Journal of Ambient Intelligence and Smart Environments 3 (2011) 193–211
IOS Press

Visual recognition of 3D emblematic gestures in an HMM framework

Jan Richarz * and Gernot A. Fink
Intelligent Systems Group, Robotics Research Institute, TU Dortmund,
Otto-Hahn-Str. 8, 44221 Dortmund, Germany
E-mail: {jan.richarz, gernot.fink}@udo.edu

Abstract. In human-machine interaction, gestures play an important role as input modality for natural and intuitive interfaces. While the usage of sign languages or crafted command gestures typically requires special user training, the class of gestural actions called "emblems" represents more intuitive yet expressive signs that seem well suited for the task. Following this, an approach for the visual recognition of 3D emblematic arm gestures in a realistic smart room scenario is presented. Hand and head positions are extracted in multiple unsynchronized monocular camera streams, combined to spatiotemporal 3D gesture trajectories and classified in an Hidden Markov Model (HMM) classification and detection framework. The contributions within this article are threefold: Firstly, a solution for the 3D combination of trajectories obtained from unsynchronized cameras and with varying frame rates is proposed. Secondly, the suitability of different alternative feature representations derived from a hand trajectory is assessed, and it is shown that intuitive gestures can be represented by projection on their principal plane of motion. Thirdly, it is demonstrated that a rejection model for gesture spotting and segmentation can be constructed using out-of-domain data. The approach is evaluated on a challenging realistic data set.

Keywords: 3D dynamic gesture recognition, human-machine interaction, smart rooms, time-series analysis

1. Introduction

The development of alternative input modalities for Human-Machine-Interaction (HMI) and smart environments has seen considerable advancement in recent years, driven by the ever growing complexity of devices and the functionalities they offer. This also implies a growing complexity of operation, pushing conventional input devices towards the limits of their capabilities, and accordingly raising a need for new types of interfaces.

Currently, two major developments can be observed: The first aims at overcoming the need for physical devices both for input and information display, leading to the development of ubiquitous or ambient touch screens (cf. e.g. [47]). However, they often rely on conventional WIMP (Windows, Icons, Menus, Pointers) interfaces, leading to interaction scenarios that are considerably less constrained, but sometimes not very intuitive in their usage. The second major current aims at improving the intuitiveness of device operation by analyzing the behavior of the users and incorporating natural and intuitive modes of human interaction. It is in the latter field where this article seeks to contribute. However, combining the best of both worlds should be the ultimate goal.

Natural inter-human communication is clearly multimodal, including facets like speech and gesture as well as gaze, facial expression and body language. While some of these modalities may be very subtle and subject to considerable variations between users, speech and gestures are much more explicit. Thus, they have been studied extensively as cues interpreting user intents and building human-centered interfaces.

This work focuses on the automatic visual recognition of dynamic arm gestures in a smart room scenario. The vision of smart environments also encompasses the absence of dedicated devices, like displays or key-

*Corresponding author. E-mail: jan.richarz@udo.edu.

boards, replacing them with ubiquitous computational services that can be accessed from anywhere at any time. Consequently, the user should not be constrained to a certain position or setting when interacting with the environment. Hence, for gestural interfaces, this implies that view- and position-invariant recognition is needed. To achieve this, this work aims at recognizing gestures in 3D space using a (potentially arbitrary) multi-camera setup.

Since the term gesture has been used in very different – and, sometimes, misleading – meanings (ranging from 2D fingertip motions to full-body actions), some clarification is needed.

In linguistics and semiotics, a variety of gesture taxonomies exist (cf. eg. [13]). According to Kendon [23], three major classes of gestures can be identified: On one end of the spectrum, there is what is generally called gesticulation. It is the most intuitive type of gestural action, and is typically accompanying other modalities such as speech. Thus, gesticulation is inherently multi-modal [12], and the mutual interactions between subconscious processes in generating speech and accompanying gesticulation are complicated and subtle. Therefore, automatically analyzing gesticulation is a very challenging task due to its subjective and unconscious nature.

At the other end of the spectrum are artificially created "alphabetic" gestures, like sign languages and crafted command gestures. They exhibit a well-defined structure and meaning and are therefore well-suited for automatic interpretation, but typically lack intuitiveness and require extensive user training in order to be used fluently.

In between resides a class of gestures that is often called "emblems". These are gestural actions that are well-defined and convey a certain meaning on their own within a certain cultural region. Therefore, their meaning and usage can be learned very easily or may even be understood intuitively. Thus, emblems seem especially suited for natural HMI.

Although directly transferring these concepts – originally related to natural fluent interaction between humans – to HMI is difficult, we adopt the term "emblem" in the following to denote simple iconic gestures that either closely resemble "natural" emblems or can convey an easily understandable meaning.

This article focuses on recognizing one-armed dynamic emblems performed by cooperative users. Within this field, the contribution is threefold: Firstly, techniques for the combination of results from unsynchronized camera streams are developed. This is challenging because hypotheses generally do not coincide temporally and may arrive with different and varying frame rates, rendering straightforward 3D reconstruction techniques ineffective in the case of moving scene objects.

Secondly, a variety of feature representations derived from the 3D trajectory of a moving hand is analyzed, assessing their suitability for the task. This includes feature types that go beyond normalized or user-centered point coordinates, which is the predominant approach in the literature. Additionally, motivated by the observation that most emblematic gestures have a simple structure and exhibit an inherent planar nature, a simple scheme for estimating their principal plane of motion is presented. It is shown that this plane, hereafter referred to as a gesture's "action plane", can be used to calculate a virtual view of the trajectory suitable for normalization and classification.

Thirdly, it is demonstrated experimentally that a meaningful rejection model can be learned using out-of-domain data.

2. Related Work

The relevance of gestures for natural HMI – either as exclusive cue or as part of multi-modal systems (cf. e.g. [40]) – is undisputed. Accordingly, there is an abundance of related publications, and it is impossible to give an exhaustive overview in this section. Most recent work in the field focuses either on the recognition of specially crafted artificial gesture alphabets and sign language in somehow restricted settings [30,45] or on the interpretation of full-body movements from video or inertial sensor data [49], generally referred to as action recognition (cf. e.g. [33,43] for recent surveys of the field).

While the shortcomings of artificial gesture alphabets for the purposes of general MMI have already been mentioned, full-body action recognition is related closely to emblematic gesture analysis, but typically operates on a higher level of abstraction. Instead of creating an input modality for HMI, it rather aims at analyzing human behavior in surveillance settings, or for scene understanding. Approaches from the field may, however, also be suitable for gestural interfaces.

Dynamic arm gestures can be defined by subsequent movements of a few prominent points (e.g. joint positions) relative to the body. Thus, given a spatiotemporal track of these points, recognition is a problem of time-series or trajectory analysis. Measurements like

in [3] indicate that, for simple stroke-like arm movements, the trajectories of the joints and hand are qualitatively very similar. This suggests that the problem can be reduced to the classification of the hand's trajectory only. Indeed, good results have been published using this approach (cf. e.g. [14,15,29]), indicating its validity. In [41], bimanual movements are classified by combining the trajectory with a shape descriptor of the hand, whereas [1] use the centroid positions of hand candidates and their mean optical flow. In [7], the spatiotemporal trajectory is mapped to discrete symbols with a Self-Organizing Map. These are then augmented with optical flow features. Combinations of 2D hand trajectories and associated inertial sensor data have also been used [5,34]. Some authors propose the usage of joint trajectories obtained from full body model tracking [46,48].

Trajectory recognition generally requires the accurate localization of certain points on the body. This is a hard and error-prone task, and some approaches therefore avoid the usage of point trajectories. Gaussian density features extracted at visual interest points are applied in [25], and gestures are classified using a protocol learning strategy. In [28], histograms of local optical flow are used, while a representation based on spatiotemporal contours is proposed in [27]. Silhouette images are utilized in [24], whereas [22] extract local motion signatures by tracking Histograms of oriented Gradients (HoG) descriptors.

Regarding classification, (Hidden) Markov Models ((H)MM) have been used extensively to model the spatiotemporal characteristics of dynamic gestures [5,6, 7,14,15,24,28,29,41]. Other classification approaches that have been proposed include Dynamic Time Warping [27], Dynamic Bayesian Networks [34] and variants of Conditional Random Fields [14,46].

Interestingly, only few approaches exist that aim at view-invariant recognition of 3D arm gestures. Some authors proposed the usage of short-baseline stereo cameras to incorporate 3D information [6,29,46]. Short baseline settings can provide some invariance against out-of-plane rotation, but cannot cope well with occlusion and therefore still require the user to approximately face the camera. In [24], eight different gestures for controlling certain functions of a smart room are recognized by learning a mapping from silhouettes of multiple viewpoints to key postures, whereas [48] utilize a multi-camera setting for tracking the 3D pose of a body model. In the latter, the intention to use the inferred model postures for gesture recognition is mentioned, but not carried out. In [8], a multi-camera setup is utilized to infer 3D pointing directions, but the method relies on strong assumptions and includes no explicit gesture recognition.

In this work, several of the approaches mentioned above are brought together in a way that, to the authors' best knowledge, has not been done before. The goal is the view-invariant recognition of emblematic arm gestures in a potentially arbitrary multi-camera setup using 3D trajectory information. The position of the user within the environment – and in relation to the cameras – should be as unrestricted as possible, and the gesture analysis should be done markerless and without tracking devices, i.e. based only on visual evidence. Recognition as well as detection is achieved within an HMM framework. In the following, the components of the system are explained briefly, starting with the 2D image processing pipeline followed by the 3D integration phase. Afterwards, the recognition framework is presented and evaluated in detail on a realistic hand gesture dataset.

3. Visual Recognition of 3D Emblematic Gestures

Intuitive usability of a gesture-based man-machine interface (MMI) implies that the interaction space should not be subject to artificial constraints. For example, restricting the interaction space to a predefined area, camera setup or specific user pose requires the user to be instructed about those restrictions. In a smart environment, however, the user ideally should not even be aware of where the sensors are. Furthermore, requiring the user to wear markers or specialized tracking gear would impose severe limitations on the general applicability of such a system, and its acceptance.

Therefore, the goal is a 3D recognition framework based on contactless visual cues utilizing off-the-shelf cameras in a principally arbitrary multi-camera setup. The only assumptions that are made are that there is only one person visible at a given time, that the user's upper body and gesturing hand are visible in at least two cameras, and that he/she is aware of the possibility of gestural interaction and willing to use it. Figure 1 shows an overview of the proposed approach. The individual components will be described shortly in the following, without going into too much detail, since the concrete acquisition of the trajectories is not the main focus of this article. A more detailed overview of an earlier version of the image processing pipeline is given in [36].

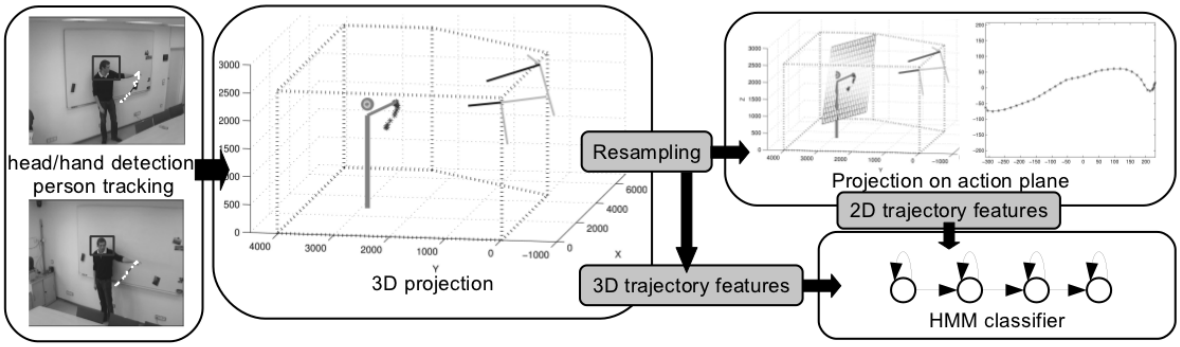

Fig. 1. Overview of the proposed approach.

The key assumption is that arm gestures may be analyzed using the trajectory of the active hand alone, which means that no expensive full-body model tracking is required. While this seems like a strong assumption, its validity – for the type of gestures considered here – is indicated by the good results reported in the related literature. The first step is the extraction of 2D spatiotemporal hand and head trajectories in the individual camera image streams. These are then combined into a 3D trajectory, which is normalized to a common reference frame afterwards. For normalization, the projection of a gesture on its action plane is proposed and assessed, additionally to conventional normalization approaches for the 3D trajectory. The obtained normalized representation is then classified in an HMM framework.

3.1. 2D Trajectory Acquisition

In order to extract gesture trajectories, persons and their gesturing hands have to be detected first. Here, a sliding window detector based on Histograms of oriented Gradients (HoG) [11] and a Multi-Layer Perceptron (MLP) classifier is utilized. The detector does not rely on skin color or face structure, but uses the shape of the head-shoulder line instead. Therefore, it is able to detect persons under a large variety of poses and viewing angles. Real-time performance is achieved by restricting the search window using adaptive background subtraction. Additionally, once the user has been detected reliably, his upper body is tracked using a Mean Shift histogram tracker [9] with an adaptive color model. The tracking result is used to further constrain the search space and also helps to eliminate false detection hypotheses, whereas the detection results can be used to detect tracking failure. By combining tracking and detection in this way, we achieve a reliable person localization with a very low number of false positives.

The centers of the head-shoulder detection rectangle in subsequent frames form the head trajectory. Hand candidates are found combining motion detection with a skin color model. Since the head-shoulder detection gives an estimate of the position of the person's head, it can be used to train a personalized adaptive color model online. Therefore, pixels are extracted in an elliptic area around the detection center, assuming that some skin should be visible most of the time. From these, a histogram model is computed using a Gaussian spatial weighting function. More details can be found in [36].

The resulting series of spatial image coordinates for head and hands are aggregated into 2D spatiotemporal trajectories. A simple greedy aggregation algorithm is used to identify valid assignments. It calculates temporal and spatial distances between hypotheses and existing trajectories, assigning to each trajectory the best-fitting hypothesis, creating new trajectories from points that were not assigned, and deleting trajectories that did not receive an assignment for a longer period. For the somewhat idealized data used here (cf. Sec. 4.1), this is sufficient. However, further work is still required on this part in order to be able to reliably deal with multiple hypotheses and false detections.

The trajectories are then postprocessed using Gaussian smoothing to eliminate detection jitter, and short tracks of duplicate points are removed. Longer tracks are kept because they indicate an idle phase of the gesticulating hand.

3.2. 3D Combination

Given spatiotemporal trajectories from at least two cameras, the original 3D gesture trajectory can be re-

constructed. At this point, it should again be pointed out that the cameras used in the data recording were not synchronized, and the obtained frame rates were not stable due to variations in the 2D processing duration. Thus, hypotheses from different cameras generally have been recorded at differing points in time. The implication of this is that the 3D reconstruction cannot be assumed to be exact (specifically, it makes a volumetric reconstruction based on silhouettes infeasible), and the individual 2D trajectories have to be aligned prior to combination. It also implies that the reconstructed 3D trajectories may be affected by considerable noise and inaccuracies, which makes the recognition task challenging.

The alignment algorithm starts by identifying the longest overlapping subsequence of two trajectories from different cameras, given their timestamps. Currently, a segmentation of continuous trajectories into gesture instances is assumed. This can be easily achieved by, e.g., searching for idle phases of a hand, but obviously poses a limitation to the applicability in an unconstrained scenario. The start and end points of alignment candidates do not have to be aligned, since only the temporally overlapping parts are considered. Note that, in a multi-camera setting, a view selection algorithm can be applied for choosing the two "best" views, which would considerably reduce the number of possible combinations. In [37], it was demonstrated that a reasonable ranking of camera views can be achieved based on simple bottom-up features.

Then, both trajectories are interpolated bilinearly at the points in time given by the data points of the respective other trajectory. The resulting interpolated trajectories have an equal number of points with pairwise matching timestamps that are then projected to 3D by ray casting.

Because of the temporal interpolation, detection inaccuracies and the discretization of the image plane, the rays will generally be skewed. Thus, there is no exact reconstruction point. An approximative projection is calculated as follows: Given two aligned points $\mathbf{p}_i(t) = (p_{xi}(t), p_{yi}(t))$, $\mathbf{q}_j(t) = (p_{xj}(t), p_{yj}(t))$ from cameras i and j at time t (we omit these indices in the following for readability), their corresponding 3D rays $\mathbf{r}_p = \mathbf{c}_i + \gamma\mathbf{p}'$ and $\mathbf{r}_q$ are obtained by backprojection using the respective camera calibration matrices Ψ_i and Ψ_j:

$$\mathbf{p}' = \hat{\Psi}_i^{-1}\mathbf{p}. \tag{1}$$

Here, $\mathbf{c}_i$ is the projection center of camera i, $\mathbf{p}'$ is the ray's directional vector, $\hat{\Psi}_i$ is the left 3×3 submatrix of Ψ_i, and similar for $\mathbf{r}_q$. The two directional vectors $\mathbf{p}'$ and $\mathbf{q}'$ along with $\mathbf{c}_i$ define a plane $\mathcal{P} : \mathbf{n}^T\mathbf{x} - \lambda = 0$, with the plane normal $\mathbf{n} = \mathbf{p}' \times \mathbf{q}'$ and the location parameter $\lambda = \mathbf{n}^T\mathbf{c}_i$. $\mathcal{P}$ contains $\mathbf{r}_p$ and is parallel to $\mathbf{r}_q$. Thus, translating $\mathbf{r}_q$ by $d = \mathbf{n}^T\mathbf{c}_j - \lambda$ along $\mathbf{n}$ moves $\mathbf{r}_q$ into the plane, and the intersection point $\mathbf{v}$ of the transformed rays can be calculated. The reconstructed 3D point $\mathbf{u}$ is then given by linear interpolation

$$\mathbf{u} = (1 - \frac{\alpha_i}{\alpha_i + \alpha_j}) \cdot \mathbf{v} + (1 - \frac{\alpha_j}{\alpha_i + \alpha_j}) \cdot (\mathbf{v} + d\mathbf{n}), \tag{2}$$

where α_i and α_j are confidence measures for the point positions in the individual images. This way, probabilities for individual hypotheses, e.g. obtained during the detection stage or assigned based on whether a trajectory point is an interpolated point or real observation, can be incorporated in the combination. Setting them to equal values yields the mean point along the direction of $\mathbf{n}$ where the two rays are closest. Also, hypothesis selection or rejection can be done based on d, assuming that, as d gets larger, the likelihood that $\mathbf{p}$ and $\mathbf{q}$ are corresponding points yielding a valid reconstruction gets smaller. This way, outliers or trajectory points stemming from false detections can be discarded. Also, a complete alignment hypothesis can be discarded if the overall reconstruction error or the ratio of discarded points is too high.

The resulting 3D trajectory $\mathcal{T} = \{\mathbf{u}_k, k = 1...n\}$ finally is resampled and smoothed using impulse resampling [18]. The general idea is to incorporate a momentum term related to the orientation $\mathbf{v}(t-1)$ of the last resampled trajectory segment

$$\begin{aligned} \mathbf{v}(t) &= \alpha \frac{\mathbf{u}_l - \hat{\mathbf{u}}(t-1)}{\|\mathbf{u}_l - \hat{\mathbf{u}}(t-1)\|} + (1-\alpha)\mathbf{v}(t-1), \\ \hat{\mathbf{u}}(t) &= \hat{\mathbf{u}}(t) + \mathbf{v}(t), \end{aligned} \tag{3}$$

where $\hat{\mathbf{u}}(t)$ is the resampled trajectory point at time t and $\mathbf{u}_l$ is the next original trajectory point that has not yet been reached. Resolution control can be achieved by only adding every mth resampled point to the new trajectory. Since the resampling depends only on the current and last trajectory segment, the algorithm can easily be applied incrementally to online data.

3.3. The Action Plane

Trajectories consisting of absolute spatial coordinates are not suited well for the recognition task because they would directly depend on the user's global pose in the environment. Thus, the interaction setup and the amount of variation allowed in the gestures would have to be seriously constrained to achieve reliable results. Normalization to some common reference frame and abstraction from the absolute positions is necessary.

A simple approach to achieve this are user-centered coordinate systems, or the usage of derivative or delta features that do not encode the absolute values of trajectory points, but rather their consecutive changes (cf. e.g. [6,29]). However, these still depend on the choice of coordinate system and the global alignment of the trajectory in 3D space.

Another possibility that is investigated here arises from the observation that typical emblematic gestures consist of relatively simple movements, where mostly one joint angle is modified at a time, the others remaining approximately constant. Consequently, the 3D trajectory exhibits an inherent planar characteristic. This suggests that the gesture trajectories may be represented without too much loss of information by projecting them on an appropriate plane. A similar assumption has been made in [34] to compensate for camera pan and tilt. Opposed to them, however, in the presented setup the plane may be oriented arbitrarily in space, and will only rarely coincide with any of the image planes. This concept will in the following be referred to as the "action plane", and it will be shown how an estimation of such a plane can be derived and used as a common reference frame for normalization.

Given a 3D trajectory $\mathbf{T} = \{\mathbf{t}_1 ... \mathbf{t}_m\}$ with m points $\mathbf{t}_i = (t_{xi}, t_{yi}, t_{zi})$, a plane $\mathcal{P} : \mathbf{n}^T\mathbf{x} - \lambda = 0$ is sought that best approximates $\mathbf{T}$. This can be formulated as a least-squares regression problem with the objective function

$$f(\mathbf{n}) = \sum_{i=1}^{m} (n_x t_{xi} + n_y t_{yi} + n_z t_{zi} - \lambda)^2 \rightarrow \text{Min}, \tag{4}$$

assuming that $\mathbf{n}$ is normalized to unit length. This is a well-known problem, and the plane's normal $\mathbf{n}$ is given by the Eigenvector corresponding to the smallest Eigenvalue of $\mathbf{\Psi} = \mathbf{M}^T\mathbf{M}$, $\mathbf{M} = (t_{x,i} - \bar{t}_x, t_{y,i} - \bar{t}_y, t_{z,i} - \bar{t}_z)$, with the data mean $\bar{\mathbf{t}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{t}_i$.

Calculating a regression plane in this way may result in a solution strongly influenced by outlier points. Therefore, the above procedure is carried out on the consensus set obtained from RANSAC [19]. Since $\mathbf{n}$ and $-\mathbf{n}$ correspond to the same global orientation of the plane and the sign depends on the choice of points, $\mathbf{n}$ is forced to always point towards the mean of head detections. When the above procedure is applied incrementally to online data, a smoothness constraint should be applied to the orientation of $\mathbf{n}$ to avoid abrupt changes. One possibility is adding a penalty term taking into account the angle between two consecutive plane normals in the model selection phase of RANSAC.

Furthermore, the procedure is problematic if $\mathbf{T}$ consists predominantly of a linear motion. In this case, the orientation of the plane is not well defined, since all planes containing the predominant line will get similar model scores. To cope with such cases, a simple heuristic is applied: Assuming that each command gesture is addressing an interaction partner – even a virtual one, e.g. some visible sensor representing the monitoring system – it will be conducted intuitively such that it can be seen and interpreted by the interaction partner. Consequently, gestures with an action plane that is oriented towards the floor or ceiling are less likely to occur. Thus, instead of selecting from RANSAC the single best model, all models $\mathcal{P}_j$ that have a score similar to the best one (within a reasonable threshold) are analyzed. Among these, the action plane $\tilde{\mathcal{P}}$ is selected according to the angle between the plane normal and the vertical axis $\mathbf{z}$:

$$\tilde{\mathcal{P}} = \mathcal{P}_k : k = \arg\min_j \mathbf{n}_j \mathbf{z}. \tag{5}$$

Projecting $\mathbf{T}$ onto $\tilde{\mathcal{P}}$ requires an 2D orthonormal coordinate system in $\tilde{\mathcal{P}}$. An obvious choice would be the Eigenvector corresponding to the largest Eigenvalue of $\mathbf{M}$ as first axis. The second axis is then given by its cross product with $\mathbf{n}$. This results also in a normalization of the global gesture orientation, which may not always be intended (e.g. horizontal and vertical gestures of the same type will no longer be separable) and would make the orientation of the coordinate axes dependent on the part of the trajectory that is currently analyzed. Indeed, previous experiments [35] showed that this is not a good choice for the considered data.

Thus, a different solution is presented here, where the global coordinate system is rotated into the plane such that the original x-axis is always parallel to the

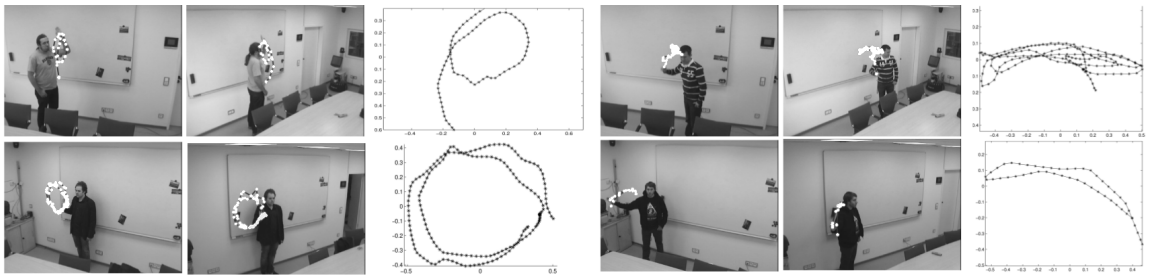

Fig. 2. Examples of action plane projections. Original images with overlaid hand trajectories and their 2D representations for "circle" (left) and "horizontal wave" (right).

ground plane and the original y-axis is parallel to the plane normal. The plane coordinate system is then formed by the x and z axes of the transformed coordinate system. The mean of projected trajectory points $\bar{\mathbf{t}}$ is chosen as coordinate origin. Figure 2 shows some projection results. The whole procedure can be interpreted as estimating a virtual camera with an optimal view of the gesture, followed by a coordinate normalization. It will be shown later that this projection indeed captures the properties of the trajectory sufficiently well.

3.4. Detection and Classification with Hidden Markov Models

HMMs are a popular tool for time-series analysis because of their ability to model temporal relationships between samples in a sound probabilistic framework. Furthermore, they provide an integrated approach for segmentation and classification. Their properties are well understood and efficient algorithms exist for training and inference. Therefore, they have been widely used, and their discriminative power has been demonstrated on a wide variety of tasks. In the following, the general model structure used in this work will be presented. For details on the theoretical background of HMM, please refer to e.g. [16].

An HMM is a probabilistic graphical sequence model that represents a two-stage stochastic process with hidden states and observable output. The first stage is a discrete stochastic process that can be viewed as a probabilistic finite state machine, i.e. state transitions occur according to transition probabilities. Each state can generate outputs according to some probability distribution. If this distribution is a continuous function that is modeled individually for each state, the model is called a continuous HMM, which represents the predominant modeling approach in the literature.

Typically, the output probability densities are modeled with Gaussian Mixture Models (GMM). In this work, so-called semi-continuous HMM are applied, which use a single codebook of Gaussians that is shared among all model states. This is advantageous when the amount of available training data is small, since the number of free parameters in the model is decreased. Discrete HMM are not considered here because the data is continuous, and a discretization to a set of symbols would result in a loss of information.

In the following, each gesture is modeled by an individual semi-continuous HMM, considering linear and Bakis model topologies (Fig. 3). The number of states in each model is initialized automatically according to the minimum observation length of the respective gesture class. The GMM is modeled with diagonal covariances, and parameter training is performed using the standard Baum-Welch algorithm. In the recognition phase, all individual gesture models are decoded in parallel (cf. Fig. 3) using Viterbi Beam Search.

Additionally, for continuous detection and recognition, a rejection criterion is needed. Here, a so-called garbage model (also called background or Null model) is adopted. This approach, which was originally developed for continuous speech recognition (cf. e.g. [2]), uses an additional HMM that is trained on non-gesture data and competes with the gesture models during decoding. All subsequences in the decoded state sequence that correspond to the garbage model are treated as rejections. An open source HMM toolbox [17] is used for the implementation of the recognition framework.

3.5. Trajectory Features

Early work on view-invariance for trajectory-based gesture recognition was done by Campbell et al. [6], who analyzed several alternative trajectory representations. Their findings still dominate the way trajecto-

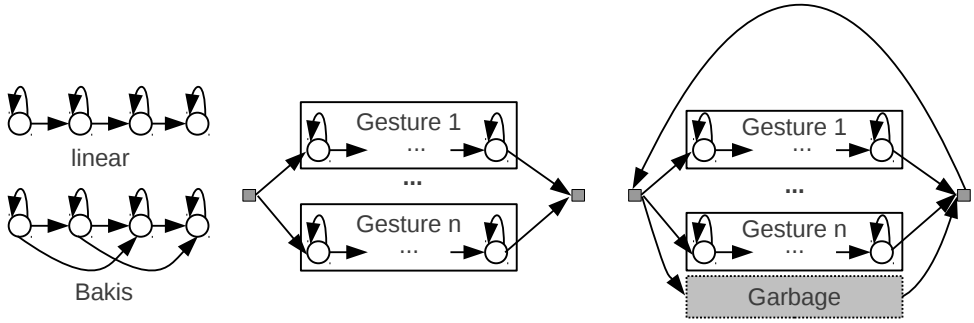

Fig. 3. HMM model structure. Left: Linear (top) and Bakis (bottom) topology. A Bakis model can skip states. Center: Model structure used for recognition: All gesture models are decoded in parallel. Right: Model structure used for segmentation/detection: A garbage model is added that is also decoded in parallel. In order to detect multiple occurrences, a backward edge has to be introduced, linking the terminal states to the start states of the models.

ries are represented in current approaches. Usually the trajectory is transformed into some common reference coordinate system and represented as Euclidean or polar normalized coordinates, sometimes along with velocity information. Similar problems – analyzing the spatiotemporal trajectory of a point or set of points – exist in other fields of research.

One example is online handwriting recognition (cf. [31] for an overview of the field). Interestingly, very different types of features have been developed there, although the classification methods used are very similar: State of the art handwriting recognizers are typically either also based on HMM [38] or on connectionist approaches [20]. But the features used to describe time-series of points are much more diverse. Examples include velocity and curvature along with shape-describing features of short trajectory segments [20] or Hu moments [10]. In [18] and [38], pen pressure, vicinity, curliness and features relating the trajectory to the baseline are used. Appearance-based descriptors and higher-level structural features, like ascenders, descenders and crossings, are also frequently combined with online trajectory features (e.g. in [20][38]).

Some of these lack a straightforward resemblance for the task of gesture recognition. E.g., pen pressure is not available, and features referring to a baseline (like ascenders and descenders) are difficult to apply, since, opposed to handwriting, it is not clear what the baseline of a gesture should be. Still, a multitude of interesting features for trajectory representation remain, and impressive results have been published in the field. To the best of the authors' knowledge, many of these features have not been applied to 3D dynamic arm gesture recognition before. One contribution of this article is to analyze the suitability of some of those, more specifically, curvature information and vicinity features, and compare their performance to common trajectory representations.

The features used for representing gesture trajectories are mostly motivated by [20]. Some changes are made to adapt them to the different characteristics of the data. In order to increase robustness against noise and detection errors, the features are calculated in a sliding window scheme.

Let $\mathbf{O}_i = \{\mathbf{o}_i, ..., \mathbf{o}_{i+w-1}\}$ be the trajectory points in the ith sliding window with size w and median point $\mathbf{o}_i^m$. Then, the features are calculated as follows:

Raw trajectory: Mean point of the window:
$\bar{\mathbf{O}}_i = \frac{1}{w}\sum_k \mathbf{o}_k, k = i...i+w-1$

Normalized trajectory:
$\hat{\mathbf{O}}_i = (\bar{\mathbf{O}}_i - \bar{\mathbf{H}}_i)/\bar{h}$, where $\bar{\mathbf{H}}_i$ is the mean 3D head position for the trajectory in the 3D case, and the coordinate origin in the 2D case (thus, the normalization of the 2D trajectory comprises only scaling, since the position is already normalized by the plane projection). $\bar{h}$ is the average height of the person calculated from the head positions. The goal here is to obtain a normalization of the overall gesture reach by assuming that it is relative to the person's height.

Normalized polar trajectory:
$\mathbf{P}_i = \{|\mathbf{r}_i|, \phi_i\}$. For the 3D case,
$\mathbf{r}_i = (\bar{\mathbf{O}}_i - \bar{\mathbf{H}}_i)/\bar{h}, \phi_i = \arctan(\sqrt{r_{xi}^2 + r_{yi}^2}/r_{zi})$,
i.e. the radius between mean trajectory and mean head point inside the window normalized by the person's height and the elevation angle of their connecting line. Note that the azimuth angle

would correspond to the global orientation of the person, so it is not included. For 2D, $|\mathbf{r}_i|$ and ϕ_i are polar coordinates in the plane relative to the coordinate origin.

Velocity: The mean velocity of data points in the window, i.e. $\mathbf{v}_i = \frac{1}{w}\sum_{k=i+1}^{i+w-1}(\mathbf{o}_k - \mathbf{o}_{k-1})/(t_k - t_{k-1})$ where t_k is the time associated with $\mathbf{o}_k$. The mean magnitude of the velocity vectors is also included. Again, recall that the trajectory points may be observed with a varying frame rate. Thus, the velocity feature is different from the trajectory delta feature.

Curvature: Curvature is defined as the cosine and sine of the angle between the vectors from $\mathbf{o}_i^m$ to $\mathbf{o}_i$ and $\mathbf{o}_{i+w-1}$.

Vicinity: These features are intended to describe the general shape of a feature window. Let $\mathbf{d}_i = \mathbf{o}_{i+w-1} - \mathbf{o}_i$ be the vector connecting the window boundaries. The vicinity features comprise the vicinity aspect $\alpha = (|d_{yi}| - |d_{xi}|)/(|d_{yi}| + |d_{xi}|)$ for 2D data and three values with permutations of the vector components for 3D data, the cosine and sine of the angle between $\mathbf{d}_i$ and the x-axis or ground plane, respectively, the ratio between the length of the connecting line and the trajectory length $l_i = |\mathbf{d}_i| / \sum_{k=i+1}^{i+w-1} |\mathbf{o}_k - \mathbf{o}_{k-1}|$ and the average sum of squared distances between trajectory points and $\mathbf{d}_i$.

Orientation change: For two subsequent windows $\mathbf{O}_i$ and $\mathbf{O}_j$, the orientation change is calculated as the cosine and sine of the angle between $\mathbf{d}_i$ and $\mathbf{d}_j$.

Head distance: The mean distance between points in $\mathbf{O}_i$ and the mean head position $\bar{\mathbf{H}}_i$, normalized by $\bar{h}$. This feature encodes some very weak representation of the spatial relation between gesturing hand and head.

All features together yield 20 and 25-dimensional feature vectors for 2D and 3D points, respectively, and twice the size including delta features.

4. Experiments

In the following, detailed evaluation results on a realistic gesture dataset will be presented. First, the feasibility of the different feature representations is assessed in a classification setting using manually segmented and annotated data. Afterwards, results obtained in a segmentation scenario are presented.

4.1. Experimental Setup and Data

The evaluation took place in a realistic setup inside a smart conference room (cf. e.g. [26]) equipped with several Sony EVI D70P pan-tilt-zoom cameras. The cameras are mounted on the ceiling and are calibrated intrinsically and extrinsically. Throughout the experiments, the same pair of cameras was used. In neutral position, their principal axes form an angle of approximately 90°, but their orientations were changed several times during data recording.

A set of nine emblematic command gestures was chosen such that they either resemble natural gestures that are commonly used, or their meaning can be understood intuitively. Some examples are shown in Fig. 4. The gestures are:

horizontal wave: Raise forearm to approximately head height, then perform several repetitions of a horizontal waving motion.

vertical wave: Wave hand vertically parallel to the body.

up: Raise forearm towards shoulder with the open hand pointing upwards, then push hand upwards above head level.

down: Raise forearm towards shoulder with the open hand pointing downwards, then push hand downwards below waist level.

circle: Perform a repetitive circular motion at approximately shoulder height.

comehere: Extend arm towards addressee, then move hand in a wide bowing motion towards shoulder. Can be repetitive.

goaway: Starting at waist level, move hand back a bit, then perform a fast, wide bowing motion forward and upwards towards the addressee.

stop: Starting from relaxed position, move hand at head height by flexing the elbow joint, hold it there for a short period.

point: Starting from relaxed position, point with extruded arm. This gesture is special in two aspects. Firstly, it is not purely emblematic, since it defines a reference to the surroundings and can only be evaluated taking into account the environmental context. Secondly, while all other gestures are performed in a similar position relative to the body, no pointing targets were specified and the subjects could point to wherever they wanted, leading to very large variations in the global trajectory orientation.

Fig. 4. Selected (cropped) examples of the gesture set with original trajectories overlaid. Gestures marked with (R) are repetitive, (r) indicates a gesture that may or may not be repetitive. From left to right: “circle”(r), “come here”(r), “down”, “go away”(r), “point”, “stop”, “up”, “horizontal wave”(R) and “vertical wave”(R).

The potential meanings conveyed by these gestures can be used in a variety of scenarios, e.g. directing a mobile robot, steering computational attention, or controlling services of the smart room. The set contains short one-stroke as well as more complicated repetitive gestures, and gestures that can be both.

Short sequences of still images were recorded from 17 different people each performing one to three instances of each gesture with their right as well as their left arm. The sequences were captured with a resolution of 378 by 278 pixels at a nominal frame rate of 20 Hz. Since the recording infrastructure was built in a decentralized and unsynchronized way in order to mimic the structure of the processing pipeline, the frame rate actually varies, and typically is considerably less than 20 Hz. No instructions on gesture speed, absolute or relative position, etc., were given. The subjects were allowed to move freely inside the cameras’ fields of view, including their orientation with respect to the cameras. Thus, the dataset is quite challenging since it contains multiple viewpoints as well as considerable variations in gesture appearance, speed and trajectory diameter. In total, it contains 51,217 images and 800 gesture instances.

The positions of hands and heads were annotated semi-automatically. First, the 2D head and hand detection algorithm described in chapter 3.1 was applied. The generated hypotheses were then inspected manually. Missing detections were added and erroneous hypotheses were corrected. In general, if a hypothesis from the detector was remotely correct, it was kept. The reason for this is that the goal of this evaluation is the assessment of the trajectory aggregation, 3D projection and classification stages, so we assume a reasonable detection of point hypotheses. Furthermore, the gesture instances were segmented manually, with considerable variations in starting and end points as well as number of repetitions. From the 2D trajectories obtained in this way, the 3D projections were computed using the described algorithm. The resampled trajectory lengths vary between 16 (“comehere”) and 364 (“vertical wave”) data points.

4.2. Classification experiment

The first experiment is carried out using the presegmented gestures. One HMM model is trained for each gesture type (left and right handed gestures are not distinguished) using 14 persons of the dataset. The models are then evaluated on the remaining 3 persons. This is repeated in a 5-fold cross validation scheme , where the respective training and validation sets are composed differently for each run. Thus, the reported results are person-independent and evaluated on 15 out of the 17 persons contained in the dataset.

Two different types of HMM topologies (see Fig. 3) were considered and trained with different parameter sets. In particular, the number of mixture components in the GMM and the model length were varied. The latter was determined by multiplying the shortest observed sequence for each gesture by a constant factor. Suitable values for those parameters were selected on a very coarse grid (150, 200 and 250 mixture components, length scaling of 0.5, 0.75 and 1.0) based on experience with HMM from other fields and previous experimental results [35]. For feature extraction, sliding window sizes of 5, 7 and 9 were applied, with 50% window overlap.

First, the different features are evaluated independently. In the following, the best results over all parameter combinations will be reported along with the corresponding parameters, if not stated otherwise. Table 1 summarizes the results for 3D trajectory and 2D projection features.

For the 3D case, using the delta of the raw trajectory yields a classification accuracy of 84.0 +/-2.7% (all confidence intervals reported in this chapter refer to a significance level of 95%), obtained with a Bakis model, a window size of 5, a codebook containing 150 densities and a model length scaling fac-

Table 1
Classification accuracy in % for single features (left) and respective derivative (Δ) features (right). The best results for each trajectory type are highlighted. Confidence intervals (+/-) are given in brackets

Feature	3D	2D	Δ3D	Δ2D
Raw traj.	56.1 (3.7)	76.2 (3.2)	**84.0 (2.7)**	78.0 (3.1)
Norm. traj.	78.5 (3.1)	79.8 (3.0)	49.0 (3.7)	76.2 (3.2)
N. polar	80.3 (3.0)	**76.7 (3.1)**	77.5 (3.1)	**81.3 (2.9)**
Curvature	39.6 (3.6)	39.8 (3.6)	43.2 (3.7)	42.7 (3.7)
Headdist.	59.7 (3.6)	61.2 (3.6)	27.7 (3.3)	25.9 (3.3)
Orientation	42.5 (3.7)	40.5 (3.6)	45.5 (3.7)	46.1 (3.7)
Velocity	**80.8 (2.9)**	71.3 (3.4)	75.5 (3.2)	68.7 (3.4)
Vicinity	71.6 (3.3)	66.7 (3.5)	67.2 (3.5)	56.8 (3.7)

Table 2
Average model lengths (number of model states) for best reported results and average reconstruction errors (in mm) per trajectory point for projected 2D gesture trajectories

Gesture	#States 3D	#States 2D	RError
h. wave	27	7	21.8
v. wave	54	14	29.7
up	11	3	8.0
down	18	5	14.8
circle	37	10	17.1
comehere	12	3	13.4
goaway	12	3	10.6
stop	16	4	8.0
point	19	5	8.7

tor of 0.75. The number of states in the best models for the different gestures, averaged over the cross validation runs, are shown in Tab. 2. With the corresponding 2D features and the same model topology and feature extraction window, 78.0 +/-3.1% accuracy is achieved. The best result for the 2D features is 81.3 +/-2.9%, obtained using the delta normalized polar trajectory with a linear model and window size of 9. This model has a codebook of size 200 and a scaling factor of 0.75. The number of states for the gesture models is considerably smaller (Tab.2). This is due to the model type (Bakis models generally are initialized at twice the size of similar linear models in order to correctly model the skipping of states) and the larger feature extraction window, which results in shorter trajectories. Apart from the normalized trajectory representations, reasonable results are also achieved using the gesture's velocity profile and the vicinity features for both 2D and 3D. Interestingly, even the very simple one-dimensional head distance feature can encode some gesture characteristics, indicating that even weak relative positional information can contribute to the task. Similar findings are reported in [4], where british sign language gestures are classified using simple linguistic features encoding relative positions.

While some loss of information for the 2D features is expected due to the projection, the results show that this loss is small. This represents a substantial improvement compared to previous experiments [35]. It appears that the choice of the coordinate system is critical for the action plane, and was not appropriate before. Hence, the assumption about the inherent planar nature of the gestures used here seems valid. This is also backed up by the small average reconstruction errors given in Tab. 2. Looking at the raw trajectories, for the 2D case, the accuracy of 76.2% is close to the best results, indicating that the proposed action plane is indeed suitable to obtain abstraction from global world coordinates. The importance of this abstraction becomes obvious when looking at the best result for the raw 3D trajectory, which is significantly worse at 56.1%. However, this accuracy is still a lot better than expected, indicating some bias in the dataset towards certain global positions. Specifically, there definitely is a strong bias in global orientation, since the delta of the raw trajectory, which yields the best result in 3D, is not invariant against this.

Revisiting the data revealed that indeed many of the subjects tended to position themselves such that they approximately faced one of the cameras frontally, without being told so. Thinking about it, this is hardly surprising: The cameras incorporate the addressee or interaction partner, and it is normal and intuitive human behavior to face the addressee during interaction. Consequently, the problem of view-invariance – at least against large out-of-plane rotations – probably is not as grave as could be expected, provided that the user is aware of the whereabouts of the system's sensory equipment (or, putting it differently, the sensors can be perceived as some kind of avatar or interface agent of the system).

Considering combinations of the above feature types, previous experiments indicated that no significant improvement can be expected [35]. Since all features are calculated from the same trajectory points, it can be assumed that they are strongly correlated. Additionally, increasing the feature dimensionality also

Table 3
Best Classification accuracy for PCA features in %. Confidence intervals for the highlighted results are provided in the text

No. of PC	1	2	3	4	5	6	7	8	9	10	12	15
3D	56.1	75.9	81.6	85.3	86.3	**88.3**	87.8	87.2	86.7	86.0	86.7	86.7
2D	45.7	70.6	82.0	86.0	85.6	**86.6**	84.6	85.0	85.3	85.6	84.6	84.4
2D+3D	55.2	76.7	81.7	86.2	86.6	87.8	**89.8**	89.2	88.6	88.5	88.2	86.5

Table 4
Explained variance for Principal Components (in %), averaged over cross validation sets and feature window sizes. The maximum standard deviation of values is less than 0.08, indicating that the variations in feature extraction and cross validation composition have only minor influence

No. of PC	1	2	3	4	5	6	7	8	9	10	12	15
3D	12.5	22.6	31.6	39.3	46.0	51.2	56.1	60.5	64.3	68.0	73.5	80.7
2D	12.8	23.1	32.8	41.5	47.7	53.7	58.6	62.7	66.7	70.5	76.6	84.0
2D+3D	12.0	21.9	30.7	37.2	42.4	46.9	50.9	54.5	57.5	60.5	66.0	73.4

leads to an increased model complexity with a larger number of free parameters. One drawback of HMM models is that they need large amounts of training data in order to reliably capture the data statistics. Given the relatively small number of training examples in the database, this property becomes critical if the feature dimension is increased beyond a certain point.

So, instead of combining several of the above features by simply concatenating them, Principal Component Analysis (PCA) is applied to achieve decorrelation. Using the same cross-validation scheme as before, the complete feature representation (all feature types + derivatives) of the respective training set is first normalized to zero mean and unit variance to account for different feature dynamics. Afterwards, the Principal Components (PC) are calculated, and the resulting transformations are also applied to the validation set. Finally, classifiers are trained using different numbers of Principal Components. Table 3 summarizes the results. Additionally, the (average) percentage of data variance explained by a certain number of PC is given in Tab. 4.

Using PCA features indeed results in an improvement in classification accuracy. Interestingly, the influence of both the feature extraction window size and the model parameters appears to be minor. The only exception is the length scaling parameter, where a value of 1.0 (i.e. setting the number of states in the HMM equal to the length of the shortest observation sequence in the respective class) considerably degrades performance. This may be due to the larger number of free parameters in combination with the relatively small amount of training data. Apart from this, similar results (around 88 to 89%) were achieved throughout most parameter combinations (not shown in the table). Specifically, the best reported classification accuracy (89.8 +/-2.3%) was given by two substantially different models: Bakis with a codebook size of 250 and a scaling of 0.75, and Linear with 150/0.5, both with a window size of 9 and using the first 7 PC (i.e. approx. 51% of the feature variance) of the combined feature set. This improvement is significant.

Using 3D and 2D features only yields a slightly worse accuracy at 88.3 +/-2.4% and 86.6 +/-2.5%, respectively (with explained variances around 51 and 54%). Although these improvements – compared to the single feature results – are not significant, the overall results indicate that the classification can benefit from the incorporation of different alternative features. Interestingly, good results are already obtained using only few Principal Components. For all cases, adding more than four PCs does not lead to significant improvements anymore. This is beneficial, since the model complexity can be kept small. When more PCs are added, the accuracy starts to drop at a certain point, indicating that the amount of data may not be sufficient anymore for estimating reliable statistics.

Confusion matrices for both best models are provided in Fig. 5, showing high accuracy for most gesture types. The worst gestures are ”pointing” and

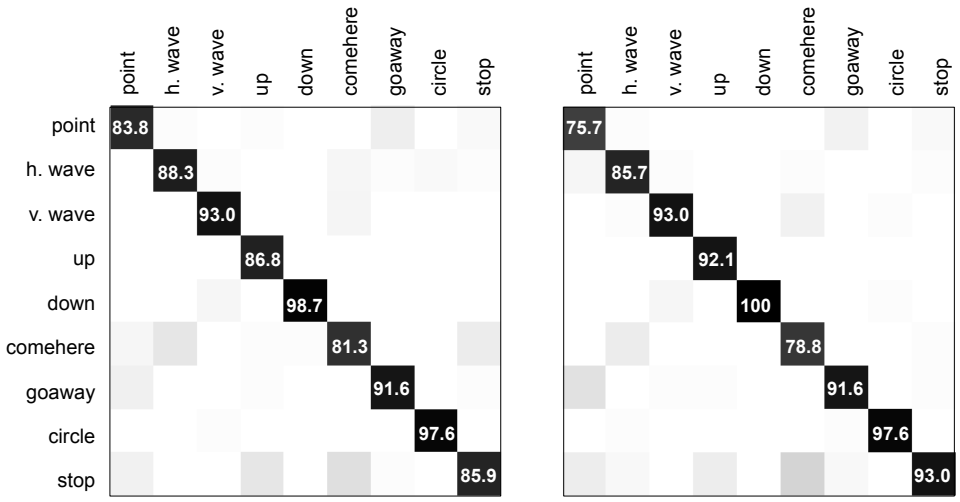

Fig. 5. Confusion matrices showing per-gesture results (in %) for best PCA models. Left: Linear model, 150 mixture components, scaling 0.5. Right: Bakis model, 250 mixture components, scaling 0.75. Note that the general confusion patterns are very similar.

"comehere". As mentioned before, pointing is the gesture with the greatest amount of variability in its global trajectory orientation. The comehere gesture may be single-stroke or repetitive, and exhibits large variations in terms of execution speed, trajectory length and number of repetitions. Thus, it is also challenging. The confusion patterns are very similar for both models. Most of the substitutions make sense, occurring between classes that show similar motion characteristics, or for short gestures that may be subgestures of a longer gesture.

4.3. Segmentation experiment

In a second experiment, the segmentation performance on longer sequences containing multiple gestures is assessed. Thus, the task becomes to segment the sequence and assign a gesture label to the segments. This requires a rejection criterion besides the gesture models. Rejection in the presented recognition framework is based on a garbage model (cf. Sec. 3.4).

4.3.1. Garbage modeling with out-of-domain data

In order to work reliably, a garbage model needs large amounts of representative non-gesture data that is not always available. Thus, the possibility of training a garbage model with out-of-domain data is investigated here. However, care has to be taken in order to obtain valid results. Firstly, the data that is used should obviously exhibit similar characteristics than the data that will be presented to the model at run-time. This means that, in the case considered here, it should resemble common spatiotemporal motions of body joints. Secondly, all data should exhibit a similar initial frame rate and be subject to the same preprocessing. Thirdly, the absolute global coordinates of trajectory points cannot be used because they encode the global coordinate system of the data domain, which can be quite different.

The data used for garbage modeling in the following experiments comes from the freely available HumanEVA dataset [42]. It consists of spatiotemporal marker trajectories captured with a commercial motion capturing system at high frame rates, along with the associated video data. Only the motion capture data is used. The dataset contains four subjects that perform several actions with three trials for each action. For more details, please refer to [42]. Only those markers that exhibit considerable motion in most trials are considered, namely the markers corresponding to the wrists, elbows and feet.

The respective joint trajectories are extracted, subsampled to 20 fps to approximately match the gesture data acquisition rate, and represented in a user-centered coordinate system by subtracting the trajectory of the head marker. They then undergo the same preprocessing – Gaussian smoothing and impulse resampling – and the same sliding window feature ex-

Table 5
Best classification accuracy for PCA features without global position in %. First value: Linear model, 150 mixture components, scaling 0.5. Second value: Bakis model, 250 mixture components, scaling 0.75. Confidence intervals are given in brackets

No. of PC	5	6	7	8	9
3D	85.7 (2.6)/85.6 (2.6)	**88.0 (2.4)**/86.9 (2.5)	87.9 (2.4)/**87.8 (2.4)**	86.9 (2.5)/85.7 (2.6)	87.5 (2.5)/85.9 (2.6)
2D	**86.3 (2.6)**/83.0 (2.8)	85.5 (2.6)/83.6 (2.8)	84.3 (2.7)/82.6 (2.8)	85.7 (2.6)/**83.7 (2.7)**	85.9 (2.6)/81.8 (2.9)
2D+3D	85.0 (2.7)/84.7 (2.7)	87.3 (2.5)/**87.6 (2.5)**	**89.8 (2.3)**/87.5 (2.5)	88.3 (2.4)/87.5 (2.5)	88.9 (2.3)/87.3 (2.5)

traction as the gesture data. The global position is excluded from the feature set, for the reason mentioned above. This results in a different PCA feature representation. Hence, the corresponding classification experiment for the gesture trajectories was repeated with the two best model configurations from above and a window size of 9 to assess the impact of this change. It turns out that the classification accuracy is not affected (cf. Tab. 5), the results are almost identical. Again, this is not surprising because the different trajectory representations contained in the feature set can be expected to be highly redundant.

Using the motion capture trajectories, the garbage model is trained in the same way as the gesture models, with one difference: An automatic initialization of the model size relative to the sequence length is not suitable for long unsegmented sequences. Therefore, the model size is manually set to a small value, the choice of which will be evaluated. The same cross validation setup as above is used, where two of the available HumanEVA subjects are assigned to each training set, and the remaining third is used for constructing the validation set (subject four is intended for testing only, and no motion capture data is available).

4.3.2. Constructing validation sequences

For the detection experiment, longer sequences containing gesture and garbage instances are required. They are constructed as follows: Firstly, for each gesture instance in the respective validation set, possible continuations in the garbage data are searched. The transitions should be smooth. Otherwise changes between gesture and garbage instances would be marked by abrupt changes in feature values, which would make the segmentation task trivial. Thus, only cases where the Euclidean distance between the gesture and garbage trajectory points are smaller than the mean sample distance of the gesture plus one standard deviation are considered. Then, one gesture instance is selected randomly as starting point. The sequence is then grown in both directions by selecting gesture instances with matching continuations, and inserting them with the corresponding subsequence of the garbage data in between. During selection, gesture classes that were selected rarely (less than average) are favored. This way, it is ensured that the distribution of gesture class appearances in the validation sequences is approximately uniform (cf. Tab. 6).

Table 6
Number of gesture instances in the validation sequences

pointing	402	h. wave	479	v. wave	525
comehere	505	goaway	500	up	424
down	531	circle	524	stop	481

Hence, the result is a randomized sequence of gesture instances separated by garbage segments of varying lengths and with smooth transitions. During gesture selection, it is ensured that the same instance is not included multiple times in one sequence. Finally, the process is aborted when a specified maximum length (in terms of gesture instances) is reached or no valid continuation is found, and the timestamps associated with the trajectory points are shifted and interpolated at the junctions to be smooth and chronological. Due to the randomized nature, however, it cannot be ensured that all instances from the validation set will be contained in the final sequences. 122 out of 694 gesture instances and 50 out of 196 garbage trajectories did not yield any valid continuations and therefore were never selected. The cross validation sets contain a total of 1,000 sequences with 4,371 gesture occurrences. The average sample count per sequence is 220 before and 548 after resampling. Gesture instances are always separated by a garbage segment of varying length. Sequences can begin and end with an instance of any class, including garbage.

4.3.3. Results

The segmentation experiment was performed using the following meta parameters: Since the amount of training data is now considerably larger due to the garbage data, a larger codebook size can be used, which can generally be expected to improve the model quality given a sufficient number of samples. The codebook size thus was set to 1,024. For model initialization, the scaling factor from the best model in the previous experiment (0.5) was kept. The feature extraction window size is a critical parameter for the segmentation task because large windows can merge together samples from different instances and the total instance length is influenced. Therefore, all three previously used window sizes are investigated. The features used are the first 7 PC of the 3D feature set.

Additionally, the choice of garbage model size is not intuitively clear, and several sizes are evaluated. The choice was guided by the following considerations: It was assumed that, in order to be flexible enough, the garbage model should not be much longer than the gesture models. Therefore, the maximum length was set to the average linear gesture model length corresponding to the respective window size. The minimum length was one in all cases, and two additional lengths distributed approximately equally between the boundaries were chosen. This leads to model lengths of 1, 3, 6 and 8 states for window size 5, 1, 2, 3 and 5 for size 7, and 1, 2, 3 and 4 for size 9.

Segmentation and classification was again done using Viterbi Beam Search and free model competition, i.e. all gesture models and the garbage model were decoded in parallel without using any prior knowledge about the sample structure. Afterwards, all garbage instances were discarded from the hypothesis. Furthermore, a rejection threshold was applied to gesture hypotheses following the log-odd scores approach [32]. The general idea is to normalize the length-normalized word-based alignment scores to some reasonable background distribution. Here, the prior probabilities of the codebook densities are used. This results in comparable normalized scores that can serve as a confidence measure for rejection.

For quality assessment, the number of detections, substitutions, insertions and deletions are reported, a measure periodically used in spotting and segmentation experiments (cf. e.g. [14,21,39]). A gesture hypothesis and annotation are treated as coincident if their overlapping length is at least 50% of the annotation's total length. A coinciding hypothesis is correct (C) if it has the same label as the annotation, otherwise it is counted as substitution (S). A hypothesis that does not coincide with any annotation is an insertion (I), whereas annotations without any coinciding hypothesis are counted as deletions (D). In order to have a single-valued selection criterion, the F_1 score [44] is calculated, which is the harmonic mean of recall and precision:

$$F_1 = 2\frac{Precision \cdot Recall}{Precision + Recall}. \quad (6)$$

Recall is defined as the number of correct detections (C) divided by the total number of annotations (C+S+D), whereas precision is the number of correct detections divided by the total number of hypotheses (C+S+I).

Figure 6 shows the resulting curves for the best configuration, chosen according to the highest F_1 score of 67.4 (precision 64.5 +/-1.4%, recall 70.5 +/-1.4%). The corresponding model has Bakis topology, the feature window size is 5 and the garbage model has 3 states. As can be seen from the C/S/I/D plot, the most common errors are insertions, i.e. false positive gesture hypotheses. It should be noted that the data used in this experiment is very challenging because it contains smooth temporal and spatial transitions between gesture and non-gesture instances. Furthermore, the gestures do not start or end in defined idle positions and the garbage data contains trajectories that also come from hand movements, and, in some cases, can be quite similar to some of the gestures used here (cf e.g. the data from the "Gestures" action of the HumanEVA data). Thus, a large number of false positives could be expected. The best overall recall rate achieved with this model was 78.5 +/-1.2% (precision 50.1 +/-1.2%).

Regarding the influence of the number of states in the garbage model, no clear tendency becomes visible from the experiments. A short garbage model generally leads to a slightly larger number of insertions and, hence, to a reduced precision, but the differences are negligible and mostly canceled by a reasonable choice of the rejection threshold. Also, the differences between results from linear and Bakis model topologies are not significant.

There is, however, a significant negative impact of large feature extraction windows. With a window of size 9, the same model parameters, and a garbage model length of 2 (which, in terms of relative length, roughly corresponds to the 3-state model from above), the F_1 score drops to 62.4 (P 60.1 +/-1.4%, R 64.8 +/-1.4%) and the best achievable recall rate to 70.1 +/-

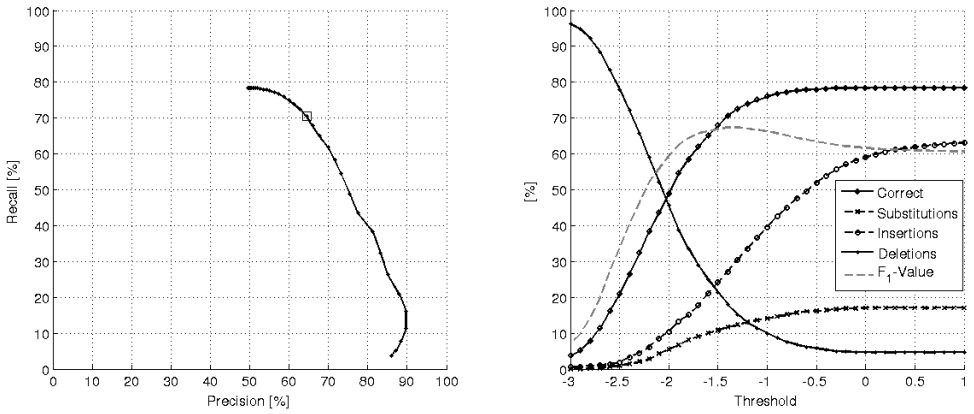

Fig. 6. Segmentation experiment: Precision-Recall curve (left) and C/S/I/D plot (right) for the best model obtained by varying the rejection threshold on the normalized alignment scores (note that these are negative log-scores, so smaller scores are better). The point corresponding to the reported F_1 score is highlighted in the left plot.

1.4%. Experiments with different parameter sets which are not presented here in detail confirmed that, on average, the scores are considerably worse. One possible reason is that, given the larger window and therefore shorter sequences, the data again is not sufficient for the increased amount of parameters in the model due to the larger codebook. Indeed, repeating the experiment with the same parameters as before, but a codebook size of 500 yielded better results (F_1 score 66.4 (P 65.8 +/-1.4%, R 67 +/-1.4%), maximum recall 74.5 +/-1.3%). The improvement is significant, but the recall rates still remain significantly worse compared to the ones obtained for the smaller window.

An interesting question is whether the results can be improved by incorporating prior knowledge concerning the sequence structure. In order to assess this, two HMM with a more constrained topology were constructed (cf. Fig. 7) using the parameter set of the best unconstrained model: The first (CHMM1) imposes weak constraints by specifying that each sequence contains an optional garbage instance at the beginning and end and has alternating sequences of one gesture instance followed by one or several garbage instances in between. This actually resembles the exact structure of the test sequences. The second (CHMM2) enforces that a single gesture instance always be enclosed between at least one garbage instance at both sides (which is not always the case in the data used here, so this model overconstrains the problem).

Using the model CHMM1 had no influence on the segmentation results, all scores remained almost identical. Due to the very flexible structure of the model – which practically allows any combination of gesture and garbage at any time – and because the HMM decoding algorithm always finds the optimal alignment in a probabilistic sense, this model becomes (almost) equivalent to the unconstrained model during decoding. Imposing stronger constraints on the problem structure by using model CHMM2, however, leads to a significant drop in the number of insertions (cf. Fig. 8) and thus to a higher average precision, but has a negative impact on the achievable detection rate. The maximum recall achieved with this model and the same meta parameters as above drops to 70.6 +/-1.3% (precision 68.9 +/-1.4%) while the maximum F_1 score increases slightly to 70.0 (precision 69.9 +/-1.4%, recall 70.0 +/-1.4%). The structural constraint effectively restricts the possible transitions between models, which is the reason for the reduced number of insertions. But it also enforces alignment of the sample sequence with the garbage model at the beginning and end of each segmented gesture instance, effectively always placing two garbage instances between gestures. This is especially critical for short gesture instances, which may be skipped.

Indeed, the drawback of reduced detection rates due to the enforced alignment becomes more severe as the garbage model length increases. Opposed to the unconstrained model, where the impact of the garbage model length was negligible, the scores drop dramatically when more states are added (cf. Tab. 7). Informal experiments with different parameter sets confirmed these findings. Thus, although slight improvements in performance could be observed given the right choice of parameters, adding strong structural constraints may

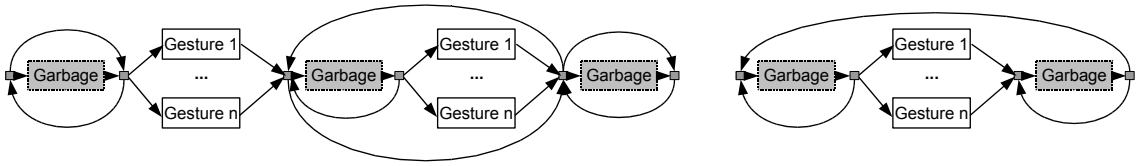

Fig. 7. Model structure of constrained HMM incorporating prior structural knowledge. Left: CHMM1, right: CHMM2.

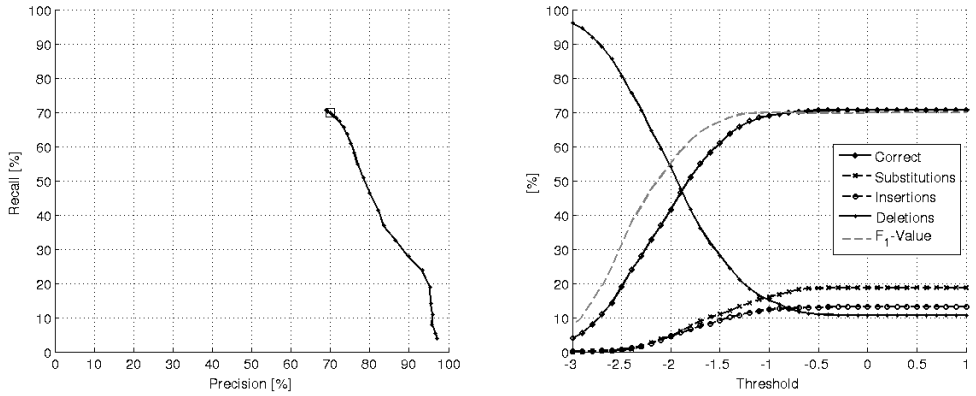

Fig. 8. Segmentation experiment: Precision-Recall curve (left) and C/S/I/D plot (right) for the constrained model CHMM2.

Table 7

Impact of the garbage model size. Top: Model CHMM2. Bottom: Unconstrained parallel decoding for comparison. All values are in %. Confidence intervals range from 1.3 to 1.8, but are omitted for space and readability reasons

Garbage model states	1	3	6	8
F_1 score (Precision, Recall)	68.2 (66.3, 70.2)	70.0 (69.9, 70.0)	60.9 (72.3, 52.7)	52.5 (71.2, 41.6)
max. Recall	74.7	70.6	52.9	42.1
F_1 score (Precision, Recall)	65.8 (62.5, 69.4)	67.4 (64.5, 70.5)	64.4 (60.8, 68.5)	65.6 (59.9, 72.4)
max. Recall	76.9	78.5	76.0	76.9

be a pitfall, and generally seems not advisable for the considered problem.

Overall, the results leave room for improvement, but are satisfactory given the realistic scenario and challenging data.

5. Summary

In this article, a system for visual interpretation of 3D arm gesture trajectories in a multi-camera scenario was presented. The focus was specifically on a type of gesture that is usually called ”emblem”: A simple sign used in natural inter-human communication, but with a well-defined meaning. It was motivated why this type of gesture is believed to be suited especially well for usage in gesture-based HMI. An image processing pipeline was presented, consisting of person detection and extraction of head and hand positions in monocular image streams, which are afterwards combined to form 3D trajectories. In particular, the problem of combining hypotheses from unsynchronized cameras obtained with variable frame rates was addressed.

Furthermore, a detailed feature study was presented, introducing trajectory-based features motivated by the field of online handwriting recognition that, to the authors' best knowledge, have not been used in gesture recognition before. It was also demonstrated that typical 3D emblematic hand gestures have an inherent planar nature, and therefore can be represented and normalized by projection on their so-called action plane. The suitability of the presented approach was evaluated on a realistic dataset containing nine different gestures and multiple persons, and satisfactory results have been achieved. Specifically, it was shown that the projection on the action plane yields competitive performance, and that the recognition performance can be improved incorporating different feature types via PCA for the proposed HMM classification framework. Additionally, the construction of a rejection model using out-of-domain motion capture data was demonstrated, and its suitability was evaluated on a challenging task.

Comparing the presented results to related work is difficult, since the authors are not aware of other existing gesture recognition systems with similar intentions, setting and preconditions. Furthermore, only very little standard evaluation datasets incorporating multi-view data of hand trajectories exist. However, the results presented in this article confirm the validity and suitability of the proposed approach.

References

[1] J. Alon et al. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1685–1699, 2009.

[2] A. Asadi, R. Schwartz, and J. Makhoul. Automatic detection of new words in a large vocabulary continuous speech recognition system. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 125–128, Albuquerque, NM, USA, 1990.

[3] C. G. Atkeson and J. M. Hollerbach. Kinematic features of unrestrained vertical arm movements. *Journal of Neuroscience*, 5(9):2318–2330, September 1985.

[4] R. Bowden, D. Windridge, T. Kadir, A. Zisserman, and M. Brady. A linguistic feature vector for the visual interpretation of sign language. In T. Pajdla and J. Matas, editors, *Proc. European Conference on Computer Vision*, volume 3021 of *Lecture Notes in Computer Science*, pages 390–401. Springer, 2004.

[5] S. Calinon and A. Billard. Recognition and reproduction of gestures using a probabilistic framework combining PCA, ICA and HMM. In *Proc. Int. Conf. on Machine Learning*, pages 105–112, Bonn, Germany, 2005.

[6] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland. Invariant features for 3-d gesture recognition. In *2nd Int. Workshop on Face and Gesture Recognition*, pages 157–162, 1996.

[7] G. Caridakis et al. SOMM: Self organizing markov map for gesture recognition. *Pattern Recognition Letters*, 31:52–59, 2010.

[8] C.-Y. Chien, Huang C.-L., and C.-M. Fu. A vision-based real-time pointing arm gesture tracking and recognition system. In *Proc. IEEE Int. Conf. on Multimedia and Expo*, pages 983–986, 2007.

[9] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 25(2):564–575, 2003.

[10] K. Daifallah, N. Zarka, and H. Jamous. Recognition-based segmentation algorithm for on-line arabic handwriting. In *Proc. Int. Conf. on Document Analysis and Recognition*, pages 886–890, Barcelona, Spain, 2009.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[12] Jacob Eisenstein and Randall Davis. Visual and linguistic information in gesture classification. In *Proc. Int. Conf. on Multimodal Interfaces*, pages 113–120, State College, PA, USA, 2004.

[13] P. Ekman and W.V. Friesen. The repertoire of nonverbial behavior: Categories, origins, usage and coding. *Semiotica*, 1:49–98, 1969.

[14] M. Elmezain, A. Al-Hamadi, and B. Michaelis. A robust method for hand gesture segmentation and recognition using forward spotting scheme in conditional random fields. In *Proc. Int. Conf. on Pattern Recognition*, 2010.

[15] M. Elmezain et al. A hidden markov model-based continuous gesture recognition system for hand motion trajectory. In *Proc. Int. Conf. on Pattern Recognition*, Tampa, FL, USA, 2008.

[16] G. A. Fink. *Markov Models for Pattern Recognition*. Springer, 2008.

[17] G. A. Fink and T. Plötz. Developing pattern recognition systems based on Markov models: The ESMERALDA framework. *Pattern Recognition and Image Analysis*, 18(2):207–215, 2008.

[18] G. A. Fink, M. Wienecke, and G. Sagerer. Video-based on-line handwriting recognition. In *Int. Conf. on Document Analysis and Recognition*, pages 226–230, Seattle, 2001. IEEE.

[19] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[20] A. Graves et al. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868, May 2009.

[21] H. Junker, O. Amft, P. Lukowicz, and G. Tröster. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recognition*, 41:2010–2024, 2008.

[22] M. B. Kaâniche and F. Brémond. Gesture recognition by learning local motion signatures. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010.

[23] A. Kendon. *The Biological Foundation of Gestures. Motor and Semiotic Aspects.*, chapter Current Issues in the Study of Gestures, pages 23–47. Lawrence Erlbaum Assoc., 1986.

[24] D. Kim, J. Song, and D. Kim. Simultaneous gesture segmentation and recognition based on forward spotting accumulative

HMMs. *Pattern Recognition*, 40:3012–3026, 2007.

[25] T. Kirishima, K. Sato, and K. Chihara. Real-time gesture recognition by learning and selective control of visual interest points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):351–364, March 2005.

[26] Christian Kleine-Cosack, Marius H. Hennecke, Szilard Vajda, and Gernot A. Fink. Exploiting acoustic source localization for context classification in smart environments. In Boris de Ruyter, Reiner Wichert, David Keyson, Panos Markopoulos, Norbert Streitz, Monica Divitini, Nikolaos Georgantas, and Antonio Mana Gomez, editors, *Ambient Intelligence*, volume 6439 of *Lecture Notes in Computer Science*, pages 157–166. Springer Berlin / Heidelberg, November 2010.

[27] H. Li and M. Greenspan. Multi-scale gesture recognition from time-varying contours. In *Proc. Int. Conf. on Computer Vision*, volume 1, pages 236–243, 2005.

[28] M. R. Malgireddy, J. J. Corso, S. Setlur, V. Govindaraju, and D. Mandalapu. A framework for hand gesture recognition and spotting using sub-gesture modeling. In *Proc. Int. Conf. on Pattern Recognition*, 2010.

[29] K. Nickel and R. Stiefelhagen. Visual recognition of pointing gestures for human-robot interaction. *Image and Vision Computing*, 25(12):1875–1884, 2007.

[30] S.C.W. Ong and S. Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):873–891, 2005.

[31] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):63–84, January 2000.

[32] T. Plötz and G. A. Fink. An efficient method for making unsupervised adaptation of HMM-based speech recognition systems robust against out-of-domain data. In *Proc. 4th Int. Workshop on Natural Language Processing and Cognitive Science*, Funchal, Portugal, June 2007.

[33] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28:976–990, 2010.

[34] J. Rett and J. Dias. Gesture recognition using a marionette model and dynamic bayesian networks (DBNs). In *Proc. Int. Conf. on Image Analysis and Recognition*, pages 69–80, Póvoa de Varzim, Portugal, 2006.

[35] J. Richarz and G. A. Fink. Feature representations for the recognition of 3d emblematic gestures. In A. A. Salah, T. Gevers, N. Sebe, and A. Vinciarelli, editors, *Proc. Int. Workshop on Human Behavior Understanding, in conjunction with 20th Int. Conf. on Pattern Recognition*, volume 6219 of *Lecture Notes in Computer Science*. Springer, 2010.

[36] J. Richarz, T. Plötz, and G. A. Fink. Real-time detection and interpretation of 3d deictic gestures for interaction with an intelligent environment. In *Proc. Int. Conf. on Pattern Recognition*, Tampa, Florida, 2008. TuBCT8.7.

[37] B. Schauerte, J. Richarz, T. Plötz, C. Thurau, and G. A. Fink. Multi-modal and multi-camera attention in smart environments. In *Proc. Int. Conf. Multimodal Interfaces and Workshop on Machine Learning for Multi-modal Interaction (ICMI-MLMI)*, pages 261–268, Cambridge, MA, USA, 2009.

[38] J. Schenk, M. Kaiser, and G. Rigoll. Selecting features in online handwritten whiteboard note recognition: SFS or SFFS? In *Proc. Int. Conf. on Document Analysis and Recognition*, pages 1251–1254, Barcelona, Spain, 2009.

[39] J. Schmidt, N. Hofemann, A. Haasch, J. Fritsch, and G. Sagerer. Interacting with a mobile robot: Evaluating gestural object references. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2008.

[40] N. Sebe. Multimodal interfaces: Challenges and perspectives. *Journal of Ambient Intelligence and Smart Environments*, 1(1):23–30, 2009.

[41] A. Shamaie and A. Sutherland. Bayesian fusion of hidden markov models for understanding bimanual movements. In *Proc. Int. Conf. on Automatic Face and Gesture Recognition*, Seoul, Korea, 2004.

[42] L. Sigal and M. J. Black. HumanEva: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical Report CS-06-08, Brown University, Providence, RI, 2006.

[43] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1473–1488, 2008.

[44] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1975.

[45] Q. Wang et al. Viewpoint invariant sign language recognition. *Computer Vision and Image Understanding*, 108:87–97, 2007.

[46] S. B. Wang, A. Quattoni, L.-P. Morency, and D. Demirdjian. Hidden conditional random fields for gesture recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2006.

[47] A. D. Wilson and H. Benko. Combining multiple depth cameras and projectors for interactions on, above, and between surfaces. In *Proc. ACM Symposium on User Interface Software and Technology*, 2010.

[48] C. Wu and H. Aghajan. Model-based human posture estimation for gesture analysis in an opportunistic fusion smart camera network. In *Proc. Int. Conf. Advanced Video and Signal based Surveillance*, pages 453–458, London, UK, 2007.

[49] A. Y. Yang, R. Jafari, S. Shankar Sastry, and R. Bajcsy. Distributed recognition of human actions using wearable motion sensor networks. *Journal of Ambient Intelligence and Smart Environments*, 1(2):103–115, 2009.