Vision Based Hand Puppet

Cem Keskin, İsmail Arı, Tolga Eren, Furkan Kıraç, Lukas Rybok, Hazım Ekenel, Rainer Stiefelhagen, Lale Akarun

Abstract— **The aim of this project is to develop a multimodal interface for a digital puppetry application, which is suitable for creative collaboration of multiple performers. This is achieved by manipulating the low- and high level aspects of 3D hierarchical digital models in real–time. In particular, the hands and the face of multiple performers are tracked in order to recognize their gestures and facial expressions, which are then mapped to kinematic parameters of digital puppets. The visualization of the puppet is provided as a feedback for the performer, and as an entertainment medium for the audience. Possible uses of this system include digital theaters, simplified animation tools, remote full-body interaction and sign-language visualization.**

The application consists of two separate hand tracking modules aimed at different shape and motion parameters, a facial feature tracker, a hand gesture and facial expression classification module, an XML based low-bandwidth communication module and a visualization module capable of handling inverse kinematics and skeletal animation. The methods employed do not depend on special hardware and do not require high computational power, as each module runs on separate computers.

I. Introduction

Puppetry is an ancient form of art and performance, which is known by most cultures in slightly different forms. Puppeteers either use sticks and ropes, or their bodies, as in hand puppetry, to animate the puppets. The forms of puppetry that do not require special devices are quite intuitive, and allow even a first time performer to succeed in animating a puppet in a visually pleasing manner.

Creative collaboration is common among most art forms, and puppetry can also be performed by multiple performers. Generally, multiple puppeteers manipulate separate puppets in order to form a theater play, but for some advanced puppets, several performers may be needed for a single puppet.

In digital puppetry, traditional puppets are replaced by 2D or 3D models that usually consist of several limbs, which can be manipulated separately and concurrently. In this work, we are concerned with 3D models that have a hierarchical skeleton with a high degree of freedom. Unless some high level animation parameters are defined, which act on several joints at the same time, these models are hard to manipulate using only low level parameters. This process is akin to digital animation, where animators create animations by carefully constructing the sequence frame by frame by manipulating each joint, which are then interpolated to form a fluent motion. This is a hard and time consuming process. Our aim is to create an intuitive interface, which allows non–expert performers to collaboratively manipulate a complex 3D model in real time.

Recent developments in technology allowed using motion capture devices to render moving humanoid models in a realistic manner. This technology is mainly used for commercial applications such as games and movies, and therefore, involves special worn devices and sensors. This method of capturing animation parameters is expensive and clumsy. In this work, we are interested in estimating animation parameters using basic sensors without using markers and without the help of special devices. Previous work in this area includes CoPuppet, a system developed by Bottoni *et. al*, which makes use of gestures and voice and allows multiple users to act on a single puppet in a collaborative manner [Bottoni]. Whereas CoPuppet captures hand gestures using a special device, we use simple cameras and also allow facial expressions and head movements.

C. Keskin, İ. Arı, F. Kıraç, and L. Akarun are with Boğaziçi University, Turkey. T. Eren is with Sabancı University, Turkey. L. Rybok, H. Ekenel and R. Stiefelhagen are with Karlsruhe Institute of Technology, Germany.

The main objective of this project is to design and implement a real-time vision-based digital puppetry system that does not rely on special sensors or markers. This work involves tracking of both hands, shape parameter estimation, motion tracking, gesture recognition, facial parameter tracking, expression classification, and also provides a graphical output that can give feedback about the efficiency and correctness of all the related modules in an eye pleasing and entertaining manner.

This report is structured as follows. In Section II we briefly describe the framework and its modules. In Section III, we give details of each module of the system. Particularly, in Section III-A we describe the stereo–vision based hand tracking module. In Section III-B, we provide the details of the facial expression detection and tracking module. Hand pose estimation module is described in Section III-C, and the recognition module is explained in Section III-D. The network protocol is given in Section III-E, and finally, the visualization module is explained in Section III-F. We provide discussions and mention our future work in Section IV.

II. System Description

This system is designed to allow multiple performers to collaborate in an intuitive manner. The only sensors used are cameras, and performers do not need to wear special markers. The number of puppeteers to perform is not limited, and they are not restricted to be at the same place. Each performer can communicate with the puppet over the network and get visual feedback at the same time.

Performers either use their hands or their faces to manipulate the puppet. In both cases, low level shape parameters are tracked, which are used to recognize certain hand gestures or facial expressions. Known gestures and expressions are used to give high level commands to the puppet, whereas shape parameters are directly used to manipulate certain limbs of the puppet.

Digital puppets are standardized by using a single skeleton for every model in order to allow seamless integration of new modules without complication. This minimizes the amount of knowledge that needs to be passed from the visualization module to the performer, as each puppet is virtually the same to the tracker module. Each module can manipulate each joint, and can give any high level command. Using a single skeleton does not constrain the shape of the puppet, but restricts the number of degrees of freedom that can be associated with a model. Specifically, we use a humanoid skeleton, which can be used to animate different objects, such as humans, animals, but also trees and buildings through rigging.

The most important criterion in choosing methodology is speed, as all the modules are required to run in real–time. The time it takes the puppeteer to perform and receive feedback should be minimal. Therefore, accuracy is sacrificed to allow rapid decision taking.

System flowchart is given in Figure 1. The framework uses three different tracking modules. The face feature tracking module uses a single camera facing the head of a single performer, and uses an active shape model to track certain landmarks on the face of the performer in real time. Hand pose estimation module uses multiple uncalibrated cameras to extract the silhouettes of the hand of the performer, and then tries to estimate the skeletal parameters that would conform to

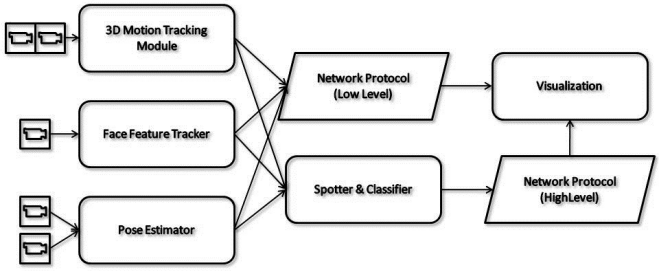

Fig. 1. System flowchart

all the silhouettes. 3D motion tracking module uses a stereo camera, or a pair of calibrated cameras to reconstruct the 3D trajectory of the hand. It also fits an ellipsoid on the estimated 3D point cloud for the hand, revealing more low level parameters associated with the hand shape.

Each of the tracking modules passes the parameter sequence to the recognition spotter and classifier module, which looks for known patterns in continuous streams of data. Face data is used to spot and recognize basic facial expressions, such as sadness and happiness. Motion data and the ellipsoid parameters retrieved from the 3D module is used to recognize 3D hand gestures. Likewise, the pose parameters supplied by the pose estimator module are used to spot certain hand posture–gesture combinations.

The visualization module continuously reads data coming from the modules and renders the puppet accordingly. The tracking and recognition modules send commands in the form of binary XML files over the network. The visualization module parses these and applies all the commands.

III. Methodology

A. Stereo–vision based hand tracking

In order to recognize hand gestures, the position of the hands first needs to be localized. To this end, we make use of a Bumblebee stereo camera system by Point Grey, allowing us the recovery of the 3D position of the hands. For hand localization, first skin-color segmentation is applied to the images captured with the left camera. Following the results form [1], we calculate for each pixel the probability of being skin-colored using Bayes' Theorem:

$$P(\text{Skin}|x) = \frac{P(x|\text{Skin}) \cdot P(\text{Skin})}{P(x)} \quad (1)$$

Here the class-conditional $P(x|\text{Skin})$ is modeled with a histogram trained on face images. Since in our scenario the hand is assumed to be the only visible skin-colored region in the image and is expected to occupy only a small fraction of it, the prior is set to $P(\text{Skin}) = 0.1$. An example of a skin-color probability map obtained using the described approach can be seen in Fig. 2.

For hand-detection, the skin-color map is thresholded, followed by the application of morphological operations to smooth out noise and skin-colored blobs are finally extracted using a connected component analysis algorithm. Further, the hand position is estimated by the center of the biggest blob. Finally, the area around the detected hand is matched in the right camera image using normalized cross-correlation and the so obtained disparity value is employed to calculate the 3D location of the hand.

Since tracking by detection is not stable enough and therefore results in noisy hand trajectories, the hand is tracked with a particle filter [2]. For the observation model again both skin-color and depth information are used. In order to achieve a low computational complexity the hand is modeled in 3D with a fixed-sized rectangle

Fig. 2. Example skin-color probability map used for hand tracking and detection

that is projected to the image plane for each particle (see Fig. 3) to evaluate the likelihood function.

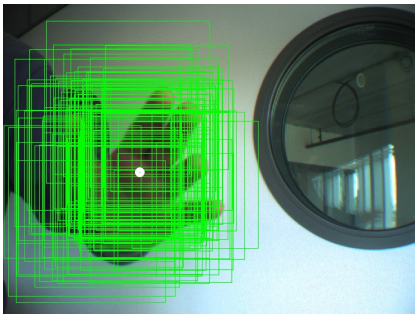

Fig. 3. Projected hand hypotheses (particles) of the tracker. The dot denotes the derived final hypothesis for the hand position.

The color cue is calculated by averaging the skin-color probability values in the region of interest defined by the projected particle multiplied by the number of pixels in the skin-color map that exceed a certain threshold. The multiplication ensures that particles that cover the hand region the most get a higher score than particles associated to a smaller region. The calculation of the depth score consists of a comparison of the disparity value defined by the particle and the disparity value obtained from matching the projected particle's area in the right camera image.

B. Vision based Emotion Recognition

The digital puppet is aimed to perform seven different emotion states in real time synchronously with the human performer. The chosen states are the six universal expressions (surprise, anger, happiness, sadness, fear, disgust) and the neutral facial expression.

With the promising results achieved in face and facial landmark detection research, emotion recognition started to attract the researchers especially in the last decade. Facial expression recognition and emotion recognition are used as overlapping terms by vision researchers since facial expressions are the visually apparent presences of internal emotions. Some surveys on the subject are available, such as Fasel and Luettin's review [3] on automatic facial expression analysis and Pantic and Rothkrantz's work [4] that examines the state of the art approaches in automatic analysis of facial expressions. Different approaches have been tried in facial expression analysis systems. All approaches share a common framework starting with face and facial landmark detection, facial feature extraction and expression classification. The main focus seems to be using static images whereas some papers discuss emotion recognition from image sequences, i.e. videos. For details, the reader may refer to Arı [5].

In this work, a similar method to Busso et al. is followed, where the authors report that they use commercial software for landmark tracking, partition the face into five regions of interest, create a histogram using PCA coefficients of these regions of interest and finally assign the video to one of the four expression classes using 3NN [6]. The whole system is run fully automatically and real time. First, we track the facial landmarks in face videos using Active Shape Model (ASM) based tracker which is modified from Wei's asmlibrary on Google code [7]. The landmark locations are used for the computation of high level features which is fed to the distance-based classifier which is an extended version of the nearest neighbor classifier.

1) Facial Landmark Tracking: ASMs are one of the state-of-the-art approaches for landmark tracking [8]. The ASM is trained using the annotated set of face images. Then, it starts the search for landmarks from the mean shape aligned to the position and size of the face located by a global face detector (in our case Viola-Jones face detector). Afterwards, the following two steps are repeated until convergence (i) adjust the locations of shape points by template matching of the image texture around each landmark and propose a new shape (ii) conform this new shape to a global shape model (based on PCA). The individual template matches are unreliable and the shape model improves the results of the weak template matchers by forming a stronger overall classifier. The entire search is repeated at each level in an image pyramid, from coarse to fine resolution using a multi-resolution approach. In the case of tracking, the model is initiated from the shape found on the previous frame instead of using the mean shape.

ASM performs better when person-specific model is trained. In this work, we had a generic model involving different subjects and a person-specific model which is trained from the face images of the test subject.

2) Feature Extraction: ASM-based tracker provides 116 facial landmarks which are seen on the left of Figure 4. Using the locations of the landmarks directly as feature seems not to be a good choice since the tracker works fine for many parts of the face such as eyebrows, eyes, chin, and nose, but not very robust for detecting the exact movements of the lips which is the most non-rigid part of the face. This phenomenon results from the fact that ASM models the face holistically and the small variations in the lips may be discarded during constraining by PCA. Moreover, the intensity difference on the lip boundaries are not as obvious as the other parts as seen in Figure 4. Thus, the landmark locations are used for computing 7 high level features seen on the right of Figure 4 as follows:

1) Average eye middle to eyebrow middle distance
2) Lip width
3) Lip height
4) Vertical edge activity over the forehead region
5) Horizontal edge activity over the lower forehead region
6) Sum of horizontal and vertical edge activity over the right cheek
7) Sum of horizontal and vertical edge activity over the left cheek

The first three features are computed using the Euclidean distance between the related landmarks. For the remaining features, the image is blurred with a Gaussian kernel and afterwards filtered with a Sobel kernel on horizontal and vertical axes separately. Then the average absolute pixel value is computed in each region. The vertical edge activity image is shown on the right of Figure 5 for surprise state. For example, the average pixel value residing in the forehead quadrilateral in the vertical edge activity image is found as the 4th feature.

In the setup we used, the test subject starts with neutral expression and waits in neutral state for about two seconds. The average values for the features are computed and the features in the remaining frames are normalized.

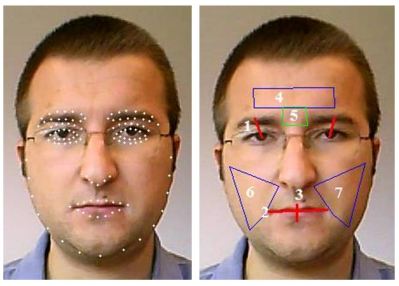

Fig. 4. Facial landmarks (on the left) and the regions of interest (on the right) used for feature extraction.

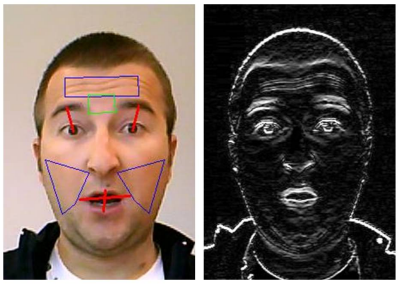

Fig. 5. A snapshot of surprise state (on the left) and corresponding vertical edge activity image (on the right).

3) Emotion Classification: Since the facial expressions vary with the subject and with the conditions of the environment such as illumination, we aimed a training setup which can be easily configured for a specific subject in a specific environment. During the training period, the subject starts by waiting in the neutral state and afterwards repeats each state five times. The interface records the feature vectors for these states, which are a total of 35 different vectors belonging to seven classes.

During testing, the subject again starts with neutral expression for normalization (and adaptation to environment). In the succeeding frames, the feature vector is computed for each frame and then its average distance to the training feature vectors are computed for each class. Let $d_i, i = 1,,7$ be the average distance computed for each class. The distances represent dissimilarity whereas $s_i = e^{-d_i}$ can be used as a similarity measure. Finally, s_i values are normalized such that their sum is one and they can be regarded as likelihoods. This method is superior to nearest neighbor (NN) classification or kNN, since it performs soft assignment instead of hard assignment.

4) Extension for Unseen Subjects: For the training of ASM, 25 face images are gathered from the test subject. A generic model is used for automatic landmarking of the faces instead of annotating them from scratch. Then the locations of the landmarks are fine-tuned and the person specific model is trained from them. As mentioned in the previous subsection, the training of the classifier can be done easily for a specific subject under the current environmental conditions. This process can be followed for extending the system to work for new subjects.

5) Results: The results of the emotion recognition system can be seen in Figure 6. The likelihoods of the related emotional states are drawn on the user interface. A core2duo 1.2 GHz laptop computer with 3GB RAM can process about 15 frames per second with 320240 pixels resolution. The built-in laptop webcam is used.

The proposed system gives promising results for handling partial occlusions as seen in Figure 7.

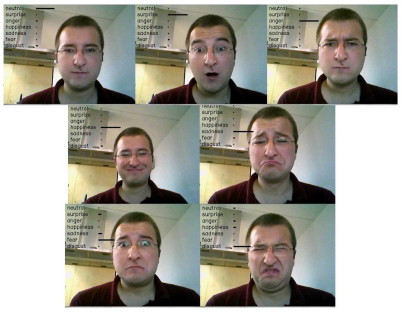

Fig. 6. Results of emotion recognition (neutral, surprise, anger, happiness, sadness, fear, disgust).

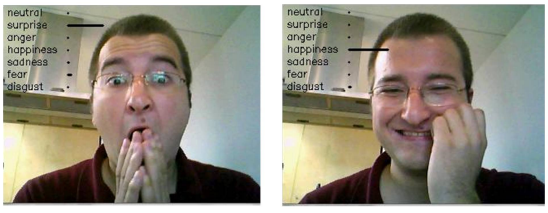

Fig. 7. Results of emotion recognition with partial occlusion.

The system is successful at discriminating seven different emotional states from a specific subject's video in real time. It is translation and scale invariant. The rotation invariance, on the other hand, depends on the performance of the tracker where ASM-based tracker provides the landmarks successfully for up to 20-30 degrees of rotation. The proposed system is open for improvements such as introducing more high level features on demand and extension for pose change.

C. Hand Pose Estimation Module

This module tracks the pose of a hand using silhouettes of the hands taken from two different cameras. Features are selected as silhouettes of a two camera setup. Dimensionality reduction is done by optimizing a Gaussian process latent variable model (GPLVM). For speeding up the optimization process Sparse GPLVM formulations have been used. Flowchart of the hand pose tracking module is shown in Figure 8.

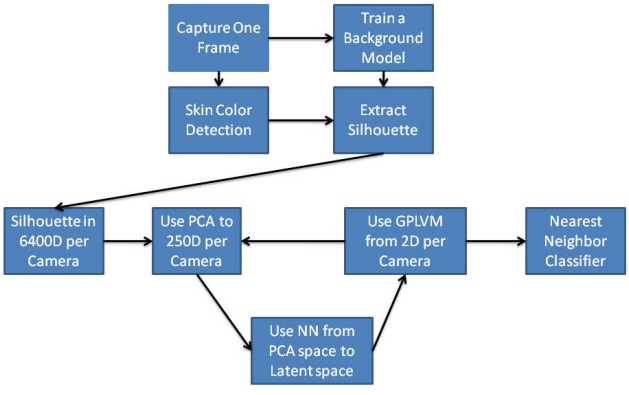

Fig. 8. Flowchart of the hand pose tracking module.

1) Training Set Generation: The ground truth hand silhouette images for both cameras are generated by rendering a 3D hand model. "Poser" software's 3D hand object is manipulated through a Python script. The silhouettes are extracted and saved as PNG image files which then are loaded into Matlab for further training. An example of a hand silhouette taken from left and right cameras are shown in Figure 9.

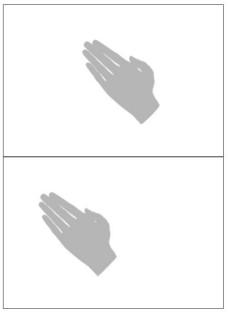

Fig. 9. An example of hand silhouettes as taken from left and right cameras respectively.

2) Dimensionality Reduction Using GPLVM: The proposed hand tracking algorithm determines the hand pose from two silhouette images without ambiguity. Hand silhouette images are 80x80 pixel resolution images. Normally a feature extraction scheme would be applied to the silhouettes. However, in this case the pixels themselves are treated as features and given directly to GPLVM for dimensionality reduction. This provides an opportunity to test the quality of GPLVM's reduction. If GPLVM is able to capture the essence of the factors generating the silhouettes, then it will be able to capture a low dimensional manifold in the low dimensional latent space.

Considering each pixel as a unique feature of the hand, we have a 6400 dimensional feature vector per camera. Since two cameras are used the feature space is in 12800 dimensions. GPLVM requires an initialization step, a choice of a non-linear function for producing covariance matrices. Then a global search algorithm is applied to optimize the non-linear mapping. GPLVM is initialized with Probabilistic PCA (PPCA) and a radial basis function (RBF) kernel is used as the non-linear covariance function generator.

The captured manifold is represented in the latent space as in Figure 10. Red crosses represent the silhouettes captured by the left camera and the green circles are the silhouettes captured from the right camera. 40 silhouette images per camera are used in the reduction phase.

As can be seen the silhouette manifolds for both of the cameras are extracted in a meaningful manner where there is only one ambiguious point in 2D latent space. The tracking algorithm can be designed in a way to handle this kind of ambiguities. Since GPLVM finds a mapping from latent space X to the feature space Y but not the other way around, for tracking the hand pose, we have to generate new silhouettes from the generative model captured by the GPLVM and match the generated model with the captured silhouette. This action involves a global search procedure where one needs to instantiate numerous variations of silhouettes from the latent space. Instantiations should be compared to the silhouette in the currently captured frame. Then the closest silhouette's pose can be considered as the pose of the currently captured hand silhouette.

3) Mapping from Feature Space Y to Latent Space X: GPLVM finds a backward mapping from latent space X to feature space Y. For the real time application of the hand pose tracking system, a mapping from feature space to data space is required. Since this mapping is not provided by the GPLVM, it is infeasible to generate all the possible

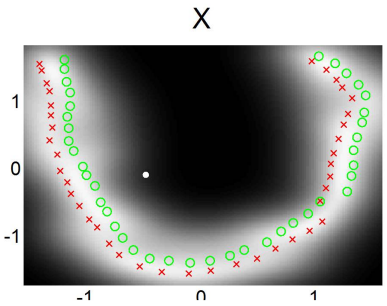

Fig. 10. Captured Manifold of Stereo Hand Silhouettes in 2D latent space X.

X to Y mappings by observing the generated silhouette with the currently captured one. Therefore a mapping from Y to X is also required. This mapping will be used as an initial point of a local search algorithm in the latent space afterwards. An MLP with one hidden layer with 15 hidden neurons is used for learning the mapping from feature space to latent space.

4) Classification in Latent Space: 2-dimensional latent space has been found in a smooth fashion by GPLVM optimization. Therefore nearest neighbor matcher has been used in the latent space as a classifier without applying a local search algorithm. Ground truth angles of the hand poses are known. An exact pose match is looked for. Any divergence from the exact angles is considered as a classification error. For the synthetic environment prepared by poser a classification performance of 94% has been reached in 2D latent space.

D. Gesture and Expression Classifier

The gesture and expression recognition module is mainly used to give high level commands to the puppet. This is either achieved by performing hand gestures or with facial expressions. Hence, performers can initiate complicated animation sequences by performing a certain hand gesture, or they can change the appearance of the puppet by making certain facial expressions. For instance, a certain hand movement and posture can make the puppet jump or dance around, and performing a happy face can *make* the puppet happy via changes in textures or posture.

Since the output is an animation that is meant to be eye pleasing, discontinuities in the animation are not desirable. Therefore, there are no predetermined gestures, hand shapes or expressions that inform the system about the start or end of a gesture. This means that all gestures or expressions are performed continuously, with no indicators for separation. Thus, the gesture classification module also needs to *spot* known gestures or expressions in continuous streams.

1) Preprocessing: Each module provides a continuous stream of data consisting of different feature vectors. Pose estimation module uses skeleton parameters, motion tracking module uses 3D motion parameters of the hand, and the facial landmark tracking module uses landmark locations as feature vectors. Even though a generic sequence classifier such as a hidden Markov model (HMM) can be used to recognize patterns in each of these streams, the characteristics of the feature vectors are significantly different, and require separate preprocessing steps before using a generic recognition module.

Gestures defined with hand motions are scale invariant, and the starting point or the speed of the gesture is not important. Therefore, absolute coordinates of the hand location make little sense. In a preprocessing step, we find the relative motion of the hand in each frame, and then apply vector quantization to simplify calculations.

Changes in hand posture do not possess the characteristics of hand motion. As the hand posture is basically the rotation parameters of each joint in the hand skeleton, scaling and translation do not affect the resulting feature vectors. Therefore, the absolute parameters can directly be used. As there are more than 30 degrees of freedom associated with each hand, most of which are either unused or correlated, we also apply PCA to reduce the dimensionality. The reduction matrix is learned from the training data retrieved from Poser.

Facial expressions are very different, as they represent states as well as processes. Therefore, facial features should be used to recover both dynamic and static aspects of the face. Also, the absolute locations of landmarks is affected by the global motion of the head. We first estimate this global motion and reverse it to find local changes in face. Then we use the first derivative of the locations and apply PCA. Each static state of the face have the same observations this way, since each of them correspond to zero motion. By an intelligent choice of number of states and training data, we correctly represent each static state via the dynamic processes that lead to them.

2) Hidden Semi Markov Models: By far the most common method used for sequence classification is by using HMMs. HMMs model sequences with latent states, and the state durations implicitly via the self transition probabilities of each state. This leads to a geometric distribution of durations of each state. As the states model subsequences of gestures or expressions, modeling every duration with a geometric distribution can have undesirable effects. The graphical model of HMMs is given in Figure 11.

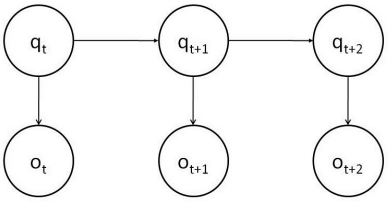

Fig. 11. Graphical model of a HMM.

HSMMs are extension of HMMs that can be thought of as graphical models consisting of *generalized states* emitting sequences, instead of states emitting a single observable [9]. For a general HSMM, there is no independence assumption for these emitted sequences and their durations. Also, the sequences emitted by the generalized states, i.e. the *segments* can have any arbitrary distribution. Different constraints on these distributions and assumptions of independence lead to different types of HSMMs. For instance, each segment can be thought of as produced from another HMM or state space model embedded in the generalized state, in which case the HSMM is known as a *segment model*. On the other hand, if the segment consists of a joint distribution of conditionally independent and identically distributed observations, the model becomes an explicit duration model [10]. Other variants include 2–vector HMM [11], duration dependent state transition model [12], inhomogeneous HMM [13], non–stationary HMM [14] and triplet Markov chains [15]. Detailed overview of HSMMs can be found in the tutorial by Yu [16].

HSMMs can be realized in the HMM framework, where each HMM state is replaced with a generalized or complex HMM state that consists of the HMM state and an integer valued random variable associated with that state, which keeps track of the remaining time. The state keeps producing observations as long as its duration variable is larger than zero. Hence, each state can emit a sequence of observations, the duration of which is determined by the length of time spent in that state. The corresponding graphical model is

depicted in Figure 12.

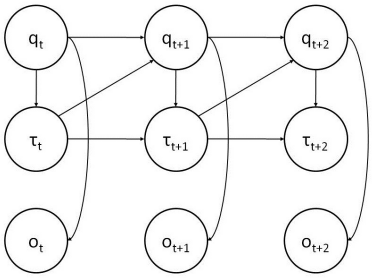

Fig. 12. Graphical model of a HSMM.

3) Continuous Recognition: HMMs and also HSMMs are quite simply applied to isolated gesture recognition problems. A separate model is trained for each class of pattern, and each candidate sample is evaluated against all the models. The class that corresponds to the model, which gives the highest likelihood is selected. This process is more complicated for continuous gesture recognition. A simple solution is to extract many candidate sequences that end at the current instance and have started at different instances. Using a threshold model for non–gestures (half-gestures, coarticulations, unintentional movements), one can determine the class of observed gestures. This is an inefficient method, as several sequences are evaluated at each frame.

In this work, we train a separate small HSMM for each gesture and expression, and then combine them to form a larger HSMM. Thus, a single (long enough) candidate sequence is evaluated with a single large model. Using Viterbi algorithm to estimate the optimal state sequence, one can determine the gesture class from the likeliest current state. The gestures and expressions are assumed to be independent. Therefore, the extracted candidate sequence can be rather short, as previous gestures have no effect on the current one.

At each frame, this module extracts the candidate sequences from each stream and applies the Viterbi algorithm for HSMMs. If the end of a pattern is recognized, a high level command is sent to the visualization module, triggering the corresponding animation sequence or posture.

E. Communication

Efficient communication between the components of the system is a crucial task. Reducing the response time helps the performers, since they receive feedback faster. To retain genericness and simplicity of the system, an XML based communication protocol is used. Visualization module accepts binary XML files from other modules, parses them and applies parameters directly to the ongoing animation in real–time. The XML-based protocol is as follows:

```
<?xml version="1.0" encoding="UTF-8" ?>
<handPuppet timeStamp="str" source="str">
    <paramset>
        <H rx="f" ry="f" rz="f" />
        <ER ry="f" rz="f" />
        <global tx="f" ty="f" ry="f" rz="f" />
    </paramset>
    <anim id="str" />
    <emo id="str" />
</handPuppet>
```

Here, keywords *timestamp* and *source* are used to identify, sort and prioritize messages received from modules. The *paramset* subtree holds the low level joint modification parameters. Each item in this subtree corresponds to a single joint. As the system uses a rigid hierarchical skeleton, joints only need the rotation parameters. Translation and scaling are not allowed for joints. In order to move the entire skeleton, the keyword *global* is used, which also allows translation parameters. The *anim* keyword is used to trigger predefined animation sequences, which are bound to hand gestures. Likewise, the *emo* keyword is used to trigger predefined changes in appearance in the 3D model, which are triggered by detected facial expressions.

F. Visualization Module

Visual output of this project is an animated avatar controlled by the performers via several input methods. In order to let the users have adequate control of this visual representation, we have utilized a skeleton based animation technique. The skeleton consists of 16 joints and accompanying bones. Using a graphical user interface, the user can create target poses for the skeleton as animation keyframes. Then it is possible to create an animation between these poses using quaternion-based spherical linear interpolation. These can be saved and then later be triggered by the modules upon recognition of certain gestures and expressions. The humanoid skeleton can be seen in Figure 13.

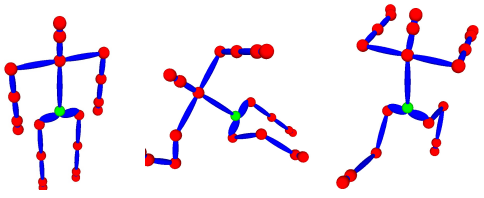

Fig. 13. Humanoid skeleton used for rigging and verification.

The skeleton has a humanoid form, but the actual 3D model does not need to be a humanoid model. The same skeleton can be bound to distinct models through rigging, which may include cartoonish creatures, or even objects with no real skeletons, such as trees or buildings.

To create a more realistic link between gesture inputs and animation output, an inverse kinematics based animation technique is implemented. This technique allows us to define end effector positions as well as particular rotations for each joint. In our context, an end effector refers to a hand, a foot or the head of the puppet. By setting goal positions for these end effectors, and applying constraints at each joint, we were able to produce a more realistic animation for the puppet.

For inverse kinematics, we have used the cyclic–coordinate descent algorithm. This algorithm starts the computation with the furthest joint away from the root. Then, each joint is traversed to minimize the difference between end effector and goal, optimally setting one joint at a time. With each joint update the end effector is also updated.

A generic model loader had also been implemented. This model loader accepts Autodesk FBX files as input, and associates model's geometric data with the underlying skeleton. When the user interacts with the skeleton, the spatial changes are automatically reflected to model's geometry, utilizing the association between model surface and skeletal bones.

IV. Conclusions

In this project, we developed a multimodal interface for digital puppetry. The design of the system allows manipulation of the low- and high level aspects of 3D hierarchical digital models in real–time. The hands and the face of multiple performers are tracked in order to

recognize their gestures and facial expressions. Each of the high and low level parameters estimated are mapped to kinematic parameters of digital puppets. The puppet is then animated accordingly, using several constraints and inverse kinematics.

Each module can run in separate workstations and communicate over the network using an XML based packet design. Also, each type of module can be used multiple times, and each such module can be associated with different aspects of the puppet. The methods employed do not depend on special hardware and do not require high computational power. Thus, the number of performers that can perform concurrently is only limited by the bandwidth of the visualization computer.

Each of the modules have been developed independently, and is shown to run in an efficient manner. Yet, the network protocol has not been adopted by every module, and the end result has not been demonstrated. When the modules can communicate with the visualization module, usability tests will be conducted and the interface will be improved accordingly. This is left as a future work.

V. Acknowledgments

This work is partially funded by the German Research Foundation (DFG) under Sonderforschungsbereich SFB 588 - Humanoid Robots - and by BMBF, German Federal Ministry of Education and Research as part of the GEMS programme. This work has also been supported by the Tübitak project 108E161.

References

[1] Son L. Phung, Abdesselam Bouzerdoum, and Douglas Chai, "Skin segmentation using color pixel classification: Analysis and comparison", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 1, pp. 148–154, 2005.

[2] Michael Isard and Andrew Blake, "Condensation - conditional density propagation for visual tracking", *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998.

[3] B. Fasel and J. Luettin, "Automatic facial expression analysis: A survey", *Pattern Recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[4] M. Pantic and L. Rothkrantz, "Automatic analysis of facial expressions: the state of the art", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424–1445, 2000.

[5] İsmail Arı, "Facial feature tracking and expression recognition for sign language", Master's thesis, Boğaziçi University, 2008.

[6] S. Yildirim M. Bulut C.M. Lee A. Kazemzadeh S. Lee U. Neumann C. Busso, Z. Deng and S. Narayanan, "Analysis of emotion recognition using facial expressions, speech and multimodal information", *Proceedings of the 6th international conference on Multimodal interfaces - ICMI '04*, p. 205, 2004.

[7] Y. Wei, "Research on facial expression recognition and synthesis", Master's thesis, Nanjing University, 2009.

[8] D. Cooper T. Cootes, C. Taylor and J. Graham, "Active shape models-their training and application", *Computer vision and image understanding*, vol. 61, pp. 38–59, 1995.

[9] Kevin P. Murphy, "Hidden semi-markov models", Tech. Rep., 2002.

[10] J.D. Ferguson, "Variable duration models for speech", *Proceedings of the Symposium on the Application of Hidden Markov Models to Text and Speech*, pp. 143–179, 1980.

[11] Vikram Krishnamurthy, John B. Moore, and Shin-Ho Chung, "On hidden fractal model signal processing", *Signal Process.*, vol. 24, no. 2, pp. 177–192, 1991.

[12] S.V. Vaseghi, "Hidden markov models with duration-dependent state transition probabilities", *Electronics Letters*, vol. 27, no. 8, pp. 625–626, 1991.

[13] P. Ramesh and J.G. Wilpon, "Modeling state durations in hidden markov models for automatic speech recognition", *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 1, pp. 381–384, 1992.

[14] Bongkee Sin and Jin H. Kim, "Nonstationary hidden markov model", *Signal Process.*, vol. 46, no. 1, pp. 31–46, 1995.

[15] Hulard C. Pieczynski, W. and T. Veit, "Triplet Markov chains in hidden signal restoration", in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, 2003, vol. 4885, pp. 58–68.

[16] Shun-Zheng Yu, "Hidden semi-markov models", *Artif. Intell.*, vol. 174, no. 2, pp. 215–243, 2010.

Cem Keskin is a Ph.D. candidate in the Computer Engineering Department at Boğaziçi University. He received his B.Sc. in Computer engineering and Physics from Boğaziçi University in 2003. He completed his M.Sc. in Computer engineering in the same university in 2006. His research interests include pattern recognition, computer vision, human computer interfaces and hand gesture recognition. He also took part in or contributed to several projects and applications, such as 3D reconstruction of buildings from multiple images, realistic and automatic coloring of buildings on real images, machine learning for genetic research, hand gesture based emergency control rooms, hand gesture based video games, synthesis of musical score from body movements and software algorithms for advanced computer graphics. He has a 2nd degree in the best student paper contest in SIU 2003. The title of his Ph.D. thesis reads "Generative vs. discriminative models for analysis and synthesis of sequential data".

İsmail Arı is a PhD candidate and teaching assistant in the Department of Computer Engineering in Boğaziçi University. His research interests include facial expression recognition, computer vision, and machine learning. Arı has an MS from the same department. Contact him at ismailar@boun.edu.tr.

Mustafa Tolga Eren is currently a PhD candidate in Computer Science and Engineering and a research assistant at Computer Graphics Laboratory in Sabancı University. He received his B.S. degree on Computer Science and Engineering Program from Sabancı University in 2006. His research interests include augmented reality, virtual environments and physically based simulations and animation.

Furkan Kıraç received the B. Eng. degree on Mechanical Engineering in 2000 and M. Sci. degree on Systems and Control Engineering in 2002 from the Boğaziçi University. He is currently a PhD candidate in Computer Engineering in Boğaziçi University. Since 2000, he has been actively working on computer vision based software design and has been the founder of a computer vision firm named Proksima which developed license plate recognition software since 2002. He has received a number of awards for his programming skills from TÜBTAK (The Scientific and Technological Research Council of Turkey). He has been given the 3rd degree in National Science Competition on Computer Science in Turkey, both in 1994 and 1995. He also has 1st and 2nd degrees in Regional Science Competitions of Marmara Territory in 1994 and 1995 respectively. Due to his performance in the national competitions he has represented Turkey in "London International Youth Science Forum '95" on computer science. He also has a 2nd degree in Best Paper Contest in SIU 2005 National Conference in Turkey. He is currently continuing his PhD research on "Human Motion Understanding" in Computer Engineering Department of Boğaziçi University, Turkey.

Lukas Rybok graduated from the University of Karlsruhe (TH), Germany in 2008 with a Diploma degree in Computer Science. He is currently a Phd candidate at the Karlsruhe Institute of Technology (KIT) working in the field of human activity recognition under the supervision of Prof. Rainer Stiefelhagen. His research interests include computer vision, pattern recognition and machine learning.

Hazim Ekenel is the head of "Facial Image Processing and Analysis" young investigator group at the Department of Computer Science in Karlsruhe Institute of Technology (KIT), Germany. He received his B.Sc. and M.Sc. degrees in Electrical and Electronic engineering from Boğaziçi University in 2001 and 2003, respectively, and Ph.D. degree in Computer Science from the University of Karlsruhe (TH) in 2009. He has been developing face recognition systems for smart environments, humanoid robots, and video analysis. He had been the task leader for face recognition in the European Computers in the Human Interaction Loop (CHIL) project and he organized face recognition evaluations within the CLEAR 2006, 2007 international evaluation campaigns. He has been responsible for face recognition in the German Humanoid Robots project. He is currently the task leader of face recognition in the French-German Quaero project. He has received the EBF European Biometric Research Award in 2008 for his contributions to the field of face recognition. In addition to the scientific work, many real-world systems have been developed based on his algorithm. With these systems, he received the Best Demo Award at the IEEE International Conference on Automatic Face and Gesture Recognition in 2008.

Dr. Rainer Stiefelhagen is a Professor at the Universität Karlsruhe (TH), where he is directing the research field on "Computer Vision for Human-Computer Interaction". He is also head of the research field "Perceptual User Interfaces" at the Fraunhofer Institut for Information and Data Processing (IITB) in Karlsruhe. His research focuses on the development of novel techniques for the visual and audio-visual perception of humans and their activities, in order to facilitate perceptive multimodal interfaces, humanoid robots and smart environments. In 2007, Dr. Stiefelhagen was awarded one of the currently five German Attract projects in the area of Computer Science funded by the Fraunhofer Gesellschaft. His work has been published in more than one hundred publications in journals and conferences. He has been a founder and Co-Chair of the CLEAR 2006 and 2007 workshops (Classification of Events, Activities and Relationships) and has been Program Committee member and co-organizer in many other conferences. Dr. Stiefelhagen received his Doctoral Degree in Engineering Sciences in 2002 from the Universität Karlsruhe (TH).

Lale Akarun is a professor of Computer Engineering in Boğaziçi University. Her research interests are face recognition and HCI. She has been a member of the FP6 projects Biosecure and SIMILAR, national projects on 3D Face Recognition and Sign Language Recognition . She currently has a joint project with Karlsruhe University on use of gestures in emergency management environments, and with University of Saint Petersburg on Info Kiosk for the Handicapped. She has actively participated in eNTERFACE, leading projects in eNTERFACE06 and eNTERFACE07, and organizing eNTERFACE07.