Real-Time Keyboard Transcription

Bilal Ahmad
Department of Electrical Engineering and Computer Science
Syracuse University

***Abstract*—The onset of the COVID19 pandemic has relegated traditionally in-person teaching methods to online communication. Remote music instruction can suffer from the audio quality of a played instrument being degraded by the video communication application. In this paper, I develop an algorithm that can transcribe in real-time the notes being played by a user using only video of the user playing a keyboard. For each frame of video, the location of the keyboard is identified and the pressed keys are determined by comparing the frame to a reference image of the keyboard. I test my method against other transcription techniques to compare accuracy and response times.**

***Index Terms*—Computer Music, AMT**

I. Introduction

I started playing the piano three years ago and regularly attend lessons every other week. In order to improve as a piano player it is important that others hear you play so that they can alert you to any errors that you are unaware of. With the onset of the quarantine orders in response to the global pandemic, my previously in-person piano lessons are now done over FaceTime. When doing piano lessons over FaceTime, the audio quality of played instruments is degraded. This degradation can make it difficult for an instructor to assess the tone and correctness of the notes being played. In order to solve this issue, I sought to develop a system that could in real-time determine what notes are being played using only the video of someone playing the keyboard.

The process of converting played music to symbolic notation (such as sheet music) is known as Automated Music Transcription (AMT). The majority of AMT work has focused on determining what notes are being played using the audio from a played instrument. Audio-based AMT methods have had difficulties transcribing polyphonic melodies, require intensive noise-reduction processing, and cannot reliably discern different octaves of the same note [1]–[3]. These issues, and the fact that the audio quality of played instruments over video communication applications is already degraded, make audio-based AMT an unfit choice for my goal of real-time, accurate note transcription. Instead, vision-based AMT systems are more apt for the task.

Within the last five years, vision-based AMT approaches have been proposed to be used in conjunction or instead of audio-based techniques for improved AMT performance [4],[5]. Fig 1 shows the camera setup for vision-based AMT approaches. A camera is placed adjacent to a keyboard at an angle between 30 and 85 degrees.

Fig. 1. Ideal Camera Setup for Vision-Based AMT [4]

The previous vision-based AMT approaches focus primarily on transcription accuracy and are applied to recorded video. The process intensiveness of the techniques used in the video-based approaches make them unsuitable for real-time transcription. For my approach, I aim to combine simplified steps from existing vision-based approaches into an algorithm that can perform in real-time while still maintaining an acceptable level of transcription accuracy.

II. Keyboard Note Detection

The note detection algorithm processes incoming frames from a video feed of a user playing the keyboard. The algorithm first determines the location of the keyboard in the video frame. Next illumination adjustments are made to account for environment differences between the current frame being processed and the previous frame. Finally, background subtraction techniques are used to determine which notes are being played in a frame. Data on which keys are being pressed each frame is recorded and saved off as a MIDI file at the end of processing. MIDI files can be converted to sheet music using commonly available applications. The algorithm sequence is shown in Fig 2.

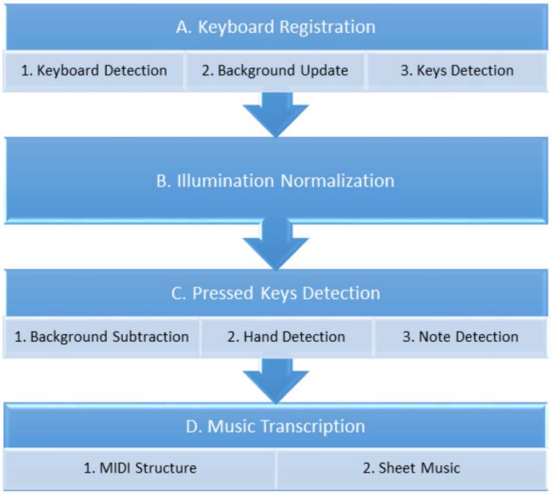

Fig. 2. Real-Time Vision-Based AMT Steps

A. Keyboard Registration

As with previous vision-based AMT approaches, the algorithm first determines the location of the keyboard using Hough line transforms (Fig 3). After the lines are found, the algorithm uses the intersection of the lines to determine potential quadrilaterals that might be the keyboard [4].

The image of the keyboard from the initial frame is saved as a reference image later used for background subtraction. In subsequent frames, after the keyboard location is determined, the algorithm determines whether the reference image should be updated. For example, say in the initial frame, parts of the keyboard are covered by the pianists' hands. If in a subsequent frame, there is less obstruction of the keyboard, the algorithm will determine that the current frame's keyboard will be better for background subtraction and update the reference image.

After the location of the keyboard is determined, a connected-component detection process is used to identify the location of the black keys. The white keys are difficult to separate using pixel gradients so their position is estimated using known information about key layouts and the distances between the identified black keys.

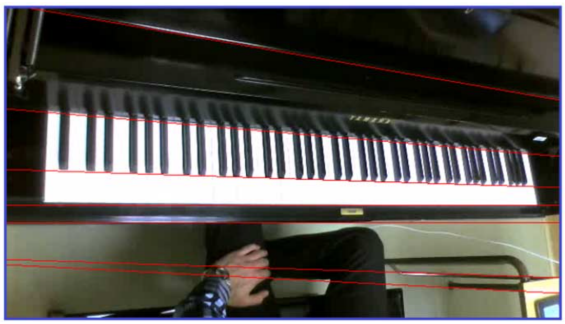

Fig. 3. Finding Keyboard Using Hough Line Transformation [4]

B. Illumination Normalization

I used the MoveTowards filter in the AForge.NET framework to account for minor illumination differences between the current video frame and the reference image [7]. The calculation the filter uses is described in Eq 1 where alpha is the current video frame, beta is the reference image, step (0-255) defines the maximum amount of change per pixel in the current image, and result is the resultant adjusted image. Causes of illumination differences include the sun being obstructed by clouds and shadows from the pianist's body being cast on the keyboard.

$$result = \alpha + min(|\beta - \alpha|, step) \times Sign(\alpha - \beta) \quad (1)$$

C. Pressed Key Detection

Key detection begins by performing background subtraction on the current video frame and reference keyboard image. The objective of this step is to use pixel differences between the current and reference images to identify keys. In the current video frame, hand positioning is determined by looking for groups of pixels that are not associated with the black and white keys. The algorithm will then only look for key detections in the area between where the hands are positioned.

When a white key is pressed, more of the adjacent black key is visible in the frame and when a black key is pressed, more of the adjacent white key is visible in the frame. The positive and negative image differences are used to determine which keys are being pressed (Fig 4).

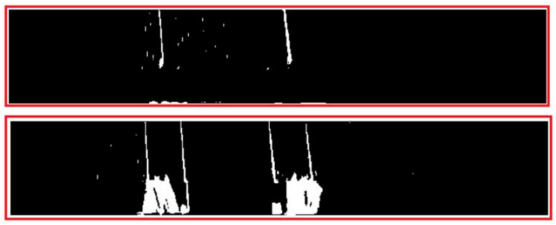

Fig. 4. Positive (top) and Negative (bottom) Image Differences

D. Music Transcription

To compare the performance of this approach with previous approaches I saved the transcription data in to a MIDI format. The ultimate goal of this approach is supposed to be aiding music instruction so for future implementations I would focus on developing a system that displays notes in real-time on the video so that an instructor has real-time feed back on what notes are being pressed.

III. Results and Comparison

To evaluate the performance of this approach I used both live footage of myself playing piano and videos posted online of other pianists playing piano. I am a beginner/intermediate level piano player so I used that level of music when testing the approaches. Approximately one hour of total video was used of varying key signature and tempo.

The focus of this approach was on real-time transcription. Ideally when a key is pressed, the algorithm should be able to identify the pressed-key before the next key is pressed. In order to achieve this, a response time of under 20ms is needed for most beginner and intermediate pieces. Because we are using simpler processing steps to improve response time, it is expected that note accuracy should not be as good as current vision-based AMT methods that focus primarily on accuracy. However, if the accuracy of our approach falls too drastically, it will not be useful for piano instruction. Ideally, this process should correctly identify at least 90% of played notes.

The tests were run on a 1.3 GHz dual-core Intel i5 processor. This CPU provides hardware performance comparable to that of current-day flagship smartphones. Video recorded from live performances was through a 720p web camera. The code for this implementation was written primarily in Python.

I tested different configurations of the algorithm to find the relationship between response time and note accuracy for different algorithm steps. The results of the tests and their comparison to current vision-based AMT approaches is shown in Table I [5]–[8]. The configurations in Table I are as follows:

I. Full Implementation
II. No Illumination Normalization
III. No Hand Detection
IV. No Illumination Normalization & No Hand Detection

TABLE I
Accuracy and Processing Performance for Different Algorithm Configurations

Configuration	Note Accuracy	Response Time (ms)
I	91%	23.22
II	88%	15.42
III	83%	17.67
IV	80%	8.44
Current AMT	98%	35-55

IV. Conclusion and Future Work

As expected, the approach (in all configurations) has a faster response time than the established video-based AMT methods while having a lower note accuracy score. All configurations met either the accuracy or response time goal criteria laid out in the previous section however no configuration met both goals simultaneously. Because of the slower nature of beginner music, the full approach (Configuration I) may be fine for beginner level instruction.

I purposely used a CPU that approximated mobile phone performance because my piano lessons are done with mobile video communication applications. As processor power increases in mobile phones, the approach may become more viable.

For future work I would like to build the display functionality mentioned earlier where an interface appears on video displaying which notes are being pressed. In addition, this approach only works for 88-key instruments. This can be expanded to larger and smaller instruments by adding a step in the note detection process to determine where the middle C note is [8]. From there, the remaining octaves can be determined based on their distance from middle C. Finally I would see how a hybrid approach combining both audio-based AMT and video-based AMT would fare. Although the sound quality is degraded over the video communication application, it may still be good enough to supplement the note detection steps of the algorithm and increase note accuracy.

Implementing additional steps will likely increase the response time so if the focus remains on real-time transcription a balance will have to be struck between keeping the processing time low and trying to increase note accuracy.

References

[1] C. Yeh, "Multiple fundamental frequency estimation of polyphonic recordings," Ph.D. dissertation, Universite Paris VI, Paris, France, 2008.

[2] K. Dressler, "Multiple fundamental frequency extraction for MIREX 2012," in Proc. 8th Music Inf. Retrieval Eval. eXchange, 2012, pp. 1–4.

[3] P. Peeling and S. Godsill, "Multiple pitch estimation using non-homo- geneous Poisson processes," IEEE J. Sel. Topics Signal Process., vol. 6, no. 5, pp. 1133–1143, Oct. 2011. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," Science, vol. 345, no. 6197, pp. 668–673, 2014.

[4] M. Sotirios and P. Georgios, "Computer vision method for pianist's fingers information retrieval," in Proc. 10th Int. Conf. Inf. Integration Web-Based Appl. Services, 2008, pp. 604–608.

[5] P. Heckbert, "Projective mappings for image warping," Master's thesis, Dept. of Comput. Sci., Univ. of California, Berkeley, CA, USA, 1989.

[6]] D. O. Gorodnichy and A. Yogeswaran, "Detection and tracking of pi- anist hands and fingers," in Proc. 3rd Can. Conf. Comput. Robot Vis., 2006, p. 63.

[7] P. Suteparuk, "Detection of piano keys pressed in video," Dept. of Comput. Sci., Stanford Univ., Stanford, CA, USA, Tech. Rep., Apr. 2014 [Online]. Available: http://web.stanford.edu/class/ee368/

[8] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A tutorial on onset detection in music signals," IEEE Trans. Speech Audio Process., vol. 13, no. 5, pp. 1035–1047, Sep. 2005.