Solfège hand sign recognition with smart glasses

Gábor Sörös, Julia Giger, Jie Song

Department of Computer Science, ETH Zurich, Switzerland

Abstract. Hand gestures are widely used in music education. The applied gesture set is well defined since the XIX. century and contains a small number of gestures only. We present a fast and robust method for recognizing solfège hand signs with smart glasses in egocentric perspective. Our method achieves above 95% classification rate and close to real time performance running on an unmodified Google Glass device.

Keywords: smart glasses, gesture recognition, solfège, music education

1 Introduction

A widely used approach in music education is sight singing, where different hand signs are associated with different tones. Solmization is the system of assigning different syllables to each note in a musical scale, and these syllables depend on the geographical and linguistical environment. Solfège is one form of solmization that is practiced in Europe and most English-speaking countries. Solfège is based on the seven syllables `do`, `re`, `mi`, `fa`, `so`, `la`, and `ti`. The teaching method with hand signs was developed by Curwen and later extended by Kodály [1]. The gesture set contains a small number of gestures only (see Fig. 1) but is challenging enough to make heuristic recognition approaches fail.

Fig. 1. The solfège gesture set. Figure adapted from [2]

The egocentric perspective of smart glasses is advantageous for learning and practicing these hand gestures. Automatic recognition of the gestures can be used to play a virtual instrument or give feedback to the student in the learning process. However, real-time and robust gesture recognition on wearable computers is challenging due to the limitations in computing resources. We present a wearable implementation of the hand gesture classification method by Song et al. [3] and demonstrate it in recognizing solfège hand signs.

2 Method

The input to our method is a single binary image of the hand silhouette. The wearable camera captures an RGB image from which the user's hand is segmented based on typical hand colors. The segmented binary silhouette is downsampled and cleaned via connected-component analysis and compensated for rotations via principal component analysis. The resulting binary mask is fed into a random forest classifier. The forest classifies each hand pixel independently to one of the pre-defined classes or one of two other classes: `no gesture` or `noise`. To make our method robust against false classifications when the hand transitions between gestures, we add a `no gesture` class which contains hand postures different from solfège signs. Furthermore, to handle errors in the simple color-based silhouette segmentation, we intentionally inject labeled `noise` pixels in the training images.

We pass each pixel from the binary mask through the forest and we make binary decisions at each split node based on comparing the mask values at two offset positions. The offset positions at each split node are defined by learnt offset vectors. The leaf nodes of the forest contain learnt probability distributions over the set of class labels. The final label of a pixel is determined by averaging the probabilities from all trees, and the performed gesture is decided by a majority vote over all foreground pixels in the image. The output of our method is a cleaned binary mask and a per-pixel gesture label mask (see Fig. 2). After temporal and spatial filtering, the recognized gestures drive a state machine that injects various input events into other GUI elements shown to the user.

Our method trades storage space for speed. The size of our random forest is about 80 MB and the algorithm runs close to real time on a Google Glass Explorer Edition (TI OMAP4430 SoC). We use a single core for processing and the GPU for hand segmentation and image scaling. As most of the pipeline consists of pixel-wise computations, the algorithm is well suited for further parallelization. Our tree storage as a matrix and tree traversal through shifting array indices are also suitable for GPU implementation, where the trees can be stored as textures. We leave the full GPU implementation for future work.

3 Training and testing

To train the classifier robust to variation in hand shapes, sizes, and distances to the camera, we asked six persons to perform the gestures under natural variation and recorded short sequences of each. In total, we recorded 3700 images per gesture covering enough variation in rotation, depth and appearance. The training set also included the `no gesture` hand class where participants casually moved their hands in front of the camera. The background `noise` samples are generated by taking hand-colored pixels from a video where the camera is moving casually with no hand in the view. These binary noise samples are added to the clean labeled images with a `noise` label prior to training.

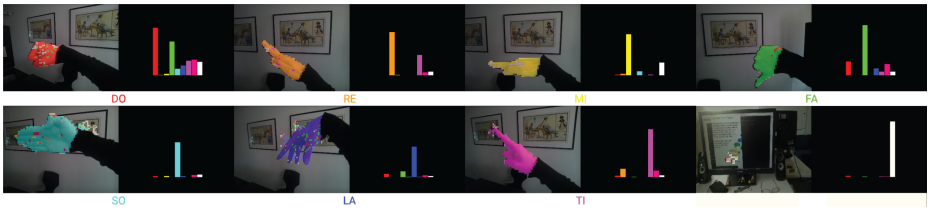

Fig. 2. Screenshots of our application running on a Google Glass. Each pixel in the image gets classified separately as one of the signs or noise and gets colored accordingly. The left hand side shows the RGB camera input and the pixel-wise classification results overlaid, while the right hand side shows the current probability distribution over all classes. The last screenshot shows how `noise` pixels are recognized.

Qualitative classification results are shown in Fig. 2. To also quantitatively test the classification forest, we used a test set consisting of 2300 images per gesture. Our method classified over 95% of the images correctly.

		predicted						
		do	re	mi	fa	so	la	ti
actual	do	96.405%	0.000%	0.000%	2.987%	0.332%	0.000%	0.277%
	re	0.000%	98.076%	1.176%	0.000%	0.214%	0.000%	0.534%
	mi	0.000%	0.000%	99.830%	0.000%	0.170%	0.000%	0.000%
	fa	0.000%	0.000%	0.000%	96.425%	0.000%	0.000%	3.575%
	so	13.608%	0.000%	0.000%	0.000%	84.734%	0.057%	1.601%
	la	0.215%	0.000%	0.000%	3.546%	0.000%	96.239%	0.000%
	ti	2.521%	0.000%	0.000%	0.000%	0.000%	0.000%	97.479%

Fig. 3. Confusion matrix of the forest for the Google Glass

4 Demo application

Built on top of our recognition pipeline, we present a solfège teaching app. The application shows a sheet of music, and the user has to practice the hand signs that reproduce that piece of music. The Glass application recognizes the gestures and gives positive or negative audio and/or visual feedback.

5 Acknowledgements

We thank Benjamin Hepp for his code for training random forests. We also thank Virág Varga from Disney Research Zurich for our inspiring discussions.

References

1. Choksy, L.: Kodály Method. Volume 1. Prentice Hall (1999)
2. Classics for Kids. www.classicsforkids.com
3. Song, J., Sörös, G., Pece, F., Fanello, S.R., Izadi, S., Keskin, C., Hilliges, O.: In-air gestures around unmodified mobile devices. In: Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology. UIST '14 (2014)