Sheet Music Reader

Sevy Harris, Prateek Verma
Department of Electrical Engineering
Stanford University
Stanford, CA
sharris5@stanford.edu, prateekv@stanford.edu

Abstract—The goal of this project was to design an image processing algorithm that scans in sheet music and plays the music described on the page. Our algorithm takes a digital image of sheet music and segments it by line. For each line, it detects the note types and locations, and computes the frequency and duration. These frequency and duration pairs are stored in a matrix and fed into the audio synthesizer, which plays back the music. For simple and non-noisy input images, the audio output for this algorithm was recognizable and played the pitches accurately, excluding the presence of accidentals.

1 Introduction

Much like the problem of optical character recognition, there are many useful applications for automating the process of reading sheet music. An optical sheet music reader allows a musician to manipulate scores with ease. An algorithm that converts an image to note frequencies can easily be transposed to other keys and octaves. Transposing music is a very tedious process when done by hand. A sheet music reader also has the potential to quickly translate handwritten music into a much more readable digital copy. Musicians can also use the sheet music reader as a practice aid to check their own sound with a digital reference for a passage. For this project, we focused just on translating the image to audio output. This algorithm takes a digital image of sheet music and converts it into audio samples that are played back to the user. Section 2 reveals the specific implementation of the algorithm, and section 3 explains the results we achieved. The conclusion and future work is detailed in section 4.

2 Implementation

This image processing algorithm works in three general steps, shown in Figure 1. First, the algorithm performs segmentation and several preprocessing functions to enhance the quality of the image and make it easier to analyze. Then, the key objects are detected using morphological operations. In the final stage, the detected notes and other objects are combined and analyzed to produce frequency and duration pairs that are sent to the synthesizer as a matrix.

2.1 Segmentation and Preprocessing

First the image is binarized using Otsu's method because it makes the image cleaner and the calculations easier. The image is inverted so that the objects of interest, the notes, have a high intensity and the background is zero. Next, the sheet is split into separate lines using the following method based off the segmenting algorithm described in [1]. First, the image is eroded by a horizontal line to emphasize the staff lines and disregard most everything else in the calculation. The horizontal rows are summed into a column vector normalized by the width of the image. These row projections are binarized using Otsu's method to find the lines that are most likely staff lines. Next, the complement of this vector is eroded with vertical lines of variable length to find the regions with lots of blank rows. The divisions are drawn roughly in the middle of the blank

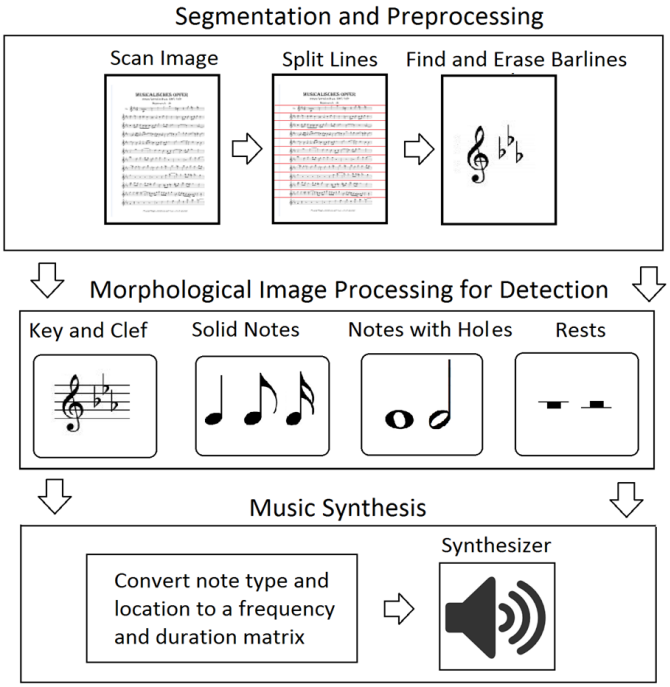

Fig. 1. Shows the flow of the image processing algorithm.

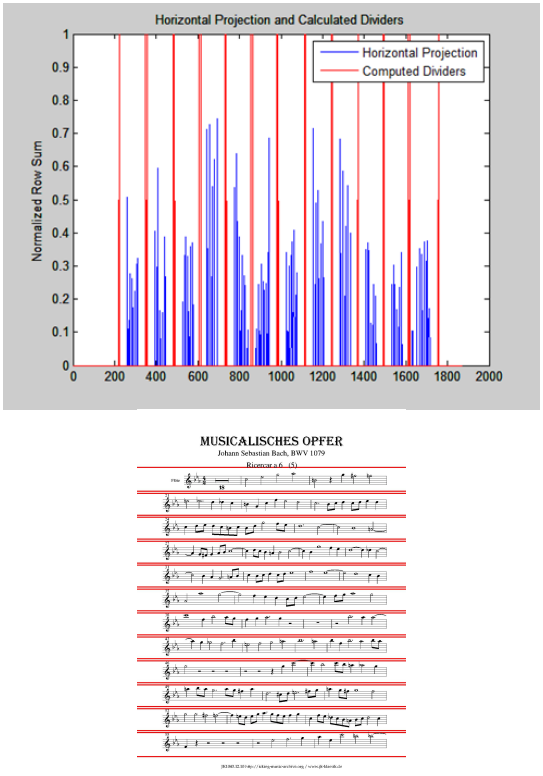

Fig. 2. Shows the row sums of a page of sheet music (top) after erosion and the corresponding image (bottom) with the dividers drawn in.

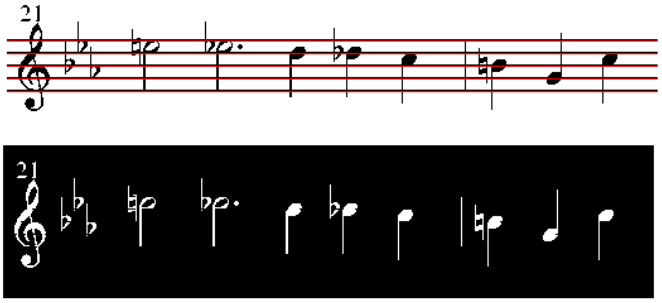

Fig. 3. Shows the detected staff lines highlighted in red (top) and then removed from the image. row sums of a page of sheet music (top) after erosion and the corresponding image (bottom) with the dividers drawn in.

spaces so that high notes and low notes are still included in the segmented image. Figure 2 shows the sum of the rows after image erosion with the calculated dividing lines drawn in red. It also shows the corresponding image with the divisions drawn in.

Once the image is segmented into different lines, the next step is to identify the individual staff lines. These positions are used later for identifying the key signature and calculating the note frequency. The staff lines for each line of music are found by summing the rows and finding the 5 equally spaced rows that maximize the sum. These rows are then subtracted from the image. The results of finding and erasing the staff lines are shown in Figure 3.

The staff line locations and images without the staff lines are then passed on to the next part of the algorithm for note detection.

2.2 Object Detection

The objects of interest (notes, key signatures, rests, etc...) were detected in the second stage using several different morphological operations important elements of the image, mainly the notes, the clef, and key signature are detected at this stage.

2.2.1 Clef Detection

Identifying the clef of the music (treble or bass for this application) is simplified by the fact

that the clef is always the first object drawn on the left side of the staff. The left edge of the staff is found by looking at sums of the vertical columns for the first major peak in activity. The end of the clef is found by looking at the vertical sums for the first transition below the median. Bass clef differs from treble clef in that it has almost no pixels between the bottom two staff lines, so we sum the pixels in this area, normalize by the spacing between staff lines, and then threshold to get the classification.

2.2.2 Key Signature Identification

The left boundary of the key signature was found in a similar manner to the boundary of the clef. The vertical sums of pixels were taken and then the boundaries were determined to be the major transitions in intensity levels. The right boundary was taken as a fixed width guaranteed to at least contain the whole key signature.

First, the algorithm detects the sharps by eroding with a vertical line and applying several filters on the resulting connected components. The algorithm looks for pairs of connected components that are thin, close together, and similar in height and vertical offset. An example of this culling is shown below in Figure 4.

Fig. 4. Shows the process of identifying sharps in a key signature. The original image (left) is eroded and then dilated with a vertical line (middle) and then extra filters based on the shape are applied (right).

If no sharps are detected, the algorithm looks for flats by eroding with a small disk shaped structuring element. The algorithm then counts the number of sharps or flats detected and uses the known key signature conventions (the order of sharps or flats must follow certain rules) to determine how many accidentals are a valid part of the key signature.

2.2.3 Quarter and Eighth Note Detection

Quarter and eighth notes were grouped together for detection because they both have filled black circles. These notes were detected by eroding and dilating by a disk shaped structuring element that was normalized to the space between the staff lines. The resulting connected components were then thresholded by eccentricity (not too circular or too linear) and area (must meet a minimum area requirement). If the variance in the area of the remaining connected components was large enough, components more than 1.75 standard deviations below the median area were removed. An example of this process is shown in Figure 5.

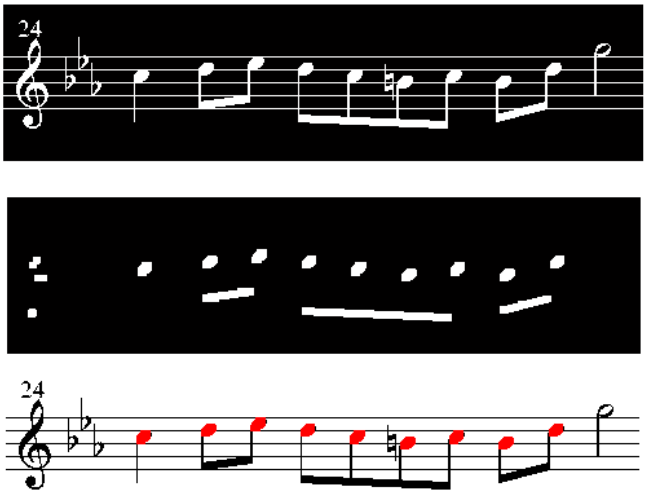

Fig. 5. Shows the process of detecting quarter and eighth notes. The original image (top) is eroded and then dilated with a small disk (middle) and then the connected components are thresholded by eccentricity and area (bottom).

2.2.4 Whole Note and Half Note Detection

The algorithm for detecting whole notes and half notes is very similar to the method used to detect quarter notes and eighth notes. The key idea is to fill up the holes in an image and then subtract this from the original image in order to get the image which has non-filled notes present.

There exist many methods for filling up regions based on connectivity mapping, closed regions etc. We use the watershed algorithm [2] in order to detect these closed regions. However the main drawback is that the staff lines form closed regions without removal which result in filling up the entire regions. Thus we use a preprocessing step to remove the staff lines. This, however opens up the connected regions which are otherwise difficult to fill and detect.

The approach we use is as follows: We first dilate the image with a structuring element proportional to a fixed ratio of the size of the width of the staff. After dilating the image, the gaps due to removal of staff lines are removed, however the original size of the image is increased. This image is then used to fill up all the closed regions using the watershed algorithm. Once we have an image containing the filled up regions, we subtract the original undilated image in order to obtain an image which has dots present only where there were initially whole notes present. The algorithm for dot detection is used in previous step in order to detect the position of the whole notes.

An example of whole and half note detection is shown in Figure 6.

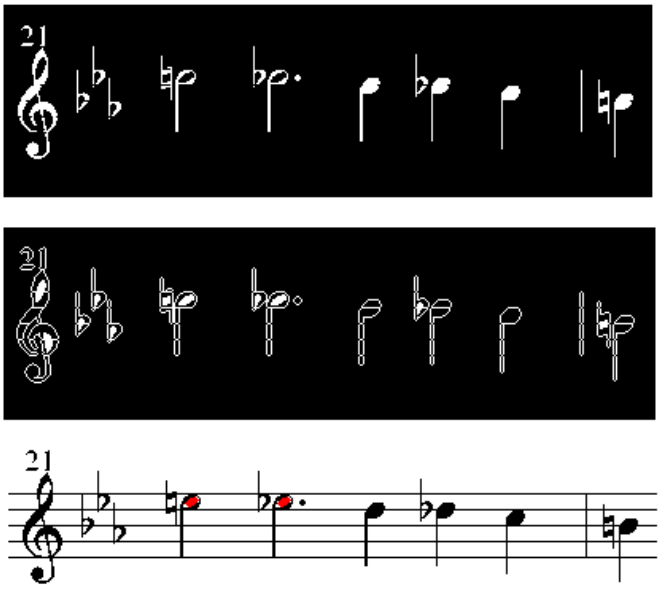

Fig. 6. Shows the process of detecting half and whole notes. The original image (top) has the holes filled (middle) and then uses erosion and thresholding to detect the notes (bottom).

2.2.5 Detecting Dotted Notes

For accidentals we have detected only the dots which are used to tell us about the changes in the timing of the note. This is a difficult task as there may occur spurious regions occurring due to errors in binarization, removal of staff lines etc. The method for dotted note detection is as follows. We first segment the image into each of the individual staffs. The vertical lines and the horizontal lines are removed to give a decomposed image consisting of only the accidental, whole and solid notes as well as the flags. This image is used as an input in order to detect the accidental notes. The characteristics that separate these dots from other symbols are the eccentricity and the size. The dots have a high value of eccentricity which we enhance further by dilating the image. The dilated image is first used to find all the connected regions using a 4-neighborhood connected definition. After finding all the connected regions, these regions are then thresholded by the length of the connected region. The accidental dots will have size smaller than the staff width. Then on all the connected regions separated by connectivity, they are separated by eccentricity to yield the desired dots.

2.2.6 Detecting Rests

Many of the rests are rectangular elements that are present in an image. In order to detect these, we take our input as the horizontal and vertical line removed image. The property that we use of the pauses is the fact that the pause elements will have zero variance when we sum up the pixels along the vertical axis in the negative binary image. All other elements have non zero variance due to changes in the width, length etc.

2.3 Audio Synthesis

Once the whole notes, quarter notes, and half notes are identified, the next step is to identify the note location and get the frequency of the corresponding notes.

We use an additive synthesis algorithm [3] instead of storing the recorded samples. This

has many advantages i.e. we can have flexibility of changing the tempo of the output, reference frequency as well as the flexibility of playing multiple instruments with a fixed number of small parameters. For real time applications, it may be faster to synthesis a sample rather than retrieve it from memory. The parameters we use in this are the envelop of the instrument as well as the temporal envelop of the volume dynamics. These give a sense of naturalness to the sound. We have not considered the temporal evolution of the timbral coefficients.

3 Results

We tested our algorithm on five simple pieces of music. While none of them was executed perfectly at the output of the synthesizer, all of them were easily recognizable. The algorithm successfully segmented and identified the key signature for every line of the five pieces we used for testing. The accuracy of the algorithm was measured by generating images that allowed us to visually inspect the notes that were detected. Two examples of these comparison images are shown below in Figure 7.

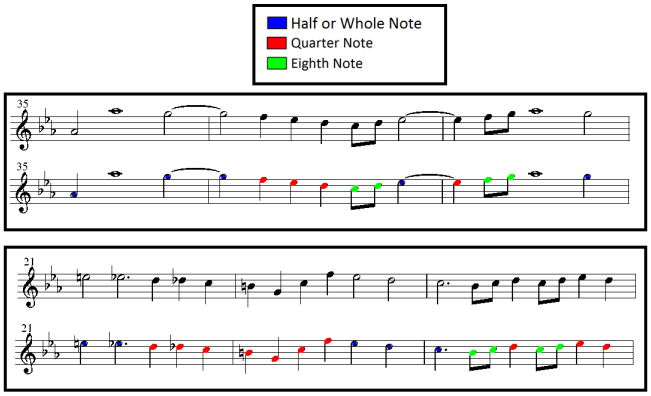

Fig. 7. Shows the original image and the types of notes detected.

The table in Figure 8 summarizes the error rate of the algorithm when detecting quarter notes, eighth notes, and whole or half notes. Whole and half notes were combined because the algorithm does not yet distinguish between the two. The error rate is calculated as the total number of insertions, deletions, and substitutions for a piece divided by the total number of notes.

Song	Quarter Notes	Whole Notes	Eighth Notes
Ode to Joy	0.0189	0.33	0.4286
Mary had a Little Lamb	0	1	0.3438
America the Beautiful	0.0227	0.75	0.875
Frere Jacques	0	0.25	0
Twinkle Twinkle Little Star	0	1	0
Overall	0.0128	0.667	0.3818

Fig. 8. Shows the accuracy of the algorithm for each song.

We found that the algorithm rarely makes mistakes on quarter notes, but regularly substitutes quarter notes for eighth notes and misses whole and half notes. This is usually due to the presence of a staff line in the middle of the whole note, which makes the hole filling method less effective. The result is that the algorithm tends to have inaccurate rhythm, but plays the correct pitches for all notes detected (excluding notes with accidentals).

4 Conclusion and Future Work

For simple and non-noisy images, this image processing algorithm can transform digital sheet music into recognizable audio output, but there is still much room for improvement.

This algorithm is largely missing the handling of accidentals that are not described in the key signature at the beginning of each line. This significantly increases the number of wrong notes as the complexity of the piece increases. Additionally, this algorithm does not yet distinguish between whole notes and half notes. This can be accomplished through the detection of note stems and the calculation of proximity to non-filled notes.

On top of the improvements that can be made in robustness, there are dozens of musical symbols that are not even mentioned in this algorithm, such as the dynamics markings, time

signatures, triplet groupings, and accents. This makes optical music sheet reading a practically limitless problem. There is great room for improvement in robustness and functionality, but this algorithm provides the initial end to end framework for an optical sheet music reader.

REFERENCES

[1] Bellini, Pierfrancesco, Ivan Bruno, and Paolo Nesi. "Optical music sheet segmentation." Web Delivering of Music, 2001. Proceedings. First International Conference on. IEEE, 2001.

[2] Bieniek, Andreas, and Alina Moga. "An efficient watershed algorithm based on connected components." Pattern Recognition 33.6 (2000): 907-916.

[3] Horner, Andrew, and James Beauchamp. "Piecewise-linear approximation of additive synthesis envelopes: a comparison of various methods." Computer Music Journal (1996): 72-95.

Appendix A: Work Breakdown

Sevy:
Segmentation and Preprocessing
Clef Detection
Key Signature Identification
Quarter and Eighth note detection
Detecting Solid Notes

Prateek:
Rests
Whole notes
Dotted notes
Synthesis