Chapter 12
Providing Feedback in Ukrainian Sign Language Tutoring Software

M.V. Davydov, I.V. Nikolski, V.V. Pasichnyk, O.V. Hodych, and Y.M. Scherbyna

Abstract This chapter focuses on video recognition methods implemented as part of the Ukrainian Sign Language Tutoring Software. At the present time the sign language training software can easily verify how users understand signs and sentences. However, currently there is no good solution to the problem of verifying how the person reproduces signs due to a large variety of training conditions and human specifics. The new approach to user interaction with the Sign Tutoring Software is proposed as well as new algorithms implementing it. The use of body posture recognition methods allows interaction with users during learning of signs and the verification process. The software provides a feedback to the user by capturing person's gestures via a web camera improving the success of training. A single web camera is used without utilising depth sensors. The process of human posture reconstruction from a web camera in real-time involves background modelling, image segmentation and machine learning methods. The success rate of 91.7% has been achieved for sign recognition on the test set of 85 signs.

Key words: Sign language, image segmentation, interactive tutoring, tutoring software, neural-networks

M.V. Davydov, I.V. Nikolski, V.V. Pasichnyk
L'viv Polytechnic National University
12 Bandery St., L'viv, Ukraine
e-mail: maks.davydov@gmail.com

O.V. Hodych, Y.M. Scherbyna
Ivan Franko National University of L'viv
1 Universytetska St., L'viv, Ukraine
e-mail: oles.hodych@gmail.com

12.1 Introduction

Sign languages are based on hand signs, lip patterns and body language instead of sounds to convey meaning. The development of sign languages is generally associated with deaf communities, which may include hearing or speech impaired individuals, their families and interpreters. The only currently viable ways to enable interactive communication between hearing impaired and not impaired people is to use services provided by interpreters (specially trained individuals or software applications), or to learn a sign language.

A manual communication has developed in situations where speech is not practical. For instance, scuba diving or loud work places such as stock exchange. In such cases learning a sign language is the most effective solution.

Teaching a sign language is a challenging problem, which often requires a professional tutor, presence of students in a class room and many hours of practice. There is a number of commercial software packages and research projects directed at developing software applications for sign language interpretation. The majority of such software is directed at interpreting a spoken to sign language, which covers all major cases where interpretation is required for deaf people (e.g. conventions, television). For example, researchers at IBM have developed a prototype system, called SiSi (say it, sign it), which offers some novel approaches by utilising avatars to communicate speech into sign language [41]. Some other applications, such as iCommunicator [42], are based on a database of sign language videos and provide means to adaptively adjust to user's speech. The are several online sign language tutoring software packages such as Handspeek ASL Dictionary [1] that can demonstrate signs, but cannot provide any feedback to the user. Some systems that provide feedback [2] require wearing coloured gloves for better hand shape processing. Other approaches to capture data for sign language feedback include the use of Kinect hardware or 3DV Systems ZCam [5].

The problem of tracking head and hands in sign language recognition was studied by Jörg Zieren and Karl-Friedrich Kraiss [39]. The tracking makes the use of several methods of ambiguity prevention such as Bayesian trust network, expectation maximisation algorithm, CAMSHIFT algorithm [4]. This approach allows tracking of hands in motion even if they overlap the face. For the dictionary of 152 signs for one user 2.4% of Word Error Rate (WER) was obtained.

The accuracy of face localisation in case of face/hand overlapping was improved in the research by Suat Akyol and Jörg Zieren [1]. In this study the Active Shape Model was applied in order to locate the face. The research conducted by Jörg Zieren and others [40] resulted in 0.7% of WER for the test based on 232 gestures of the same person segmented manually in ideal conditions. WER of 55.9% was achieved for the user who did not participate in the system training based on gestures by six speakers under controlled light conditions.

[1] www.handspeak.com

In the study by Morteza Zahedi [38] for the video camera-based sign recognition utilised the hidden Markov model and two image comparison methods (the comparison method with deformations and the tangent distance method). Based on BOSTON50 signs the 17.2% of WER was reached. This method was improved by Philippe Dreuw and others [14] by introducing the trigram-based linguistic model, motion path determination of the main hand movement and the reduction in the number of characteristics using the principle component analysis method. The 17.9% of WER was reached for the RWTH-Boston-104 signs. The application of the special algorithm of hand shape tracking [15], the model for image creation and the sentence model for the sign recognition in the study [16] resulted in 11.24% of WER for the RWTH-Boston-104 signs. This has been the best result so far.

In the research conducted by authors [10], [11], [12], [9], [13] achieved 91.7% sign recognition rate on the test set of 85 signs.

This chapter provides a review of the developed by authors methodology for sign language recognition with an emphasis on video recognition methods used for developing the Ukrainian Sign Language Tutoring Software. In relation to our previous publications, this is the first time where the problem of providing feedback as part of the developed software is presented.

12.2 Problem Formulation

We are working on a generic solution, which could be available to every Ukrainian School for hear-impaired students. The requirement for such solution is to be inexpensive. Thus, it has been decided to utilise a commodity hardware – PC and a web-camera as a video input device. One of the main goals formulated for the software it to enable a tutor-like experience, where students would be provided with a feedback about the correctness of the sign they're trying to reproduce. In order to reflect this the term *tutoring software* (or simply *tutor*) is used instead of *training software*.

The developed sign language tutoring software incorporates solutions to the following problems:

- system setup in a new environment;
- tracking position for left and right hands;
- hand shape matching;
- providing feedback during tutoring.

The Ukrainian Sign Language Video Dictionary, which was originally developed by Ivanusheva and Zueva, has been used to collect the necessary data. This dictionary consists of three continuous DVD video files, which have indexed in order to extract separate signs and phrases. For the purpose of providing feedback, the tutoring software needs to know the shape and the location of hands and the face in

the video tutorials. Although, it is possible to pre-process the video data, the developed software processes video streams from both the video tutorial being played back and the web-camera capturing the student simultaneously in real-time during the tutoring process. This approach provides an easy way to extend the database of videos featuring additional signs without the need for special preparation.

The following sections provide more details on how each of the specified above problems are addressed in the developed tutoring software.

12.3 System Setup in the New Environment

The initial stage of using the developed tutoring software requires tuning of the system parameters to the conditions of the new environment. The feedforward neural network classifier and Inverse Phong Lighting Model [36] are utilised to identify the face and hands from the video. Both classifiers require a training set for adjustment to the new environment.

The problem of hand classifier setup can be solved in different ways. One of the commonly used approaches requires user to put the hand in a particular place of the web-camera frame. The approached utilised in this research requires a user to move the hand in front of and in close proximity to the web-camera in order to make it the largest object captured by the camera. During this process a method based on the Self-Organising Map (SOM) is applied to separate the hand from the background [24, 25].

12.3.1 SOM-based image segmentation

It is well known that SOM operates based on the principles of the brain. One of the key SOM features is the preservation of the topological order of the input space during the learning process. Some relatively recent research of the human brain has revealed that the response signals are obtained in the same topological order on the cortex in which they were received at the sensory organs [32]. One of such sensory organs are eyes, thus making the choice of SOM for analysing the visual information one of the most natural.

Colour is the brain's reaction to specific visual stimulus. Therefore, in order to train SOM for it to reflect the topological order of the image perceived by a human eye, it is necessary to choose the colour space, which closely models the way sensors obtain the visual information. The eye's retina samples colours using only three broad bands, which roughly correspond to red, green and blue light [17]. These signals are combined by the brain providing several different colour sensations, which are defined by the CIE (Commission Internationale de l'Eclairage (French), International Commission on Illumination) [28]: Brightness, Hue and Colourfulness. The CIE commission defined a system, which classifies colour according to the human

visual system, forming the tri-chromatic theory describing the way red, green and blue lights can match any visible colour based on the eye's use of three colour sensors.

The colour space is the method, which defines how colour can be specified, created and visualised. As can be deduced from the above, most colour spaces are three-dimensional. There are more than one colour space, some of which are more suitable for certain applications than others. Some colour spaces are perceptually linear, which means that an n-unit change in stimulus results in the same change in perception no matter where in the space this change is applied [17]. The feature of linear perception allows the colour space to closely model the human visual system. Unfortunately, the most popular colour spaces currently used in image formats are perceptually nonlinear. For example, BMP and PNG utilise RGB[2] colour space, JPEG utilises YCbCr, which is a transformation from RGB, HSL[3] is another popular space, which is also based on RGB.

The CIE based colour spaces, such as CIELuv and CIELab, are nearly perceptually linear [17], and thus are more suitable for the use with SOM. The CIEXYZ space devises a device-independent colour space, where each visible colour has non-negative coordinates X, Y and Z [27]. The CIELab is a nonlinear transformation of XYZ onto coordinates $L*, a*, b*$ [27].

The image format used in our research is uncompressed 24-bit BMP (8 bit per channel), which utilises the RGB colour space. In order to convert vectors $(r,g,b) \in RGB$ into $(L*,a*,b*) \in CIELab$ it is necessary to follow an intermediate transformation via the CIE XYZ colour space. These transformations are described in details in [26] and [27]. Application of the two-step transformation to each pixel of the original image in RGB space produces a transformed image in CIELab space used for further processing.

It is important to note that when using SOM it is common to utilise Euclidean metric for calculation of distances during the learning process [32][4]. Conveniently, in CIELab space the colour difference is defined as Euclidean distance [27].

Instead of using every image pixel for the SOM training process, the following approach was employed to reduce the number of data samples in the training dataset. The basic idea is to split an image into equal segments $n \times n$ pixels. Then for each such segment find two the most diverged pixels and add them to the training dataset. Finding the two most diverged pixels is done in terms of the distance applicable to the colour space used for image representation. Due to the fact that each pixel is a three dimensional vector, each segment is a matrix of vector values. For example, below is an image A of 4×4 pixels in size represented in the CIELab space, and split into four segments 2×2 pixels each.

---

[2] Uncompressed BMP files, and many other bitmap file formats, utilise a colour depth of 1, 4, 8, 16, 24, or 32 bits for storing image pixels.

[3] Alternative names include HSI, HSV, HCI, HVC, TSD etc. [17]

[4] The selection of the distance formula depends on the properties of the input space, and the use of Euclidean metric is not mandatory.

$$A = \left(\begin{array}{cc|cc} (L_1^1,a_1^1,b_1^1)^T & (L_2^1,a_2^1,b_2^1)^T & (L_3^1,a_3^1,b_3^1)^T & (L_4^1,a_4^1,b_4^1)^T \\ (L_1^2,a_1^2,b_1^2)^T & (L_2^2,a_2^2,b_2^2)^T & (L_3^2,a_3^2,b_3^2)^T & (L_4^2,a_4^2,b_4^2)^T \\ \hline (L_1^3,a_1^3,b_1^3)^T & (L_2^3,a_2^3,b_2^3)^T & (L_3^3,a_3^3,b_3^3)^T & (L_4^3,a_4^3,b_4^3)^T \\ (L_1^4,a_1^4,b_1^4)^T & (L_2^4,a_2^4,b_2^4)^T & (L_3^4,a_3^4,b_3^4)^T & (L_4^4,a_4^4,b_4^4)^T \end{array}\right)$$

Thus, the first segment is:

$$S_1 = \left(\begin{array}{cc} (L_1^1,a_1^1,b_1^1)^T & (L_2^1,a_2^1,b_2^1)^T \\ (L_1^2,a_1^2,b_1^2)^T & (L_2^2,a_2^2,b_2^2)^T \end{array}\right)$$

The above approach can be summarised as the following algorithm. Let n denote the size of segments used for image splitting, the value of which is assigned based on the image size. T – the training set, which is populated with data by the algorithm. Let's also denote jth pixel in segment S_i as $S_i(j)$. Further in the text both terms *pixel* and *vector* are used interchangeably.

Algorithm 12.1 Training dataset composition

Initialisation. Split image into segments of $n \times n$ pixels; $N > 0$ – number of segments; $T \leftarrow \varnothing$; $i \leftarrow 1$.

1. Find two the most diverged pixels $p' \in S_i$ and $p'' \in S_i$ using Euclidean distance.
   - 1.1 $max \leftarrow -\infty$, $j \leftarrow 1$
   - 1.2 $k \leftarrow j+1$
   - 1.3 Calculate distance between pixels $S_i(j)$ and $S_i(k)$: $dist \leftarrow \|S_i(j) - S_i(k)\|$
   - 1.4 If $dist > max$ then $p' \leftarrow S_i(j)$, $p'' \leftarrow S_i(k)$ and $max \leftarrow dist$
   - 1.5 If $k < n \times n$ then $k \leftarrow k+1$ and return to step 1.3
   - 1.6 If $j < n \times n - 1$ then $j \leftarrow j+1$ and return to step 1.2
2. Add $p' \in S_i$ and $p'' \in S_i$ to the training set: $T \leftarrow T \cup \{p', p''\}$
3. Move to the next segment $i \leftarrow i+1$. If $i \leq N$ then return to step 1, otherwise stop.

The above algorithm provides a way to reduce the training dataset. It is important to note that an excessive reduction could cause omission of significant pixels resulting in poor training. At this stage it is difficult to state what rule can be used to deduce the optimal segment size. The segmentation used for the presented results was obtained though experimentation. However, even applying segmentation 2×2 pixels to an image of 800×600 pixels in size reduces the training dataset from 460000 down to 240000 elements, which in turn enables the use of a smaller lattice and reduces the processing time required for SOM training.

There are several aspects to a successful application of SOM, among which are:

- Self-organisation process, which encompasses a problem of selecting a learning rate and a neighbourhood function.
- The size and structure of the SOM lattice.

In this research the guidelines from [32] and [22] were followed to conduct the self-organisation process. The structure of the SOM lattice may differ in its dimensionality and neighbourhood relation between elements. The use of 2-dimensional lattice with hexagonal neighbourhood relation proved to be the most efficient in our research producing more adequate clustering results comparing to other evaluated configurations.

Once the SOM structure and parameters for self-organisation process are selected, the SOM is trained on the training set T, which is composed for the image to be clustered. The trained SOM is then used for the actual image clustering.

The topology preservation SOM feature is fundamental to the developed image segmentation approach. Its basic underlying principles are:

- Image pixels represented by topologically close SOM elements should belong to the same cluster and therefore segment.
- The colour or marker used for segment representation is irrelevant as long as each segment is associated with a different one.

These two principles suggest that the position of SOM elements in the lattice (i.e. coordinates on the 2D plane) can be used for assigning a marker to a segment represented by any particular element instead of the elements' weight vectors. This way weight vectors are used purely as references from 2D lattice space into 3D colour space, and locations of SOM elements represent the image colour distribution. As the result of a series of conducted experiments the following formulae for calculating an RGB colour marker for each element have produced good results.

$$R_j \leftarrow x_j + y_j \times \lambda; G_j \leftarrow x_j + y_j \times \lambda; B_j \leftarrow x_j + y_j \times \lambda; \tag{12.1}$$

Values x_j and y_j in formula (12.1) are the coordinates of SOM elements in the lattice $j = \overline{1,M}$, where M is the total number of elements. Constant λ should be greater or equal to the diagonal of the SOM lattice. For example, if SOM lattice has a rectangular shape of 16×16 elements then λ could be set to 16. Applying the same formula for R, G, and B components produces a set of grayscale colours. However, each element has its own colour, and one of the important tasks is grouping of elements based on the assigned colours into larger segments. There are several approaches, which are being currently developed to provide automatic grouping of SOM elements into clusters and have shown good results [23, 25]. One of the possible approaches is to apply a threshold to the segmented with SOM image, which requires human interaction in specifying the threshold value. This image segmentation approach can be summarised as the following algorithm.

The following figures depict the original and segmented images corresponding to several frames of the same video, which have been processed using the same trained SOM. The recorded video captured an open palm closing and opening again during a period of several seconds. The recording was done using an ordinary PC

Algorithm 12.2 Image segmentation

Initialisation. $p_j = (R_j, G_j, B_j)$ – pixel j; $j = \overline{1,K}$; $K > 0$ – total number of pixels; $j \leftarrow 1$; $i^*(p_j) = (R_{i^*}, G_{i^*}, B_{i^*})$ – a weight vector of the best matching unit (BMU – winning element) for input vector p_j; (x_{i^*}, y_{i^*}) – coordinates of element i^*; choose appropriate values for λ.

1. Find $BMU(p_j)$ for vector p_j in the trained SOM utilising the distance used for training (Euclidean for CIELab).
2. Calculate marker for pixel p_j: $R_j \leftarrow x_{i^*} + y_{i^*} \times \lambda$, $G_j \leftarrow R_j$, $B_j \leftarrow R_j$.
3. Move to the next image pixel: $j \leftarrow j+1$;
4. If $j \leq K$ return to step 1, otherwise stop.

web camera capable of 30FPS throughput with a frame size of 800×600 pixels. The background of the captured scene is nonuniform, which increases the complexity of image segmentation.

Figure 12.1 depicts a fully open palm (frame 25), contracted fingers (frame 35) and a fully closed palm (frame 40).

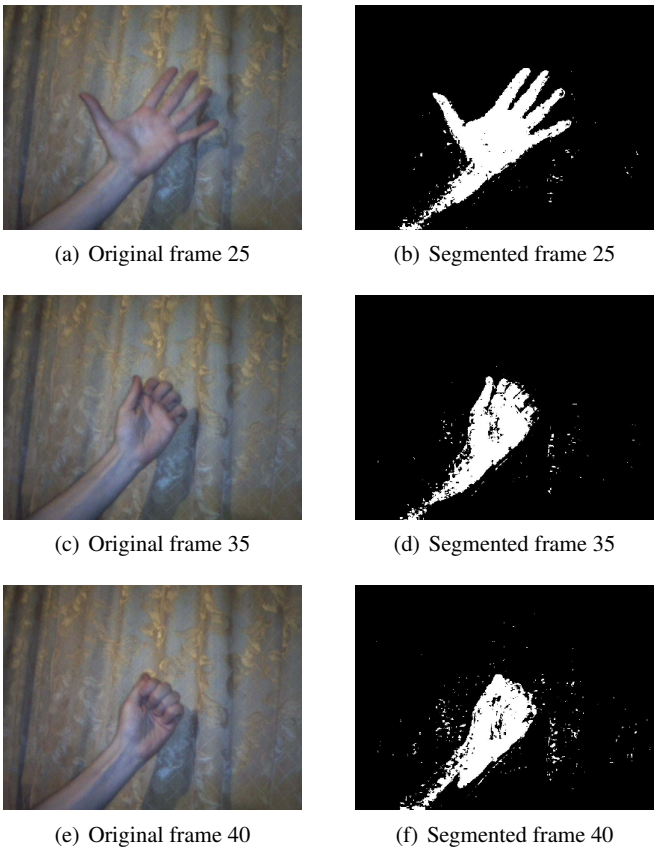
The important aspect of the presented results is the use of SOM trained only on a single frame. This initial frame as well as all subsequent ones have been successfully segmented with clear separation of the human palm from the nonuniform background. The use of only one frame for SOM training allows much faster dynamic image segmentation needed for video, avoiding SOM retraining for every frame.

The trends of the past decade in architecture of the central processing unit show a clear direction towards multi-core processors with the number of cores increasing every eighteen months according to the Moore's law. The shift from fast single-core to slower multi-core CPUs poses a question of scalability for a vast class of computational algorithms including algorithm for image processing.

The developed sing tutoring sofrware provides effective utilisation of multi-core processors, which includes parallelisation of SOM training based on methods for SOM decomposition published in [18, 19, 33, 34]. The flowchart diagram depicted on Fig. 12.2 outlines the SOM training algorithm for multi-core processors, which is incorporated into the tutoring software.

More details on the computational aspects of the developed software is provided in our monograph [25], which also includes the caching algorithm used to speedup segmentation of frames in the video stream eliminating the need to search for BMU for each input pixel.

Once the regions containing hands are identified, the feedforward neural network classifier or Inverse Phong Lighting model classifier is used to segment skin coloured areas in real-time.



(a) Original frame 25 (b) Segmented frame 25

(c) Original frame 35 (d) Segmented frame 35

(e) Original frame 40 (f) Segmented frame 40

Fig. 12.1 Frames 25, 35 and 40

12.3.2 Feedforward neural network classifier

In order to segment skin sections the neural network classifier has been applied to the surrounding area of each pixel in the image matrix $M = \{m_{ij}\}$, $i = 1,2,\ldots,w$, $j = 1,2,\ldots,h$, where w and h are image width and height respectively.

The significant features of the area are selected from the cross-shaped or square surrounding areas (Fig. 12.3). The best result was achieved when pixel are represented in the YCbCr colour space.

The feature extraction can be achieved by utilising rough sets for calculating the upper and lower approximations of the surrounding areas [43]. The developed software utilises neural network-based classifiers with one hidden layer and one or two outputs. For the neural network with a single output the area is considered to be of skin colour if the output value is greater than 0.5.

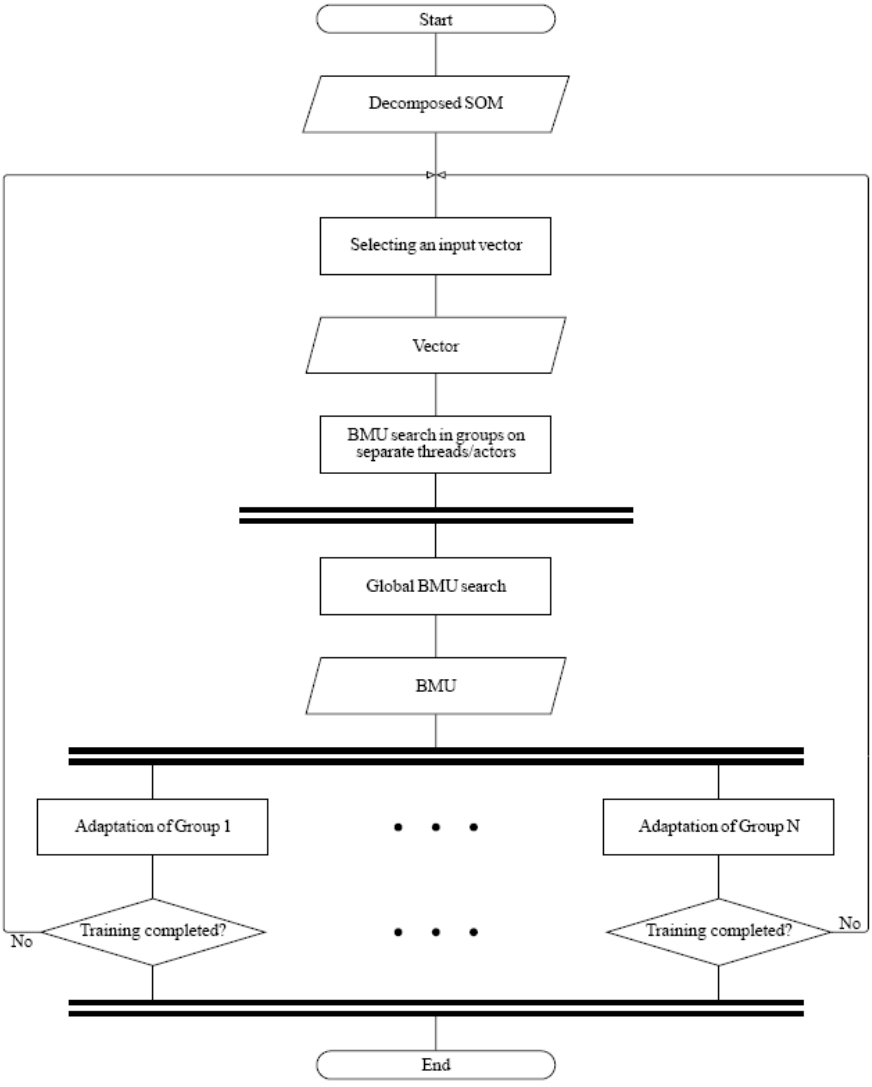


Fig. 12.2 Decomposed SOM training

When the neural network with two outputs is applied the area is considered to be of skin colour if the output value of the first neuron is greater than the output value of the second one. The neural network classifier with two outputs proved to be susceptible to noise. Thus, additional smoothing is required, which can be achieved by averaging each pixel by four neighbouring pixels. The smoothing prevents faulty reactions. The set of positive training samples is formed from the skin area and the

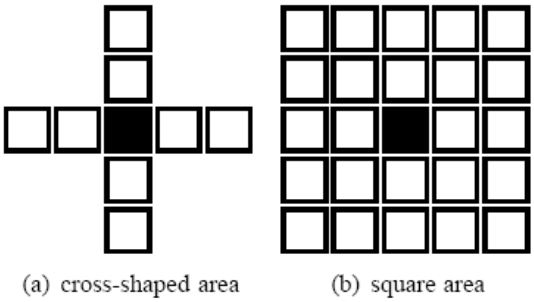


Fig. 12.3 The cross-shaped (a) and square (b) surrounding area for image feature extraction. The image pixel, the surrounding area of which is considered, is marked black. The surrounding area has radius $r = 2$.

negative training samples are formed from the image area not marked as a skin. The complete training set consists of positive and negative training samples. The modification of the back-propagation of error algorithm was developed for training of the feedforward neural network-based classifier [9].

This modification includes the following:

1) training samples are divided into groups according to the value of the required output providing the uniform selection of training samples from each group;
2) the groups for the areas with the worst results are represented by larger sets of training samples;
3) additional random value from the interval $[-\varepsilon, +\varepsilon]$ is added to the neural network weights; the value of this parameter is decreased during the training from the initial value ε_0 to zero;
4) in case where the error cannot be reduced during three iterations of the training algorithm the weight values of the neural network are reset to original;
5) an additional parameter has been introduced to accelerate the scale shift of the neural network weights of the hidden network layer.

The application of the developed method reduces the training time 10x in comparison to the classical back-propagation of error algorithm [9]. This result is significant for the development of interactive software and software trained during the usage. The result of the trained neural network classifier application is depicted on Fig. 12.4.

12.3.3 Inverse Phong lighting model classifier

The skin-colour segmentation can be implemented by using the simplified Inverse Phong lighting model. In this research only the diffuse and ambient components of

(a) Original image (b) Processed image

Fig. 12.4 Skin segmentation by neural network classifier

the Phong lighting model is utilised to estimate the lighting parameters.

$$\begin{pmatrix} I_R \\ I_G \\ I_B \end{pmatrix} = \begin{pmatrix} k_{aR} & i_{aR} \\ k_{aG} & i_{aG} \\ k_{aB} & i_{aB} \end{pmatrix} + (\overline{L} \cdot \overline{N}) \begin{pmatrix} k_{dR} & i_{dR} \\ k_{dG} & i_{dG} \\ k_{dB} & i_{dB} \end{pmatrix} + \begin{pmatrix} \varepsilon_R \\ \varepsilon_G \\ \varepsilon_B \end{pmatrix}, \tag{12.2}$$

where $I = (I_R, I_G, I_B)^T$ – the intensity of red, green and blue pixel components, $k_a = (k_{aR}, k_{aG}, k_{aB})^T$ – ambient dispersion coefficients, $i_a = (i_{aR}, i_{aG}, i_{aB})^T$ – the intensity of ambient lighting, $k_d = (k_{dR}, k_{dG}, k_{dB})^T$ – diffuse dispersion coefficients, $i_d = (i_{dR}, i_{dG}, i_{dB})^T$ – the intensity of diffusive lighting, $\overline{L}$ – the unit vector denoting lighting direction, $\overline{N}$ – the unit vector normal to the surface, $\varepsilon = (\varepsilon_R, \varepsilon_G, \varepsilon_B)^T$ – the normally diffused error.

In order to setup the illumination model parameters the hand area is used. It is assumed that the illumination properties are constant for the whole area of the hand.

Equation (12.2) can be rewritten as

$$I = c_a + \alpha c_d + \varepsilon \tag{12.3}$$

where $c_a = (k_{aR} \cdot i_{aR}, k_{aG} \cdot i_{aG}, k_{aB} \cdot i_{aB})^T$ – the permanent intensity of ambient lighting in the frame, $\alpha = (L \cdot N)$, $\alpha \in [0,1]$ – the coefficient determining how surface is oriented in respect to the source of light, $c_d = (k_{dR} \cdot i_{dR}, k_{dG} \cdot i_{dG}, k_{dB} \cdot i_{dB})^T$ – the permanent intensity of the diffuse illumination in the frame, $\varepsilon = (\varepsilon_R, \varepsilon_G, \varepsilon_B)^T$ – the normally distributed error. Parameters of the illumination model c_a and c_d are determined by the method of linear regression based on the training samples.

Thus, having a training set of pixel colours I_i corresponding to the skin colour of hand it is possible to calculate c_a and c_d:

$$I = \frac{1}{n} \sum_{i=1}^{n} I_i \tag{12.4}$$

$$k = norm\left(\sum_{i=1}^{n}(I_{iR} - I_R)\cdot(I_i - I)\right) \tag{12.5}$$

$$v_{min} = \min_i(I_i - I, k) \tag{12.6}$$

$$v_{max} = \max_i(I_i - I, k) \tag{12.7}$$

$$c_a = I + v_{min}\cdot k \tag{12.8}$$

$$c_d = (v_{max} - v_{min})\cdot k \tag{12.9}$$

The pixel of image with intensity of colour I is classified as similar to the colour of skin if there is a value $\alpha \in [0,1]$ such that

$$|I - c_a - \alpha\cdot c_d| \leq \theta \tag{12.10}$$

where θ is the threshold value.

Once lighting model is determined it is used to calculate the truth level t_p of pixel coloured $P = (p_r, p_g, p_b)^T$.

$$t_p = max\left(0, \frac{1}{\theta}\cdot(\theta - |P - c_d\cdot\max(0, \min(1, (P - c_a, c_d)))|)\right) \tag{12.11}$$

For each pixel the degree of truth is calculated utilising formula (12.11) in order to classify the pixel as skin coloured. Formula (12.11) calculates how far is the pixel from the skin diffuse illumination model. The result of this calculation is a grayscale image where lighter pixels represent hand or face segments.

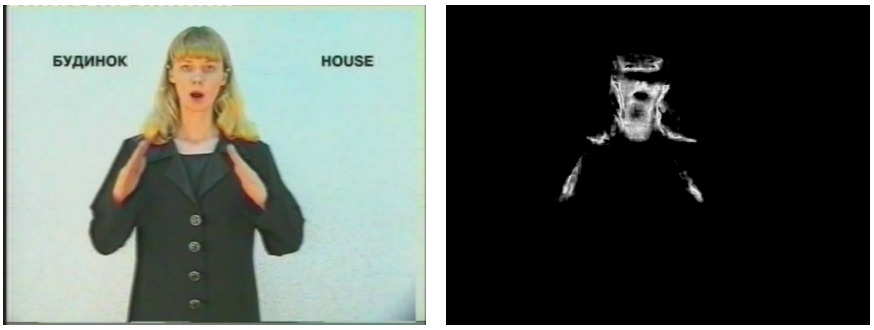
Fig. 12.5 depicts the result of skin segmentation with the simplified Phong illumination model. As can be observed the result achieved for depicted example from the sign dictionary with the illumination model is slightly better for elimination of the hair colour.

12.4 Hands and Face Extraction

The next step is to find centroids of left and right hand. The K-means clustering and connected points labelling algorithm are used for this purpose.

There are several cases that should be considered for the application of K-means: only the head is in the frame, the head and one hand, the head and two well distinguished hands, the head and two hands located close to each other. Three clusters are specified for the initial K-means clustering and then the result is checked as to how well are the clusters connected with each other.

A better result is achieved when using metric space similarity joins of the skin regions [29] by providing minimal distance d between the clusters and minimal



(a) Original image (b) Processed image

Fig. 12.5 Skin segmentation with the simplified Phong illumination model

weight of each cluster w. The task is to divide objects into clusters, so that the minimal distance between each cluster is more that d, and the cluster cannot be divided into smaller clusters in order to preserve this property.

The comparison of the results produced by k-means and similarity joins algorithms is depicted on Fig. 12.6.

(a) K-means clustered image (b) Partitioning by similarity joins

Fig. 12.6 Using k-means (a) and similarity joins (b) for locating hands and face

The K-means clustering produces more errors in comparison to the result produced by the similarity joins partitioning for the head and neck when hands are located closer to the face.

The Haar Cascade Classifier face detector from OpenCV library was used for the face location. The face detection classifier from OpenCV library, which is based on the integral image classifiers, was used in the dictionary pre-processing stage to improve the face recognition. However, two major drawbacks have been identified for the face classifier:

1) high computational complexity – it takes 100 ms per frame to calculate;
2) inadequate results for cases of hands intersecting the head.

It was proposed to utilise the fact that the head is a large skin-coloured cluster at the top of an image, which is well separated from segments representing hands. This was used to apply a fast similarity joins partitioning algorithm for identifying the exact location of hands and face.

The example of the extracted hand area is depicted on Fig. 12.7.

Fig. 12.7 Example of automatically extracted hand shapes

12.5 Feedback During Tutoring

The results of the undertaken studies [25] lay the foundation for the developed Ukrainian Sign Language Tutoring software. The software consists of the sign dictionary and verification modules. The student can observe simultaneously the sign video and his/her own feedback-image in the main window of the program (Fig. 12.8).

The extracted hand shapes and the positions of the tutor on the video and the student hands are used to provide feedback to the student.

As depicted on Fig. 12.8, the tutoring software consists of three windows – the window on the left listing the words from the dictionary, the window in the middle plays back the video with a sign corresponding to the selected in the list word, the

Fig. 12.8 Main window of the tutor application

window on the right displays the video stream of the student as captured by a web-camera.

The provided feedback allows the student to repeat the sign after the reference video (middle window) at any speed and correct the way it is being reproduced by adjusting hand positions. The flowchart of the sign tutoring algorithms is depicted on Fig 12.9.

The sign dictionary video and the student video from the web-camera are processed simultaneously in two different hardware threads. The hand and head positions are extracted and normalised according the frame size. The best correspondence of hands and face positions is estimated and if the distance from the student and tutor hands locations is too large then the sign video is paused. If the positions are close then the hand shapes are compared using pseudo 2-dimensional image deformation model (P2DIDM) [10]. In case where the student makes a mistake in reproducing the sign, the video with the tutor is paused, and the system awaits the student to correct the positions of the hands.

The user can playback the sign at different speeds, frame-by-frame or in a loop mode. The provided feedback allows synchronisation of the sign performed by the student and tutor. The interactive part of the tutoring software is intended to improve the practice experience of speaking the sign language.

12.6 Conclusion

The developed tutoring software has been successfully trialled in several Ukrainian schools for hearing impaired. The main advantage of the incorporated into the software methods is their ability to adjust to different light conditions and skin colour.

The interactive tutoring process with the help of gestures is much more interesting than traditional, and provides the opportunity for students to practice gestures on their own. This is especially important for distance learning.

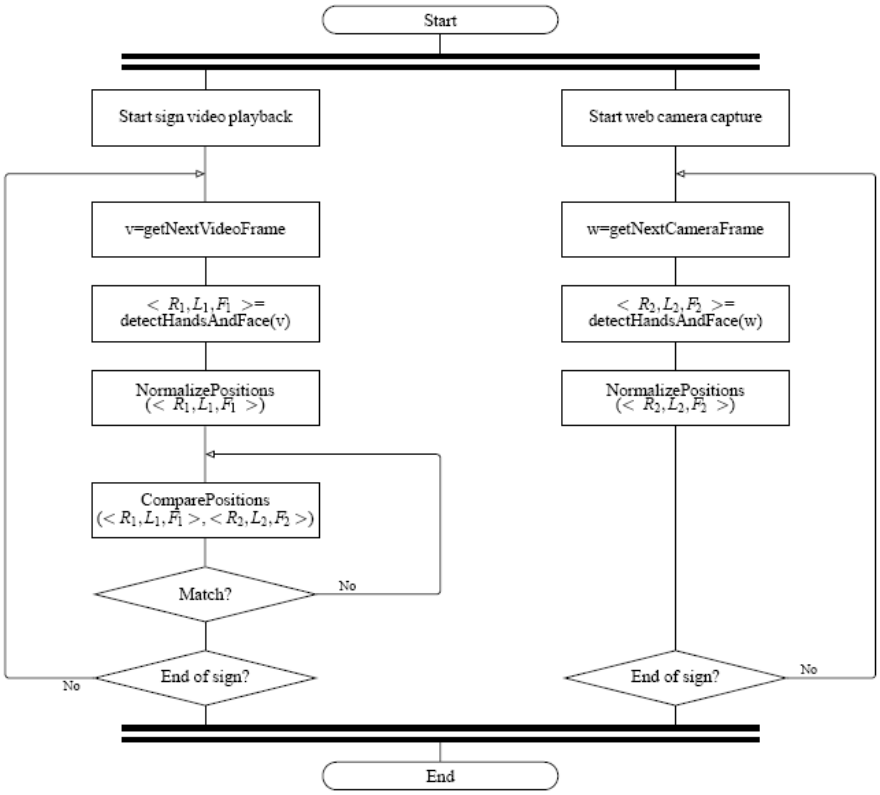


Fig. 12.9 Sign tutor feedback algorithm

The proposed algorithm is not robust enough when processing skin coloured objects. Another complex problem, which is not fully resolved with the developed software is the recognition of hand shapes in cases where hands overlap the head. The robustness of the software can be improved by utilising a depth camera. Currently the developed algorithms are being adjusted to make the use of additional depth information provided by the Microsoft Kinect device.

The sign language tutoring software allows teacher to select the necessary gestures for teaching quickly as well as to control gesture reproduction, which increases the efficiency of the sign language classes in comparison to the use of video materials on tapes or DVDs.

The developed software makes an effective use of the multi-core processors. The Amdahl's law $\frac{1}{(1-P)+\frac{P}{N}}$, where P is the portion of the program that can be paralleled and N – the number of hardware threads, provides a way to calculate the maximum expected

improvement to an overall system. Formula (12.12) can be used to estimate the value of P.

$$P_{est} = \frac{\frac{1}{SU} - 1}{\frac{1}{NP} - 1}, \tag{12.12}$$

where SU – empirically calculated speedup coefficient for N hardware threads. Specifically, the estimation for the developed software running on the four-core processor provided by formulae (12.13) and (12.14).

$$P_{est} = \frac{\frac{1}{2.8} - 1}{\frac{1}{4} - 1} \approx 0.86 \tag{12.13}$$

$$\lim_{N\to\infty} \frac{1}{(1-0.86) + \frac{0.86}{N}} \approx 7.14 \tag{12.14}$$

Fig. 12.10 depicts the charts for different values of P, where the black solid chart corresponds to the discussed here software.

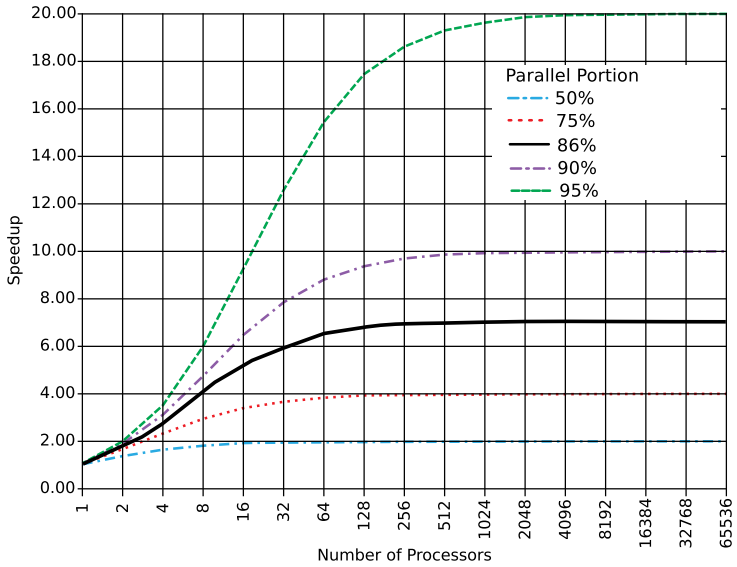


Fig. 12.10 Amdahl's Law

The developed tutoring software is not limited to Ukrainian Sign Language and can easily be adapted to any sign language as it only utilises recorded sign videos as an input.

References

1. Akyol, S., Zieren., J.: Evaluation of ASM head tracker for robustness against occlusion. Proceedings of the International Conference on Imaging Science, Systems, and Technology (CISST 02), June 24-27, Las Vegas, Nevada, Volume I, CSREA Press (2002)
2. Aran, O., Keskin, C., Akarun, L.: Sign language tutoring tool. Proceedings EUSIPCO'05 (2005)
3. Akgül, C. B.: Cascaded self-organizing networks for color image segmentation, 2004, `http://www.tsi.enst.fr/~akgul/oldprojects/CascadedSOM\_cba.pdf`. Cited 15 Apr 2011
4. Bradski, G.: Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal **Q2'98**, (1998)
5. Brashear, H., Zafrulla, Z., Starner, T., Hamilton, H., Presti, P., Lee, S.: CopyCat: A corpus for verifying american sign language during gameplay by deaf children. 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies, Proceedings of the 7th (2010)
6. Brown, D., Craw, I., Lewthwaite, J.: A SOM Based Approach to Skin Detection with Application in Real Time Systems, University of Aberdeen, 2001, `http://www.bmva.ac.uk/bmvc/2001/papers/33/accepted\_33.pdf`. Cited 15 Apr 2011
7. Campbell, N.W., Thomas, B.T., Troscianko, T.: Neural networks for the segmentation of outdoor images. International Conference on Engineering Applications of Neural Networks, pp. 343–346. (1996)
8. Campbell, N.W., Thomas, B.T., Troscianko, T.: Segmentation of Natural Images Using Self-Organising Feature Maps. University of Bristol (1996)
9. Davydov, M. V., Nikolski, I. V.: Real-time Video Object Classification Using Neural Network Classifier. Herald of National University "Lvivska Polytechnica", No. **549**, 82–92 (2005) (Lviv, Ukraine, in Ukrainian)
10. Davydov, M. V., Nikolski, I. V.: Automatic identification of sign language gestures by means on dactyl matching. Herald of National University "Lvivska Polytechnica", No. **589**, 174–198 (2007) (Lviv, Ukraine, in Ukrainian)
11. Davydov, M. V., Nikolski, I. V., Pasichnyk, V. V.: Software training simulator for sign language learning. Connection, 98–106 (2007) (Kyiv, Ukraine, in Ukrainian)
12. Davydov, M. V., Nikolski, I. V., Pasichnyk, V. V.: Selection of an effective method for image processing based on dactyl matching for identification of sign language gestures. Herald of Kharkiv National University of Radio-Electronics, No. **139**, 59–68 (2008) (Kharkiv, Ukraine, in Ukrainian)
13. Davydov, M. V., Nikolski, I. V., Pasichnyk, V. V.: Real-time Ukrainian sign language recognition system. Intelligent Computing and Intelligent Systems (ICIS), 2010, IEEE International Conference, pp. 875–879. (2010)
14. Dreuw, P., Rybach, D., Deselaers, T., Zahedi, M., Ney. H.: Speech recognition techniques for a sign language recognition system. ISCA best student paper award Interspeech 2007, Antwerp, Belgium, pp. 2513–2516. (2007)
15. Dreuw, P., Stein, D., Ney., H.: Enhancing a sign language translation system with vision-based features. Gesture in Human-Computer Interaction and Simulation (GW07), Lisbon, Portugal, pp. 108–113. (2007)
16. Dreuw, P., Forster, J., Deselaers, T., Ney, H.: Efficient approximations to model-based joint tracking and recognition of continuous sign language. IEEE International Conference Automatic Face and Gesture Recognition, Amsterdam, The Netherlands (2008)

17. Ford A., Roberts A.: Colour Space Conversions, 1998, http://www.poynton.com/PDFs/coloureq.pdf. Cited 15 Apr 2011
18. García, C., Prieto, M., Pascual-Montano, A. D.: A speculative parallel algorithm for self-organizing maps. PARCO, pp. 615–622. (2005)
19. Hämäläinen, T. D.: Parallel implementation of self-organizing maps. In: Udo Seiffert, Lakhmi C. Jain (eds.) Self-Organizing Neural Networks: Recent Advances and Applications, pp. 245-278. Springer-Verlag, New York (2002)
20. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, (2001)
21. Hodych, O., Nikolskyi, Y., Shcherbyna, Y.: Application of Self-Organising Maps in medical diagnostics. Herald of National University "Lvivska Polytechnica" No. **464**, 31–43 (2002) (Lviv, Ukraine, in Ukrainian)
22. Hodych, O., Nikolskyi, Y., Pasichnyk, V., Shcherbyna, Y: Analysis and comparison of SOM-based training algorithms. Control Systems and Machines No. **2**, 63–80 (2006) (Kyiv, Ukraine, in Ukrainian)
23. Hodych, O., Nikolskyi, Y., Pasichnyk, V., Shcherbyna, Y: High-dimensional data structure analysis using self-organising maps. CAD Systems in Microelectronics, CADSM apos;07. 9th International Conference, 218–221 (2007)
24. Hodych, O., Hushchyn, K., Nikolski, I., Pasichnyk, V., Shcherbyna, Y.: SOM-based dynamic image segmentation for sign language training simulator. In: Wil van der Aalst (eds.) Information Systems: Modeling, Development, and Integration, pp. 29-40. Springer, Heidelberg (2009)
25. Hodych, O.V., Davydov, M.V., Nikolski, I.V., Pasichnyk, V.V., Scherbyna, Y.M.: Ukrainian Sign Language: the aspects of computational linguistics. Piramida, Lviv (2009) (in Ukrainian)
26. Hoffmann G.: CIE Color Space, 2000, http://www.fho-emden.de/~hoffmann/ciexyz29082000.pdf. Cited 15 Apr 2011
27. Hoffmann G.: CIELab Color Space, 2003, http://www.fho-emden.de/~hoffmann/cielab03022003.pdf. Cited 15 Apr 2011
28. Hunt R.W.G.: Measuring Colour. 3rd edition, Fountain Pr Ltd (2001)
29. Jacox, E.H., Hanan S.: Metric space similarity joins. ACM Trans. Database Syst. **33**(2), Article 7 (2008)
30. Jander, M., Luciano, F.: Neural-based color image segmentation and classification using self-organizing maps, 1996, http://mirror.impa.br/sibgrapi96/trabs/pdf/a19.pdf. Cited 15 Apr 2011
31. Jiang, Y., Chen, K.-J., Zhou, Z.-H.: SOM Based image segmentation. *Lecture Notes in Artificial Intelligence* **2639**, pp. 640–643. Springer, Heidelberg (2003)
32. Kohonen, T.: Self-Organizing Maps. 3rd edition, Springer, Berlin (2001)
33. Lawrence, R. D., Almasi, G.S., Rushmeier, H.E.: A Scalable Parallel Algorithm for Self-Organizing Maps with Applications to Sparse Data Mining Problmes. http://www.research.ibm.com/dar/papers/pdf/scalableSOM.pdf (Cited 15 Apr 2011)
34. Rauber, A., Tomsich, P., Merkl, D.: parSOM: A parallel implementation of the self-organizing map exploiting cache effects: making the som fit for interactive high-performance data analysis. Neural Networks **6**, 61–77 (2000)
35. Reyes-Aldasoro, C.C.: Image Segmentation with Kohonen Neural Network Self-Organising Maps, 2004, http://www.cs.jhu.edu/~cis/cista/446/papers/SegmentationWithSOM.pdf (Cited 15 Apr 2011)
36. Tominaga, S., Takahashi, E. Tanaka, N.: Parameter estimation of a reflection model from a multi-band image. Proceedings of the 1999 IEEE Workshop on Photometric Modeling for Computer Vision and Graphics (1999)
37. Wu Y., Liu, Q., Huang, T.S.: An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization. In: Proc. Asian Conf. on Computer Vision, Taiwan (2000)
38. Zahedi, M., Keysers, D., Deselaers, T., Ney, H.: Combination of tangent distance and an image distortion model for appearance-based sign language recognition. Pattern Recognition, pp. 401–408. Springer, Heidelberg (2005)

39. Zieren, J., Kraiss, K.-F.: Non-intrusive sign language recognition for human-computer interaction. Proceedings of the 9th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems, Atlanta, Georgia (2004)
40. Zieren, J., Kraiss. K.-F.: Robust person-independent visual sign language recognition. Proceedings of the 2nd Iberian Conference on Pattern Recognition and Image Analysis IbPRIA 2005, June 7-9, Estoril, Portugal. *Lecture Notes in Computer Science* **3522**, Springer, Heidelberg (2005)
41. IBM Research Demonstrates Innovative 'Speech to Sign Language' Translation System, Pressrelease, 12 Sep 2007, `http://www-03.ibm.com/press/us/en/pressrelease/22316.wss`. Cited 15 Apr 2011
42. The iCommunicator User's Guide, 2005, `http://www.myicommunicator.com/downloads/iCommunicator-UserGuide-v40.pdf`. Cited 15 Apr 2011
43. Senthilkumaran, N., Rajesh, R.: A study on rough set theory for medical image segmentation. International Journal of Recent Trends in Engineering **2**(2), 236–238 (2009)