Multimed Tools Appl (2018) 77:25513–25535
https://doi.org/10.1007/s11042-018-5803-1

A real-time system for online learning-based visual transcription of piano music

Mohammad Akbari[1] · Jie Liang[1] · Howard Cheng[2]

Received: 10 March 2017 / Revised: 2 February 2018 / Accepted: 14 February 2018 /
Published online: 23 February 2018

Abstract In order to deal with the challenges arising from acoustic-based music information retrieval such as automatic music transcription, the video of the musical performances can be utilized. In this paper, a new real-time learning-based system for visually transcribing piano music using the CNN-SVM classification of the pressed black and white keys is presented. The whole process in this technique is based on visual analysis of the piano keyboard and the pianist's hands and fingers. A high accuracy with an average F_1 score of 0.95 even under non-ideal camera view, hand coverage, and lighting conditions is achieved. The proposed system has a low latency (about 20 ms) in real-time music transcription. In addition, a new dataset for visual transcription of piano music is created and made available to researchers in this area. Since not all possible varying patterns of the data used in our work are available, an online learning approach is applied to efficiently update the original model based on the new data added to the training dataset.

Keywords Music information retrieval · Real-time piano music transcription · Image and video processing · Convolutional neural networks · Support vector machines · Online learning

✉ Mohammad Akbari
akbari@sfu.ca

Jie Liang
jiel@sfu.ca

Howard Cheng
howard.cheng@uleth.ca

[1] School of Engineering Science, Simon Fraser University, 8888 University Drive, Burnaby, BC, Canada

[2] Department of Mathematics and Computer Science, University of Lethbridge, 4401 University Drive, Lethbridge, AB, Canada

1 Introduction

The growth of digital music has considerably increased the availability of content in the last few years. In order to analyze this amount of data, Musical Information Retrieval (MIR) has become an important field of research [15, 26, 55], which deals with problems such as genre detection [6, 22], music generation [17], music recognition [32], music recommendation [20, 34], etc. Since music is usually composed from sound, most existing MIR methods only utilize audio information for the analysis of music content. However, sound is not the only important element in representing music. The visual representation is also a vital aspect of music in our daily life since music is often accompanied by a visualization [7, 16, 60]. The visual language of music genres can be a very good example in which costumes and gestures play important roles [39]. That is why music video directors use a wide range of visual effects to draw attention and manipulate our perception of a song. As a result, the visual layer of music videos also provides a wide range of music information that is very useful for the analysis of music content [8, 40, 49, 50].

One important problem in MIR is Automatic Music Transcription (AMT), which is the conversion process from music to some symbolic notation such as a music score or a Musical Instrument Digital Interface (MIDI) file [12, 44, 58]. The core problem in AMT is multi-pitch estimation, which is the task of finding multiple pitches in polyphonic music [43]. Previous works have performed this task mostly by analyzing the acoustic signal of a piece of music. However, it is difficult to obtain an accurate transcription from audio alone in a number of important situations (Section 2.1). In order to handle the difficulties arising from analyzing audio alone, the visual aspect of music can be employed.

This paper introduces an online learning-based system to perform multi-pitch estimation of piano music by only visually analyzing the piano keyboard and pianist's hands. Previous algorithms [2–4] do not work well under non-ideal conditions. The contribution of this paper is a new real-time visual piano music transcription algorithm that is accurate even when the previous algorithms are not. An improvement of over 0.44 (black keys) and 0.29 (white keys) in F_1 scores is observed over the method in [4] in these non-ideal conditions. The statistical properties of the data in our work vary according to different camera angles, lighting conditions, skin colors, fingering and hand shapes, positions, and postures. Since not all possible varying patterns are available and there is always a possibility for having new variations (e.g., playing styles used by different pianists), an online learning strategy is incorporated to dynamically adapt the model to new data added to the training set. In addition, a new dataset for visual transcription of piano music is created and made available. To the best of our knowledge, this is the first such dataset available to researchers.

The paper is organized as follows. In Section 2, different acoustic and visual approaches to AMT developed by other researchers. As described in Section 3, the first stage in our proposed approach is feature extraction in which keyboard detection, initial background image identification, keys localization, and hand area detection are initially performed. In order to deal with varying illuminations during the performance, a dynamic background update method is used to update the background image in each video frame. Convolutional neural networks (CNNs) are then utilized to extract features by feed-forwarding the difference images to the network. In Section 4, two separate binary support vector machines (SVMs) used for classifying pressed black and white keys are formulated and discussed. The experimental results will finally be evaluated in Section 5. The overall framework of the different steps in our system is illustrated in Fig. 1.

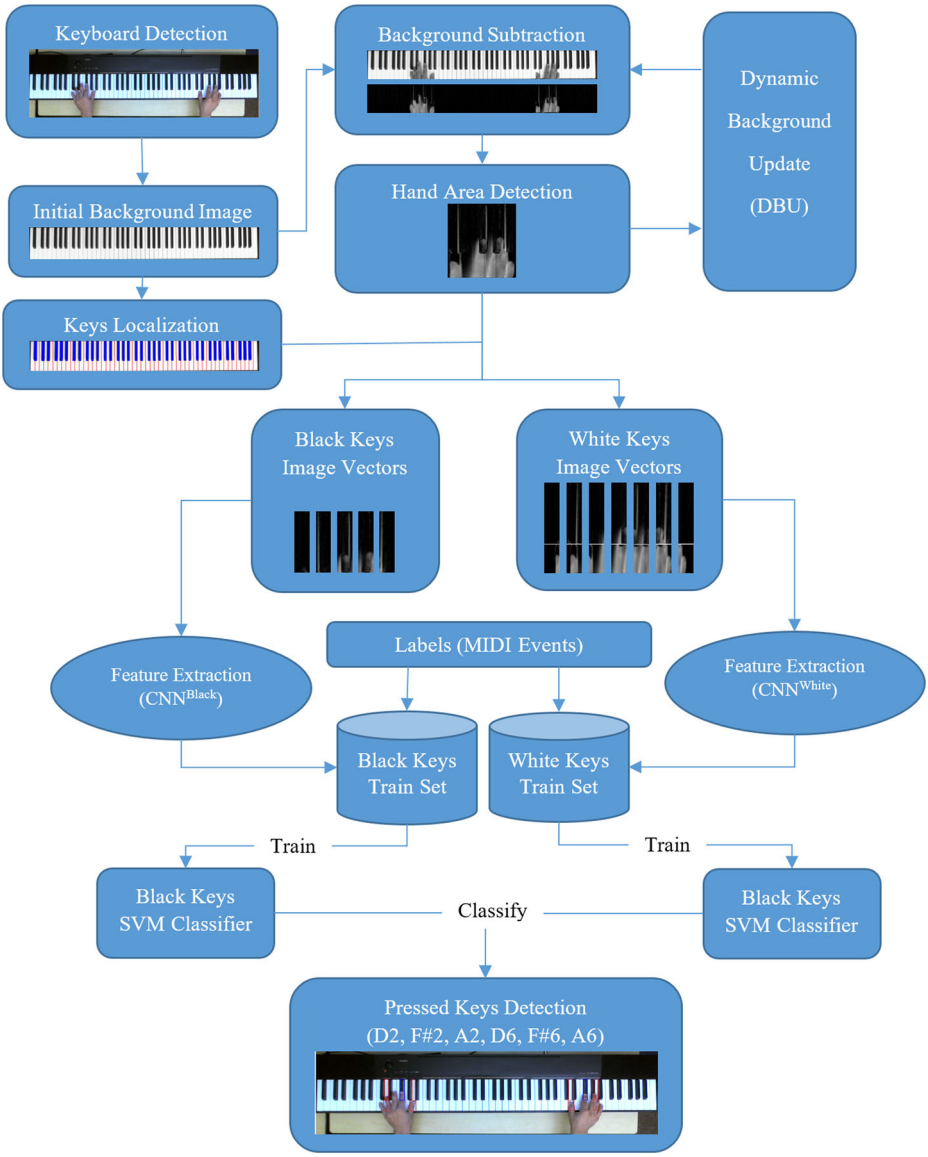


Fig. 1 The overall framework of the learning-based AMT approach proposed in this paper

2 Related works

A variety of acoustic-based AMT methods has been proposed. In Section 2.1, we will give a brief background of these methods and discuss their limitations. Following that, we describe

some previous works based on image and video processing for visually analyzing and transcribing music. Finally, some existing multi-modal developments employing both audio and visual information for transcribing music are given.

2.1 Acoustic music transcription

The main task in the AMT problem is multi-pitch estimation in which concurrent pitches in a time frame are estimated. This analysis can be performed in time and/or frequency domains. The key idea in time domain methods is to use an auto-correlation function to find the periodicity of harmonic sounds. For instance, Davy and Godsill [24], and Cemgil et al. [18] used probabilistic modeling to perform this process. On the other hand, frequency domain methods basically deal with recognizing the harmonic patterns corresponding to each pitch. Iterative spectral subtraction [35], spectral peak modeling based on maximum likelihood [27, 43], probabilistic-based full spectrum modeling [62], and spectral decomposition using Non-negative Matrix Facorization (NMF) [13] and Probabilistic Latent Component Analysis (PLCA) [10, 11] are some approaches in the frequency domain.

Most of the recent audio-based polyphonic music transcription research have focused on learning-based (or classification-based) methods in which multiple binary classifiers (each corresponding to one note) are trained with short-time acoustic features in a supervised manner [14, 53, 59]. Then, these classifiers are used in predicting notes in new input data. These methods have shown some promising results in AMT compared to other approaches. However, like any other supervised learning approaches, when the training set is limited overfitting can happen. Acoustic variations such as different timbre, tuning, and recording conditions in the test set make it difficult to obtain good transcription results. In order to handle this problem, unsupervised feature learning methods can be utilized in which robust feature representations invariant to acoustic variations can be learned [52].

Although there has been some recent progress in multi-pitch estimation techniques, it remains a research challenge [12, 36] because of a number of difficulties such as the presence of multiple lines of music (e.g, simultaneous played notes or instruments), audio noise, exact onset/offset detection, out-of-tune harmonics (incorrect recorded pitches), octave ambiguity, etc. In order to deal with these acoustic-based challenges in AMT, other types of music representations such as visual information can be employed [23, 45].

2.2 Visual music transcription

An automated approach was proposed for visually detecting and tracking the piano keys played by a pianist using image differences between the background image and current video frame [56]. The background image containing only the piano keyboard was required to be manually taken by the user at first. The algorithm had a 63.3% precision rate, a 74% recall rate, and an F_1 score of 0.68 for identifying the pressed keys [56]. Gorodnichy and Yogeswaran [31] demonstrated a video recognition tool called C-MIDI to detect which hands and fingers were used to play the keys on a piano or a musical keyboard. Instead of detecting the played notes using transcription methods, the musical instrument was equipped with a MIDI interface to be able to transmit MIDI events associated to the played notes to the computer [31].

In 2007, Zhang et al. proposed a visual method for automatically transcribing violin music in which some visual information of fingering and bowing on the violin was used to improve the accuracy of audio-based music transcription [64]. Their method had an accuracy of 92.4% in multiple finger tracking and 94.2% in string detection. However, automatic

note inference including the pitch and onset/offset detection of an inferred note had an accuracy of only 14.9%.

Extracting fingering information for many musical instruments such as piano [31, 41, 54] and guitar [42, 48] can be considered an important task used for a better transcription of the corresponding music. However, they often assume that the pitches of the played notes are already available from MIDI interfaces. Some researchers have considered the extraction of fingering information as a gesture recognition problem [50]. For example, Sotirios and Georgios [54] used Hidden Markov Models (HMM) to classify hand gestures and retrieve fingering information from the right hand of a pianist.

Akbari and Cheng proposed a computer vision-based automatic music transcription system named claVision to perform piano music transcription in real-time [4, 5]. Instead of processing the music audio, their system performed the transcription only from the video performance captured by a camera mounted over the piano keyboard. Their method used the following four-stage procedure to perform music transcription:

1. **Keyboard Registration**: keyboard detection was done at first, which included locating, transforming, and cropping the piano keyboard image. After that, the background update procedure was performed to identify the best background image (the keyboard image without any hand covering it). Having the adjusted keyboard image, the locations of the black and white keys as well as their musical features were calculated to be used in the next stages.
2. **Illumination Normalization**: this process was applied to decrease the minor differences between the background image and the other video frames. As a result, issues caused by varying illumination, noise, and shadows were significantly reduced.
3. **Pressed Keys Detection**: given the background image and the normalized image in each video frame, the difference image between them was computed, and was used in analyzing the pressed keys and the location of the pianist hands.
4. **Note Transcription**: at the final stage, the obtained musical information (e.g., note, octave, onset/offset events, etc.) was transcribed to the MIDI structure and sheet music.

Although the proposed method in claVision [4] had a high accuracy (F_1 score over 0.95) and a very low latency (about 7.0 ms) in real-time music transcription, it had a number of limitations as described below.

- In situations where the images are very dark or bright or the lighting conditions change sharply during the performance, claVision cannot function properly.
- Pressed keys that are directly below the camera are not accurately detected. In this case, the key press and release events cannot be reliably detected using the pressed keys detection algorithm.
- If a large portion of a piano key is covered by the pianist's hands, pressed keys detection for that key may not be accurate. Moreover, the pressed keys under the hands cannot be seen by the camera or even human viewers.
- If the camera vibrates or moves slightly during the performance, the keyboard and the key cannot be properly localized, which resulted in a low accuracy in pressed key detection.

2.3 Audio-visual music transcription

Recent developments have focused on multi-modal transcription systems utilizing both audio and visual information [29, 46, 47]. Frisson [29] implemented a multi-modal system

for extracting information from guitar music. However, this system requires artificial markers on the performer for visual tracking. In 2008, Paleari et al. [42] presented a complete multi-modal approach for guitar music transcription. Their method first used visual techniques to detect the position of the guitar, the fret-board, and the hands. Then, the extracted visual information was combined with the audio data to produce an accurate output. According to the results described in [42], this technique had an 89% accuracy in detecting the notes.

Tavares et al. [57] proposed an audio-visual vibraphone transcription technique to reduce the errors caused by polyphony and noise. First, audio detection was applied to estimate the notes played on vibraphone. Different audio detection techniques such as PLCA and Non-Negative Least Squares Fitting (NNLSQ) were tested. In parallel, the position of the mallets was visually detected using color filtering and contour detection algorithms. Finally, a note was detected as played if the mallet was located over the corresponding bar.

In 2009, a violin transcription system was introduced by Zhang and Wang [63] to enhance the audio-only onset detection of violin music. In their approach, a visual analysis of bowing and fingering of the violin performance was utilized to infer the onsets. The proposed method was evaluated with feature-level (concatenating audio and visual features in early stage) and decision-level (fusing the classification results of audio and video analysis obtained separately) fusion techniques. The best F_1 score was 93% achieved by decision-level data fusion.

3 Feature extraction

In this section, the process of extracting features used for training our models and classifying the pressed keys is discussed. First, the video frames are preprocessed to identify the background image, localize the piano keys, and compute the difference image. A dynamic background update method is then employed to handle varying illumination conditions in the difference image. Finally, the feature vectors corresponding to the black and white keys are extracted using CNNs.

3.1 Preprocessing

Similar to the approach in [4], the keyboard is detected and localized by utilizing the Hough line transform and homogeneous transform. The image of the detected keyboard is considered as the initial background image denoted by G^0, which is an image with no hand coverage (Fig. 2).

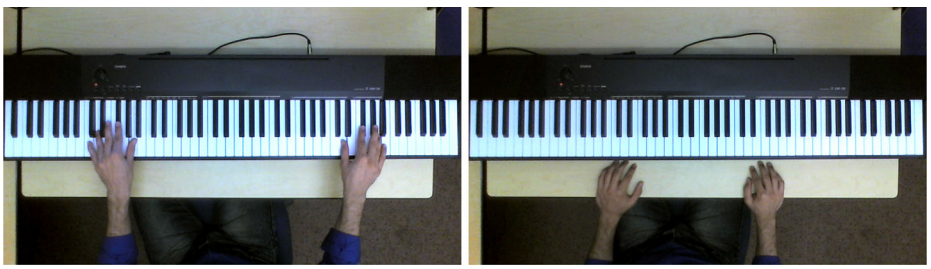


Fig. 2 Video frames with and without hands coverage

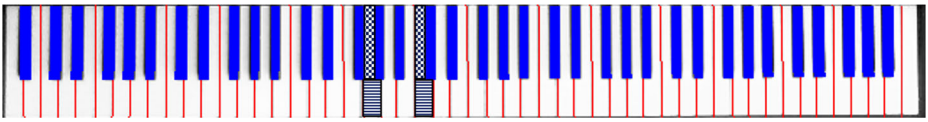


Fig. 3 Localization of black and white keys. The lines separating the white keys are estimated based on their adjacent black keys. The rectangles highlighted using dots and striped lines show the upper and lower parts of two sample white keys

Next, keys localization is performed in which the black keys are detected as objects in white background using a connected component labeling algorithm, and the white keys are localized using the dividing lines estimated from the positions of their adjacent black keys. Each of the detected keys is identified by its bounding rectangle. More formally, let $N = \{k_1^{black}, \ldots, k_n^{black}\}$ and $M = \{k_1^{white}, \ldots, k_m^{white}\}$ be the set of localized black and white keys, respectively. Each of the black keys is identified by the upper left and bottom right coordinates of the bounding rectangles identified in G^0. White keys require a more complicated description, as its shape can be considered to be the union of two rectangles. We divide each key into its upper bounding rectangle and its lower rectangle (Fig. 3).

The bounding rectangle corresponding to each black key is rescaled to a fixed size 10×40 for all samples. Similarly, the upper and lower bounding rectangles corresponding to each white key are rescaled to 10×40 and 10×20, respectively.

The musical information such as the octaves and the notes corresponding to all located keys is determined as follows. The octave numbers can be estimated by considering the middle visible octave as octave 4 including the Middle C key. The other octaves located on the left and right can be numbered accordingly. The black keys on the piano are divided into groups of two black keys (C♯) and D♯) and three black keys (F♯, G♯, and A♯), which are repeated in the whole keyboard. Since there is no black key between these two groups, the space separating them is doubled in comparison to that between the black keys inside each group. Thus, these two groups can be distinguished to determine their associated musical notes. The natural notes corresponding to the white keys are then determined based on the notes of the black keys and the estimated separating lines. For example, the black key G♯ lies between the white keys corresponding to the notes G and A.

Given the background image and the video frame at time t, G^t and I^t, the difference image, D^t, is computed as follows to identify the changes caused by key presses as well as hand movements at time t (Fig. 4):

$$D^t = I^t - G^t. \tag{1}$$

If there are no drastic changes in illumination, we may use the same background image at each step; that is $G^t = G^0$ for all t. In the next section, we will discuss how the background image can be updated to handle changing illumination conditions.

In [4], it was discussed that the black and white key presses usually result in different sign changes in the difference image. So, the positive and negative values in D^t were processed separately. However, this is not always true, especially for the pressed keys that are directly below the camera. In this work, we utilize positive and negative values for both black and white keys to improve detection.

The keys are usually pressed in the areas that the pianist's hands and fingers are present. Thus, the search domain for finding the pressed keys can be limited by detecting the location of the hands on the keyboard [4]. In addition, we can remove noisy detection results from other parts in which no hands exist. This process is performed by applying the connected

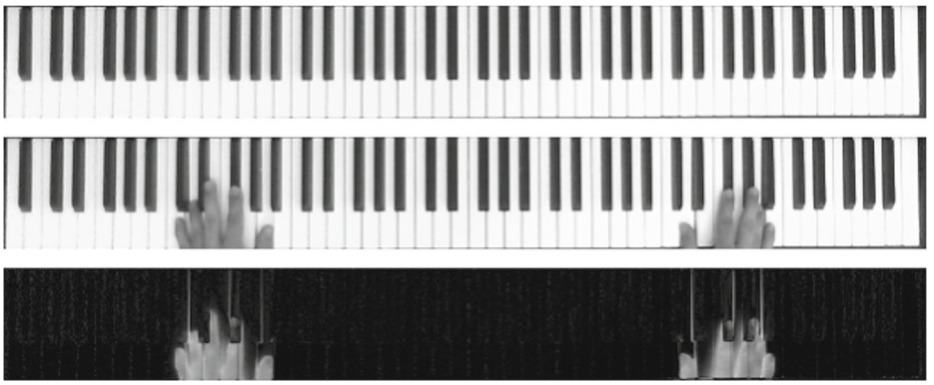


Fig. 4 Background Subtraction (from top to bottom: the background image G^t, the video frame I^t, and the difference image D^t)

component labeling algorithm to the binarized difference image, D^t, to extract a set of column indices, H^t, containing the pianist's hands and fingers (Fig. 4).

3.2 Dynamic Background Update (DBU)

Varying illumination can be a significant issue in our approach if we compare each frame to a fixed initial background image. Different types of noise and shadows cause difficulties in this approach. These variations cause an inconsistency between the background image and the other video frames, which results in noisy difference images after background subtraction. In order to make the background image consistent with the current probable variations (i.e., illumination changes, noise, and shadows), a Dynamic Background Update (DBU) algorithm is employed, which improves over the illumination correction procedure in claVision [2, 4]. The updated background image at time t, G^t, is defined by

$$G^t_{i,j} = \begin{cases} I^{t-1}_{i,j} & i \in H^{t-1}, \\ G^{t-1}_{i,j} & \text{otherwise.} \end{cases} \tag{2}$$

In other words, for columns that include the pianist's hands and fingers, the updated background image contains pixels from the previous frame I^{t-1}. Otherwise, it contains pixels from the previous background image G^{t-1}. This allows the background to adapt to changes in illumination especially around the areas where the pianist's hands are. In order to reduce computational complexity, we may choose to perform this update every k frames. Otherwise, it contains pixels from the previous background image G^{t-1}.

3.3 CNN-based feature extraction

Recent works for image classification tasks have shown that generic feature descriptors extracted from Convolutional Neural Networks (CNNs) can be effective [51]. In order to extract robust feature representations from our data, we propose a simple CNN model whose architecture is summarized in Table 1.

We use two convolutional layers (with 8 and 16 small filters of size 3×3 with stride 1) in which the ReLU activation function is utilized. Since the stride used in the convolutional layers is 1, we also set the zero-padding to 1 to ensure that the input and output of the

Table 1 CNN architecture used in this work

Layer	Type	Filters	Size
1	Convolutional + ReLU	8	3×3, stride $= 1$, pad $= 1$
2	Maxpool	–	2×2, stride $= 2$
3	Dropout (50%)	–	–
4	Convolutional + ReLU	16	3×3, stride $= 1$, pad $= 1$
5	Maxpool	–	2×2, stride $= 2$
6	Dropout (50%)	–	–
7	Fully Connected	–	2 hidden units
8	Softmax	–	2 way

layer will have the same size. Each convolution layer is followed by a maxpool layer. Since pooling sizes with larger receptive fields may be too destructive, we use small, but common 2×2 filter size and stride 2 to downsample the input by half. In order to avoid over-fitting, we apply dropout regularization with the dropout ratio 0.5 after each maxpool layer. The last two layers include a fully-connected layer with 2 hidden units and a 2-way softmax loss.

The input to our CNN network for the black keys is the cropped gray-scale image (concatenation of the gray-scale intensities in the bounding rectangle) corresponding to the ith black key in the difference image. We use the output activations from Layer 6 as the feature vector for the ith black key k_i^{black}, denoted by Φ_i. For the white keys, the concatenation of the upper and lower bounding rectangles corresponding to the ith white key is considered as the input to the CNN. The feature vector for the ith white key k_i^{white}, denoted by Ψ_i, is similarly extracted from Layer 6. Figure 5 shows some sample cropped difference images used as the inputs to our CNN.

As explained in Section 3.1, black and white keys have different sizes, shapes, structures, and contexts. Each white key is represented by the union of two rectangles, upper bounding rectangle whose structure is similar to the black keys and lower bounding rectangle which usually includes the hands and fingers used for playing that key. So, in order to extract more robust features corresponding to each type of the keys, we train two separate CNN models with different input sizes 10×40 and 10×60 (concatenation of the lower and upper rectangles), which result in Layer 6 outputs of sizes $2 \times 10 \times 16$ and $2 \times 15 \times 16$, respectively. The black keys CNN, denoted by CNN^{black}, is pre-trained with all black keys samples in the training dataset and the white keys CNN, denoted by CNN^{white}, is pre-trained with all the white keys samples (see Section 5.1 for more details about the dataset). In each video frame, we used the MIDI events (transmitted from the digital piano keyboard to the computer using a MIDI cable) as the labels for training the CNNs.

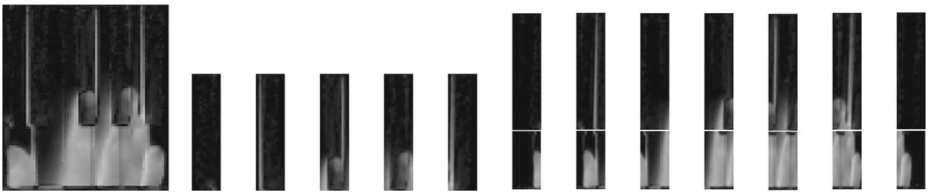


Fig. 5 From left to right: the cropped difference image of the pianist's right hand covering octave 6 in Fig. 4 with its corresponding 5 cropped images for the black keys and 7 cropped images for the white keys)

Note that for training and classification, our algorithm only considers feature vectors overlapping with the columns covered by the pianist's hands and fingers. In other words, a feature vector is only considered if it includes pixels with column indices in H. In order to standardize the range of features before training and classification, a min-max normalization is applied to scale the values to the range [0, 1].

4 Online learning-based visual AMT

4.1 Pressed keys classification

Having the normalized CNN-based feature vectors Φ and Ψ extracted in the previous section, we model our pressed keys detection problem with two binary Support Vector Machines (SVMs) [9], which are separately trained in a supervised manner using the labels provided. They are used for linear classification on whether or not a black or a white key is pressed in each video frame.

Learning algorithms can be categorized into batch and online learning. In batch learning, all examples are available at once and the model is trained over the entire set. In contrast, online learning is the technique in which training data becomes available in a sequential order, which requires the model to be updated accordingly. This type of learning is very useful in situations where the model needs to dynamically adapt to new patterns in the data, when the data is created as a function of time, or when it is not computationally feasible to train over the entire dataset. Moreover, online learning is beneficial for dealing with large or non-stationary data and real-time processing tasks for continuous data streams [37]. For classification, when a small amount of data is added to or removed from the training set, the model is not required to change much. Incremental and decremental learning algorithms can be used to efficiently update the model without retraining over the entire data set from the beginning [61].

Different camera angles, illumination conditions, style of pianists' hands and fingers in playing (i.e., fingering and hand shapes, positions, and postures), skin colors of the hands, etc. result in varying patterns in the data in our work. These variations could happen in the same video performance (e.g., varying illumination) or between the performances (e.g., camera view). A complete dataset including all these possible patterns is not available because there is always a possibility for having new variations, for example, playing styles used by different pianists. As a result, we apply an online learning technique to efficiently update the original model to incorporate new patterns in the data added to the training set in a sequential order over time.

The process of adding new training data and updating the model can be done when the classification performance in specific situations is not satisfactory to the user. In other words, the model is updated such that it can also deal with those specific cases whose data patterns have not been previously learned while the generality of the updated model is preserved (see Section 5.3 for experimental results). One way to collect the new training data is to ask the user play different pieces of music on a digital keyboard (with a MIDI cable providing the labels) in any desired situation described in Section 5.1 (e.g., different camera angles). It is also possible to collect the data using an acoustic piano. In this case, the user is asked to play some predetermined keys whose corresponding labels are already available to the program.

In order to efficiently update the models online, an alternate formulation of the SVM optimization problem is needed. In [4], it was discussed that the separate detection of the

pressed black and white keys results in a better performance in pressed keys detection since it is unlikely that the white keys are incorrectly detected as their adjacent black keys and vice versa. In this work, we follow the same strategy and consider two separate binary SVMs for black and white keys. In this work, we use trust region Newton (TRON) method, which is one of the most efficient Newton approaches for solving linear SVM [38]. This method is only applicable to the L2-loss function. Thus, we will only consider the L2 loss function in our work.

Let d_v and d_w be the dimensions of the feature vectors for the black keys and white keys, respectively. Given the training sets $\{(\Phi_i, \phi_i)\}$, $\Phi_i \in [0,1]^{d_v}$, $\phi_i \in \{+1,-1\}$, $i = 1, ..., n$ and $\{(\Psi_i, \psi_i)\}$, $\Psi_i \in [0,1]^{d_w}$, $\psi_i \in \{+1,-1\}$, $i = 1, ..., m$, the two-class linear classification problems for the black and white keys are represented as the following primal L2-loss SVM optimization problems [19].

$$\begin{aligned} \min_{v,v_0} f^{black}(v) &= \frac{1}{2}\|v\|^2 + C_v^+ \sum_{\phi_i=+1} \gamma_i + C_v^- \sum_{\phi_i=-1} \gamma_i, \\ \text{s.t.} \quad & \phi_i(v^T \Phi_i + v_0) \geq 1 - \gamma_i, \quad \gamma_i \geq 0, i = 1, ..., n \end{aligned} \tag{3}$$

and

$$\begin{aligned} \min_{w,\omega_0} f^{white}(w) &= \frac{1}{2}\|w\|^2 + C_w^+ \sum_{\psi_i=+1} \xi_i + C_w^- \sum_{\psi_i=-1} \xi_i, \\ \text{s.t.} \quad & \psi_i(w^T \Psi_i + \omega_0) \geq 1 - \xi_i, \quad \xi_i \geq 0, i = 1, ..., m \end{aligned} \tag{4}$$

where $\gamma_i = \max(0, 1-\phi_i v^T \Phi_i)^2$ and $\xi_i = \max(0, 1-\psi_i w^T \Psi_i)^2$ are the L2-loss functions. ϕ_i and ψ_i are the labels representing the status of the key in the ith training instance such that the key is considered as unpressed (key-up) if the label is -1 and it is considered as pressed (key-down) if the label is $+1$. Since the number of positive and negative instances in our work may be unbalanced [1, 21, 28], we use two different penalty parameters in each SVM formulation to balance the data [19]. The penalty parameters C_v^+, C_v^-, C_w^+, and C_w^- are estimated using the following heuristic rules:

$$\begin{aligned} C_v^+ &= \lambda^+ \frac{n}{\sum_{i=1}^n \Phi_i \Phi_i^T}, \phi_i = +1, \\ C_v^- &= \lambda^- \frac{n}{\sum_{i=1}^n \Phi_i \Phi_i^T}, \phi_i = -1, \\ C_w^+ &= \lambda^+ \frac{m}{\sum_{i=1}^m \Psi_i \Psi_i^T}, \psi_i = +1, \text{and} \\ C_w^- &= \lambda^- \frac{m}{\sum_{i=1}^m \Psi_i \Psi_i^T}, \psi_i = -1 \end{aligned} \tag{5}$$

where λ^+ and λ^- are the positive and negative class weights indicating how much of the the parameter C should be applied to instances carrying the positive and negative labels.

In order to efficiently update the models formulated in (3) and (4) when new data is added to the dataset, the warm start setting proposed in [61] is utilized to incrementally learn the patterns in the new data. In this setting, the optimal solutions v^* and w^* obtained from the most recent training and optimization are considered as the initial solutions of the new training and optimization problem as in below.

$$\bar{v} \equiv v^* \quad \text{and} \quad \bar{w} \equiv w^* \tag{6}$$

As a result, the number of iterations used for retraining the new data set is reduced. As discussed in [61], the effectiveness of this strategy is dependent on the optimization method used for training. They showed that if an initial solution is close to the optimal solution, the warm start setting significantly speeds up a high-order optimization method such as TRON [61], which is the method we use for solving the SVM problems defined in (3) and (4).

In each video frame, the black and white keys k_i^{black} and k_i^{white} are classified to the pressed $(+1)$ or unpressed (-1) classes using the decision functions $\text{sgn}(v^T \Phi_i + \nu_0)$ and $\text{sgn}(w^T \Psi_i + \omega_0)$, respectively, if their corresponding feature vectors Φ_i and Ψ_i overlap with the columns covered by the pianist's hands and fingers. Otherwise, they are classified to the unpressed (-1) class.

4.2 Music transcription

For transcribing all classified pressed keys to symbolic notation (e.g., MIDI structure), information such as note name, octave number, and note onset/offset is required. The note names and their octave numbers corresponding to the pressed keys are determined from the keys localization step (Section 3.1). The onset (attack time) of each played note is the time (frame) t at which its corresponding key is classified as pressed, and the offset (release time) is the one at which the key is classified as unpressed. As soon as a pressed key is released, a new MIDI event (containing the required Note On and Note Off messages) is created from the obtained note information (i.e., name, octave number, and onset/offset). The event is then added to a MIDI structure representing the transcription of the played music at the end of the performance.

5 Experimental results

In this section, the results obtained from our experiments are discussed and the efficiency of the proposed method is analyzed and compared to state-of-the-art. The method proposed in this work was evaluated on a PC with an Intel Core i7-4470 CPU (3.40 GHz) and 8.00 GB RAM. All the videos and images used in this evaluation were captured using an HD 720p webcam with resolution and frame rate of 640 × 360 pixels and 30 FPS. A stable stand was used to hold the camera at the top of the piano keyboard to capture a video of the pianist's performance (similar to [4]). In order to implement the proposed method, the ConvNetSharp [33], Accord.NET [25], and LIBSVM [19] libraries were utilized.

5.1 Dataset

Since there is no existing dataset for visual piano music transcription, we created a basic dataset including videos of piano performances on an 88-key digital piano.[1] This dataset consists of videos captured in different situations as described in below (Fig. 6).

– **Camera Positions**: variations in camera positions lead to different rotated and angled views of the keyboard. We considered these variations (the angles between -45 and +45 degrees from vertical) in our dataset except the drastic perspective views because they may result in incorrect registration of all visible piano keys located on the piano keyboard. As a consequence, inappropriate feature vectors are produced.

[1] All videos can be downloaded from http://www.sfu.ca/akbari/MTA/Dataset.

Fig. 6 Example frames from four sample videos in the dataset

- **Lighting Conditions**: A variety of illumination conditions was taken into account. We did not consider the situations where the images are very dark or bright such that the black and white keys become unclear and difficult to be distinguished from each other.
- **Pianists**: we included the performances by 14 pianists with different playing styles, size of hands and fingers (e.g., male/female or kid/adult), and skin colors.
- **Music**: for each of the above-mentioned situations, two types of performances are provided: 1) each key is pressed once sequentially and 2) real pieces of piano music including different chords and melodies are played.
- **Positive and Negative Examples**: the dataset includes both positive and negative training samples for black and white keys. The negative examples are usually produced when the pianist's hands and fingers cover or move over the keyboard without pressing any key (Fig. 6).

We extracted $n \approx 600,000$ samples for the black keys and $m \approx 400,000$ samples for the white keys from 71 recorded videos with 70,540 frames in total.

5.2 Training

In a real piano performance, the number of unpressed keys over the entire keyboard is usually more than the number of pressed keys. Since the training samples in our dataset were extracted from real piano performances, the dataset contains more negative instances (unpressed keys) than the positive ones (pressed keys), i.e., $\frac{n^+}{n^-} < 1$ and $\frac{m^+}{m^-} < 1$. In order to deal with this unbalance, the following settings are applied to the positive and negative class weights used in (5): $\lambda^- = 1$, $\lambda^+ = \frac{n^+}{n^-}$ for black keys, and $\lambda^+ = \frac{m^+}{m^-}$ for white keys.

Each black key sample $k^{black}_{i\in[1,m]}$ is feed-forwarded to our pre-trained CNN^{black} network up to Layer 6 whose output is considered as the feature vector Φ_i of dimension $d_v = 320$. The same process is performed to construct the feature vector Ψ_i of dimension $d_w = 480$ corresponding to the white key sample $k^{white}_{i\in[1,n]}$ from the pre-trained CNN^{white} network. In each video frame, we used the MIDI events (transmitted from the digital piano keyboard to the computer using a MIDI cable) as the labels ϕ_i and ψ_i corresponding to each sample feature vector extracted. The same labels were used for training the CNNs.

The SVM models formulated in (3) and (4) were separately trained with the training sets $\{(\Phi, \phi)\}$ and $\{(\Psi, \psi)\}$ to be later used for the classification of the pressed black and white keys, respectively.

5.3 Analysis of online learning method

In order to evaluate the computational complexity of the incremental learning approach described in Section 4, we divide the black and white datasets into 11 parts where the first part is considered as the original data and the other 10 parts are considered as new data. In order to analyze the effectiveness of the warm start strategy, we choose different levels of data changes from a small to a large increase of the data in the new subsets In this evaluation, we analyze and compare the offline and online learning fashions as described in below.

In offline mode, the model is first trained using the original dataset (OS). Then, the next new data subset is added and the model is retrained from scratch. In other words, no warm start setting is used in this mode, which means the solutions obtained from the previous training and optimization process are not utilized for the next training. This process continues until all subsets are merged and the final trained model is obtained.

However, in the online learning approach in which the warm start strategy is employed, the solution obtained in each training and optimization process is considered as the initial point for the training procedure in the next iteration. For example, the solution of training OS is used as the initial point for training OS plus the first new data subset (S1), the solution of training $OS \cup S1$ is used as considered as the initial solution for training $OS \cup S1 \cup S2$, and so on.

The convergence times for training the black and white keys models using offline (without using the warm start strategy) and online (with employing the warm start strategy) learning approaches were then analyzed. From the results obtained, it was seen that the online learning approach provided a faster convergence for both black and white models in general. For S1, S2, S4, and S5 in the black keys model and S1, S2, and S4 in the white keys model, the convergence in the online mode was almost 5 times faster because the new data statistics do not significantly change. On the other hand, the data (e.g., S3 in the white model) that causes the model to change more results in online training converging about twice as fast. On average, the results showed that the warm start strategy used for incrementally learning the new data and updating the models formulated in (3) and (4) resulted in convergence approximately three times faster than the offline method.

In order to evaluate the effectiveness of incremental learning of new data patterns in our classification problem, we set up another experiment in which different fingering styles, hand shapes and postures are taken into account. 7 recorded videos of different piano performances (denoted by OT, T1, T2, T3, T4, T5, and T6) on a 60-key electronic keyboard are considered as our training sets.[2] The properties of the videos are summarized in below.

OT: in this video, all black and white keys on the keyboard are subsequently played with the index fingers of the left and right hands such that only the middle part of the keys are pressed. This video also includes some negative examples produced by moving the hands over the keyboard while no key is pressed. We consider the samples extracted from this video as the original dataset.

[2]The videos can be downloaded from http://www.sfu.ca/akbari/MTA/OnlineLearningExperiments.

T1: the same style of playing as in OS is done in this video. However, only the tips of the keys are pressed.
T2: various white-white-white triads (the chords that consist of three distinct white keys) such as C Major and D Minor are played in this video sample. These chords are played with both hands and in different octaves.
T3: this video is composed of different black-black-black triads such as F-sharp Major and D-sharp Minor.
T4: other types of triads such as C Minor, D Major, E-flat Major, etc. are included in this video performance.
T5: this performance includes a variety of dyads (two-note chords) in which the two notes have the same pitch class, but fall into two subsequent octaves (e.g., C4 and C5). These two notes are played using the thumb and the little finger of left and right hands.
T6: this video consists of a piece of music with some chords and melodies, which are played with different fingering and hand shapes and postures.

A single test video including all the variations described in the above 7 videos is used. The training process is first performed over the samples extracted from OT. Then, the resulting trained models are used to classify the pressed black and white keys in the test video. This process is iterated 7 times in which the T1 to T6 videos are sequentially added to the training dataset. The classification performance in each iteration is illustrated in Fig. 7. As shown in the figure, adding data samples with new patterns of varying fingering and hand postures improves the overall classification performance for both black and white keys (without sacrificing the classification performance of the previous general patterns). For example, once T3 is added, the F_1 score related to the black keys increases from 0.535 to 0.689. It is because T3 provides the properties corresponding to the black-black-black triads, many of which appear in the test video.

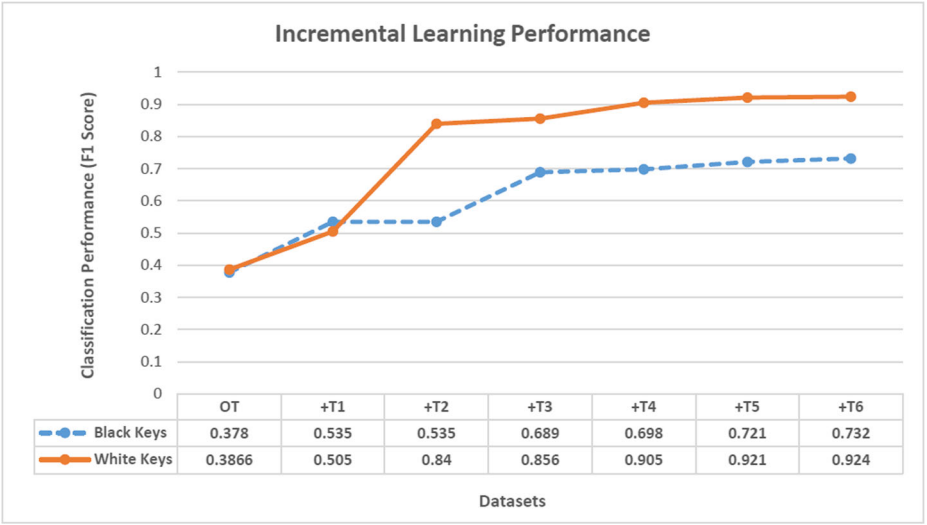


Fig. 7 The performance of incrementally learning data with new fingering and hand shapes and postures

Table 2 The F_1 scores of pressed black and white keys classification in different modes (**View**: the camera view to the keyboard, which can be **R**otated, **A**ngled, or none. **claVision**: the results using the method in [4]). The best F_1 scores are highlighted in bold

	View	CNN-SVM		CNN		SVM		SVM (NoDBU)		claVision [4]	
		B	W	B	W	B	W	B	W	B	W
V1	RA	**0.88**	**0.97**	0.87	0.88	0.77	0.90	0.38	0.89	0.69	0.64
V2	A	**0.91**	**0.95**	0.89	0.83	0.81	0.74	0.32	0.63	0.63	0.59
V3	-	0.95	**0.95**	**0.97**	0.93	0.91	0.92	0.66	0.91	0.66	0.69
V4	-	**0.97**	**0.97**	0.92	0.94	0.76	0.93	0.64	0.92	0.45	0.74
V5	RA	**0.96**	**0.98**	0.92	0.94	0.90	0.91	0.90	0.90	0.63	0.76
V6	-	**0.96**	0.96	0.91	**0.98**	0.91	0.94	0.35	0.82	0.24	0.66
V7	A	**0.97**	**0.91**	0.95	0.88	0.95	0.86	0.54	0.75	0.64	0.58
V8	RA	**0.97**	**0.93**	0.94	0.87	0.83	0.84	0.70	0.71	0.69	0.67
V9	R	0.94	**0.96**	0.93	0.95	**0.99**	0.92	0.86	0.87	0.10	0.73
V10	–	0.94	**0.97**	**0.97**	0.92	0.91	0.87	0.73	0.75	0.28	0.64
Average:		**0.94**	**0.96**	0.93	0.91	0.87	0.88	0.61	0.74	0.50	0.67

5.4 Frame-level classification of pressed keys

We consider 10 test videos including music performances with different ranges of speed, complexity, and camera positions (angles and rotations).[3] Four modes are used to evaluate the proposed method:

- **CNN-SVM:** the feature representations extracted from Layer 6 of the pre-trained CNNs are considered as the input to the SVM models (as described in Section 3). DBU is also applied in this setup.
- **CNN:** the CNN proposed in Section 3 is considered for both feature extraction and classification (using the last 2-way softmax layer). DBU is also applied in this setup.
- **SVM:** we use the gray-scale intensities in the bounding rectangles corresponding to the keys as our features and take them as the input to the SVMs. DBU is also applied in this setup.
- **SVM (NoDBU):** the same setup as in the third mode is used, but without applying DBU.

The classification results of the pressed black and white keys in terms of F_1 score are shown in Table 2 and Fig. 8. Since the offline and online learning methods described in Section 4.1 provide the same classification performance, we only report the results of the offline mode in the table.

As shown in Table 2, the average F_1 scores for the classification of the pressed black and white keys using SVM (NoDBU) are 0.61 and 0.74, respectively. However, if the DBU (Dynamic Background Update) method is used, the results are improved to 0.87 and 0.88 in the SVM setup, which demonstrates the effectiveness of DBU in adjusting varying illumination during the video performance. This is also illustrated in Fig. 8 in which claVision and SVM yield to more fluctuating and unstable results due to the varying illumination conditions in different test videos. However, by applying DBU the results become more stable,

[3]The test videos and the classification results can be downloaded from http://www.sfu.ca/akbari/MTA.

Fig. 8 Black and white pressed keys classification results summarized in Table 2

especially for CNN-SVM. From the results, it is seen that CNN outperforms SVM by ≈5%, which shows the effectiveness of using deep feature representations.

In Section 4.1, it was explained that the training samples in our dataset are unevenly distributed among the two classes (pressed and unpressed keys). As discussed in [30], when the dataset is unbalanced, CNNs usually perform better on the majority than the minority class. However, the classification ability on the minority class (pressed keys) is essential in our work. In order to address this problem, the CNN-SVM model is proposed in which two different penalty parameters in each SVM formulation is considered to balance the dataset (Sections Sections 4.1 and 5.2). Our experiments show that the hybrid CNN-SVM model (in which SVM is trained using the features extracted from the 6th layer of CNN) achieves the best performance with an average F_1 score of 0.95.

Compared to the F_1 scores 0.50 and 0.67 obtained from the pressed keys detection method used in [4], the classification-based approach proposed in this work provides a

Table 3 The maximum and average processing times of different methods in each video frame

Method	Processing time (ms)	
	Maximum	Average
CNN-SVM	51.0	21.0
SVM	25.0	7.5
SVM (NoDBU)	25.0	7.5
claVision [4]	23.0	6.6

better performance. In [4], it was discussed that the average F_1 score of the pressed keys detection in the ideal situation (30–45 degrees from vertical) is 0.974. However, as summarized in Table 2, it does not have a good performance in non-ideal situations (e.g., when the camera angle is around 0 and the pressed keys are located right under the camera). Another difficulty in [4] was the changing illumination, which had made the results noisy and less accurate. In particular, when pianists' hands cover a large portion the keyboard, the method in [4] resulted in many false positives (incorrect detection of the hands as pressed keys) and false negatives (pressed keys covered by the hands cannot be seen and detected).

In the training dataset indicated in Section 5.1, we included different camera positions (especially the challenging degrees around 0), illumination conditions, and also a variety of negative examples including partially covered keyboards. Having all these variations used for training the models in (3) and (4) resulted in a more accurate classification of the pressed and unpressed keys compared to the approach in [4].

The maximum and average processing times of the proposed method in different modes are given in Table 3. The claVision method in [4] with the maximum and average of 23.0 ms and 6.6 ms has the lowest processing times compared to the other methods. SVM is however shown to result in a 9 ms more latency than claVision, which is due to the extra time required for the classification of the pressed keys. By employing the CNN-based feature extractor, the maximum and average processing times increase to 51.0 ms and 21.0 ms, respectively. The average time required for forwarding each data sample (which corresponds to the image vector of one black or white key) to the CNNs up to Layer 6 and extract the features is 1.4 ms. CNN-SVM performs 13.5 ms slower than SVM, which is due to the extra forward times required for about 10 samples per video frame on average. The computational complexity of keyboard detection, keys localization, and DBU did not affect the real-time processing because they are performed in separate threads.

In order to analyze the performance of using more filters in the convolutional layers used in our CNNs (Section 3.3), we run a new experiment with 32 and 48 filters in Layers 1 and 4, respectively. The classification results obtained by using CNN-SVM with 8/16 and 32/48 choices are compared in Fig. 9. As illustrated in the figure, both 8/16 and 32/48 choices result in almost the same classification performance on average. The average black keys F_1 score is 0.945 for 8/16 and 0.943 for 32/48. For the white keys, the F_1 scores are 0.955 and 0.962, respectively.

Increasing the number of filters in CNN will result in higher forward time as well as higher SVM classification time due to the increase of the size of the feature vectors extracted from the CNNs with more filters (i.e., $d_v = 960$ and $d_w = 1440$). The forward times of the CNN^{black} and CNN^{white} networks with 32 and 48 filters is 11.0 ms and 18.0 ms. The average processing time of CNN-SVM with the choice of 32 and 48 filters is 145.12 ms, which is about 7 times slower than 8/16 with 21.0 ms. As a result, our choice of 8 and 16 filters is shown to give the best trade-off for this work, especially for real-time processing.

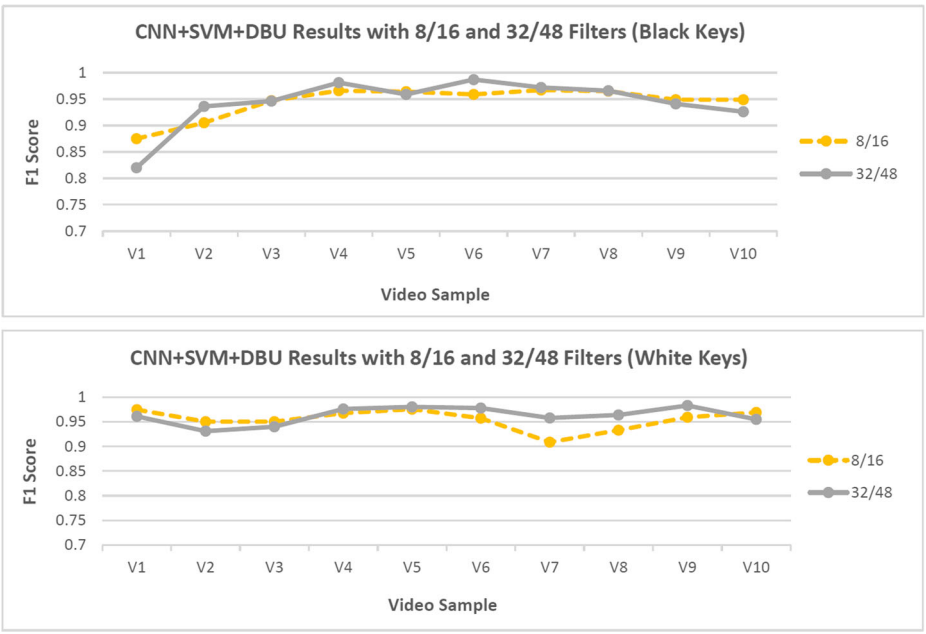


Fig. 9 The comparison of CNN-SVM classification results with 8 & 16 filters and 32 & 48 filters

Utilizing GPUs can significantly reduce the computational complexity and processing time of the proposed method, especially CNN-based feature extraction. However, we chose to run our algorithms on a PC with a regular CPU to reduce the cost of the system and make it more accessible to the users.

6 Conclusion

It was shown that the visual analysis of music performances can be very useful for dealing with MIR problems such as AMT. In this paper, a learning-based visual approach for transcribing the piano music in real-time was presented. The proposed CNN-SVM approach achieved F_1 scores of 0.94 and 0.96 in the classification of the pressed black and white keys, respectively. Compared to the state-of-the-art, an improvement of over 0.44 (black keys) and 0.29 (white keys) in F_1 scores under non-ideal camera view, hand coverage, and illumination conditions was achieved. In order to adapt our classifiers with the new patterns in the data added to training set, an incremental learning strategy named warm start was employed to efficiently retrain and update the models. A new dataset for research in this area has also been created.

The proposed system has a number of limitations such as intense darkness or brightness of the images, sharp lighting changes during the performance, drastic camera views, expressive aspects of music (e.g., dynamics), and camera movements. Some of these limitations are intrinsic to the approach and cannot be removed. However, there are some that can be removed using more sophisticated and often more time-consuming algorithms (e.g., keyboard tracking when the camera or the keyboard moves). We have chosen not to use these approaches in this method in order to allow real-time transcription.

One potential direction of this work is to utilize both audio and visual information in order to analyze music content. Such multi-modal systems can improve the performance of music transcription as well as other problems in MIR, especially by using machine/deep learning approaches.

Acknowledgements This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant RGPIN312262, STPGP447223, RGPAS478109, and RGPIN288300.

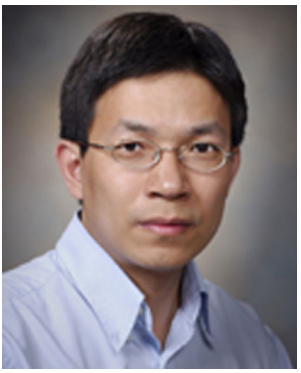
References

1. Akbani R, Kwek S, Japkowicz N (2004) Applying support vector machines to imbalanced datasets. In: European conference on machine learning, pp 39–50
2. Akbari M (2014) claVision: bisual automatic piano music transcription. Master's thesis, University of Lethbridge, Lethbridge
3. Akbari M, Cheng H (2015) claVision: visual automatic piano music transcription. In: Proceedings of the international conference on new interfaces for musical expression. Louisiana State University, Baton Rouge, pp 313–314
4. Akbari M, Cheng H (2015) Real-time piano music transcription based on computer vision. IEEE Trans Multimed 17(12):2113–2121
5. Akbari M, Cheng H (2016), Methods and systems for visual music transcription. http://www.google.com/patents/US9418637. US Patent 9,418,637
6. Baniya BK, Lee J (2016) Importance of audio feature reduction in automatic music genre classification. Multimed Tools Appl 75(6):3013–3026
7. Baur D, Seiffert F, Sedlmair M, Boring S (2010) The streams of our lives: visualizing listening histories in context. IEEE Trans Vis Comput Graph 16(6):1119–1128
8. Bazzica A, Liem C, Hanjalic A (2016) On detecting the playing/non-playing activity of musicians in symphonic music videos. Comput Vis Image Underst 144:188–204
9. Ben-Hur A, Weston J (2010) A user's guide to support vector machines. In: Data mining techniques for the life sciences, pp 223–239
10. Benetos E, Dixon S (2012) A shift-invariant latent variable model for automatic music transcription. Comput Music J 36(4):81–94
11. Benetos E, Weyde T (2015) An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In: International society for music information retrieval, pp 701–707
12. Benetos E, Dixon S, Giannoulis D, Kirchhoff H, Klapuri A (2013) Automatic music transcription: challenges and future directions. J Intell Inf Syst 41:407–434
13. Bertin N, Badeau R, Vincent E (2010) Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. IEEE Trans Audio Speech Lang Process 18(3):538–549
14. Böck S, Schedl M (2012) Polyphonic piano note transcription with recurrent neural networks. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 121–124
15. Borjian N, Kabir E, Seyedin S, Masehian E (2017) A query-by-example music retrieval system using feature and decision fusion. Multimed Tools Appl 1–25. https://doi.org/10.1007/s11042-017-4524-1
16. Brown S (2006) The perpetual music track: the phenomenon of constant musical imagery. J Conscious Stud 13(6):43–62
17. Cao X, Sun L, Niu J, Wu R, Liu Y, Cai H (2015) Automatic composition of happy melodies based on relations. Multimed Tools Appl 74(21):9097–9115
18. Cemgil AT, Kappen HJ, Barber D (2006) A generative model for music transcription. IEEE Trans Audio Speech Lang Process 14(2):679–694
19. Chang C, Lin C (2011) Libsvm: a library for support vector machines. ACM Trans Intell Syst Technol (TIST) 2(3):27
20. Chang H, Huang S, Wu J (2016) A personalized music recommendation system based on electroencephalography feedback. Multimed Tools Appl 1–20. https://doi.org/10.1007/s11042-015-3202-4
21. Chawla NV, Japkowicz N, Kotcz A (2004) Editorial: special issue on learning from imbalanced data sets. ACM Sigkdd Explor Newsl 6(1):1–6
22. Corrêa DC, Rodrigues FA (2016) A survey on symbolic data-based music genre classification. Exp Syst Appl 60:190–210

23. Dannenberg RB (1993) Music representation issues, techniques, and systems. Comput Music J 17(3):20–30
24. Davy M, Godsill SJ (2003) Bayesian harmonic models for musical signal analysis. Bayesian Stat 7:105–124
25. de Souza C (2014) Accord.net framework. http://www.accord-framework.net
26. Downie JS (2003) Music information retrieval. Annu Rev Inf Sci Technol 37(1):295–340
27. Duan Z, Pardo B, Zhang C (2010) Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. IEEE Trans Audio Speech Lang Process 18(8):2121–2133
28. Farquad M, Bose I (2012) Preprocessing unbalanced data using support vector machine. Decis Support Syst 53(1):226–233
29. Frisson C, Reboursière L, Chu W, Lähdeoja O, Mills Iii J, Picard C, Shen A, Todoroff T (2009) Multimodal guitar: performance toolbox and study workbench. QPSR of the Numediart Res Progr 2(3):67–84
30. Geng M, Wang Y, Tian Y, Huang T (2016) Cnusvm: Hybrid cnn-uneven svm model for imbalanced visual learning. In: IEEE second international conference on multimedia big data (BigMM), pp 186–193
31. Gorodnichy DO, Yogeswaran A (2006) Detection and tracking of pianist hands and fingers. In: 2006 The 3rd Canadian conference on computer and robot vision, p 63
32. Gutiérrez S, García S (2016) Landmark-based music recognition system optimisation using genetic algorithms. Multimed Tools Appl 75(24):16905–16922
33. Karpathy A (2016) Convnetsharp. https://github.com/cbovar/ConvNetSharp
34. Katarya R, Verma OP Efficient music recommender system using context graph and particle swarm. Multimed Tools Appl 1–15. https://doi.org/10.1007/s11042-017-4447-x
35. Klapuri AP (2003) Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. IEEE Trans Speech Audio Process 11(6):804–816
36. Klapuri A (2004) Automatic music transcription as we know it today. J New Music Res 33(3):269–282
37. Laskov P, Gehl C, Krüger S, Müller K (2006) Incremental support vector learning: analysis, implementation and applications. J Mach Learn Res 7:1909–1936
38. Lin C, Weng R, Keerthi S (2008) Trust region newton method for logistic regression. J Mach Learn Res 9:627–650
39. Maler A (2013) Songs for hands: analyzing interactions of sign language and music. Music Theory Online 19(1):1–15
40. Nanni L, Costa YM, Lumini A, Kim MY, Baek SR (2016) Combining visual and acoustic features for music genre classification. Exp Syst Appl 45:108–117
41. Oka A, Hashimoto M (2013) Marker-less piano fingering recognition using sequential depth images. In: 2013 19th Korea-Japan joint workshop on frontiers of computer vision, (FCV), pp 1–4
42. Paleari M, Huet B, Schutz A, Slock D (2008) A multimodal approach to music transcription. In: 15th IEEE international conference on image processing, pp 93–96
43. Peeling PH, Godsill SJ (2011) Multiple pitch estimation using non-homogeneous poisson processes. IEEE J Sel Top Sign Process 5(6):1133–1143
44. Pertusa A, Iñesta JM (2005) Polyphonic monotimbral music transcription using dynamic networks. Pattern Recogn Lett 26(12):1809–1818
45. Poast M (2000) Color music: visual color notation for musical expression. Leonardo 33(3):215–221
46. Quested G, Boyle R, Ng K (2008) Polyphonic note tracking using multimodal retrieval of musical events. In: Proceedings of the international computer music conference (ICMC)
47. Reboursière L, Frisson C, Lähdeoja O, Mills Iii J, Picard C, Todoroff T (2010) MultimodalGuitar: a toolbox for augmented guitar performances. In: Proceedings of the New Interfaces for Musical Expression++ (NIME++)
48. Scarr J, Green R (2010) Retrieval of guitarist fingering information using computer vision. In: 25th international conference of image and vision computing New Zealand (IVCNZ), pp 1–7
49. Schindler A, Rauber A (2016) Harnessing music-related visual stereotypes for music information retrieval. ACM Trans Intell Syst Technol (TIST) 8(2):20
50. Seger RA, Wanderley MM, Koerich AL (2014) Automatic detection of musicians' ancillary gestures based on video analysis. Exp Syst Appl 41(4):2098–2106
51. Sharif Razavian A, Azizpour H, Sullivan J, Carlsson S (2014) Cnn features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 806–813
52. Sigtia S, Benetos E, Boulanger-Lewandowski N, Weyde T, d'Avila Garcez AS, Dixon S (2015) A hybrid recurrent neural network for music transcription. In: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 2061–2065
53. Sigtia S, Benetos E, Dixon S (2016) An end-to-end neural network for polyphonic piano music transcription. IEEE/ACM Trans Audio Speech Lang Process (TASLP) 24(5):927–939

54. Sotirios M, Georgios P (2008) Computer vision method for pianist's fingers information retrieval. In: Proceedings of the 10th international conference on information integration and web-based applications & services, iiWAS '08. ACM, pp 604–608
55. Stober S, Nürnberger A (2013) Adaptive music retrieval–a state of the art. Multimed Tools Appl 65(3):467–494
56. Suteparuk P (2014) Detection of piano keys pressed in video. Tech. rep., Department of Computer Science, Stanford University
57. Tavares TF, Odowichuck G, Zehtabi S, Tzanetakis G (2012) Audio-visual vibraphone transcription in real time. In: 2012 IEEE 14th international workshop on multimedia signal processing (MMSP), pp 215–220
58. Tavares TF, Barbedo JGA, Attux R, Lopes A (2013) Survey on automatic transcription of music. J Braz Comput Soc 19(4):589–604
59. Taweewat P, Wutiwiwatchai C (2013) Musical pitch estimation using a supervised single hidden layer feed-forward neural network. Exp Syst Appl 40(2):575–589
60. Thompson WF, Graham P, Russo FA (2005) Seeing music performance: visual influences on perception and experience. Semiotica 2005(156):203–227
61. Tsai C, Lin C, Lin C (2014) Incremental and decremental training for linear classification. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, pp 343–352
62. Yoshii K, Goto M (2012) A nonparametric bayesian multipitch analyzer based on infinite latent harmonic allocation. IEEE Trans Audio Speech Lang Process 20(3):717–730
63. Zhang B, Wang Y (2009) Automatic music transcription using audio-visual fusion for violin practice in home environment. Tech. Rep. TRA7/09, School of Computing, National University of Singapore
64. Zhang B, Zhu J, Wang Y, Leow WK (2007) Visual analysis of fingering for pedagogical violin transcription. In: Proceedings of the 15th international conference on multimedia, pp 521–524

Mohammad Akbari received his M.Sc. degree in Computer Science from University of Lethbridge, AB, Canada. He is currently a Ph.D. student in Engineering Science at Simon Fraser University, BC, Canada. His research interests include image/video/audio processing, audio-visual analysis of music content, and machine/deep learning.



Jie Liang is a professor and the associate director at the School of Engineering Science, Simon Fraser University, BC, Canada. His research interests include Image and Video Coding, Multimedia Communications, Sparse Signal Processing, Computer Vision, and Machine Learning.

Howard Cheng is an associate professor in the Department of Mathematics and Computer Science at the University of Lethbridge, Alberta, Canada. His research interests include image and video processing, computer vision, as well as computer algebra and symbolic computation.