Open access • Journal Article • DOI:10.1109/MMUL.2009.41

A Folk Song Retrieval System with a Gesture-Based Interface — Source link

Attila Licsár, Tamás Szirányi, László Kovács, Balázs Pataki

Institutions: University of Pannonia, Hungarian Academy of Sciences

Published on: 01 Jul 2009 - IEEE MultiMedia (IEEE Computer Society)

Topics: Gesture

Feature Article

A Folk Song Retrieval System with a Gesture-Based Interface

Attila Licsár
University of Pannonia

Tamás Szirányi, László Kovács, and Balázs Pataki
Computer and Automation Research Institute of the Hungarian Academy of Sciences

This article describes how a folk song retrieval system uses a gesture-based interface to recognize Kodály hand signs and formulate search queries.

Information technologies can open new avenues for preserving and circulating of culture. This article presents a folk song search-and-retrieval system that relies on a gesture-based interface. The system preserves both the multimedia content (the folk songs) and the core part of the Kodály music-teaching approach, which is a hand sign-based system of naming the notes of a musical scale by syllables (that is, do, re, mi, fa, sol, la, ti) instead of letters (see ''The Kodály Approach'' sidebar). One goal of this system is to demonstrate that this type of user interface can serve as a new option for human–computer interaction while also helping to maintain the digital future of a powerful teaching method that has proved its success worldwide.

Our system uses the Tillarom archive, which is a comprehensive collection of Hungarian folk songs collected during the last centuries. In 1896, Béla Vikár was the first person in Europe to use a phonograph to collect folk songs. Béla Bartók, Zoltán Kodály, and others continued this type of work in Hungary, building one of the most comprehensive folk song collections recorded on wax cylinders. They recorded more than 4,500 wax cylinders during the first years of the 20th century, the last period of living folk art. The Tillarom archive contains a selection of folk songs from this collection and from other collections of the last century. The Music Institute of the Hungarian Academy of Sciences created a professional catalog of bibliographic records based on the organization's unique research in clustering methods of folk songs melodies. Hungarian folk songs can be clustered into more than 2,300 different types.

The Tillarom archive provides different types of search-and-retrieval interfaces. For example, users can search folk songs via a traditional interface based on metadata as well as the full text of folk songs together with the original recordings. In addition, the archive presents the different types of folk songs as MIDI recordings, which presents the opportunity of developing MIDI and melody-based search-and-retrieval functions. Moreover, a sequence of notes in the usual music notation can define the retrieval target, which opens the way to create a hand-gesture-based interface for this multimedia archive. In special circumstances, such as in noisy rooms, voice commands for human–computer interaction aren't effective and gesture-based controls are more suitable.

Human gestures can be used as a query interface in Web-based search-and-retrieval systems. The decreasing prices of hardware devices have made it possible for more people to own, for example, webcams, to facilitate this technology. The performance of standard computers makes the application of computer-vision techniques possible at home as well. Computer-vision-based methods have the advantage over other approaches in that users don't have to carry any special equipment with physical sensors, such as a data glove. In this article, we present a general framework for gesture-based communication between a Web browser and a standalone application. This framework enables the integration of any type of human gestures into Web-based applications, opening up the possibility for new types of input interfaces.

1070-986X/09/$26.00 © 2009 IEEE Published by the IEEE Computer Society

The Kodály Approach

Named after Hungarian composer and teacher, Zoltán Kodály, the Kodály method uses hand signs to indicate vocal pitch (see Figure A). Though the basic concept of using gestures to represent notes is ancient, during the 20th century the concept was formalized as a standard teaching method. John Curwen largely defined this method. This approach treats melodies as a sequence of musical notes represented by hand signs.

Gestures are effective as a pedagogical tool because they visually reinforce the high and low and intervallic relationship between the pitches being sung. This reinforcement makes the technique effecting for developing music reading skills and as a mnemonic device for training singers. Furthermore, the listening and movement activities, as a part of the Kodály approach, offer opportunities for the development of children's perceptual function, concept formation, and motor skills. Furthermore, the technique also helps improve the intonation and pitch accuracy of musical tones.

By teaching the Kodály philosophy, principally at the beginning of a music education, the program comfortably leads a child into his or her school music program. Using hand signs helps the teacher know what the student is singing when the environment (such as the classroom) is noisy.

The books of songs and exercises, collectively known as the *Kodály choral method* (although Kodály himself never fashioned any method), have transformed music education in Hungarian schools and made their mark on the country's musical and educational institutions at all levels.[1] Kodály's concepts of music education have been adopted with great success by many schools in the US, Canada, Japan, Argentina, the Baltic States, and Carpatho-Ukraine.[2]

The numerous universities and conservatories basing their training on Kodály's relative sol-fa system, as well as the Kodály institutes founded all over the world—in Kecskemét, Hungary; Tokyo, Boston, Ottawa, and Sydney—further prove the popularity of Kodály's approach. Kodály hand signs were also used in the film *Close Encounters of the Third Kind* (see http://www.imdb.com/title/tt0075860/trivia), where the music and also hand signs were the basis of the intergalactic communication between people and aliens.

References

1. *New Grove Dictionary of Music and Musicians,* Macmillan, 2001.
2. E. Szőnyi, *Kodály's Principles in Practice,* Corvina Press, 1973, pp. 70-71.

Figure A. Vocabulary of Kodály hand signs. (A. Licsár et al., "Tillarom: An AJAX Based Folk Song Search and Retrieval System with Gesture Interface Based on Kodály Hand,* Proc. Workshop Human-Centered Multimedia *© 2006 ACM; http://doi.acm.org/10.1145/1178745.1178760.)

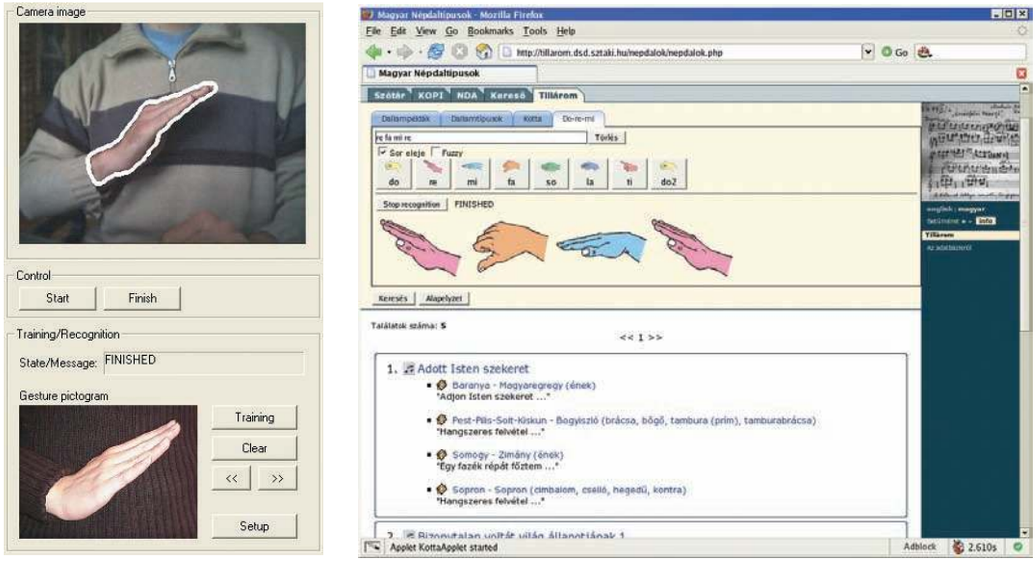
System fundamentals

Our proposed system contains a Web server with the Tillarom database, a standard PC with an Internet connection, and a simple video camera. The required software components on the user's PC are a Web browser and a standalone application running in Microsoft Windows. The software recognizes hand signs with computer-vision techniques, providing a gesture-based interface for the browser. After the query input, the browser retrieves the corresponding data from the digital archive.

Figure 1 shows the general camera configuration and the hand position in our system. Users perform gestures in front of the chest, as is consistent with the original requirements of this technique, which is natural and comfortable. The hand needs to be properly illuminated and located in the center of the image. Thus, gesture position information is not considered during the recognition process. We should note that the processing speed and the efficiency depend on the appropriate segmentation of gestures against the background. Sophisticated backgrounds (sharp textures, skin-colored patches, shadows, camera motion, or other objects) can be processed at a much higher computational power, but it's better to avoid them.[1]

Tillarom

Tillarom is implemented as an Asynchronous JavaScript and XML (AJAX)-based search application.[2] The software searches for folk song types and recordings using various search methods. These search methods can be plugged



(a) **(b)**

Figure 1. System configuration: (a) user interface of the gesture-recognition software and (b) Tillarom user interface with gesture retrieval displayed in a Web browser.

into the application, appearing as a separate tab in the GUI (as shown in Figure 1). The system currently supports several types of search:

- song recordings (using their associated metadata fields),
- song types (using their associated metadata fields),
- score (visually editing score in a score writing applet), and
- do-re-mi (entering do-re-mi signs using buttons or as hand gestures via video).

The search-interface forms for each search method are dynamically loaded on demand using JavaScript's XmlHttpRequests. Searches and details of folk songs presented in modal dialogs also use AJAX technologies to enhance the user experience. At the backend of Tillarom, there is a MySQL database containing metadata of folk song types, recordings, and their connections.

As the result of search using any of the supported methods, the user receives a list of matching folk song types. By clicking the song title, the metadata associated with the song is displayed together with its score. For folk song types, besides descriptive information (for example, the number of syllables in the song's first line), the score and MIDI are available; while for recordings, the actual MP3 and illustrative pictures together with their metadata (recorder, performer, recording date, collection place and geographical area, and so on) are available.

The database contains 2,291 types of folk songs. A folk song type can have various versions because the same song might have been sung with different lyrics or with some minor modifications in their tune in various parts of Hungary and the Carpathian Basin (Romania, Slovakia, Ukraine, Slovenia, and Austria). For each folk song type we have the MIDI and the internal representation of the songs, and for 611 song types we have 1,834 MP3 samples in the Tillarom database.

Gesture recognition client software

The client software has to be manually installed on the user's machine. We designed the user interface and software to meet requirements related to clarity of function and ease of use. The user interface provides only minimal functionality to help the user quickly survey and understand the program's operation. These functions include the following:

- The hand's relative position in the camera's view is continuously shown. In the configuration-and-recognition phase, the user can adjust and permanently supervise the hand's position.
- The contour line of the segmented hand is drawn on the camera image. Hence, the system can verify the segmentation algorithm

and if the result isn't satisfactory it can tune the segmentation parameters.

- The pictogram of the detected hand sign is displayed, letting the system validate the recognition's accuracy.
- The current state of the recognition and hints for the user are displayed.
- Several control buttons that handle training, segmentation, and recognition functions are provided.

The program involves configuration, preliminary training, and the recognition phase. In the configuration phase, the user positions the camera and adjusts the segmentation parameter. Before using the software for the first time, the user needs to train the application. The user chooses the musical note to be trained and then performs the related gesture, while the system automatically grabs the sample images and automatically saves them along with the classification models parameters. If the recognition rate of gestures decreases during use, the user can repeat the training by collecting new gesture samples.

The system captures and processes gesture images at a resolution of 320 × 240 pixels at about 15 frames per second with nonoptimized C++ code. The hardware environment is a 2.4-GHz Pentium processor and a conventional webcam. The minimal pixel size of the hand can be 20 percent of the camera image size.

Data interface

Tillarom's search engine and the gesture recognition application are two separate programs in our current prototype. To search for the matching folk song types, we needed to bridge the two systems, as Figure 2 depicts.

Our prototype provides a gesture-recognition terminal, which involves three applications:

- a browser with the Tillarom service loaded,
- the gesture-recognition application, and
- an Apache process and PHP application that provides the gesture information for the Tillarom Web server

The gesture recognition application continuously writes codes of recognized gestures into a specific file in the file system. A small PHP application running in the recognition terminal for the Tillarom service via a simple URL call provides this gesture file to the server. Moreover, to make the actually recognized hand signs appear in the users' browser, the Tillarom AJAX client periodically polls the Tillarom server for updated hand sign codes.

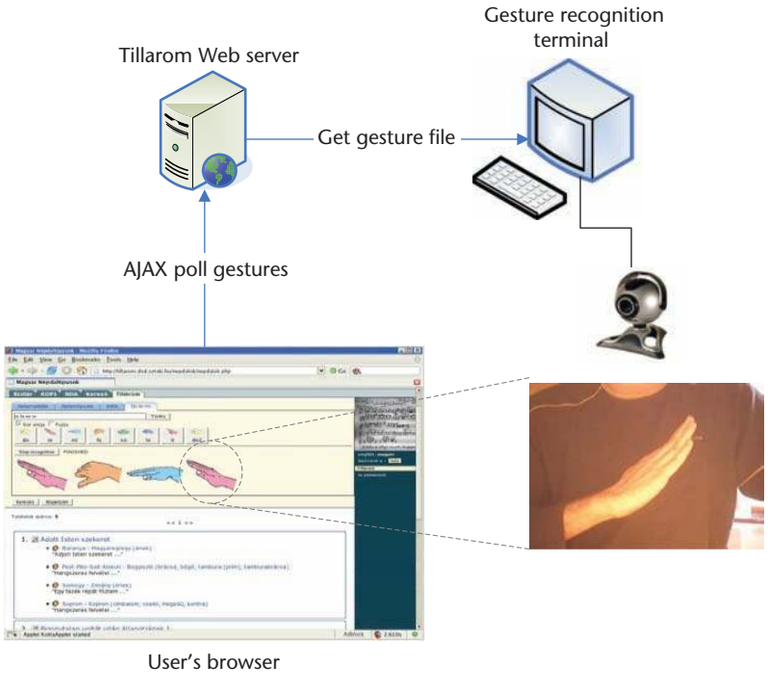


Figure 2. Architecture of Tillarom system. (A. Licsár et al., "Tillarom: An AJAX Based Folk Song Search and Retrieval System with Gesture Interface Based on Kodály Hand, **Proc. Workshop Human-Centered Multimedia** *© 2006 ACM; http://doi.acm.org/10.1145/1178745.1178760.)*

In our actual prototype setup, a search involves the following actions:

1. The user loads Tillarom in his or her browser and starts the gesture-recognition application and the Apache/PHP gesture service.
2. In the Tillarom page, the user clicks "start recognition." This makes the Tillarom AJAX client in the user's browser poll for gesture data generated by the recognition application via the Tillarom server.
3. The user makes the gestures in a webcam, which are recognized by the gesture-recognition application.
4. As each hand sign is recognized, the pictogram of the hand sign appears in the Tillarom page.
5. When the user finishes the hand-sign recognition, he or she then clicks "search" in

the Tillarom page. The application then executes a do-re-mi search based on the hand sign inputs.

Search engine

The Tillarom database contains folk songs in MIDI and in an internal representation. The internal representation is derived from the MIDI files. In this representation, we assigned a number to each note, starting from great C at 0 and then incremented by one each note in half note steps (C = 0, C# = 1, D = 2, D# = 3, and so on). Thus, notes in the score snippet in Figure 3 are represented in Tillarom with the numbers 14, 19, 18, 19, 23, 19.

Based on this numerical scheme, we created six representations for each score:

- score without rhythm,
- score with rhythm,
- score with rhythm only,
- score with tune only,
- difference, and
- simple difference.

In the score without rhythm representation, only the numbers assigned to each note (the pitch value) are present. This format is used when searching songs while ignoring rhythm. Taking our example from the Figure 3, this represents the score as ''14, 19, 18, 19, 23, 19.''

The score with rhythm representation converts each note to a 32nd note. The number representing the pitch of the note is repeated as many times for as many 32nd notes the given note represents. A quarter note (one fourth the duration of a whole note), for example, is represented by writing the numeric pitch value four times, one after the other. Taking the previous example, the internal representation with rhythms looks like this: ''14 14, 19 19, 18 18, 19 19, 23 23 23 23, 19 19 19 19.''

In the rhythm-only case, only the rhythms are represented with a number assigned to the length of each note. A whole note is represented as ''1,'' a half note as ''2,'' a quarter note as ''4,'' and so on. Taking our example, it looks like this: ''8 8 8 8 4 4.''

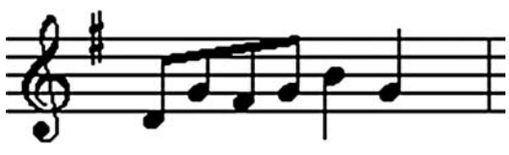


Figure 3. Example score snippet.

A score with tune alone is similar to the score without rhythm representation, with the difference being that repeating notes are ignored and are represented with a single numerical value. In our example, there are no consecutive repeating notes, so it would be the same as the score-without-rhythm example.

The difference representation provides only the numeric differences between each note of the song. This representation ignores rhythm. In our example snippet, difference numbers are ''5 −1 1 4 −4 −1.''

The simple difference representation is used to show whether a note following another note is higher, lower, or the same. Change to a higher note is represented with a 1, change to a lower with a −1, and no change with a 0. In our example, simple difference numbers are ''1 −1 1 1 −1 −1.''

When searching for songs by score (using the score-writing applet in Tillarom), we use the first four representations based on user-defined search criteria. We use the difference and simple difference representations to implement a do-re-mi search. The well-known C, D, E, F, G, A, B naming scheme in scores provides an absolute representation of notes. The do-re-mi scheme, however, is a relative representation, which means that ''do'' doesn't have to match the note C, but can match E, for example. In this case ''re'' would be matched to F, ''mi'' to G, and so on.

This means that when we search using solmization signs, we should only take into account the pitch difference of notes rather then their absolute values. We use the difference representation for this purpose, while the simple difference provides a kind of fuzzy or contour-search facility still using solmization hand signs. Parsons showed that this simple encoding of tunes (up, down, same), which ignores most of the information in the musical signal, can still provide enough information to make a distinction between a large number of tunes.[3]

The do-re-mi search method provides a much simpler, yet less accurate way of searching, than the score-writing applet. When searching with hand signs, half notes can't be provided and rhythm is ignored as well. There is also a

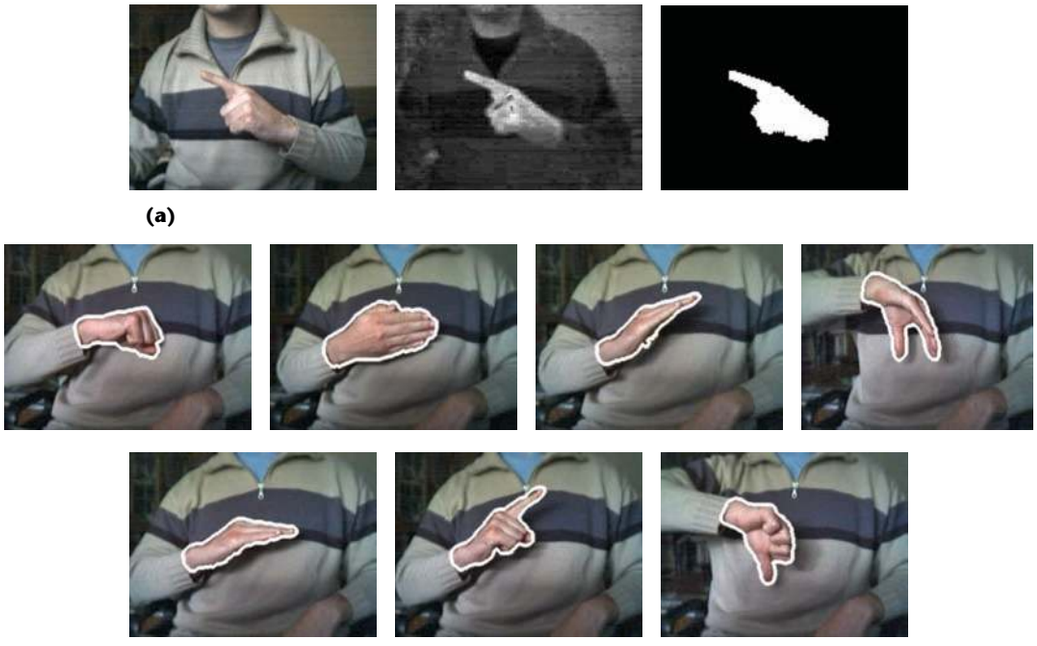


Figure 4. (a) Camera image (left), I-component image of YIQ color space (middle), and segmented and labeled hand blob (right). (b) Camera images with contours of segmented gestures outlined by white boundaries.

limitation that notes can be entered in one octave at once. These limitations, however, don't really affect the use of do-re-mi based search, because most old-style folk songs in our database are pentatonic (five-note scale and music). This means that not all the seven supported solmization notes are required, only five of them for a specific song can suffice. Additionally, it also indicates that searching without rhythm has little impact on search result accuracy.

The Tillarom do-re-mi search algorithm allows some errors in the note-input sequence and helps find songs that are the closest matches to the input. To achieve this functionality, we use Levenshtein distance calculation.[4] The Levenshtein algorithm (also called *edit-distance*) calculates the least number of edit operations necessary to modify one string to obtain another string. To achieve this, we convert the note difference representation of every song to a string and compare the representation with the user's do-re-mi input also converted to a string. The Levenshtein calculation gives a percentage value of how similar the user's input is to a specific song's notes. The higher this value, the closer it matches. With this approach, we are able to correct one error (one note missing, one extra note added, or one wrong note specified) of the user's input.

Hand sign recognition

There are two main problems in the recognition of Kodály hand signs: spatial and temporal segmentation. Spatial segmentation entails separating points belonging to the hand object from the points of the background or other objects, such as the head. When the user performs several musical notes with Kodály signs, two consecutive gestures are isolated in time by the arm's movement. A hand signal is recognized as a musical note when the user holds the hand after a movement, resulting in temporal segmentation. After the segmentation processes, our appearance-based recognition method analyzes static gestures by extracting the hand blob contour. The standard vocabulary of solmization ensures the global usage of the proposed system without gesture definition and training problems.

Spatial segmentation

Our previously described configuration involves dynamic backgrounds where the user performs the gesture in the middle of the image. We made some constraints to ensure the correct detection of the hand silhouette and applied skin-color-based detection of the hand. The camera image is converted to YIQ color space because the ''I'' component is sensitive to skin colors (see Figure 4).[5] Intensity

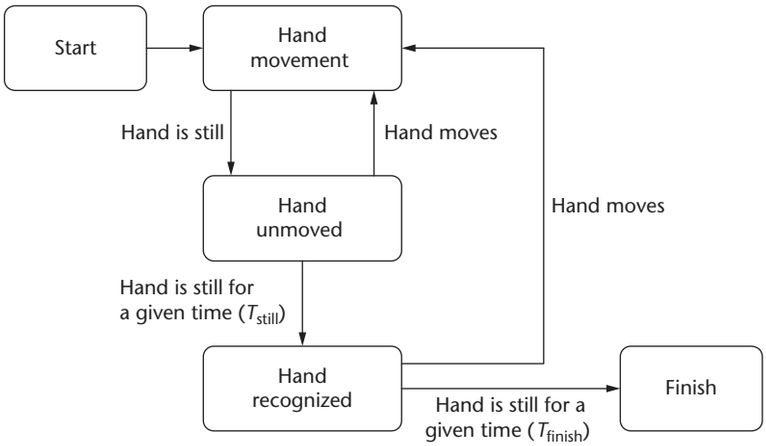


Figure 5. Finite state machine model: temporal segmentation of gestures. (A. Licsár et al., "Tillarom: An AJAX Based Folk Song Search and Retrieval System with Gesture Interface Based on Kodály Hand,* Proc. Workshop Human-Centered Multimedia *© 2006 ACM; http://doi.acm.org/10.1145/1178745.1178760.)

thresholding of the channel "I" detects the mask of the skin color pixels:

$$\Phi(\overline{z}) = \begin{cases} 1: & I(\overline{z}) > \Theta \\ 0: & \text{otherwise} \end{cases}$$
$$I(\overline{z}) = 0.5957^{*}\mathrm{R}(\overline{z}) - 0.2745^{*}\mathrm{G}(\overline{z}) - 0.3212^{*}\mathrm{B}(\overline{z}) + 151.9 \quad (1)$$

where $I,R,G,B \in [0,255]$ denotes the related channel of YIQ and RGB color spaces at a given position z, and Θ is the threshold value. We set the value of Θ in our experiment between 155 and 210 depending on the lighting conditions. The user should wear a long-sleeved shirt to exclude the lower arm from detection (as shown in Figure 4). Gestures are performed by placing the hand in front of the clothing, which shouldn't have skin-like colors.

After this step, the segmented image can involve several objects that have skin-like colors, such as head, other hand, or any object in the background (see left and middle images in Figure 4a). Our blob-labeling method filters these objects by their position and perimeter and selects the hand blob (see right image in Figure 4a).

According to the system configuration, a blob is selected as a hand silhouette if it appears closest to the center point of the camera image and its perimeter is greater than the half of the height of the image (in our experiments it was 120 pixels).

Temporal segmentation

The segmentation of consecutive gestures entails detecting a moving or unmoved hand. We realize the temporal segmentation with a finite state machine (see Figure 5). The possible states are hand movement, hand unmoved, hand recognized, start, and finish. The system only recognizes a gesture as a musical note if the hand is still for a predefined period (T_{still}). When the performed sign is recognized as a note, the system stores it (the hand-recognized state in Figure 5). The next note is only detected when the state of the recognition turns into hand movement by the detection of the hand movement.

The input mechanism can be stopped when the recognition is in the hand-recognized state but the user's hand is still for a given period (T_{finish}). The current state and the actually recognized note are displayed on the user interface (see the state and pictogram sections shown in Figure 1) giving continuous feedback for users. Users can delete falsely recognized notes with a button in the GUI displayed in the Web browser.

For example, if the user wishes to perform the melody "re, re, do", then the he or she would

1. perform the sign "re" for a predefined length of time (T_{still}) without any movement,
2. move his or her hand without any changes in hand configuration,
3. keep the position of the hand still for a given time (T_{still}),
4. move his or her hand and perform gesture "do," and
5. hold his or her hand in a fixed position to allow for recognition.

The client application displays hints to the user about the possible transitions between recognition states. For example, if the hand is still and the recognition is in the hand-unmoved state, the system indicates that it is waiting for the recognition of a musical note. These hints are designed to help the user with the application and give continuous feedback for direct manipulation. Our method continuously analyses hand movement by function *Move(t)*:

$$Move(t) = \begin{cases} 1: & S_t > \Omega \\ 0: & \text{otherwise} \end{cases}$$
$$S_t = \max\left[\underset{i=0}{\overset{N}{median}}(|v_{t-i} - v_{t-i-1}|), \underset{i=0}{\overset{N}{median}}(|h_{t-i} - h_{t-i-1}|)\right]$$

where S calculates the maximum of horizontal and vertical velocities in a given period. The hand moves in frame t if $Move(t) = 1$. The value N denotes the time window's length of the temporal analysis, v_t and h_t give the vertical and horizontal position of the center of the segmented hand silhouette in frame t, and Ω is a threshold value to detect movement. Their median is calculated for filtering the detection noise due to failures in determining the spatial segmentation.

In our experiments, we tested different parameters: $T_{\text{still}} = 1.5$ seconds and $T_{\text{finish}} = 2$ seconds (see Figure 5). These intervals offered enough time for users to perform gestures and weren't too long to cause fatigue. We set the threshold $\Omega = 4$ and the size of the temporal window as $N = T_{\text{motion}} * FPS$, where $T = 0.5$ seconds and FPS is frames per second and gives the actual processing speed of the system.

Hand signs recognition

The system extracts the contours of the segmented hand blobs used for the hand sign classification and generates a sequence of complex numbers, denoted by $S_k = m_k + j * n_k$, where m and n are vertical and horizontal coordinates of boundary points and k is the position in the sequence. Then the software calculates the Fourier descriptors of the complex sequence by discrete Fourier transform (DFT) and constructs descriptors as the magnitude of DFT coefficients, denoted by $|D_k|$:

$$F_k = \frac{|D_{k+2}|}{|D_1|}$$

where F_k (k = 0, 1, 2, 3, ...) gives the kth Fourier descriptor. D_0 is discarded from the computation to remove positional information of the hand while $|D_1|$ is used to remove scale sensitivity. The magnitude of DFT coefficients ensures that descriptor is rotation invariant. We used the first 50 descriptors in the classification process so high frequencies are removed to decrease noise sensitivity.

A support vector machine (SVM) carries out the classification by the selected feature vectors.[6] The classifier has 50 input parameters and returns the identifier of the recognized gesture sign. The SVM kernel is a radial basis function:

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$$

where x is the training samples and the gamma parameter is determined as 0.125 by our experiments.

In Figure A in "The Kodály Approach" sidebar, we can see that that the silhouette of sign "re" and "mi" are similar but their orientations are different. The angles of "re" is about 45 and "mi" is approximately 0 degree compared to the horizontal. For this reason if "re" or "mi" notes are detected by our SVM classifier, then a second classification step is needed by their orientation. This orientation is automatically calculated by Horn moments from the contour of the silhouette.[7] In the second classification step, the gesture is recognized as "re" if the calculated orientation is above 15 degrees.

Results and experiments

We tested the usability and the performance of our system with 10 users. These people were selected from colleagues and friends who weren't involved in the development of Tillarom and have at least basic computer skills. Because in Hungary everyone learns solmization in primary schools, all of these users except for one (who learned the tool in about 10 minutes) were familiar with do-re-mi hand signs. We performed the experiments in three rooms equipped with different hardware configurations (camera and computer).

Typically, users want to apply the gesture-based interface as soon as possible after the installation of the software. Because the hand silhouette depends on the hand physiology of different users and because the sleeve length of the clothes could also modify the contour, preliminary training is necessary. In our first experiments, users manually collected each training sample by making gesture snapshots.

To avoid uncomfortable and slow manual control of the training process (training took more than five minutes), we developed a method that automatically acquires samples from the camera input. The user selects the sign to be trained and then performs it for five seconds while the system grabs 10 sample images. The software indicates the starting time and completion of the sampling period. This training is effective as it's simple and quick (taking about one to two minutes).

In our experiment, each user trained the system with 280 samples of Kodály signs (40 samples per musical note). The testing set involved 3,318 samples. The average recognition rates of musical

Table 1. Average recognition rates (%) summarized in a confusion matrix: rows indicate signs to be performed, while columns depict the distribution of recognition results.

Signs	Do	Re	Mi	Fa	Sol	La	Ti
Do	**88.8**	0	1.4	2.0	3.5	1.7	2.7
Re	3.7	**95.5**	0	0.2	0.3	0.2	0
Mi	2.5	0	**96.7**	0	0	0	0.8
Fa	8.7	0	0	**87.9**	0	3.1	0.2
Sol	5.8	0	6.8	0	**87.4**	0	0
La	0.2	1.7	0	0	0	**89.1**	9.1
Ti	10.4	0	0	8.5	0	0.3	**80.7**

notes are summarized in a so-called confusion matrix (see the bolded diagonal in Table 1).

The recognition rates of gestures were above 87.4 percent except for the note ''ti,'' which was frequently misrecognized as ''do'' and ''fa.'' This misrecognition could be caused by the fact that contours of gestures ''fa'' and ''ti'' are similar (see Figure A in ''The Kodály Approach'' sidebar). If the recognition of a given sign is not sufficient, then the user can complement the database of training samples with more patterns to improve the classification performance.

Furthermore, if the spatial segmentation is incorrect, then the user can improve it by the proper adjustment of the segmentation parameter (see Θ in Equation 1). From our 10-subject test group, two persons adjusted the segmentation parameter and two used the collection of additional gesture samples to improve the recognition efficiency. After these steps, the system repeated the performance analysis on the previously used testing set (see Table 2).

Following the training, the performance of the recognition of gestures ''ti'' and ''fa'' considerably increased and recognition rates of all notes were above 89.3 percent. We observed that the most important factor in the training phase was the fast and efficient gathering of representative samples. We also observed that our method could improve the recognition rate by following the preliminary training phase with an additional training phase.

Table 2. The average recognition rates of musical notes after adjustment of the segmentation parameter and improvement of the training set.

Gesture	Recognition rate (%)
Do	94.8
Re	99.2
Mi	94.3
Fa	89.3
Sol	95.2
La	93.9
Ti	95.4

In addition to conducting a training phase prior to users gaining access to the full system, we evaluated the efficiency of our temporal segmentation method. In our test, each user performed 40 gestures, one after the other, to create a melody consisting of 40 musical notes. The aim of the segmentation was the correct detection of the point of time when the user starts to perform the consecutive musical note of the melody.

In the experiment, the correct recognition rate of the temporal segmentation was 90 percent when users didn't receive any feedback about the state of the temporal segmentation. The rate of the false negative recognition was 10 percent, which means that the actual gesture wasn't recognized as a new musical note. The reason for this issue is the user didn't hold his or her hand in place for the necessary period (value T_{still}), and the finite state model didn't get to the hand-recognized state. There was no false positive recognition, which indicates that the actual gesture was erroneously detected as a consecutive musical note.

In practice, when false negative detection occurred, the user could see on the user interface that the system hadn't recognized the performed gesture. In these cases, the user would repeat the input and hold the hand still for a longer period, until the system could recognize the actual hand sign. Consequently, the continuous interaction between user and computer is substantial during the recognition process.

The Tillarom database contains approximately 9.4 notes per row, and each song involves about four lines. From our experiments, we observed that it was enough to search among the notes of the first line of each song. Because the Levenshtein distance is able to correct one wrongly specified note in the query, and the average recognition rate is about 90 percent, we found the matching process to be error tolerant.

Hand Gestures in Multimedia Applications

Appearance-based systems have two categories: posture- and motion-based recognition. Posture-based recognition handles the characteristic features of the hand's configuration without analyzing its movements. Motion-based recognition recognizes the movement trajectory and/or hand configuration changes in time. Posture based systems use static hand gestures, while motion-based systems use dynamic gestures.

Early research focused on dynamic gestures that specify commands by simple drawings made by mouse or pen.[1] These systems recognized objects by statistical pattern recognition, such as Rubine's single-path algorithm.[2] Mouse gesture approaches analyze the movements and events of a computer mouse that the computer recognizes as specific commands.[3] These approaches provide access to common functions and are designed to help people who have difficulty with typing on a keyboard. Typical applications of this technique include Web browsers, such as Opera or Firefox, where the user can navigate and perform complex tasks without using a keyboard, menus, or toolbars.

Computer-vision research has analyzed the movement and the configuration of human gestures. Freeman, for example, used computer-vision techniques to find the user's open hand from across the room to control a television.[4] This approach applied the detected hand sign's position on the camera image to control the cursor on the screen. Other methods have focused on recognizing hand-movement trajectory, like those that form the basis of mouse gestures. Lee's method, for example, focused on recognizing alphanumeric characters and graphic primitives by the analysis of trajectories with hidden Markov models.[5] Lee used gestures to control a Microsoft PowerPoint presentation.

Spatiotemporal gesture models fuse static and dynamic gestures. These methods analyze the position and the configuration of the hand in time, performing complex gestures.[6] Ng, for example, tested gestures in a simulated desktop system, where the user manipulated windows and objects without using a mouse.[7] Gestures can also be used in virtual- or augmented-reality applications. Starner used gestures in a VR application using the arm's 3D position and direction to control objects.[8]

Static and dynamic gestures can be divided into two groups: for example, boundary-based methods involve edge-based contours;[9] while region-based techniques involve image moments[10] and image eigenvectors.[11] Other techniques use second-order moments. One example is Zernike's method, which relies on a recognition technique based on a shape's rotation.[12] Another method uses orientation histograms, which are invariant to lighting conditions but sensitive to rotation and contour scaling.[13] The disadvantage of invariant methods is the high computational cost because features are computed using the whole region of the given shape.

Boundary-based methods, such as Fourier descriptors,[14] use only contour points and tend to result in a more efficient feature extraction and low computational cost. Boundary analysis of gestures by Fourier descriptors realizes translation, rotation, and scale invariant descriptors. Taking these facts into consideration, we have chosen the boundary-based method because it's simple and efficient and thus suitable for real-time applications.

References

1. M.L. Coleman, ''Text Editing on a Graphic Display Device Using Hand-Drawn Proofreader's Symbols,'' *Proc. 2nd Univ. of Illinois Conf. Computer Graphics,* Fairman & Nievergelt, 1969, pp. 283-290.
2. T.R. Henry, S.E. Hudson, and G.L. Newell, ''Integrating Gesture and Snapping into a User Interface Toolkit,'' *Proc. ACM Siggraph,* ACM Press, 1990, pp. 112-122.
3. M.S. Dulberg, R. St. Amant, and L.S. Zettlemoyer, ''An Imprecise Mouse Gesture for the Fast Activation of Controls,'' *Proc. Int'l Conf. Human–Computer Interaction,* IOS Press, 1999, pp. 375-382.
4. W.T. Freeman and C. Weissman, ''Television Control by Hand Gestures,'' *Proc. Int'l Workshop Automatic Face and Gesture Recognition,* MIT Press, 1995, pp. 179-183.
5. H. Lee and J. Kim, ''An HMM Based Threshold Model Approach for Gesture Recognition,'' *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 10, 1999, pp. 961-973.
6. F.S. Chen, C.M. Fu, and C.L. Huang, ''Hand Gesture Recognition Using a Real-Time Tracking Method and Hidden Markov Models,'' *Image and Vision Computing,* vol. 21, no. 8, 2003, pp. 745-758.
7. C.W. Ng and S. Ranganath, ''Real-Time Gesture Recognition System and Application,'' *Image and Vision Computing,* vol. 20, no. 13, 2002, pp. 993-1007.
8. T. Starner et al., ''The Perceptive Workbench: Computer-Vision Based Gesture Tracking, Object Tracking, and 3D Reconstruction for Augmented Desks,'' *Machine Vision and Applications,* vol. 14, no. 1, 2003, pp. 59-71.
9. K. Cho and M. Dunn, ''Learning Shape Classes,'' *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 9, 1994, pp. 882-888.
10. T. Starner and A. Pentland, ''Visual Recognition of American Sign Language using Hidden Markov Models,'' *Proc. Int'l Workshop Automatic Face and Gesture Recognition,* MIT Press, 1995, pp. 189-194.
11. K. Imagawa et al., ''Appearance Based Recognition of Hand Shapes for Sign Language in Low Resolution Images,'' *Proc. 4th Asian Conf. Computer Vision,* Springer, 2000, pp. 943-948.
12. J. Schlenzig, E. Hunter, and R. Jain, ''Vision Based Hand Gesture Interpretation Using Recursive Estimation,'' *Proc. 28th Asilomar Conf. Signals, Systems, and Computer,* vol. 2, IEEE Press, 1994, pp. 1267-1271.
13. W. Freeman and M. Roth, ''Orientation Histograms for Hand Gesture Recognition,'' *Int'l Workshop Automatic Face and Gesture Recognition,* IEEE CS Press, 1995, pp. 296-301.
14. C.T. Zahn and R.Z. Roskies, ''Fourier Descriptors for Plane Closed Curves,'' *IEEE Trans. Computers,* vol. 21, no. 3, 1972, pp. 269-281.

Conclusions

We've presented a new way to maintain the popularity of the Kodály method of teaching music. While our approach can be considered as a new way to protect the core of the Kodály approach, an important entity of our cultural heritage, it must be noted that this system could be used for the preservation of other dynamic entities of cultural heritage, such as dances, where human body movement could be applied as input for retrieving dance information.

In the future we plan to extend the proposed gesture-based interface with a voice input mechanism where users hum or sing the tune of the song. This multimodal interface could help users to formulate queries in the Web browser and/or to learn the Kodály Approach. **MM**

Acknowledgments

We acknowledge the support of Ferenc Liszt Academy of Music—Zoltán Kodály Pedagogical Institute of Music, Kecskemét, Hungary.

References

1. A. Licsár and T. Szirányi, ''User-Adaptive Hand Gesture Recognition System with Interactive Training,'' *Image and Vision Computing,* vol. 23, no. 12, 2005, pp. 1102-1114.
2. J.J. Garrett, *Ajax: A New Approach to Web Applications,* Adaptive Path, 2005; http://www.adaptivepath.com/ideas/essays/archives/000385.php.
3. D. Parsons, *The Directory of Tunes and Musical Themes,* Spencer Brown, 1991.
4. A. Apostolico, Z. Galil, and A. Apostolico, *Pattern Matching Algorithms,* Oxford Univ. Press, 1997, p. 337.
5. H. Lee and J. Kim, ''An HMM Based Threshold Model Approach for Gesture Recognition,'' *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 10, 1999, pp. 961-973.
6. K.R. Müller et al., ''An Introduction to Kernel Based Learning Algorithms,'' *IEEE Trans. Neural Networks,* vol. 12, no. 2, 2001, pp. 181-201.
7. B.K.P. Horn, *Robot Vision,* MIT Press, 1986.

Attila Licsár is a researcher at the Department of Image Processing and Neurocomputing at the University of Pannonia, Hungary. His research interests include human–computer interaction, hand-gesture analysis, and digital film restoration including correction of blotches and image vibration. Licsár has a PhD in information technology from the University of Pannonia. Contact him at licsara@almos.uni-pannon.hu.

Tamás Szirányi is a scientific advisor at the Computer and Automation Research Institute, Hungarian Academy of Sciences, where he is the leader of Distributed Events Analysis Research Group. His research interests include texture and motion segmentation, surveillance systems for panoramic and multiple camera systems, measuring and testing image quality, digital film restoration, Markov random fields and stochastic optimization, image rendering, and coding. Szirányi has a PhD in electronics and computer engineering and a D.Sci. in image processing from the Hungarian Academy of Sciences. He is the founder and past president of the Hungarian Image Processing and Pattern Recognition Society; associate editor of *IEEE Transaction on Image Processing*; and recipient of the 2001 Master Professor award. He is a senior member of IEEE and a Fellow of the IAPR and the Hungarian Academy of Engineering. Contact him at sziranyi@sztaki.hu.

László Kovács is the founder and head of the Department of Distributed Systems at the Computer and Automation Research Institute, Hungarian Academy of Sciences. His research interests include R&D of digital library and archiving systems, collaborative knowledge management systems, computer-supported cooperative Work, groupware, social computing, human–computer interaction, as well as philosophy of sciences. Kovács has a technical doctoral degree in software engineering from Budapest University of Technology and Economics. Contact him at laszlo.kovacs@sztaki.hu.

Balázs E. Pataki is a senior research associate of the Department of Distributed Systems at the Computer and Automation Research Institute, Hungarian Academy of Sciences. Pataki has a BS in informatics from Dennis Gabor Applied University. His research interests include innovative user interfaces, human–computer interaction, social computing, and groupware and awareness technologies. Contact him at pataki@sztaki.hu.