J Multimodal User Interfaces (2011) 4:61–79
DOI 10.1007/s12193-011-0059-3

ORIGINAL PAPER

# Automatic fingersign-to-speech translation system

**Marek Hrúz · Pavel Campr · Erinç Dikici · Ahmet Alp Kındıroğlu · Zdeněk Krňoul · Alexander Ronzhin · Haşim Sak · Daniel Schorno · Hülya Yalçın · Lale Akarun · Oya Aran · Alexey Karpov · Murat Saraçlar · Milos Železný**

Received: 4 February 2011 / Accepted: 29 April 2011 / Published online: 5 July 2011


**Abstract** The aim of this paper is to help the communication of two people, one hearing impaired and one visually impaired by converting speech to fingerspelling and fingerspelling to speech. Fingerspelling is a subset of sign language, and uses finger signs to spell letters of the spoken or written language. We aim to convert finger spelled words to speech and vice versa. Different spoken languages and sign languages such as English, Russian, Turkish and Czech are considered.

**Keywords** Fingerspelling recognition · Speech recognition · Fingerspelling synthesis · Speech synthesis

M. Hrúz (✉) · P. Campr · Z. Krňoul · M. Železný
Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czech Republic
e-mail: mhruz@kky.zcu.cz

P. Campr
e-mail: campr@kky.zcu.cz

Z. Krňoul
e-mail: zdkrnoul@kky.zcu.cz

M. Železný
e-mail: zelezny@kky.zcu.cz

E. Dikici · A.A. Kındıroğlu · H. Sak · H. Yalçın · L. Akarun · M. Saraçlar
Bogazici University, Istanbul, Turkey

E. Dikici
e-mail: erinc.dikici@boun.edu.tr

A.A. Kındıroğlu
e-mail: alpkindirogl@gmail.com

H. Sak
e-mail: hasim.sak@gmail.com

H. Yalçın
e-mail: hulyayalcin@gmail.com

L. Akarun
e-mail: akarun@boun.edu.tr

M. Saraçlar
e-mail: murat.saraclar@boun.edu.tr

A. Ronzhin · A. Karpov
SPIIRAS Institute, St. Petersburg, Russia

A. Ronzhin
e-mail: ronzhinal@iias.spb.su

A. Karpov
e-mail: karpov@iias.spb.su

## 1 Introduction

The main objective of this paper is to design and implement a system that can translate fingerspelling to speech and vice versa, by using recognition and synthesis techniques for each modality. Such a system enables communication with the hearing impaired when no other modality is available.

Fingerspelling is a representation of letters or numerals, using only the hands. In most sign languages it has been adopted to sign out-of-vocabulary words. In terms of automatic recognition, fingerspelling has the advantage of using limited number of finger signs corresponding to the letters/sounds in the alphabet. In this paper we present a system incorporating several modules. The modules are:

– Fingerspelling recognition

O. Aran
Idiap Research Institute, Martigny, Switzerland
e-mail: oaran@idiap.ch

D. Schorno
STEIM, Amsterdam, Netherlands
e-mail: dnl@xs4all.nl

– Speech recognition
– Fingerspelling synthesis
– Speech synthesis

The resulting modules can then be used as a part of a sign language to speech translation systems.

## 2 Literature survey

### 2.1 Fingerspelling recognition

The vision based fingerspelling recognition task involves conversion of fingerspelling hand gestures in an image sequence to letters of a written alphabets. Like automatic sign language recognition, fingerspelling recognition is still an active and challenging research topic. As fingerspelling is a subset of sign languages that makes use of hand gestures, same recognition methodologies are used for both. Therefore, we review techniques for both. It is composed of several subtasks such as segmentation of hands, extraction of features from segmented hand images and classification and temporal modeling of hand features. For hand segmentation, no single method to our knowledge performs unconstrained segmentation of hand images from cluttered backgrounds. Therefore studies in the literature make use of different cues and restrictions to perform hand segmentation. Popular methods for hand segmentation include using skin color cues [5], motion cues [37], shape cues [9] or depth information [39].

Vision based fingerspelling recognition systems usually use two different approaches to represent segmented hands. One approach is to use appearance based descriptive features to model human hands. In [58], an excellent survey on shape based descriptors is presented. Common appearance based features used in hand representation tasks include Histograms of Oriented Gradients [37], Local Binary Patterns [42], Elliptic Fourier Descriptors [35] or SURF [16]. Another approach to hand representation is the usage of generative 3D models to represent hands [39].

Sign gesture recognition is achieved through the classification and temporal modeling of these extracted hand features. In sign languages, there are two different kinds of gestures—static and dynamic gestures. While it is possible to recognize static gestures using just a single snapshot with classifiers such as k-Nearest Neighbors or Gaussian Mixture Models, dynamic gestures are recognized using either motion models or models that utilize temporal information such as Hidden Markov Models [37].

### 2.2 Fingerspelling synthesis

The goal of an automatic sign language synthesizer is the reproduction of human behavior during the signing. The sign language synthesizer should express manual components (position and shape of hands) as well as non-manual components (facial expression, lip articulation, etc.) of the performed signs. Sign language synthesis is implemented in several steps. Firstly the source utterance has to be translated into the corresponding sequence of signs since sign language has different grammar than the spoken one. Then the relevant signs have to be concatenated to form a continuous utterance.

The straightforward solution of sign language synthesis is to record the video of a signing human. The video records capture signed speech with very good quality and realism but simple concatenation of these records is not possible. Image based synthesis of facial movements or simple body pose could be considered [15, 56]. On the other hand, we can find virtual character (avatar) animations allowing low-bandwidth communication, arbitrary 3D position and lighting or replacement of the animation model or the character itself.

Sign language synthesis systems that use avatar animations are based on motion-capture (MoCap) or parametric data, or combination of both. The systems adopt techniques of MoCap often together with data gloves to obtain the movement data of sign languages. ViSiCAST project [13] was aimed at voice to English Sign Language translation and SignCom project [11] aimed at French Sign Language (LSF). The second type of systems generally rely on parametric models and scripts. These synthesis systems decode input such as special designed XML documents, HamNoSys or SignWriting notations to sequences of scripted animation commands: eSIGN [13], MUSSLAP [29] projects, the ILSP [17] and EMBR [21] systems, and synthetic corpus of American Sign Language (ASL) [49]. A system that supports the translation from Italian to Italian Sign Language (LIS) and generates avatar animation by combined approach is being designed [38].

### 2.3 Speech recognition

Human speech refers to the processes associated with the production and perception of sounds used in spoken language. Automatic speech recognition (ASR) is the process of converting a speech signal to a sequence of words, by means of an algorithm implemented as a software or hardware module. Several kinds of speech are identified: spelled speech (with pauses between letters or phonemes), isolated speech (with pauses between words), continuous speech (when a speaker does not make any pauses between words) and spontaneous natural speech in an inter-human dialogue. The most common classification of ASR by recognition vocabulary (lexicon) is the following:

– small vocabulary (10–1000 words);
– medium vocabulary (up to 10 000 words);

– large vocabulary (up to 100 000 words);
– very (or extra) large vocabulary (>100 000 words that is adequate for ASR for inflective or agglutinative languages);
– unlimited vocabulary (attempts to model all existing and potential words of the language).

Recent automatic speech recognizers exploit mathematical techniques such as Hidden Markov Models (HMM) [61], Artificial Neural Networks (ANN) [51], Dynamic Bayesian Networks (DBN) [53], Dynamic Time Warping (DTW) or dynamic programming [24], Support Vector Machines (SVM) [52] or some hybrid models [18, 55]. The most popular ASR models apply speaker independent speech recognition, though in some cases (for instance, personalized systems that have to recognize owner only) speaker dependent systems are more adequate, as in the case of personalized systems.

In the framework of the given project a multilingual ASR system is constructed by applying the Hidden Markov Model Toolkit (HTK version 3.4) [60]. Language models based on statistical text analysis and finite-state grammars are implemented for ASR of continuous phrases or messages [45].

### 2.4 Speech synthesis

Speech synthesis is the artificial production of human speech. A speech synthesis (text-to-speech) system translates normal orthographic text into symbolic linguistic representations like phonetic transcriptions, which are then converted into speech. Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database [7], or by using a parametric model like HMM [12, 20, 62]. Articulatory Synthesis, where a model of the vocal tract is considered to generate the speech, is another but less popular approach. Systems for speech concatenation differ in the size of the stored speech units; a system that stores allophones or diphones provides acceptable speech quality but the systems that are based on unit selection methods provide a higher level of speech intelligibility. The quality of a speech synthesizer is judged by its similarity to human voice (naturalness) and by its ability to be understood (intelligibility). Concatenation and HMM synthesis are the state-of-the-art methods. They have been developed in recent years i.e. faster unit selection [10], expressive speech synthesis [19] or application [40].

### 2.5 Properties of the considered languages

The Czech, English, Russian and Turkish languages are included in the speech scope of the system and the Czech, Turkish and Russian fingerspelling alphabets are included in the visual scope.

Turkish is an agglutinative language with relatively free word order. Due to their rich morphology Czech, Russian and Turkish are challenging languages for ASR. The out-of-vocabulary (OOV) rates for a fixed vocabulary size are significantly higher in these languages. The higher OOV rates lead to higher word error rates (WERs). Having a large number of words also contributes to high perplexity numbers, an undesired situation since it increases the average number of possibilities for each word in the model. Turkish, being an agglutinative language with a highly productive inflectional and derivational morphology is especially prone to these problems.

Recently, large vocabulary continuous speech recognition (LVCSR) systems have become available for Turkish broadcast news transcription [6]. An HTK based version of this system is also available [8]. LVCSR systems for agglutinative languages typically use sub-word units for language modeling.

The Russian language belongs to the Slavonic branch of the Indo-European group of languages, which are characterized by a tendency to combine (synthesize) a lexical morpheme (or several lexical morphemes) and one or several grammatical morphemes in one word-form. So, Russian is a synthetic inflective language with a complex mechanism of word-formation. For large vocabulary Russian ASR, it is required to apply a recognition vocabulary in several orders larger than for English or French ASR because of existence of prefixes, suffixes and endings that essentially decreases both accuracy and speed of recognition. Grammatical dictionary of Russian contains above 150 thousand words and due to word-formation rules it allows extracting all the dictionary entries and obtaining over two million various word-forms. For instance, verbs can generate up to two hundred word-forms, which have to be taken into account in speech recognition. Besides, most word-forms of the same word differ in endings only, which are pronounced in continuous speech not as clearly as the beginning parts of words. Misrecognition in endings results in misrecognition of the word and the whole sentence because of word discordance. Moreover, word order in Russian sentences is not restricted by hard grammatical constructions, like in English or German that complicates creation of statistical language models or essentially decreases their effectiveness. Statistical corpus-based language models for Russian have perplexity and entropy estimations three-four times higher with respect to English [57].

SAMPA phonetic alphabet for the Russian language includes 42 phonemes: 36 units for consonants and six for vowels, so consonants ambiguity is rather high in Russian. An automatic phonetic transcriber is required for the creation of the recognition vocabulary for Russian ASR. Rules for transformation from orthographic text to phonemic representation are not very complicated for Russian; however,

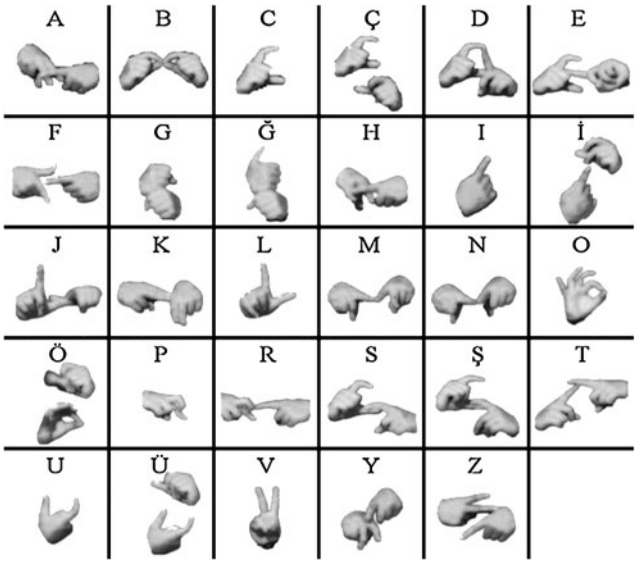



**Fig. 1** Turkish fingerspelling alphabet


**Fig. 2** Russian fingerspelling alphabet

the main problem is to find the position of stress (accent) in a word-form. There exist no common rules to determine stress positions; moreover, compound words may have several accents at once. Only knowledge-based approaches can solve this challenge.

The three different fingerspelling alphabets included in the system contain varying characteristics that make their combined recognition a challenging problem. The Turkish Fingerspelling Alphabet (TFA), seen in Fig. 1, contains seven gestures performed by one hand and twenty two gestures performed by two hands.

In contrast to the Turkish alphabet, all signs of the Russian fingerspelling alphabet (Fig. 2) are performed by one (dominant) hand. The Czech sign alphabet in Fig. 3, contains a one handed and a two handed sign for each letter. Therefore, this combination of gestures creates a need to handle the processing of different kinds of letters separately.

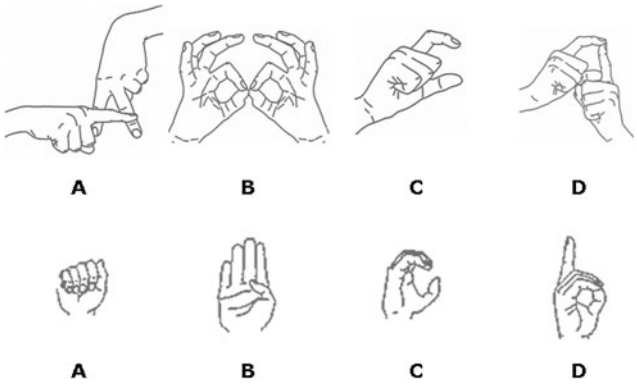



**Fig. 3** Example form Czech fingerspelling alphabet

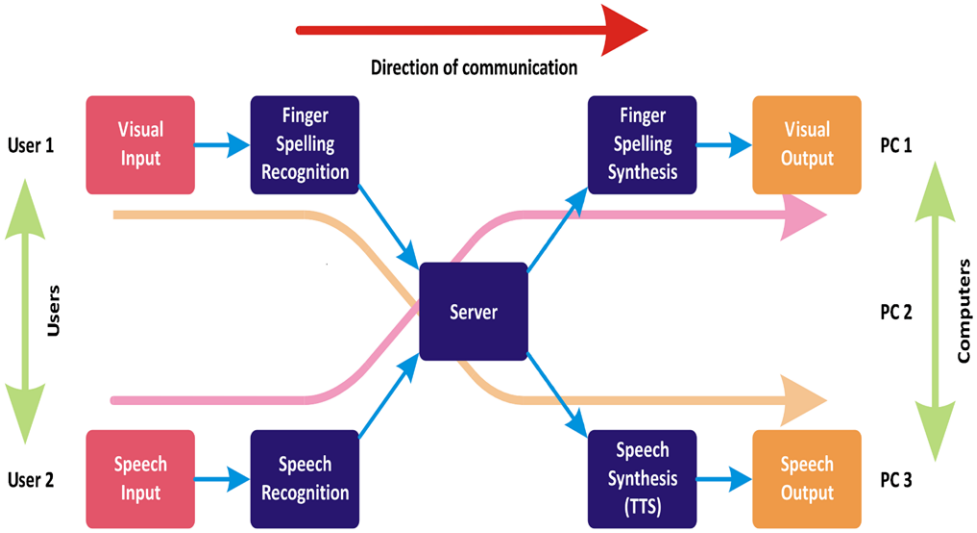
## 3 System overview

The system is implemented in a client-server architecture. Individual modules are client applications which are communicating through the server. The system is operating close to real time on an average PC. The tests were done on several machines with average specifications similar to Intel(R) Core(TM)2 Duo @ 2 GHz, 2 GB of operating memory. The system takes visual hand gesture input from the camera or the speech input from the microphone and converts it to synthesized speech or fingerspelling. The input and output can be selected among the supported languages for each module. The translation between different languages is handled via Google translate API. The system flowchart can be seen in Fig. 4.

A screenshot of the usage of hand gesture and speech modalities with the system can be seen in Fig. 5. A simple game scenario that involves the usage of speech and fingerspelling modalities is defined. The purpose of the game is for a deaf person and blind person to play a simple child's game: City games are a natural case where finger spelling needs to be used: While there are specific fingersigns for local cities, there are no specific signs for foreign cities; and therefore, their names need to be fingerspelled. We tried to make this justification in the text. The goal of the game is to give a name of a city using one of the input modalities. Then the next player has to give a name of a city beginning with the letter with which the last city ended. The game ends when one of the players cannot think of a city. In that case the other player wins. An example of a game scenario:

SP- Hi, I am Alexander, from Russia
FS- Hi, I am Alp, from Turkey
SP- Do you want to play city names game?
FS- Yes
SP- Ok, I start. London
FS- Naples
SP- St. Petersburg ...

**Fig. 4** System flowchart



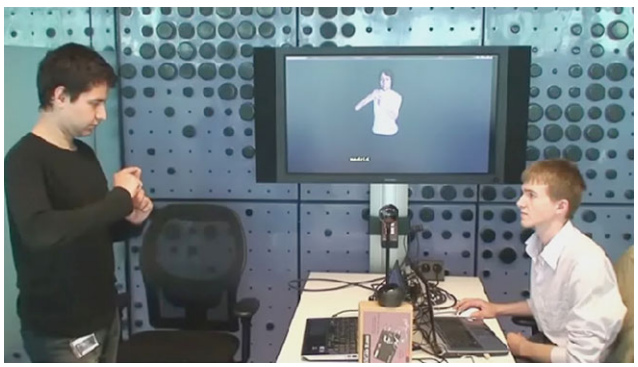



**Fig. 5** Users communicating through system with each other using fingerspelling and speech

### 3.1 Client-server architecture

Since the aim of this project is to help the communication of two people, the use of a computer network is essential to allow remote communication. As seen in Fig. 4, the whole communication system has two input parts (one for each user), the central server and two output parts (again, one for each user). The central server, located outside of the user computers, runs a stand-alone application, which communicates with the applications located in the users computers (one input and one output application for every user). Since these applications connect to the server, we can call them clients. The server has these features:

- handles multiple discussions (sessions), i.e. multiple user pairs can discuss separately on the same server and receive text messages from input clients,
- stores recent messages,
- translates messages to another languages using Google Translate API,
- sends messages to the output clients.

The message can be a single letter (e.g. received from a fingerspelling recognition client), a single word or a sentence (e.g. from a speech recognition client). The server automatically concatenates letters into words and words into sentences. The server is implemented as a web server that receives and delivers content using HTTP (Hypertext Transfer Protocol) over the Internet. The server receives requests from the clients and sends a response back.

For example, to retrieve messages by an output client: http://[server_url]/dialogue?list=all&session=tom_and_bob&format=json&tran_lang=cs_CZ lists all messages from tom_and_bob session, translates all messages into cs_CZ language and sends the response in JSON format.

Another example to send a new message by an input client: http://[server_url]/dialogue?language=en_GB&user_id=Tom&sentence=Hello+world&session=tom_and_bob.

This adds the new sentence “Hello world” in en_GB language by user Tom into tom_and_bob session.

The advantage of this client-server architecture is the possibility to have multiplatform client applications created in any programming language which supports HTTP communication.

## 4 Fingerspelling recognition

The automatic fingerspelling recognition system captures hand gestures in continuous image sequences and converts them to letters of written alphabets. Hand gestures of the users are captured from the signing environment using a single camera. The user is not required to wear any markers on his/her hands. The supported languages of the system are Turkish fingerspelling, Russian fingerspelling and Czech fingerspelling. Some additional signs such as delete and end of word are defined as well to improve system usability.

**Fig. 6** Multilingual fingerspelling database

### 4.1 Dataset collection

For training of our system and testing its accuracy, we have collected a multilingual fingerspelling database from five different users [27]. All five of these users performed Turkish fingerspelling gestures and three of them performed Czech and Russian sign alphabets. Samples from videos of different users are shown in Fig. 6.

The videos are recorded by a mini-dv camera at 25 fps at a resolution of 640 × 480. The signs are performed in front of a black background using a constant camera distance and angle. The signers wear dark colored clothes with long sleeves. Each letter is repeated five to eight times depending on the complexity of the gesture, allowing us to discard any mistakes on the users part. The total length of the database for each signer changes between 30 to 45 minutes. Due to different recording environments the lighting conditions and camera calibration settings vary slightly from subject to subject.

### 4.2 Hand tracking and segmentation

The highly mobile and self occluding nature of the hands makes tracking of hand gestures a challenging task. While signing in a natural manner, hands often tend to interact with each other, cross over the face and make movements that are sudden and rapid. In order to handle many of the exceptional cases without excessive computational burden, we used a tracking algorithm based on the Continuously Adaptive Meanshift (Camshift) method [4] to jointly track the hands and face of a signer.

Our tracking method replaces the manual initiation of Camshift by implementing a motion based skin color model initialization module. The module requires the user to wave his hands for a few seconds before commencing signing to generate a person specific color histogram. The location of the hand is localized by calculating image differences between consecutive frames. By using double differencing, the location of the hand is obtained, which is used to train a skin color model to track hands. This skin color model assumes certain user and environment conditions and needs to be reinitialized for each new user.

During tracking, we handle issues caused by simultaneous two hand tracking and tracking failures with a hierarchical hand re-detection module [14]. With this module, we can robustly detect new foreground objects to track in our image. Detection of new objects to track allows us to handle oc-

cluding, merging, disappearing and reappearing hands, thus allowing continuous tracking of the hands and the face. By marking the merging and separation of tracking boxes, we keep track of the number of hands in the search boxes. The tracking is performed on a reduced resolution video of size $320 \times 240$ to boost system performance. The final system performs between 8 to 10 fps.

A limitation of the skin color based hand recognition method is that the subject should not allow any other skin colored objects on the scene. He/she should wear long sleeves and make as minimal interactions with his/her face as possible. Although this seems like a limitation, this is clearly a very minor limitation when similar systems are considered: Such systems usually require studio set-up with controlled illumination. Some systems require the user to wear special gloves. Our system is adaptable; as it performs adaptation to the illumination conditions and the skin color at the start-up. As a result, the users need not wear special markers or gloves.

### 4.3 Feature extraction

On the segmented hand images, we used the following shape descriptors as features to mathematically represent the hand images. We chose our appearance based features based on their strengths on representing different properties of hand shapes, such as shape, texture and external contours. In this study, we used shape based features such as image moments, texture representation based features such as Local Binary Patterns and external contour based features such as Elliptic Fourier Descriptors and Radial Distance function. Since fingerspelling gestures limit the use of arm and body gestures, we refrained from using any motion based features.

#### *4.3.1 Local binary patterns*

Local Binary Patterns (LBP) were introduced by Ojala [43] for texture representation. LBP is used across various computer vision fields (e.g. image synthesis, light normalization, face detection, face/expression recognition). It has been successfully used for hand detection in cluttered images [42]. We use LBP for hand shape description.

First an LBP image is computed. The algorithm moves a defined patch along all the pixels in an image. The evaluated pixel is in the center of the patch. Depending on the size and shape of the patch the resulting LBP image changes. We use a circular 8-neighborhood patch with radius one and two pixels. The patches can be seen in Fig. 7. If the brightness of a pixel in the center is greater or equal to the evaluated pixel's brightness we assign the binary label 1 to the proper location in the patch. If the patch brightness is lower than the evaluated brightness we assign the label 0 to it. In each patch we evaluate eight locations (or combinations of locations)

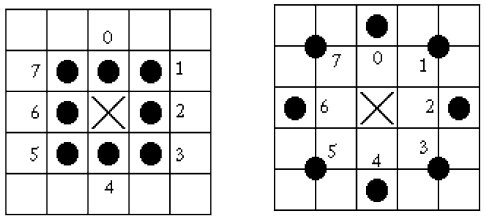



**Fig. 7** Examples of LBPs. We use LBP with radius of one (*left*) and two (*right*). Numerical values represent the position of the patch in the binary representation of the pattern

which yields an 8-bit number. This number is assigned to the location of the evaluated pixel and the patch moves to the next pixel. Next, we compute a histogram of the LBP image, which we use as a feature vector. For normal LBPs there are 256 histogram bins, each bin for one pattern. In practice it has been shown that all the patterns are not important for recognition. Most of the information is in the patterns that have two or fewer changes between 0 and 1 in its binary representation. Such LBPs are called uniform [44]. There exist 58 such patterns and all the other patterns are moved to the 59th bin of the histogram. This means that for the uniform LBPs that we use in our system, the feature vector is of size 59.

We implemented both uniform and non-uniform LBP both with the patch radius one and two. We used uniform LBPs with patch radius size two to test our system.

#### *4.3.2 Elliptic Fourier descriptors*

Elliptic Fourier descriptors on shape signatures are widely used for shape analysis and recognition [33, 35]. These descriptors represent the shape of the object in the frequency domain. The lower frequency descriptors contain information about the general features of the shape, and the higher frequency descriptors contain information about finer details of the shape. Although the number of coefficients generated from the transform is usually large, a subset of the coefficients is enough to capture the overall features of the shape. For an n-harmonic elliptic Fourier descriptor representation of a 2-D closed shape is given in Equation 1. In the equation, the center of the curve is $(a_0, c_0)$. $(a_k, b_k, c_k, d_k)$ for $k = 1 \ldots n$ are elliptic Fourier coefficients of the curve up to $n$ Fourier harmonics. $T$ is the perimeter of the closed curve.

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} a_0 \\ c_0 \end{bmatrix} + \sum_{k=1}^{N} \begin{bmatrix} a_k \cos(\frac{2k\pi t}{T}) + b_k \sin(\frac{2k\pi t}{T}) \\ c_k \cos(\frac{2k\pi t}{T}) + d_k \sin(\frac{2k\pi t}{T}) \end{bmatrix} \quad (1)$$

The elliptic Fourier descriptors $(a_k, b_k, c_k, d_k)$ are calculated for each point $k$ of the curve as seen in (2)–(5).

$$a_k = \frac{1}{2k^2\pi^2} \sum_{i=1}^{n} \frac{\Delta x_i}{\Delta t_i} \left[ \cos\left(\frac{2k\pi t_i}{T}\right) - \cos\left(\frac{2k\pi t_{i-1}}{T}\right) \right] \quad (2)$$

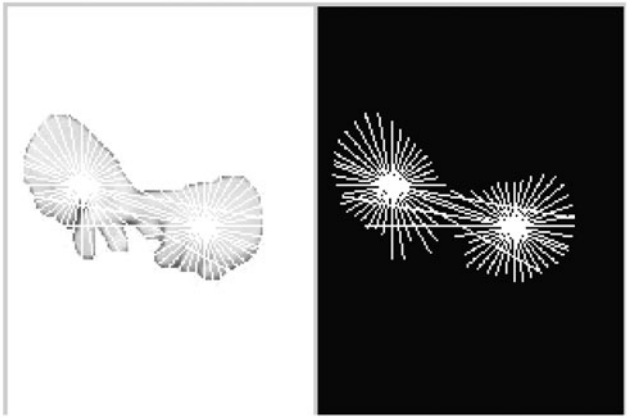


**Fig. 8** Radial distances for a two-handed gesture

$$b_k = \frac{1}{2k^2\pi^2}\sum_{i=1}^{n}\frac{\Delta x_i}{\Delta t_i}\left[\sin\left(\frac{2k\pi t_i}{T}\right) - \sin\left(\frac{2k\pi t_{i-1}}{T}\right)\right] \quad (3)$$

$$c_k = \frac{1}{2k^2\pi^2}\sum_{i=1}^{n}\frac{\Delta y_i}{\Delta t_i}\left[\cos\left(\frac{2k\pi t_i}{T}\right) - \cos\left(\frac{2k\pi t_{i-1}}{T}\right)\right] \quad (4)$$

$$d_k = \frac{1}{2k^2\pi^2}\sum_{i=1}^{n}\frac{\Delta y_i}{\Delta t_i}\left[\sin\left(\frac{2k\pi t_i}{T}\right) - \sin\left(\frac{2k\pi t_{i-1}}{T}\right)\right] \quad (5)$$

For each calculated Fourier harmonic, we obtain an ellipse that yields four invariant features [54]. The first two features are the major and minor axis lengths of the calculated ellipses. The latter two features can be derived from the first two features and vice versa. In our experiments with hand contours, we found 10 harmonics sufficient to represent hand gestures, yielding feature vectors of size 40. The formulas for the features are given in (6)–(9).

$$A_k^2 = \frac{I_k + \sqrt{I_k^2 - 4J_k^2}}{2} \quad (6)$$

$$B_k^2 = \frac{J_k^2}{A_k^2} \quad (7)$$

$$I_k = a_k^2 + b_k^2 + c_k^2 + d_k^2 \quad (8)$$

$$J_k = (a_k d_k) - (b_k c_k) \quad (9)$$

#### 4.3.3 Radial distance function

The radial distance function method, presented in [59] is a contour based method. Using the distance of a seed-point (possibly the center of mass) in all directions to the closest background pixel (Fig. 8), we compute a feature vector of the image. The calculated image descriptors are invariant to translation, size and rotation. Rotation invariance is achieved by choosing the angle with the smallest radial distance as the point for each seed point.

While describing an isolated hand, the radial distance function is an efficient measure as it is possible to represent finger locations and notable extensions of the hand. However, when describing blobs consisting of not completely overlapping, but touching hands, obtaining a point which has a straight line distance to both hands may not be possible. For this reason, we attempt to find the centers of gravity belonging to both hands. By using the image moments $M_{ij}$, we calculate the parameters of the smallest ellipse that covers both hands using the formulas (10)–(13).

$$a = \frac{M_{20}}{M_{00}} - x^2 \quad b = 2\frac{M_{11}}{M_{00}} - x_c y_c \quad c = \frac{M_{02}}{M_{00}} - y^2 \quad (10)$$

$$l_1 = \frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2} \quad (11)$$

$$l_2 = \frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2} \quad (12)$$

$$\Theta = \frac{1}{2}\tan^{-1}\left(\frac{b}{a-c}\right) \quad (13)$$

In the calculations, $l_1$ and $l_2$ are the principal axes of the bounding ellipse centered on the centroid of the image and $\Theta$ is their rotation angle. By dividing the image using the minor axis $l_2$ as a separator, we effectively divide the blob into two approximately equal parts belonging to different hands. After calculating the centroid of each part using image moments, we obtain seed locations for two radial distance functions that are sufficient to describe a hand blob consisting of two hands, as seen in Fig. 8. We chose to calculate two radial distance functions every five $(2\pi/72)$ degrees and obtain a feature vector of size 144.

#### 4.3.4 Hu moments

We make use of Hu moments as shape descriptors and compare the performance with our other features. The invariant moments of Hu are calculated from normalized central image moments using the formulas below [23]:

$$I_1 = \eta_{20} + \eta_{02}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + (2\eta_{11})^2$$

$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

**Table 1** Effects of dimensionality reduction on Turkish fingerspelling recognition using kNN

| | HU | EFD | RDF | LBP |
|---|---|---|---|---|
| **NO PCA** | 0.4924 | 0.6465 | 0.7606 | **0.7639** |
| **PCA %95** | 0.4633 | **0.6964** | 0.3896 | 0.2343 |
| **PCA %99** | 0.4924 | **0.7704** | 0.6643 | 0.5649 |
| **PCA %100** | 0.4924 | 0.7629 | 0.7609 | **0.7636** |

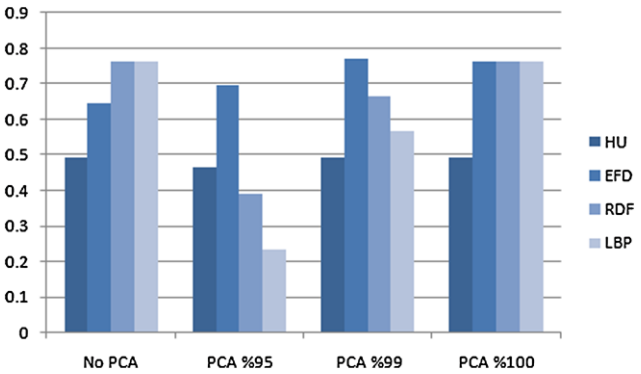



**Fig. 9** Effects of PCA on fingerspelling recognition accuracy

$$I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Calculated from binary shape masks, we use the seven Hu moments as rotation, scale and translation invariant feature vectors.

#### *4.3.5 Feature selection and dimensionality reduction*

In Table 1, we display results on the effects of dimensionality reduction on key-frame based recognition accuracy. For dimensionality reduction, we make use of the principal component analysis algorithm. In these tests, the kNN algorithm is used to classify the features belonging to the key-frames of the Turkish Fingerspelling alphabet. Since these tests are performed on individual key-frames rather than image sequences, recognition accuracy is lower than our sequence-wise recognition results. We have chosen to preserve energy at different levels by choosing different number of coefficients whose eigenvalues sum up to a certain percentage of the total energy. Figure 9 shows that except for Elliptic Fourier Descriptors, no feature shows performance improvement when projected onto a lower dimensional space.

We explicitly feel the need to note here that each EFD coefficient is of a different magnitude giving higher weight to higher harmonic features. Performing z- normalization on the vector actually reduces overall descriptor accuracy by negating high level shape characteristics. Thus, the success of PCA algorithm to converge on features with higher harmonics increases the descriptors overall efficiency.

**Fig. 10** Motion blur example

### 4.4 Recognition and results

For modeling and recognizing our gestures, we make use of a two layered classification method. First, we use a motion model to decide where a certain gesture starts and ends in a continuous image sequence. Then we combine the classification results of frame-wise classifiers to make a decision for the sequence of images belonging to a gesture.

#### *4.4.1 Motion modeling and key frame selection*

In continuous hand gesture recognition, an important issue is designating frames in which the signer actually performs a gesture and frames where the signer does not perform it. We use a key-frame based motion model, where consecutive key-frame sequences of certain lengths are assumed to belong to the same gesture. Given a sequence of images, we classify the images into two categories depending on their semantic content. If the frame contains a snapshot of a hand gesture which can be used to represent a concept, we call the frame a key-frame. If the frame does not contain any hand gestures or is a transition between two gestures, we call the frame a transition frame.

In key-frame selection, the goal is to distinguish probable sign language hand gestures from transition gestures that occur while moving the hands from one gesture to another. Although the motion of hands seems to be the key distinguishing feature for such a case, individual usage of such feature does not produce extremely accurate results. It is mainly because the motion of hands varies with the signer, with the accuracy of hand tracking, with the current sign and with the co-articulation of previous and following sign. Therefore, supplementing hand motion with a feature that is less dependent on the above conditions and more dependent on the quality of captured images becomes useful in the selection of meaningful and accurate key-frames. For that reason, we use motion blur in addition to hand motion to automatically model and separate key-frames from transition frames.

In images containing hand gestures, we search for the presence of blur around the external contours of the hand. Compared to unblurred images, a major characteristic of

**Fig. 11** Trace image

blurred images is that edges tend to be smoother and contain smaller gradient values (Fig. 10). Therefore, focusing on the distribution of gradient values in a certain image patch can give us an idea about the presence of partial motion blur. Using the image derivative masks,

$$D_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad D_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{14}$$

we convolve the images to obtain the derivative images $I_x$ and $I_y$. Using a Gaussian window, we obtain the smoothed squared image derivatives $I_x^2$ and $I_y^2$. Then we calculate the trace of the image (see (15)) in the same manner that is used to calculate the image trace for the Harris corner detector (Fig. 11).

$$A = k(I_x^2 + I_y^2)^2 \tag{15}$$

The trace image yields the most significant results in the edge areas that separate the hands from the background. We compute the trace image for small windows around the hand contour. We capture $n \times n$ sized small images around the hand contours that are at least $n^2/4$ pixels apart from each other ($n = 7$). Since we are looking for motion blur which is nonexistent in the direction of motion, the magnitude of the difference of the maximum and minimum gradient strength values of the same image can be used to hint an increase or decrease in the amount of motion blur [36].

To model hand gestures in continuous videos, we make use of key-frame information. For the classification of movement and blur features, we use two methods, one gaussian and one heuristic based. In the gaussian method, we train the system with positive and negative key-frame examples. However, in the online system, the heuristic method where the user can manipulate the thresholds to adjust system performance to his speed was deemed more usable, as it was easier to calibrate for different users.

To clearly analyze the effect of automatic keyframe recognition on fingerspelling recognition, we performed isolated fingerspelling recognition experiments on manual and automatically selected hand gesture keyframes. In the first group of images, snapshots of each sign language video were extracted from fingerspelling videos. Using a fusion of hand displacement and partial motion blur detection methods with pre-determined thresholds, each frame was classified as a keyframe or a transition frame. As these keyframets are manually extracted from image sequences to represent gestures, they are not affected by possible propagating image segmentation or temporal segmentation errors. Therefore, the tests with annotated snapshot images provide us information on feature descriptor and classifier performances free of segmentation and keyframing errors. In Table 2 we compare the recognition accuracies for the Turkish fingerspelling alphabet using both manually selected and automatically extracted keyframes. While comparing the results in Table 2 one must take into account that isolated snapshots contain no temporal ordering information. Therefore, transition frames that are selected and reduce recognition accuracy in automatic keyframe selection should not be considered useless, as they are simply left out in this kind of modeling.

**Table 2** Turkish fingerspelling recognition accuracy with different keyframe selection using kNN

| | HU | EFD | RDF | LBP | Fusion |
|---|---|---|---|---|---|
| **Manual** | 0.5416 | 0.7041 | 0.7283 | 0.8662 | **0.8814** |
| **Auto** | 0.4924 | 0.6465 | 0.7606 | 0.7639 | **0.7793** |

In our motion model, when processing hand images in a continuous video, the system marks the first encountered keyframe as the start of a gesture. The consecutive frames following this start frame are all accepted as a part of this gesture until a certain number of consecutive transition frames arrives. Two adaptive thresholds are used to distinguish the start and the end of a gesture sequences. The minimum sequence length indicates the minimum number of keyframes to be considered a gesture. This is useful for separating the background noise and the movement of dynamic keyframes from larger gesture sequences. Likewise the minimum cutoff length threshold separates movements between two separate frames from the movements of dynamic keyframes. Figure 12 displays an overview of the two layered classification model that combines different classifiers using this motion model.

#### *4.4.2 Hand gesture recognition*

In hand gesture classification, hand gesture features of different types are handled individually and are classified using the K-nearest neighbor algorithm with 10-fold cross validation. For each recognized key-frame, a decision is obtained by fusing the classification results from Hu moments (Hu), Elliptic Fourier Descriptors (EFD), Radial Distance Functions (RDF) and Local Binary Patterns (LBP). The results from the underlying features are subjected to score level fusion using weighted voting to obtain a frame-wise decision. As gestures belonging to successive key frames are classified, the results belonging to a hand gesture sequence are gathered using the motion model described in Sect. 4.4.1.

**Fig. 12** Classification pipeline

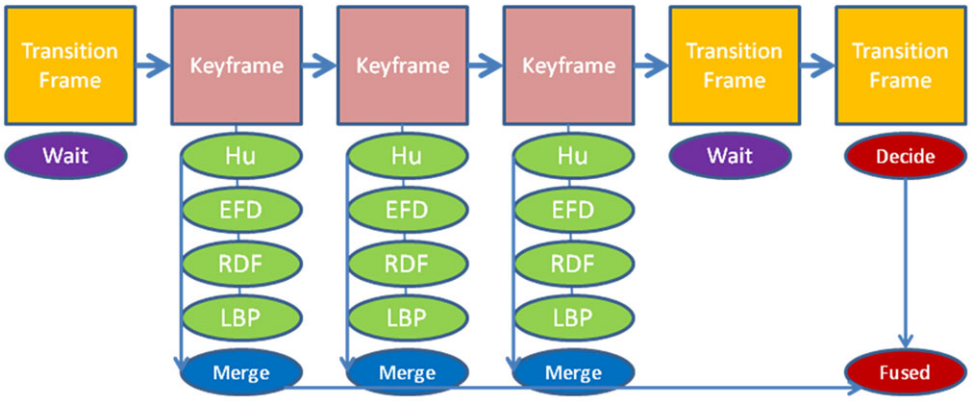



**Table 3** Multilingual signer dependent recognition accuracy

| | Language | HU | EFD | RDF | LBP | Fused |
|---|---|---|---|---|---|---|
| Subject1 | **Turkish** | 0.53 | 0.63 | 0.67 | 0.85 | **0.88** |
| Subject2 | **Turkish** | 0.56 | 0.70 | 0.78 | 0.91 | **0.93** |
| Subject3 | **Turkish** | 0.54 | 0.75 | 0.77 | **0.87** | **0.87** |
| Subject4 | **Turkish** | 0.48 | 0.61 | 0.61 | 0.74 | **0.78** |
| Subject5 | **Turkish** | 0.59 | 0.79 | 0.76 | 0.94 | **0.96** |
| Average | **Turkish** | 0.54 | 0.70 | 0.72 | 0.86 | **0.88** |
| Subject1 | **Russian** | 0.64 | 0.61 | 0.72 | 0.69 | **0.75** |
| Subject2 | **Russian** | 0.38 | 0.54 | 0.75 | **0.8** | **0.8** |
| Subject3 | **Russian** | 0.63 | 0.61 | 0.64 | 0.69 | **0.73** |
| Average | **Russian** | 0.55 | 0.59 | 0.7 | 0.73 | **0.76** |
| Subject1 | **Czech** | 0.60 | 0.52 | 0.62 | **0.75** | 0.72 |
| Subject2 | **Czech** | 0.48 | 0.61 | 0.78 | 0.76 | **0.84** |
| Subject3 | **Czech** | 0.47 | 0.65 | 0.67 | 0.73 | **0.78** |
| Average | **Czech** | 0.52 | 0.59 | 0.69 | 0.75 | **0.78** |

**Table 4** Multilingual signer independent recognition accuracy

| | Language | HU | EFD | RDF | LBP | Fused |
|---|---|---|---|---|---|---|
| Subject1 | **Turkish** | 0.31 | 0.15 | 0.39 | 0.28 | **0.43** |
| Subject2 | **Turkish** | 0.23 | 0.06 | 0.24 | 0.21 | **0.28** |
| Subject3 | **Turkish** | 0.36 | 0.30 | **0.48** | 0.29 | 0.45 |
| Subject4 | **Turkish** | 0.07 | 0.28 | 0.61 | 0.48 | **0.54** |
| Subject5 | **Turkish** | 0.32 | 0.20 | **0.45** | 0.23 | 0.40 |
| Average | **Turkish** | 0.26 | 0.20 | **0.43** | 0.30 | 0.42 |
| Subject1 | **Russian** | 0.36 | 0.16 | 0.44 | 0.44 | **0.47** |
| Subject2 | **Russian** | 0.36 | 0.25 | 0.39 | **0.42** | 0.4 |
| Subject3 | **Russian** | 0.27 | 0.28 | **0.43** | 0.31 | 0.33 |
| Average | **Russian** | 0.33 | 0.23 | **0.42** | 0.39 | 0.4 |
| Subject1 | **Czech** | 0.39 | 0.34 | 0.46 | 0.41 | **0.48** |
| Subject2 | **Czech** | 0.27 | 0.18 | **0.49** | 0.44 | 0.46 |
| Subject3 | **Czech** | 0.22 | 0.18 | 0.37 | 0.37 | **0.38** |
| Average | **Czech** | 0.29 | 0.23 | **0.44** | 0.4 | **0.44** |

The gathered classification results of different key frames are then fused together using majority voting (Fig. 12).

For offline testing, we have used the fingerspelling videos for three fingerspelling alphabets (see Sect. 4.1). Images belonging to each subject in the dataset have been divided into equal sized training and test sets. The recognition data for each subject's test videos were tested in two settings; signer dependent setting, where the system is trained using the training videos of the test subject and signer independent setting, where the training videos of the test subject are excluded from the dataset. The results of signer dependent classification are shown in Table 3 and the results of signer independent classification are presented in Table 4.

As it can be inferred from the average results, having the user's own data in the recognition set doubles the accuracy of recognition from 42% to 88%. The main reasons of this difference can be inferred as the variance in illumination and the minor differences in the performances of the signers.

However, since the overall aim of the system is to provide an online word level recognition system, such accuracy rates on their own were deemed insufficient. For that reason we opted to implement a vocabulary list of 2000 words and match the signed letters to the closest word using Levenshtein distance [34]. We compared the sequence of recognized letters when the user returns to rest position (hands separated and lowered to opposing sides of the body), to each word in the word list using a normalized Levenshtein distance to increase word level recognition accuracy.

## 5 Speech recognition

We have implemented two different speech recognition modules. First one is continuous speech recognition (implemented for Turkish) and the other is spelled speech recognition (implemented both for Russian and English). The former module has been substantially realized before the eNTERFACE'10, and adopted for the developed application. The latter component for spelled speech was principally de-

veloped during the workshop based on the recent SIRIUS Russian ASR system [46].

### 5.1 Continuous speech recognition

Automatic speech recognition (ASR) is needed in the first stage for the one-way communication from a speaking person to a hearing impaired person by converting spoken words to text that gets synthesized to sign language in later stages. We integrated a Weighted Finite-State Transducer (WFST) based large-vocabulary continuous speech recognition system developed at Bogazici University into this multimodal communication platform [6, 48]. The integrated system is currently capable of recognizing just Turkish utterances since language and acoustic models were readily available only for Turkish.

The speech recognition problem is treated as a transduction from input speech signal to a word sequence in the WFST framework [41]. The WFSTs provide a unified framework for representing different knowledge sources in ASR systems. A typical set of knowledge sources consists of a transducer $H$ modeling context-dependent phones as hidden Markov models (HMMs), a context-dependency network $C$ transducing context-dependent phones to context-independent phones, a lexicon $L$ mapping context-independent phone sequences to words, and an n-gram language model $G$ assigning probabilities to word sequences. The composition of these models $H \circ C \circ L \circ G$ results in an all-in-one search network that directly maps HMM state sequences to weighted word sequences, where weights can be combinations of pronunciation and language model probabilities. The WFST also offers finite-state operations such as composition, determinization and minimization to combine all these knowledge sources into an optimized all-in-one search network.

The morphology as another knowledge source can be represented as a WFST and can be integrated into the WFST framework of an ASR system. The lexical transducer of the morphological parser maps the letter sequences to lexical morphemes annotated with morphological features [47]. The lexical transducer can be considered as a computational dynamic lexicon in ASR in contrast to a static lexicon. The computational lexicon has some advantages over a fixed-size word lexicon. It can generate many more words using a relatively smaller number of root words (55 278) in its lexicon. So it achieves lower OOV rates. In the WFST framework, the lexical transducer of the morphological parser can be considered as a computational lexicon $M$ replacing the static lexicon $L$. Since $M$ outputs lexical morphemes, the language model $G$ should be estimated over these lexical units. Then with the morphology integrated, the search network can be built as $H \circ C \circ M \circ G_{\text{morpheme}}$. The decoding for the best path in the resulting network is a single-pass Viterbi search.

Note that the word-based models output word sequences such as "merhaba saat on uc haberleri ajanstan aliyorsunuz", while the morphology-integrated model outputs lexical morpheme sequences such as; merhaba[Noun] saat[Noun] on[Adj] üç[Adj] haber[Noun] + lAr[A3pl] + SH[P3sg] ajans[Noun] + DAn[Abl] al[Verb] + Hyor[Prog1] + sHnHz[A2pl]. Therefore, we use the morphological parser as a word generator to convert the recognition output to words.

We evaluated the performance of the speech recognition system on a Turkish broadcast news transcription task. The acoustic model uses hidden Markov models (HMMs) trained on 188 hours of broadcast news speech data [6, 8]. In the acoustic model, there are 10 843 triphone HMM states and 11 Gaussians per state with the exception of the 23 Gaussians for the silence HMM. The test set contains 3.1 hours of speech data that has been pre-segmented into short utterances (2 410 utterances and 23 038 words). We used the geometric duration modeling in the decoder.

The text corpora that we used for estimating the parameters of statistical language models are composed of 182.3 million-words BOUN NewsCor corpus collected from news portals in Turkish [47] and 1.3 million-words text corpus (BN Corpus) obtained from the transcriptions of the Turkish Broadcast News speech database [6].

As a baseline word language model, we built 200K vocabulary 3-gram language model. Our previous study showed that higher vocabulary sizes than 200K and higher $n$-gram orders did not improve the accuracy significantly [6]. The OOV rate for 200K word vocabulary is about 2% on the test set. For the morphology-integrated model, the optimal n-gram order of the language model over lexical morphemes was chosen as four. The OOV rate of the morphological parser is 0.68% on the test set.

Figure 13 shows the word error rate versus run-time factor for 200K vocabulary word model Word-200K and the morphology-integrated model MP. The improvement in OOV rate for the morphology-integrated model translates to WER reductions.

For this communication platform, we implemented a voice activity detection (VAD) system to prevent false triggers and improve recognition accuracy. A binary supervised classification methodology has been adopted for this purpose. The *speech* class is trained with improvised talk and readings on a silent background and the *nonspeech* class contains silence, noise and some other noisy activities (cough, tapping on the microphone, mouse clicks, etc.). We use 13 dimensional MFCC vectors as features and GMMs with 16 components for training. Testing is done online and the decision is given by the likelihood ratio test.

### 5.2 Spelled speech recognition

Spelled speech input (letter-by-letter input) is widely used by humans for rare and out-of-vocabulary words (for in-

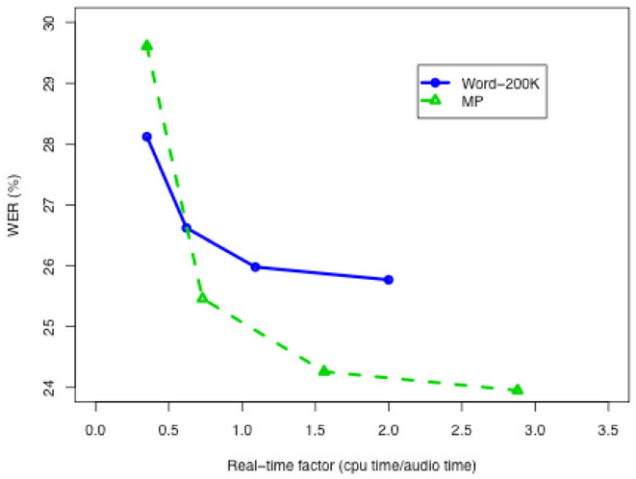



**Fig. 13** Word error rate versus real-time factor obtained by changing the pruning beam width from 9 to 12

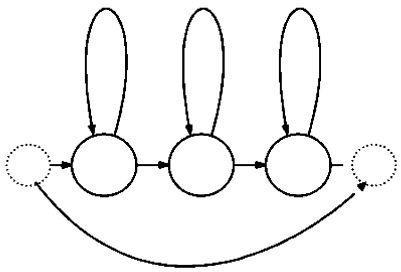


**Fig. 14** Topology of HMM-based acoustical model for a phoneme

stance, personal names, city names, e-mail addresses, etc). A speaker-dependent automatic speech recognition (ASR) system has been implemented and embedded into the communication platform.

A single stationary microphone located at 30–40 cm away from the speaker's mouth is used for speech input. As acoustic features we used 13-dimensional Mel-Frequency Cepstral Coeffs. (MFCC), including the 0-th coeff., with the first and second derivatives calculated from 26 channel filter bank analysis of 20 ms long frames with 10 ms overlap. Thus, the frequency of audio feature vectors is 100 Hz. Cepstral Mean Subtraction is applied to audio feature vectors. Acoustic modeling and recognition of phonemes and words of the recognition vocabulary are based on Hidden Markov Models (HMM). The acoustical models are realized as HMMs of the context-independent phones with mixture Gaussian probability density functions (GMMs). HMMs of phones have three meaningful states (and two additional states intended for concatenation of the phones in the letter models), see Fig. 14.

Figure 15 shows an example of a complex HMM-based model for the isolated word "SEVEN". One can see that there are five phones in this word and the second /e/ may sometimes disappear in pronunciations of some people.

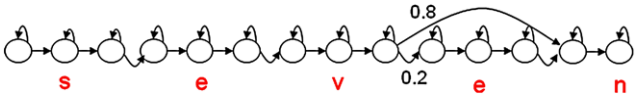



**Fig. 15** Topology of HMM for the isolated word SEVEN

The developed ASR system is multilingual and able to recognize letters/graphemes pronounced both in English and Russian. The lexicon of ASR contains 26 English letters, plus 31 Russian letters (there exist 33 letters in the Russian language in total, but we recognize 31 of them only, because two graphemes (the soft sign and the hard sign) do not have own phonetic representations, but affect on the previous letter(s) pronunciation in the spoken language [26]), plus all the digits for both languages looped in the null-gram model. Moreover, two system commands were additionally introduced into the system: "DEL" (backspace) used in order to delete the latest pronounced but misrecognized letter/word, and "DOT" (point) needed to indicate on the completion of the sentence input. The English and Russian vocabularies use a mutual pool of trained phone models. A pause (silence) between two letter inputs that lasts more than five seconds means space symbol (break of two words). A general architecture of the spelled speech recognition system is shown in Fig. 16, one can notice that there are two main work modes: model training and speech decoding.

The stage of system training includes the following steps:

- transcribing items of the lexicon of an applied domain;
- creation of a stochastic language model or grammar for possible phrases;
- coding the speech data (feature extraction);
- definition of topology of HMM (prototyping);
- creation of initial HMMs for all the monophones by the flat-start method;
- re-estimation of HMMs parameters using a speech corpus and the Baum-Welch algorithm;
- creation of context-dependent phones (triphones) from monophones;
- mixture splitting and parameters re-estimation.

In order to train the speech recognizer a speech corpus was recorded in office conditions using one distant-talking directed microphone. Totally, we have recorded about 20 minutes of speech data from one speaker, these data were labeled semi-automatically in the terms of phonemes.

The spelled speech decoder (Fig. 17) uses the Viterbi algorithm [45]. An input phrase syntax is described in a simple grammar loop that allows recognizing one vocabulary item in a hypothesis. The audio speech recognizer operates very fast (less than 0.1xRT) so results of speech recognition (in the form of N-best list of recognition hypotheses) are available almost immediately after detection of speech end by the energy-based voice activity detector (VAD).

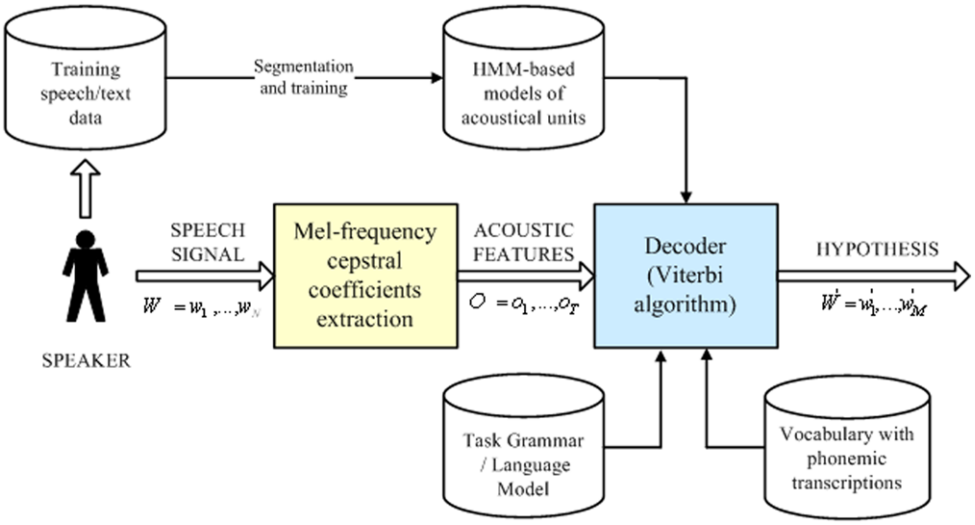



**Fig. 16** Architecture of the spelled speech recognition system

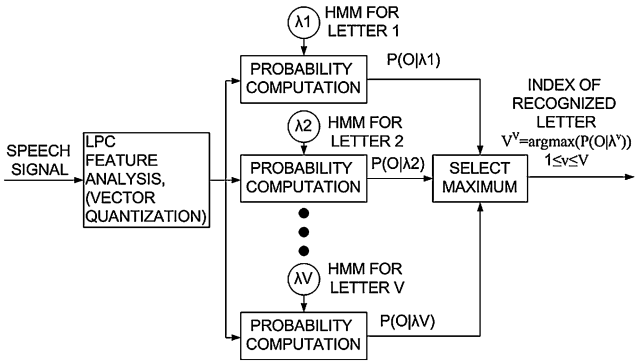



**Fig. 17** A general algorithm for spelled speech decoding

Performance of the spelled speech recognition system has been evaluated using a portion of other speech data in two languages, collected in the same office conditions as the training part. Training and testing databases for the automatic speech recognition system were recorded in one session and each letter of both languages was repeated 20 times for the training purpose and 10 more times for the system evaluation and testing. Figures 18 and 19 show accuracy of speech recognition (in the form of confusion matrices, where the first columns are reference letters) for the English and Russian letters, correspondingly. In these confusion matrices, the sign "+" in cells denotes 100% recognition rate (10 instances of 10) and empty cells mean 0% (0 instances recognized of 10 pronounced). A color coding describes these confusion matrices as well: dark green cells mean 100% recognition rate and, on the contrary, absolutely white cells mean 0%. The most of the pronounced letters in the test data were recognized very well; however, some letters (for example, English consonants B /b'i/ and D /d'i/ or vowels A /ei/ and I /ai/) are rather confusing. The recognition accuracy rate for all the English letters was 93.1% on the average, and 90%—for the Russian letters. English spelled speech is recognized a bit better than Russian, because of the smaller alphabet, moreover many Russian vowels are represented phonetically by corresponding monophones (for instance, /a/,/e/,/o/,/i/,/u/,/1/—in notation of the SAMPA phonetic alphabet) in contrast to the English vowels, which are longer and usually represented by allophones or diphthongs. Moreover, consonant ambiguity is somewhat higher in Russian.

## 6 Fingerspelling synthesis

### 6.1 Animation model

Fingerspelling synthesis system creates 3D animation of the upper half of a human figure. The baseline system incorporates 3D articulatory model approximating skin surface by polygonal meshes, see Fig. 20 on left. The meshes are divided into body segments describing arms, forearms, palm, knuckle-bones, face, inner mouth, etc. The animation model allows expressing both manual and non-manual components of sign languages. The manual component is fully expressed by rotations of the body segments. The body segments are connected by joints and hierarchically composed into a tree structure. Every joint is attached to at least one body segment. Thus the rotation of one body segment causes rotations of other body segments in lower hierarchy. Joint connection incorporates rotation limits to prevent non-anatomic poses of the animation model.

Synthesis of the non-manual component employs the joint connections as well as moving of control points and morph targets [28]. The joint connections ensure movements of shoulders, neck, skull, eyeballs (eye gaze) and jaw. In contrast, the control points and the morph targets allow us to change the local shape of polygonal meshes describing

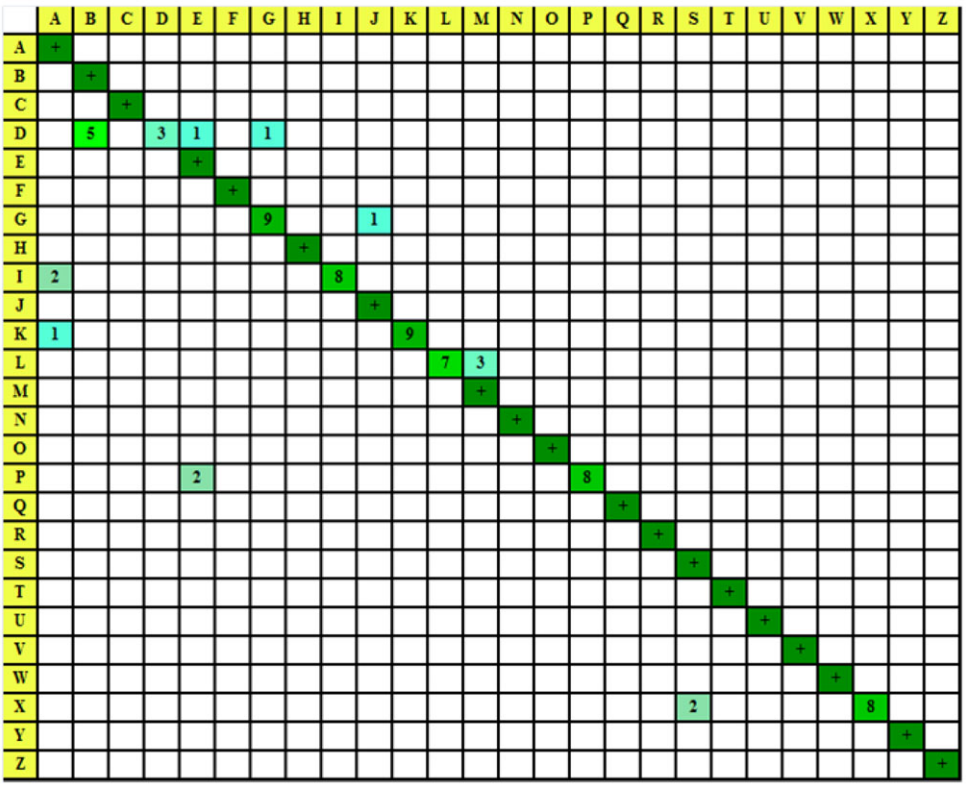



**Fig. 18** Confusion matrix for English letters recognition

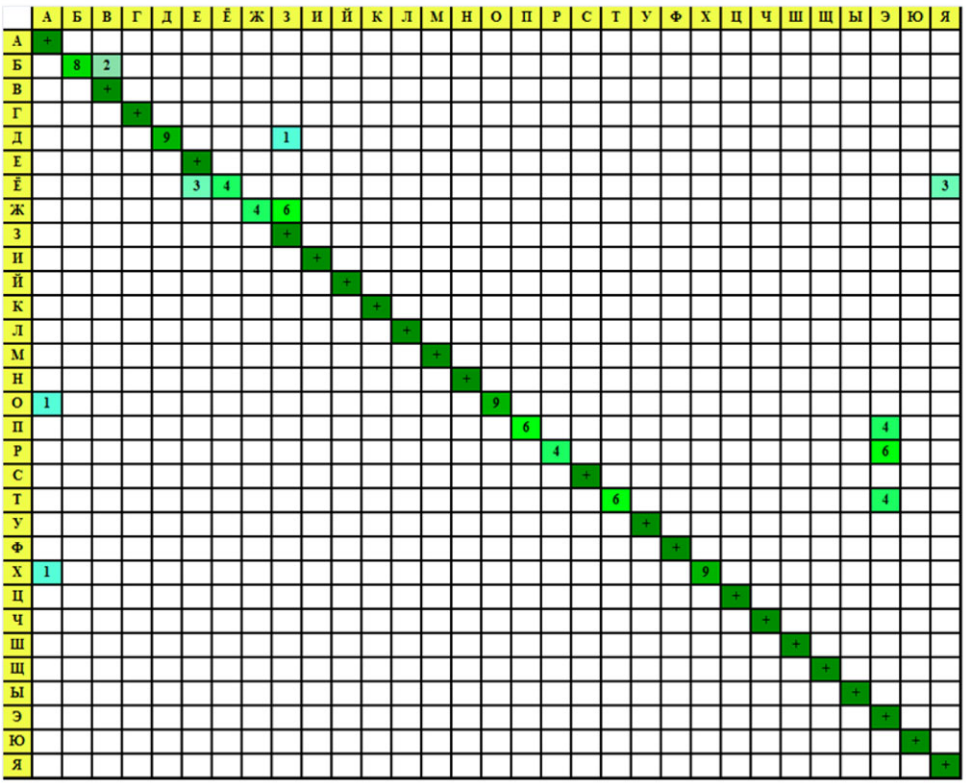



**Fig. 19** Confusion matrix for Russian letters recognition

the face, lips, or tongue. The control points have fixed positions in the particular polygonal meshes and their translation in 3D causes local deformations in face and tongue [30] and the morph targets are manually remodeled 3D shape of polygonal meshes. A complex non-manual gesture simultaneously incorporates a weighted combination of several

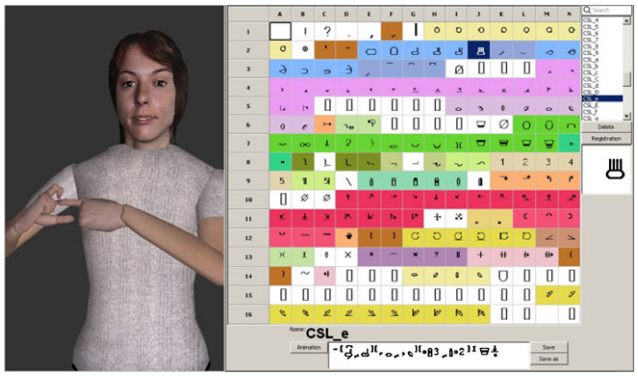


**Fig. 20** The screen shot of SignEditor. On *left* the avatar, on *right* table of symbols separated to groups by colors

morph targets, local deformation via control points and rotation of joint connections.

### 6.2 Control model

We consider the data-driven approach for phonetic transcriptions of signs [32] and the rule based approach for HamNoSys (HNS—Hamburg Notation System) notation [29]. An algorithm automatically transferring HNS notation to control trajectories is the most important part of the sign speech synthesis system. Current state of the algorithm accepts most of the valid combinations of HNS symbols like symmetry of signs, arbitrary initial configuration of hands as well as actions (direct, circular, ...), redefinitions of the initial configuration, parallelism of the actions, etc. The algorithm requires information about the signing space, the orientation of palms and fingers, the size of actions, etc. It has to be collected manually in relation to the animation model.

Animation model is controlled via animation frames that are composed into animation trajectories. The animation trajectories store time sequences of values controlling a particular rotation axis of joint connection, $(x, y, z)$ translation of control point or weight of morph target. The animation frames do not directly control joints of shoulder, elbow and wrist (7 DOF) but include pose matrices $P_{4\times4}$. $P_R$ and $P_L$ matrices determine the locations of the wrist, the direction of fingers and twist of palm separately for both arms. The inverse kinematics module (IK) determines the final poses of the arms using these matrices.

### 6.3 Collection of signs

SignEditor is used to get symbolic descriptions of the letters [25], see Fig. 20. A new sign must be manually entered in accordance with the notation rules. Graphic interface includes all HNS symbols and notation process ensures the conversion of all letters to symbolic strings. Furthermore SignEditor allows backward editing and verification. SignEditor incorporates both the animation model and the control model. This feedback immediately transfers input string into animated trajectories and exports created letter to the animation trajectories. The designer of the system can verify final animation of the created letters.

Target languages for fingerspelling synthesis are Czech and English. We consider lexicon incorporating 26 letters and 10 numerals for American Sign Language (ASL) and 46 letters and 10 numerals for Czech Sign Language (CSL). CSL allows using both one- and two-handed fingerspelling alphabet. We chose the two-handed variant because these two-handed signs incorporate simpler hand shapes. The dominant hand forms the shape of the letter. The non-dominant hand has the same or a simpler shape and is in contact with the dominant hand. Seven letters of CSL use only the dominant hand. Others are expressed with both hands located in front of the body. CSL numerals 0–5 are one-handed and 6–9 are two-handed. Several letters in CSL also include a simple movement of arms. This movement allows expressing the diacritic of some CSL letters. HNS provides a rich repertoire and all CSL letters can be successfully converted to the avatar animation.

Different situation occurs for letters, and numerals in ASL. The gesture is expressed by the dominant hand only and requires very complex hand shapes. For example, letter E includes multiple contacts between index finger, ring finger, middle finger and the thumb, letter D and numerals 6–9 incorporate a touch of thumb on all remaining fingers of the hand. M and N letters include an intersection of thumb between remaining fingers and the letter R uses crossing fingers. Current state of the control module does not allow automatic conversion of all ASL letters. On the other hand, a precise animation of these signs is very important, for example crossing fingers distinguishes R and U letters. Hence the ASL letters must be manually corrected. For this purpose we have extended SignEditor allowing direct editing of all joint connections and saving corrected animation frames.

### 6.4 Continuous speech

The lexicon is directly loaded into fingerspelling synthesis during startup. An input of the fingerspelling synthesis system is an utterance expressed by text of the target language. First the synthesis system finds signs in the lexicon for whole word of the input utterance. Thus words, digits and isolated letters separated in the input string by space key are directly chosen from the lexicon. Other unknown words, names, abbreviations etc. have to be spelled. Because a clear separation of these spelled words from neighboring signs is needed, a special "space" sign is inserted at the beginning and end of each spelled word. This special sign puts down the avatar's hands but the letters within spelled words are directly connected without an interruption.

The synthesis system concatenates loaded trajectories to one continuous trajectory in real time. Piecewise linear interpolation is used to get fluent transition between two concatenate signs (letters). The duration of the transition is automatically determined from adjacent animation frames of concatenated signs to get natural transition [31]. Animation frames are generated with a fixed frame rate and directly determine the speed of the animation. Since the speed of fingerspelling for different sign languages differs, the number of animation frames produced by SignEditor must be checked to get the natural rate of resulting animation.

## 7 Speech synthesis

Two TTS systems are applied in our global system: Open MARY TTS [50] for the English and Turkish languages developed by DFKI (Germany), and the Russian TTS engine developed by UIIP (Belarus) and SPIIRAS (Russia) [22]. Unit selection speech synthesis method is used for English, HMM-based speech synthesis method is applied for Turkish, and compilative allophone-diphone based synthesis method was realized for Russian. Male and female voices are available for English and Russian and there are only male voices for Turkish. TTS was realized as a web-based service, which waits for messages from the web-server.

## 8 Conclusion

We have developed a multi-modal communication system that allows the interaction of people with hearing and visual disabilities by translating fingerspelling to speech and speech to fingerspelling. In the real-time operating integrated system, different spoken languages and sign languages such as Czech, Russian and Turkish are used for input and output. Since this is a first attempt for such a multimodal and multi-lingual system, our system has certain shortcomings: On the vision side, the camera is sensitive to skin colored objects in the background and we require that the background does not contain such colors. Hand and face interaction sometimes causes errors for the same reason. Speech recognition is sensitive to excessive environmental noise. Since the speech recognition system has been trained with a news corpus, this makes the recognition biased towards certain words that frequently appear in the news. The system is yet to be extended to some language combinations. We have tested the system with non-disabled general users. General opinion of the users is that the system is practical but fingerspelling is a slow method of communication. Future studies will extend this system for more applications and perform large scale usability tests with handicapped users. The demo videos of the system can be found in [1–3].

**Acknowledgements** This work was developed in the eNTERFACE'10 Summer Workshop on Multimodal Interfaces, Amsterdam, the Netherlands. This work has been supported by: the Scientific and Technological Research Council of Turkey (TUBITAK) project No. 108E113; Grant of the President of Russia No. MK-64898.2010.8; the Ministry of Education and Science of Russia contracts No. 2579 and No. 2360; RFBR project No. 09-07-91220-CT-a; UWB project No. SGS-2010-054; project No. GAČR 102/09/P609; MECR project No. ME08106 and by the Grant Agency of Academy of Sciences of the Czech Republic project No. 1ET101470416.

## References

1. http://www.cmpe.boun.edu.tr/pilab/pilabfiles/demos/enterface2010/Conversation.m4v (2010)
2. http://www.cmpe.boun.edu.tr/pilab/pilabfiles/demos/enterface2010/Avatar.mp4 (2010)
3. http://www.cmpe.boun.edu.tr/pilab/pilabfiles/demos/enterface2010/Integrated_System.wmv (2010)
4. Allen J, Xu R, Jin J (2004) Object tracking using camshift algorithm and multiple quantized feature spaces. In: Proceedings of the Pan-Sydney area workshop on visual information processing, vol 36. Australian Computer Society, Inc, pp 3–7
5. Aran O, Ari I, Akarun L, Dikici E, Parlak S, Saraclar M, Campr P, Hruz M (2008) Speech and sliding text aided sign retrieval from hearing impaired sign news videos. J Multimodal User Interfaces 2(2):117–131. http://www.springerlink.com/index/XX0443800N585126.pdf
6. Arısoy E, Can D, Parlak S, Sak H, Saraçlar M (2009) Turkish broadcast news transcription and retrieval. IEEE Trans Audio Speech Lang Process 17(5):874–883
7. Beutnagel M, Mohri M, Riley M (1999) Rapid unit selection from a large speech corpus for concatenative speech synthesis. In: ESCA, pp 607–610
8. Can D, Saraçlar M (2009) Turkish broadcast news transcription with open-source software. In: IEEE 17th signal processing and communications applications conference (SIU), pp 325–328
9. Chen F (2003) Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vis Comput 21(8):745–758. http://linkinghub.elsevier.com/retrieve/pii/S0262885603000702
10. Daniel T, Jiří K, Jindřich M (2010) Enhancements of viterbi search for fast unit selection synthesis, pp 174–177. http://www.kky.zcu.cz/en/publications/TihelkaDaniel_2010_Enhancementsof
11. Duarte K, Gibet S (2010) Heterogeneous data sources for signed language analysis and synthesis: the signcom project. In: LREC—language resources and evaluation
12. Dutoit T, Bozkurt B (2009) Speech Synthesis, 1st edn. Springer, New York, pp 557–585
13. Elliott R, Glauert JRW, Kennaway JR, Marshall I, Safar E (2008) Linguistic modelling and language-processing technologies for avatar-based sign language presentation. Univers Access Inf Soc 6:375–391
14. Exner D, Bruns E, Kurz D, Grundh A (2005) Fast and reliable CAMShift tracking
15. Ezzat T, Poggio T (1999) Visual speech synthesis by morphing visemes, pp 45–57
16. Fang Y, Cheng J, Wang K, Lu H (2007) Hand gesture recognition using fast multi-scale analysis. In: Fourth international conference on image and graphics (ICIG 2007), pp 694–698. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4297171
17. Fotinea SE, Efthimiou E, Caridakis G, Karpouzis K (2008) A knowledge-based sign synthesis architecture. Univers Access Inf Soc 6:405–418

18. Ganapathiraju A, Hamaker J, Picone J (2000) Hybrid svm/hmm architectures for speech recognition. In: Proceedings of speech transcription workshop, pp 504–507
19. Grůber M, Tihelka D (2010) Expressive speech synthesis for Czech limited domain dialogue system—basic experiments, vol 1, Institute of Electrical and Electronics Engineers, Beijing, pp 561–564. http://www.kky.zcu.cz/en/publications/GruberM_2010_ExpressiveSpeech
20. Hanzlíček Z (2010) Czech hmm-based speech synthesis. In: Text, speech and dialogue. Lecture notes in computer science, vol 6231. Springer, Berlin, pp 291–298. http://www.kky.zcu.cz/en/publications/ZdenekHanzlicek_2010_CzechHMM-Based
21. Heloir A, Kipp M (2010) Real-time animation of interactive agents: Specification and realization. Appl Artif Intell 24:510–529
22. Hoffmann R, Jokisch O, Lobanov B, Tsirulnik L, Shpilewsky E, Piurkowska B, Ronzhin A, Karpov A (2007) Slavonic TTS and STT conversion for let's fly dialogue system. In: 12th international conference on speech and computer SPECOM, Moscow, Russia, pp 729–733
23. Hu M (1962) Visual pattern recognition by moment invariants. IRE Trans Inf Theory 8(2):179–187
24. Jing Z, Min Z (2010) Speech recognition system based improved DTW algorithm. In: Proceedings of the international conference on computer, mechatronics. Control and electronic engineering CMCE-2010, vol 5, pp 320–323
25. Kanis J, Krňoul Z (2008) Interactive HamNoSys notation editor for signed speech annotation. In: ELRA, pp 88–93
26. Karpov A, Ronzhin A, Markov KMZ (2010) Viseme-dependent weight optimization for CHMM-based audio-visual speech recognition. In: Interspeech-2010 proceedings. ISCA Association, Makuhari, pp 2678–2681
27. Kindiroglu AA, Yalcin H, Aran O, Hruz M, Campr P, Akarun L, Karpov A (2010) Multi-lingual fingerspelling recognition for handicapped kiosk. In: Pattern recognition and image analysis, St Petersburg, pp 33–37
28. Krňoul Z (2010) New features in synthesis of sign language addressing non-manual component. In: 4th workshop on representation and processing of sign languages: corpora and sign language technologies
29. Krňoul Z, Kanis J, Železný M, Müller L (2008) Czech text-to-sign speech synthesizer. Mach Learn Multimodal Interact 1, 180–191
30. Krňoul Z, Železný M (2004) Realistic face animation for a Czech Talking Head. Lecture notes in artificial intelligence, vol 3206, pp 603–610
31. Krňoul Z, Železný M (2007) Translation and conversion for Czech sign speech synthesis. Lecture notes in artificial intelligence, vol 4629, pp 524–531
32. Krňoul Z, Železný M, Müller L, Kanis J (2006) Training of coarticulation models using dominance functions and visual unit selection methods for audio-visual speech synthesis. In: Proceedings of INTERSPEECH 2006 - ICSLP. Bonn
33. Kuhl F, Giardina C (1982) Elliptic Fourier features of a closed contour. Comput Graph Image Process 18:236–258
34. Levenshtein V (1966) Binary codes capable of correcting deletions, insertions and reversals. Sov Phys Dokl 10(8):707–710
35. Lin C, Hwang C (1987) New forms of shape invariants from elliptic Fourier descriptors. Pattern Recognit 20:535–545
36. Liu R, Li Z, Jia J (2008) Image partial blur detection and classification. In: 2008 IEEE conference on computer vision and pattern recognition, pp 1–8
37. Liwicki S, Everingham M (2009) Automatic recognition of fingerspelled words in British Sign Language. In: 2009 IEEE computer society conference on computer vision and pattern recognition workshops (iv), pp 50–57
38. Lombardo V, Nunnari F, Damiano R (2010) A virtual interpreter for the Italian sign language. In: Proceedings of the 10th international conference on intelligent virtual agents IVA'10. Springer, Berlin, pp 201–207
39. Marnik J (2007) The Polish finger alphabet hand postures recognition using elastic graph matching. In: Computer recognition systems 2, pp 454–461. http://www.springerlink.com/index/J44W02H1J7U2NXG0.pdf
40. Matoušek J, Hanzlíček Z, Tihelka D, Méner M (2010) Automatic dubbing of tv programmes for the hearing impaired. In: Proceedings of IEEE 10th international conference on signal processing, vol 1. Institute of Electrical and Electronics Engineers, Beijing, pp 589–592. http://www.kky.zcu.cz/en/publications/MatousekJ_2010_AutomaticDubbingof
41. Mohri M, Pereira F, Riley M (2002) Weighted finite-state transducers in speech recognition. Comput Speech Lang 16(1):69–88
42. Nguyen TT, Binh ND, Bischof H (2008) An active boosting-based learning framework for real-time hand detection. In: 2008 8th IEEE international conference on automatic face & gesture recognition, pp 1–6
43. Ojala T (1996) A comparative study of texture measures with classification based on featured distributions. Pattern Recognit 29(1):51–59
44. Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell 24(7):971–987
45. Rabiner L, Juang BH (1993) Fundamentals of speech recognition. Prentice-Hall, Englewood Cliffs
46. Ronzhin A, Karpov A (2007) Russian voice interface. Pattern Recognit Image Anal 17(2):321–336. http://www.springerlink.com/content/376u33001458177p/?p=ad4e76356897411e90554fc1094cb60d&pi=3, pleiades publishing
47. Sak H, Güngör T, Saraçlar M (2010) Resources for Turkish morphological processing. Lang Resour Evaluation
48. Sak H, Saraçlar M, Güngör T (2010) On-the-fly lattice rescoring for real-time automatic speech recognition. In: Interspeech, Makahuri, Japan
49. Schnepp J, Wolfe R, McDonald JC (2010) Synthetic corpora: a synergy of linguistics and computer animation. In: Fourth workshop on the representation and processing of sign languages: corpora and sign language technologies
50. Schröder M, Trouvain J (2003) The German text-to-speech synthesis system MARY: A tool for research, development and teaching. Int J Speech Technol 6(4):365–377
51. Schwarz P, Matejka P, Cernocky J (2006) Hierarchical structures of neural networks for phoneme recognition. In: Proceedings of the IEEE international conference on acoustics, speech and signal processing ICASSP-2006, Toulouse, France
52. Solera-Urena R, Martn-Iglesias D, Gallardo-Antoln A, Pelaez-Moreno C, Daz-de Mara F (2007) Robust ASR using support vector machines. Speech Commun 49(4):253–267
53. Stephenson TA, Escofet J, Magimai.-Doss M, Bourlard H (2002) Dynamic Bayesian network based speech recognition with pitch and energy as auxiliary variables. Idiap-RR Idiap-RR-24-2002 (0 2002). In: IEEE international workshop on neural networks for signal processing NNSP-2002
54. Tort A (2003) Elliptical Fourier functions as a morphological descriptor of the genus Stenosarina (Brachiopoda, Terebratulida, New Caledonia). Math Geol 35(7):873–885
55. Trentin E, Gori M (2001) A survey of hybrid ANN/HMM models for automatic speech recognition. Neurocomputing 37(1–4):91–126
56. Vanaken C, Hermans C, Mertens T, Fiore FD, Bekaert P, Reeth FV (2008) Strike a pose: image-based pose synthesis. In: VMV, pp 131–138

57. Whittaker E (2000) Statistical language modelling for automatic speech recognition of Russian and English. PhD thesis, Cambridge University, Cambridge, UK
58. Yang M, Kpalma K, Ronsin J (2008) A survey of shape feature extraction techniques. Pattern Recognit. http://hal.archives-ouvertes.fr/hal-00446037/
59. Yörük E, Konukolu E, Sankur B, Darbon J (2006) Shape-based hand recognition. IEEE Trans Image Process 15(7):1803–1815
60. Young S, Evermann G, Gales M, Kershaw D, Moore G, Odell J, Ollason D, Povey D, Valtchev V, Woodland P (2006) The HTK book version 3.4
61. Young S (2008) HMMs and related speech recognition technologies. In: Springer handbook of speech processing. Springer, Berlin, pp 539–557
62. Zen H, Braunschweiler N, Buchholz S, Knill KĆSK, Latorre J (2010) HMM-based polyglot speech synthesis by speaker and language adaptive training. In: Proceedings of the 7th ISCA workshop on speech synthesis, pp 186–191