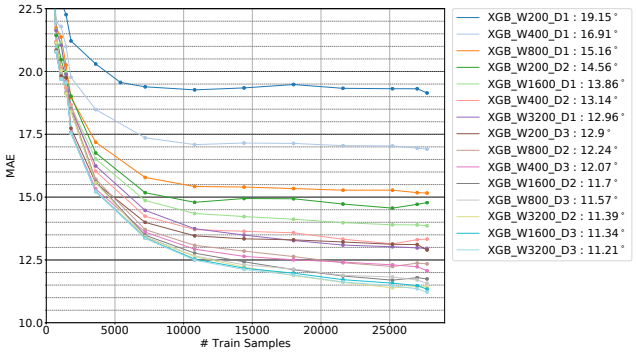



**Figure 12:** MAE values achieved by various XGBoost parameter configurations on the balanced test-set. The x-axis encodes the number of samples used for training. The best MAE for each parameter configuration is given in the legend.

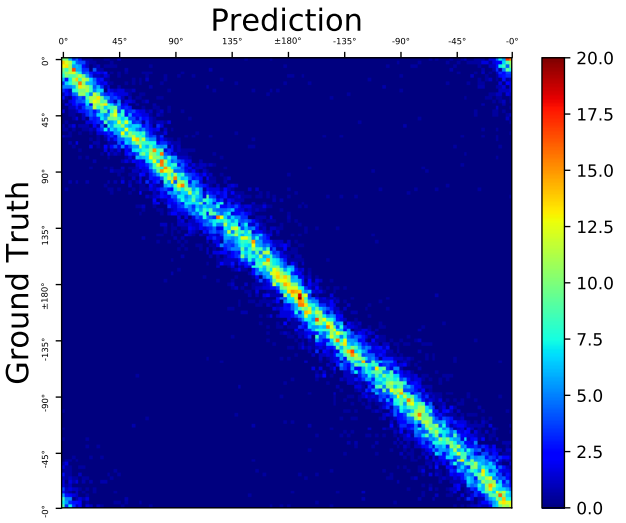



**Figure 13:** Confusion matrix of our best XGBoost regression model discretized to 128 bins.

| Approach | test MAE bal. | comp. time (detection) | comp. time (orient. est.) |
|---|---|---|---|
| [17] Tome et al. | 22.60° | 1127.48ms* | |
| [18] Zimmermann et al. | 14.66° | 606.33ms* | |
| ours-classification-fast | 15.40° | **72ms•** | **13.7ms•** |
| ours-classification-precise | 12.21° | | 105.9ms• |
| ours-regression-cv | 15.17° | | 20.8ms• |
| ours-regression-xgb | **11.52°** | | - |

**Table 3**
Comparison of our approaches to two recent deep learning-based 3D skeleton estimation approaches on the orientation test-set. For runtime comparison, we applied the deep learning approaches on the Jetson Xavier(*), NVIDIAs most recent graphics card designed for mobile autonomous systems. For our approach, we measured the average runtime on our robot's i7-7700T(•) CPU, using four threads. Best results regarding accuracy and computation time are highlighted in bold.

gressor, the MAE decreased from 15.17° to 12.64°. We are currently not able to give a run time comparison for XGBoost since we need to re-implement the optimized feature calculation for this back end. However, since feature calculation is the computational bottleneck, the run time should be similar to the OpenCV results with equal model complexities. With XGBoost as back end and a parameter grid search (see Fig. 12) over the number of weak learners (ranging from 200 to 3200) and their maximum tree depth (ranging from 1 to 3), our best regression approach (see Fig. 13) achieved a MAE of 11.52° (XGB-WL3200-D3 in the following referred as ours-regression-xgb). This is 0.69° better than our precise classification approach. Moreover, we analyzed the model complexity and the dependency of the model performance to the amount of samples used during training (see Fig. 12). There, it can bee seen that more complex models perform better in general. The enlargement of training set size from 14,400 to 27,700 samples gives a performance boost of about 1° MAE. However, increasing the model complexity above a tree depth of two and 1,600 weak learners just give a minor performance boost. Therefore, we conclude that our approach is near the maximum of its achievable accuracy. Enhancing the training set with further samples could give a slight performance boost, but for more precise estimations more complex features are required.

## 6.4. Comparison to 3D Skeleton Estimation

In order to give an insight on how other approaches perform on our data, we applied two recent state-of-the-art deep learning-based 3D skeleton estimation approaches from [17] and [18] on our test-set. Even though we are not able to retrain these approaches on our training set, we assume that the amount of data they were trained on provides good generalization capabilities. To calculate the person's orientation from the estimated 3D skeleton, we used the cross product from the left to right shoulder vector and the vector from the left shoulder to the spine base. Note: For the COCO model in [18] the spine base is interpolated from both hip joints. The resulting vector is then projected onto the ground plane and the acrtan function is used to calculate the orientation angle. Results are depicted in Tab. 3. It can be seen that our estimations are much more accurate than the ones extracted from [17]. This is reasonable since we use depth as an additional source of information. Orientation estimations results from [18], are slightly better than our fast approach but worse than our precise one. However, when it comes to computation times, the presented pipeline clearly outperforms the deep learning competitors without the need of specialized hardware. In combination with the preferred detector from [27], our fast approach achieves a frame rate of ∼12fps while our precise one runs at ∼6fps, when one person is in the scene.

## 6.5. Application and Tracking Results

First, we want to give a general statement on how well our orientation estimation approaches generalize on the tracking dataset without limitations on calculation times. Therefore, we performed an offline experiment. We generated point clusters for every point cloud in this dataset using the processing pipeline described in Sec. 3. Results can be

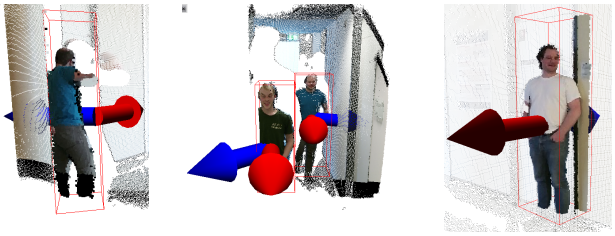


**Figure 14:** Typical failure cases for hard examples from our tracking dataset including a high level of occlusions and persons close to the background. Segmented clusters are surrounded with a red bounding box. The orientation ground truths (red) and predictions (blue) are indicated with arrows.

| Approach | MAE (all) | samples (all) | MAE (user) | samples (user) |
|---|---|---|---|---|
| ours-class-fast | 25.70° | | 21.54° | |
| ours-class-precise | 21.66° | 3019 | 18.66° | 1983 |
| ours-reg-cv | **20.99°** | | **18.39°** | |

**Table 4**
Offline results for our tracking dataset. MAE and amount of samples generated without consideration of calculation times. Best results are highlighted in bold.

| Approach | MAE (all) | estimates (all) | MAE (user) | estimates (user) |
|---|---|---|---|---|
| ours-class-fast | 25.64° | 93% | 21.48° | 94% |
| ours-class-precise | 23.12° | 54% | 20.21° | 57% |
| ours-reg-cv | 21.09° | 91% | 18.21° | 92% |
| tracking-class-fast | 21.95° | **99%** | 19.58° | **99%** |
| tracking-class-precise | 19.94° | 90% | 18.45° | 93% |
| tracking-regression-cv | **16.49°** | **99%** | **14.85°** | **99%** |

**Table 5**
Results for the MAE and amount of estimates which are available for a ground truth at a given time. Evaluated on our tracking dataset with consideration of calculation times. Middle row: results for our classification and regression approaches. Bottom row: results for our classification and regression approaches extended with the proposed orientation tracking. Best results are highlighted in bold.

found in Tab. 4. There, it can be seen that the estimations are generally more imprecise than the ones on our orientation dataset. Our fast classification approach just achieves an MAE of 25.70° for all persons. For the user samples, which are more likely in a good perception range of the Kinect2, the MAE drops to 18.66° for our precise classification approach. However, this is about 5.5° MAE worse than the result we achieved on our orientation test-set. Surprisingly, our OpenCV regression approach performs best on this dataset with a MAE of 18.39°. Nonetheless, this result is also less precise than the one this approach achieves on the orientation dataset. Reasons for this are various. First the tracking dataset was recorded in summer with people wearing shirts and shorts while the training dataset was recorded in winter with a lot of appearances of people wearing jackets and long trousers. Second, the ground truth data of the tracking dataset was labeled manually. This might adds larger errors to the ground truth than the precise automatic labeling of our orientation dataset. Third, the tracking dataset includes more difficult person appearances, like samples with a high level of occlusion and people standing close to walls, where the segmentation step fails (see Fig. 14). However, even though we do not achieve the same precision on this dataset, we have shown that our estimation approaches produce usable results under different conditions in a real-world scenario on a moving robot. Next, we want to give a statement about the performance of our approaches in the application phase and compare the results of the estimation approaches to the extended tracking. Therefore, we applied the whole processing pipeline (Sec. 3) and re-played the data in real-time. We evaluated how much estimates are dropped when new data arrives and the processing pipeline is still operating on data from previous timesteps. Additionally, we report the changes of the MAE that are generated by the delay of the processing pipeline. Averaged results over three runs can be found in Tab. 5 (middle row). There, it can be seen that the MAE for our fast classification and regression approach are nearly equal to the offline experiment. This indicates that the delay is small enough and rapid orientation changes of persons can be handled adequately. In contrast to that, the mean average error for the precise classification approach decreases by 1.46° for all persons and by 1.55° for the current user compared to the offline experiment. This shows, that the precise classification approach is too slow to be used in scenarios with rapid orientation changes. Looking at the amount of estimates, which are available on average, our fast approach is able to give prediction for 93% of the ground truths for all persons compared to the offline experiment. The precise classification approach, is just able to deliver estimates for 54% of the ground truths. This confirms the assumption that the precise classification approach is not suitable for highly dynamic scenarios, even though other applications could benefit from it. At last, we give results when an additional tracking of the orientations is performed. Therefore, we added the tracking step, as described in Sec. 5, to the processing pipeline. The data was re-played in real-time, like in the previous experiment. Averaged results over three runs can be found in Tab. 5 (bottom row). It can bee seen, that a tracking of orientations with our fast classification approach decreases the MAE by 3.69° for all persons and by 1.90° for the user. Also with the other approaches the angular error decreases by 3.5° in average. Best results for the tracking dataset are achieved by our orientation tracking using estimations from the OpenCV regression approach with an MAE of 14.85°. Additionally, using the tracking approaches, we are now able to retrieve estimations of the orientation at nearly ever timestep like in our offline experiments. Therewith, we have shown that a combination of fast orientation estimation and tracking can handle the problem of person orientation estimation adequately. This holds true even in a highly dynamic real-world application with difficult person appearances and rapid direction changes of the persons.

## 7. CONCLUSIONS

We presented a fast and accurate approach for person orientation estimation based on colored point clouds. Our approach achieves real-time estimation rates with low computational costs on a consumer CPU and is, therefore, particularly suitable for mobile robotic applications. We compared the common orientation estimation methods of multi-class classification and regression on a dataset recorded with highly precise labels for orientation estimation. Moreover, we have shown that the former approach for attribute estimation [28] is also able to work in the continuous domain. Hence, we expect it will also work for other real-valued person attributes, like age or weight. Since we aim to track persons up to a range of 10m we currently experiment with supsampling using a voxel grid filter to get a uniform data representation. Qualitative results look promising, but a quantitative evaluation will be part of future work. We also plan to train the former work for attribute estimation with multiple models based on their estimated orientation and hope to achieve better results than the previous work [28] We have extended the system for orientation estimation with an approach to track orientations over time using a Bayes filter with a discrete probability distribution. The performance was evaluated on a real-world dataset recorded on a mobile robot. This dataset contains hard samples with heavy occlusion and therefore gives a good assessment about the applicability of the proposed approach. The results have shown that the temporal integration of orientation estimates leads to more accurate predictions of up to $4,9°$ less MAE in comparison to a framewise evaluation. In addition, when tracking is applied, we are able to provide real-time orientation estimates on this dataset recorded at a frame rate of 10Hz. Even with our precise but relatively slow classification approach we achieve an estimation rate of about 9Hz. This is particularly important in highly dynamic environments such as our target scenarios. In Future work, we plan to combine the tracked orientations with a personal space cost-map for our navigation strategies to achieve a more polite navigation behavior.

## References


[1] P. Papadakis, A. Spalanzani, C. Laugier, Social mapping of human-populated environments by implicit function learning, in: International Conference on Intelligent Robots and Systems (IROS), 2013, pp. 1701–1706.

[2] T. Amaoka, H. Laga, S. Saito, M. Nakajima, Personal space modeling for human-computer interaction, in: Entertainment Computing (ICEC), 2009, pp. 60–72.

[3] X. Truong, T. Ngo, "To Approach Humans?": A Unified Framework for Approaching Pose Prediction and Socially Aware Robot Navigation, IEEE Transactions on Cognitive and Developmental Systems (TCDS) 10 (2018) 557–572.

[4] E. Avrunin, R. Simmons, Using human approach paths to improve social navigation, in: International Conference on Human-Robot Interaction (HRI), 2013, pp. 73–74. doi:10.1109/HRI.2013.6483507.

[5] P. Papadakis, P. Rives, Binding human spatial interactions with mapping for enhanced mobility in dynamic environments, Autonomous Robots 41 (2017) 1047–1059.

[6] T. Wengefeld, M. Eisenbach, Th. Q. Trinh, H.-M. Gross, May I be your Personal Coach? Bringing Together Person Tracking and Visual Re-identification on a Mobile Robot, in: Int. Symposium on Robotics (ISR), VDE, 2016, pp. 141–148.

[7] A. Scheidig, B. Jaeschke, B. Schuetz, Th. Q. Trinh, A. Vorndran, A. Mayfarth, H.-M. Gross, May i keep an eye on your training? gait assessment assisted by a mobile robot., in: IEEE/RAS-EMBS Intern. Conf. on Rehabilitation Robotics (ICORR), IEEE, 2019, pp. 701–708.

[8] H.-M. Gross, H.-J. Boehme, Ch. Schroeter, St. Mueller, A. Koenig, E. Einhorn, Ch. Martin, M. Merten, A. Bley, TOOMAS: Interactie shopping guide robots in everyday use – final implementation and experiences from long-term field trials, in: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), IEEE, 2009, pp. 2005–2012.

[9] T. Wengefeld, B. Lewandowski, D. Seichter, L. Pfennig, H.-M. Gross, Real-time person orientation estimation using colored pointclouds., in: Europ. Conf. on Mobile Robotics (ECMR), Prague, Czech Republic, IEEE, 2019, p. 7 pages.

[10] Ch. Weinrich, Ch. Vollmer, H.-M. Gross, Estimation of Human Upper Body Orientation for Mobile Robotics using an SVM Decision Tree on Monocular Images, in: International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 2147–2152.

[11] L. Fitte-Duval, A. A. Mekonnen, F. Lerasle, Upper body detection and feature set evaluation for body pose classification, in: VISAPP 2015 - 10th International Conference on Computer Vision Theory and Applications, volume 2, 2015, pp. 439–446.

[12] F. Shinmura, D. Deguchi, I. Ide, H. Murase, H. Fujiyoshi, Estimation of Human Orientation using Coaxial RGB-Depth Images., in: VISAPP 2015 - 10th International Conference on Computer Vision Theory and Applications, 2015, pp. 113–120.

[13] J. Choi, B.-J. Lee, B.-T. Zhang, Human body orientation estimation using convolutional neural network, arXiv preprint arXiv:1609.01984 (2016).

[14] L. Beyer, A. Hermans, B. Leibe, Biternion Nets: Continuous Head Pose Regression from Discrete Training Labels, in: German Conference on Pattern Recognition (GCPR), 2015, pp. 157–168. doi:10.1007/978-3-319-24947-6_13.

[15] Y. Kohari, J. Miura, S. Oishi, CNN-based Human Body Orientation Estimation for Robotic Attendant, Workshop on Robot Perception of Humans (2018).

[16] Z. Cao, T. Simon, S. Wei, Y. Sheikh, Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1302–1310. doi:10.1109/CVPR.2017.143.

[17] D. Tome, C. Russell, L. Agapito, Lifting From the Deep: Convolutional 3D Pose Estimation From a Single Image, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[18] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, T. Brox, 3D Human Pose Estimation in RGBD Images for Robotic Task Learning, in: International Conference on Robotics and Automation (ICRA), 2018.

[19] H. M. Gross, A. Scheidig, St. Mueller, B. Schuetz, C. Fricke, S. Meyer, Living with a mobile companion robot in your own apartment - final implementation and results of a 20-weeks field study with 20 seniors, in: IEEE Int. Conf. on Robotics and Automation (ICRA), Montreal, Canada, IEEE, 2019, pp. 2253–2259.

[20] H.-M. Gross, S. Meyer, R. Stricker, A. Scheidig, M. Eisenbach, St. Mueller, Th. Q. Trinh, T. Wengefeld, A. Bley, Ch. Martin, Ch. Fricke, Mobile Robot Companion for Walking Training of Stroke Patients in Clinical Post-stroke Rehabilitation, in: International Conference on Robotics and Automation (ICRA), 2017, pp. 1028–1035.

[21] H.-M. Gross, St. Mueller, Ch. Schroeter, M. Volkhardt, A. Scheidig, K. Debes, K. Richter, N. Doering, Robot Companion for Domestic Health Assistance: Implementation, Test and Case Study under Everyday Conditions in Private Apartments, in: International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 5992–5999.

[22] C. Chen, A. Heili, J.-M. Odobez, Combined estimation of location and body pose in surveillance video, in: Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference

on, IEEE, 2011, pp. 5–10.
[23] I. Ardiyanto, J. Miura, Partial least squares-based human upper body orientation estimation with combined detection and tracking, Image and Vision Computing 32 (2014) 904–915.
[24] M. Vàzquez, A. Steinfeld, S. E. Hudson, Parallel detection of conversational groups of free-standing people and tracking of their lower-body orientation, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 3010–3017. doi:10.1109/IROS.2015.7353792.
[25] M. Volkhardt, Ch. Weinrich, H.-M. Gross, People tracking on a mobile companion robot, in: IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC), IEEE, 2013, pp. 4354–4359.
[26] O. H. Jafari, D. Mitzel, B. Leibe, Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras, in: International Conference on Robotics and Automation (ICRA), 2014, pp. 5636–5643.
[27] B. Lewandowski, J. Liebner, T. Wengefeld, H.-M. Gross, A Fast and Robust 3D Person Detector and Posture Estimator for Mobile Robotic Applications, in: International Conference on Robotics and Automation (ICRA), 2019, pp. 4869–4875.
[28] T. Linder, K. O. Arras, Real-time full-body human attribute classification in RGB-D using a tessellation boosting approach, in: International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 1335–1341. doi:10.1109/IROS.2015.7353541.
[29] E. Einhorn, T. Langner, R. Stricker, Ch. Martin, H.-M. Gross, MIRA – middleware for robotic applications, in: International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 2591–2598.
[30] L. Spinello, M. Luber, K. O. Arras, Tracking people in 3D using a bottom-up top-down detector, in: International Conference on Robotics and Automation (ICRA), 2011, pp. 1304–1310. doi:10.1109/ICRA.2011.5980085.
[31] T. Wengefeld, S. Müller, B. Lewandowski, H.-M. Gross, A multi modal people tracker for real time human robot interaction, in: Proceedings of the IEEE Int. Symp. on Robot and Human Interactive Communication (RO-MAN), New Delhi, India, IEEE, 2019.
[32] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1997) 119–139.
[33] J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of statistics (2001) 1189–1232.
[34] Y. Kawanishi, D. Deguchi, I. Ide, H. Murase, H. Fujiyoshi, Misclassification tolerable learning for robust pedestrian orientation classification, in: International Conference on Pattern Recognition (ICPR), 2016, pp. 486–491. doi:10.1109/ICPR.2016.7899681.
[35] P. Sudowe, H. Spitzer, B. Leibe, Person Attribute Recognition with a Jointly-trained Holistic CNN Model, in: International Conference on Computer Vision Workshop (ICCVW), 2015.
[36] W. Liu, Y. Zhang, S. Tang, J. Tang, R. Hong, J. Li, Accurate estimation of human body orientation from RGB-D sensors, Transactions on Cybernetics 43 (2013) 1442–1452.
[37] B. Lewandowski, D. Seichter, T. Wengefeld, L. Pfennig, H. Drumm, H.-M. Gross, Deep Orientation: Fast and Robust Upper Body Orientation Estimation for Mobile Robotic Applications, in: will be published on International Conference on Intelligent Robots and Systems (IROS), 2019.
[38] Advanced Realtime Tracking GmbH, ART DTrack2, 2018. URL: https://ar-tracking.com/.
[39] K. Pentenrieder, P. Meier, G. Klinker, et al., Analysis of tracking accuracy for single-camera square-marker-based tracking, in: Proc. Dritter Workshop Virtuelle und Erweiterte Realität der GI Fachgruppe VR/AR, Koblenz, Germany, 2006.
[40] G. Bradski, The OpenCV Library, Dr. Dobb's Journal of Software Tools (2000).
[41] T. Chen, C. Guestrin, XGBoost: A Scalable Tree Boosting System, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794. doi:10.1145/2939672.2939785.

**Tim Wengefeld** received his Bachelor's and Master's Degree in Computer Science from Technische Universität Ilmenau in 2010 and 2014, respectively. He is currently employed at the Neuroinformatics and Cognitive Robotics Lab, while doing his Ph.D. in robotics. He is working on research projects with a high focus on real-world applications. This includes rehabilitation robots for stroke patients in the project ROREAS (2014-2015), companion robots for the elderly in the projects SYMPARTNER (2015-2017) and MORPHIA (2020-2023) and the developement of a robot for out-of-stock detections in a supermarket environment in the project ROTATOR (2017-2019). His research is focused on people-detection and -tracking, machine learning and pattern recognition.

**Benjamin Lewandowski** received his Bachelor's and Master's Degree in Computer and Systems Engineering from Technische Universität Ilmenau in 2014 and 2016, respectively. Since 2016 he is a research fellow and a Doctoral student at the Neuroinformatics and Cognitive Robotics Lab of the Technische Universität Ilmenau. His research interests include human perception and social robotic navigation for real-world applications. He works with robots deployed in supermarkets and hospitals in the research projects ROTATOR (2016-2019) and FRAME (2019-2020).

**Daniel Seichter** received his Bachelor's and Master's Degree in Computer and Systems Engineering from Technische Universität Ilmenau in 2015 and 2017, respectively. Since 2017 he is research a fellow and a Doctoral student at the Neuroinformatics and Cognitive Robotics Lab of the Ilmenau University of Technology. His research interests include deep learning-based semantic segmentation and scene understanding for robots in real-world applications.

**Lennard Pfennig** received his M.Sc. in Computer Science from the Technische Universität Ilmenau in 2019 working on image recognition. Since then he is working as Software Tester in Hamburg at imbus AG. There he is researching possibilities of testing with AI and testing of AI.

**Steffen Müller** received his Diploma degree (M.Sc.) in Computer Science from the Technische Universität Ilmenau in 2005. Then he joined the Neuroinformatics and Cognitive Robotics Lab at this university as research assistant and got his Ph.D. in robotics in 2016. His research interests are mobile robotics, personalization of Human-Robot Interaction, and adaptation of robot behavior by learning.

**Horst-Michael Groß** is full professor for Computer Science at Technische Universität Ilmenau and head of the research lab for Neuroinformatics and Cognitive Robotics. He received his Diploma degree (M.Sc.) in Technical & Biomedical Cybernetics (1985) and his Doctorate degree (Ph.D.) in Neuroinformatics (1989) from the Ilmenau University. His research mainly focuses on the development of robust and adaptive techniques for mobile navigation and HRI that cover aspects such as mapping, SLAM, multi-modal user detection and tracking, human-aware navigation, situation recognition, and dialog adaptation. As challenging application field, his research is focused on socially assistive mobile robots for public and domestic environments. Research highlights of his lab are the shopping guide robot TOOMAS (2000-2008), a robot companion for patients suffering from MCI as part of the EU-project CompanionAble (2008-12), the mobile health robots Max and Tweety developed within the SERROGA project (2012-2015), and the robot coach ROREAS for walking training of stroke patients (2013-2016).