Optical Character Recognition for Musical Notes and Playback

Kartik B. Bhargav, *Software Engineer, Airwatch,* Kaushik M.K., *Software Engineer, Y-MediaLabs*, Kaustubh B. Bhargav, *Software Engineer, Scientific-Games,*, Nataraj C.R., *Professor, Dept. of E&C SJCE Mysore,* and Neelanjana E.K., *Hardware Engineer, General Motors*

***Abstract*—OCR for music recognition and playback aims to address the problem of music data acquisition. It accomplishes this by using OCR to recognize the music score and converts it into MusicXML file format. MusicXML file is the industry standard way of representing western musical notation. The conversion of the recognized symbols to MusicXML file format enables the user to store, edit, play and share his music. The main motivation behind the project is to develop software like "Cam Scanner" for musical score sheets that can be used by learners, professional musicians and publishers with ease. The character recognition has been implemented through template matching using normalized cross correlation. The Music table of recognized symbols with (x,y)coordinates, staff number, duration, stem, octave, step and alter is generated, which will be helpful to test the correctness of the recognized symbols. Currently 10 different instruments such as guitar, violin, mandolin, melodica, banjo, koto, kazoo, electric guitar, sitar, mezzo-soprano are supported for music playback by using Muse Score software. The implementation was done through in Matlab(version R2014a - 8.3.0.532).**

***Keywords*—OCR, MusicXML, Cam Scanner, Template matching, Normalized cross correlation, Matlab.**

I. Introduction

A. Subsection Heading Here

II. MusicXML

MusicXML (Music Extensive Markup Language) is a digital sheet music interchange and distribution format. The goal is to create a universal format for common Western music notation, similar to the role that the MP3 format serves for recorded music.

The musical information is designed to be usable by notation programs, sequencers and other performance programs, music education programs and music databases. It is designed from the ground up for sharing sheet music files between applications and for archiving sheet music files for use in the future. MusicXML files are readable and usable by a wide range of music notation applications. MusicXML complements the native file formats used by several musical score writing programs, which are designed for rapid and interactive use. MusicXML files are the standard for sharing interactive sheet music. Using MusicXML, users can create music in one program and share the results back and forth with people using other programs.

MusicXML was based primarily on two academic music formats:

- The MuseData format, developed by Walter Hewlett at the Center for Computer Assisted Research in the Humanities (CCARH), located at Stanford University .
- The Humdrum format, developed by David Huron, based at Ohio State University.

MusicXML has two different top-level Document Type Defitions (DTDs), each with its own root element. If the partwise DTD is used, the root element is ¡score-partwise¿. The musical part is primary, and measures are contained within each part. If the timewise DTD is used, the root element is ¡score-timewise¿. The measure is primary, and musical parts are contained within each measure. The MusicXML XML Schema Definition (XSD) includes both of the top-level document elements in a single XSD file.Having two different structures does not work well if there is no automatic way to switch between them. MusicXML provides two EXtensible Stylesheet Language Transformations (XSLT) stylesheets to convert back and forth between the two document types.

Score header contains some basic metadata about a musical score, such as the title and composer. It also contains the part-list, which lists all the parts or instruments in a musical score. MusicXML music data contains two main types of elements. One set of elements is used primarily to represent how a piece of music should sound. These are the elements that are used when creating a MIDI file from MusicXML. The other set is used primarily to represent how a piece of music should look.These elements are used when creating a MuseScore file from MusicXML. The terminology used in XML are as follows.

1) Tag: A markup construct that begin with "¡ and ends with "¿. There are three types of tags namely start tags, end tags and empty element tag.

2) Element: An XML element is the central building block of any XML document. It is a logical document component which begins with a start tag and ends with an end tag.

3) Root element: The root element is the first named tag of every XML file and is a container for all other elements.

4) Parent element: An XML element that contains another element.

5) Child element: An XML element that is contained within a parent element. A child element sits inside of the parent and further itemizes the tags within the file.

6) Attribute: A markup construct consisting of a name & value pair that exists within a start tag or element tag.

7) Data String: A data string is the information that the veiwer can see. For example, a description of an inventory item would be a data string. Data strings sit between the opening and closing tags of element.

8) XML declaration: The declaration statement gives the browser information to recognize the language and syntax of the file. Without a declaration statement, the Internet processor is unable to compute the code. This is the first line of any XML document and defines the language, version, specifies encoding and declares the standalone status of the file. Only the language definition and version are required for a declaration statement. Encoding and standalone are optional attributes.

III. Software Requirements

The input image acquired has to be subjected to various processing operations for recognising the characters in the music score sheet. Thus, the platform used for the implementation should support image processing operations. Matlab has extensive support for image processing operations through Image Processing Toolbox". Matlab also has Data and File Management Toolbox which can be used to read, write and generate XML files. MusicXML being one of the types of XML file, the toolbox will be able to handle it's basic operations also. Hence Matlab becomes a good choice for implementation. Matlab version R2014a - 8.3.0.532 has been used for the implementation of the algorithm. For reading MusicXML files MuseScore can be used. MuseScore is a free and opensource score writer software with rich features.

IV. Implementation

This section presents the steps involved in converting the captured image of a score sheet into MusicXML file.

A. Image Acquisition

The music score sheet image is acquired from MuseScore software. The file format used is "png". A sample image of score sheet is shown in figure2

B. Image Preprocessing

The image acquired will be in rgb format and is converted into grey scale image. The purpose of converting an RGB image into gray scale is due to the fact that it eliminates the hue and saturation information, as they contribute very less for the total appearance of the image and retains the needed luminance (intensity). The main advantages of image preprocessing are it helps in reducing the noise in images, Variations in illumination and viewing geometry between images (for optical sensors) can be corrected by image pre-processing techniques and helps to convert the image into a form that is more suitable for further operations like morphological operations and feature extraction.

C. Image Segmentation

Image segmentation is the process of partitioning an image into parts or regions. This division into parts is often based on the characteristics of the pixels in the image. Thresholding method of segmenting method is used here to convert a

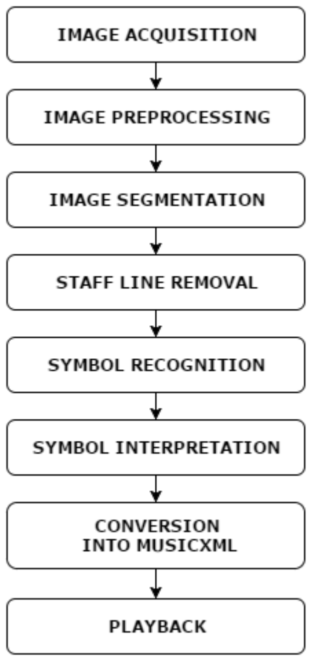

Fig. 1. Flow chart for the steps involved conversion of image of score sheet to MusicXML file

grey scale image into binary image. This transformation is important because the morphological operations like dilation, erosion, etc. can be done efficiently on a binary image. Examples of a segmented image and a non-segmented image are shown in Figure 4.4 and Figure 4.3. Figure 4.3 contains pixel values in the gray scale and each pixel can take any value out of the available 256 gray scale levels. As seen from figures 4.3 and 4.4, image segmentation converts the gray scale image into a binary image.

D. Morphological Image processing

Morphological image processing is a collection of non-linear operations related to the shape or morphology of features in an image. Morphological operations rely only on the relative ordering of pixel values, not on their numerical values and therefore are especially suited to the processing of binary images. Morphological techniques typically probe an image with a small template known as a Structuring Element (SE). The SE is very small compared to the image size. They are usually made up of ones and zeroes. The SEs usually have odd orders like 3x3, 5x5, 7x7 etc. because, while performing any morphological operations, the centre pixel of the SE is placed on each and every pixel of the image and the operation is thus performed. The SE is positioned at all possible locations in the image and it is compared with the corresponding neighborhood pixels. By marking the locations where SE fits or hits the image, information about the structure of the image can be obtained. The SE "fits image if for each of its pixels that is set to 1(foreground), the corresponding image pixel is also set to 1

. Dilation and erosion are morphological image processing techniques which are extensively used in staff line removal.

E. Staff Line Removal

Removing staff lines is important, because in the score sheet essentially the same symbol will be placed on different staff lines. Without this step, the same symbol will be treated as different when they are on different staff lines. For removal of staff lines morphological image processing techniques such as dilation and erosion are used. The image is subjected to dilation two times which is then followed by erosion.

1) Dilation: The dilation of an image f by a structuring element s (denoted by $f \oplus s$) produces a new binary image

$$g = f \oplus s$$

with ones in all locations (x,y) of a structuring elements origin at which that structuring element s hits the the input image f, i.e. g(x,y) = 1 if s hits f and 0 otherwise, repeating f, i.e. g(x,y) = 1 if s hits f and 0 otherwise, repeating.

The holes enclosed by a single region and gaps between different regions become smaller, and small intrusions into boundaries of a region are filled in. Figure 4.5 shows the effects of dilation on an input image.

2) Erosion: The erosion of a binary image f by a structuring element s (denoted by $f \ominus s$) produces a new binary image

$$g = f \ominus s$$

with ones in all locations (x,y) of a structuring elements origin at which that structuring element s fits the input image f, i.e. g(x,y) = 1 is s fits f and 0 otherwise, repeating for all pixel coordinates (x,y).

Erosion with small square structuring elements shrinks an image by stripping away a layer of pixels from both the inner and outer boundaries of regions. The holes and gaps between different regions become larger and small details are eliminated.

Larger structuring elements have a more pronounced effect, the result of erosion with a large structuring element being similar to the result obtained by iterated erosion using a smaller structuring element of the same shape. If s1 and s2 are a pair of structuring elements identical in shape, with s2 twice the size of s1, then Erosion removes small-scale details from a binary image but simultaneously reduces the size of regions of interest, too. By subtracting the eroded image from the original image, boundaries of each region can be found:

$$b = f - (f \ominus s)$$

where f is an image of the regions, s is structuring element, and b is an image of the region boundaries. Figure 4.6 shows the effects of erosion on an input image.

F. Symbol Recognition

The symbol recognition is the critical and most important step OCR systems. This step actually determines the performance of the overall project. Along with the symbol recognition, vertical translation has to be determined to get the output. To recognize the symbols, template matching is used. A template is a small image (sub-image), the goal is to find occurrences of this template in a larger image. Template matching techniques compare portions of images against one another. Sample images are used to recognize similar objects in source image.

Template matching has been a classical approach to the problems of locating and recognizing of an object in an image. Among several matching methods, Normalized Cross Correlation and Square root of Sum of Square Differences have been used as the measure for similarity. Moreover, many other template matching techniques, such as Sum of Absolute Differences (SAD) and Sequential Similarity Detection Algorithm (SSDA) have been adopted in many applications for pattern recognition. In the project Normalized Cross Correlation technique is used for template matching.

Correlation is a measure of the degree to which two variables agree, not necessary in actual value but in general behavior. The two variables are the corresponding pixel values in two images, template and source. The cross-correlation of two real continuous functions x(t) and y(t) , is defined by

$$\phi_{xy}(t) = \int x(t-\tau)y(\tau)d\tau$$

In comparison to convolution

$$x(t) * y(t) = \int_{-\infty}^{\infty} x(\tau - t) * y(\tau)d\tau$$

It can be seen that the only difference is that for the cross correlation, one of the two functions is not reversed.

$$\phi_{xy}(t) = x(-t) * y(t)$$

Since the operation of time reversal is the same as taking the complex conjugate in the frequency domain.

$$\phi_{xy} = FT[\phi|xy(t)] = X(f) * Y(f)$$

In the discrete domain, the correlation of two real time series x_i, i = 0, 1, . . . , M-1 and y_j, j = 0,1, . . . , N-1 is by analogy to equation (1) given by

$$\phi_{xy,k} = \sum_{j=max(0,k)}^{min(M-1+k,N-1)} x_{j-k} + y_j$$

where k = -[M+1],0,(N-1)

G. Normalized Cross Correlation

The normalized correlation for two time series can be defined as

$$\phi'_{xy}(t) = \frac{\phi_{xy}(t)}{\sqrt{\phi_{xx}(0)\phi_{yy}(0)}}$$

The normalized quantity will vary between -1 and 1. A value of 1 indicates that at the alignment, the two time series have the exact shape (the amplitudes may be different) while a value -1 indicates that they have the same shape except that they have the opposite signs. A value of 0 shows that they are completely uncorrelated. In practice when one applies this normalization to real discrete signals, one will find that a correlation coefficient greater than about 0.7 or 0.8 indicates a pretty good match.

The main advantage of the normalized cross correlation over the ordinary cross correlation is that it is less sensitive to linear changes in the amplitude of illumination in the two compared images. Furthermore, the Normalized Cross Correlation is confined in the range between 1 and 1. The setting of detection threshold value is much simpler than the cross correlation. There are several set of templates which are used to determine the symbols using the normalized cross correlation method. The database of symbols includes flat, sharp, rests, half note, whole note and quarter note as shown in Figures 4.7 to 4.11.

1) Vertical Translation: Vertical Translation of a symbol is used to denote its pitch. It is also a crucial parameter in determining the actual symbol. Any error in this step will distort music. Hence it is one part of the recognition where error is intolerable. The position of staff lines in the score sheet is fixed. This principle is useful in finding out the vertical translation. After the normalized cross correlation, the (x,y) coordinate value of the centre pixel will be available. Comparing this with the standard value, the vertical translation can be easily obtained. Combining both the symbol and its vertical translation the perfect output is obtained. After this step finally the symbols will be ready for Music XML conversion and playback.

H. Symbol Interpretation

The recognized symbol needs to be interpreted to complete the recognition part. The interpretation part mainly depends on the clef that is used by the music instrument and also it needs to be combined with the vertical translation value to get the final result. When the input is normalized cross correlated with the template, the symbols corresponding to the template will be recognized. After this step the vertical translation is determined by using the bottom most staff line as a reference. But after this step the output that is produced will have similar symbols next to each other, hence there is a need to put it all together.

I. Conversion to MusicXML

For converting the recognized symbols into xml file, the symbols are arranged according to their (x,y) coordinate values. A structure consisting of (x,y) coordinates, staff number, duration, stem, octave, step, voice and alter is created. The structure will be converted into table. The table is sorted and the XML file is generated. The structure for a half note with stem down is shown in Figure 4.12. Similarly the structure parameters vary for different notes. The output xml file defines each and every symbol present on the input music score sheet. This file can be played using MuseScore to generate the musical tone. The XML file generated consits of the following

1) Declaration:

APPENDIX A
PROOF OF THE FIRST ZONKLAR EQUATION

Some text for the appendix.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LATEX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

Kartik Bhargav Top entrepreneur