Hanfei Mei
Qiuwei Shou
Alan Wu
CSCI-B657: Computer Vision
2/9/16

Assignment 1 Report

I. Instructions on running the code

1. We used music1.png to produce all images submitted on github. A discussion of results using other images takes place in a latter section.
2. To reproduce scores4.png, run line in template_match_4 function, which returns a vector of symbols
    a. write_detection_image("scores4.png, blank, scores4);
3. To reproduce detected4.png, run line in the main() function
    a. write_detection_image("detected4.png", total_4, input_image);
    b. fyi, total_4 is a vector of all the symbols reflecting all 3 templates
4. To reproduced edges.png, run line in template_match_5 function, which returns a vector of symobls
    a. write_detection_image("edges.png", output, image_edges);
5. To reproduce detected5.png, run line in the main() function
    a. write_detection_image("detected5.png", total_5, input_image);
6. To reproduce staves.png, run Hough in the main() function which finds the highest votes for top_row and row_spacing. Then run line in main()
    a. write_png_file("staves.png", input_image, input_image, output); [this is an RGB function]
7. To reproduce detected7.png, run line in main()
    a. write_detection_image("detected7.png", total_7, input_image);
    b. please make sure to use the library of templates, each of which is resized and labeled according to the image to which it corresponds
        i. ex. template1.png is for music1.png
        ii. ex. template1_4.png is for music4.png
    c. detected7.txt is output with detected7.png

II. Assumptions and design decisions

1. Assumptions
    a. For the Hough transform, the two axes in Hough space are top_row and row_spacing. Voting is based on the number of pixels accumulated from a sweep across the image for various top_row and row_spacing values of 5 lines at a time.
    b. The clefs caused many false positives when running using the method from #5.

However, a lower score threshold caused many missing notes. We opted to detect as many notes as possible despite getting numerous false positives on the clefs.

c. We used Mac's preview tool to rescale the templates, creating our own library of templates to use on music(n>1).png files. We assumed this approach is acceptable.

2. Design decisions
   a. Setting the optimal threshold for edge detections proved very important in how well the detection worked. With too low a threshold, thick lines get passed. Then those thick lines get interpreted as having 2 edges in the ensuing scoring function, which in turn, leads to a poor result.
      i. Individual thresholds needed to be set for the different templates
      ii. Also, each input image needs custom thresholds set
   b. We found that it is nearly always beneficial to run edge detection – Sobel is nice with built-in lowpass filter and gradient functionalities – prior to performing any scoring or Hough transforms. We did not have time to implement a (full-fledge) Canny edge detector, which could possibly have yielded better results.

III. Accuracy of program on test images; when they work well and when they don't

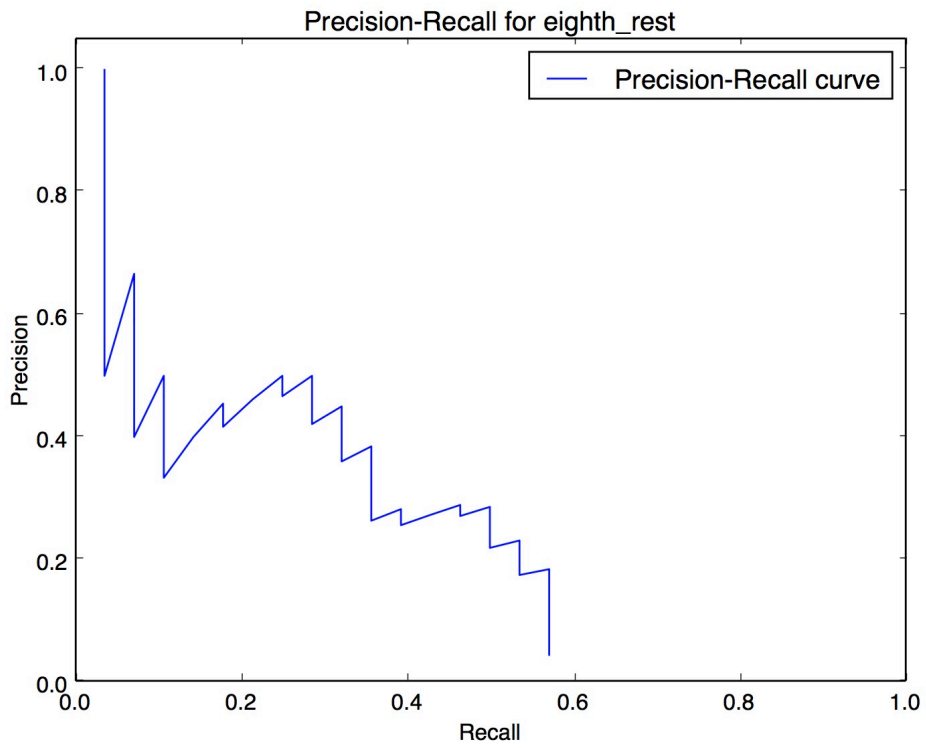
For music1.png, using method 4, we were able to detect all 3 symbols with no false positives. With method 5, we were able to detect roughly 80% of the notes without any false positives over the clefs and the curved note stems. With a lower threshold, we detected more notes, but the false positives increased significantly. We were able to detect all of both types of rests.

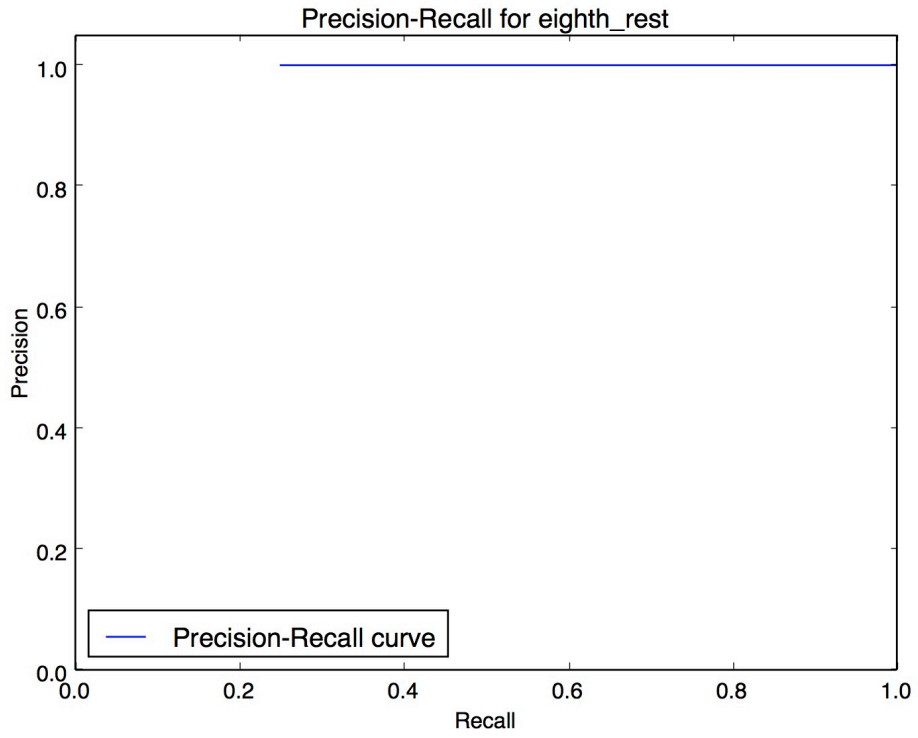
We did not observe a superior performance of method 5 over method 4 in detecting music2.png, which was noisier than music1.png, although method 5 was expected to handle noise better. With re-scaling of the symbols (i.e. modified templates), method 4 performed superior to method 5. However, the notes detected dropped substantially compared to music1.png. All the rests were detected.

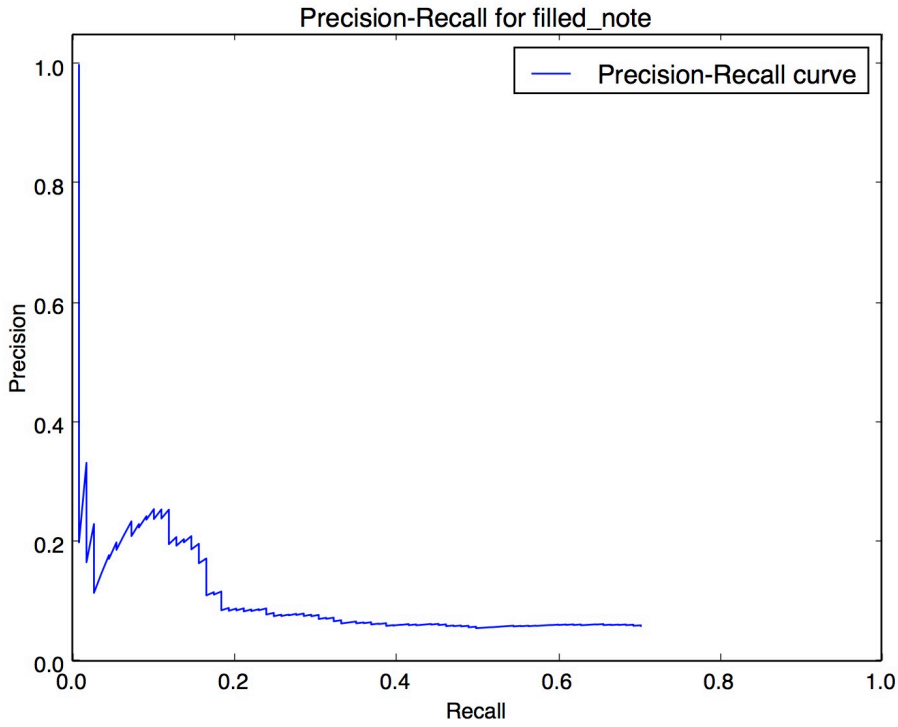
Detecting music4.png was challenging, and we achieved moderate results using method 4, which was far superior to method 5. Notes outside of the staves proved more difficult to detect. Also, the curved stems on some notes fooled the algorithm to think that they are notes, creating false positives. We also detected the eighth rests on the first stave, but not the second stave. This may be due to a low resolution re-scaled template.

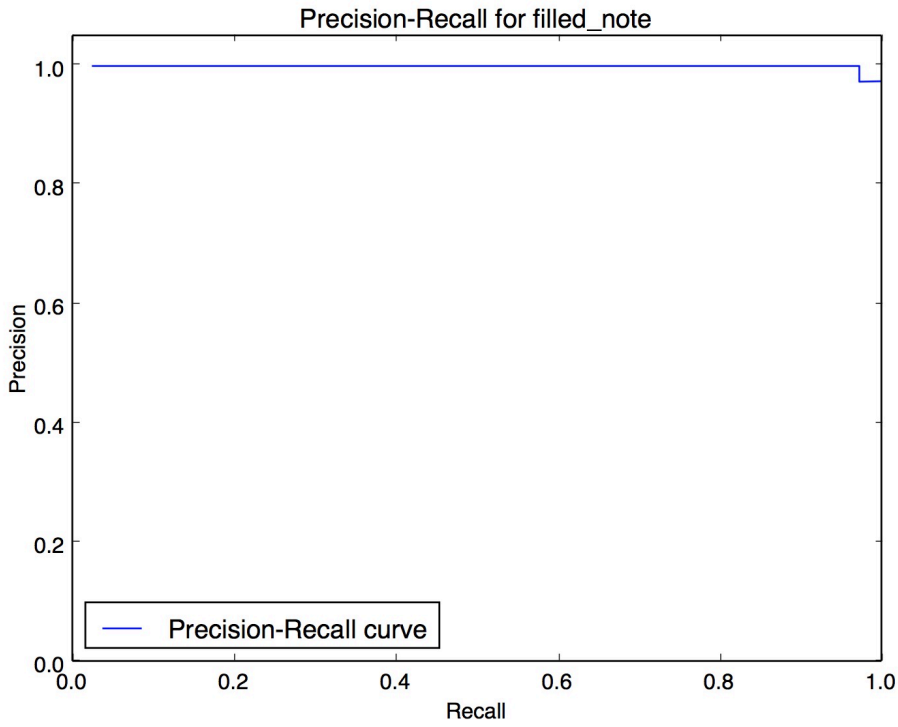
We were able to consistently detect the staves on all test images provided and the spacing for each stave were all accurately reported using our Hough transform. Although the double thick lines on the sixteenth notes could add confusion as representing a line, sweeping the entire width of the image for the scoring is able to circumvent this possible distraction.

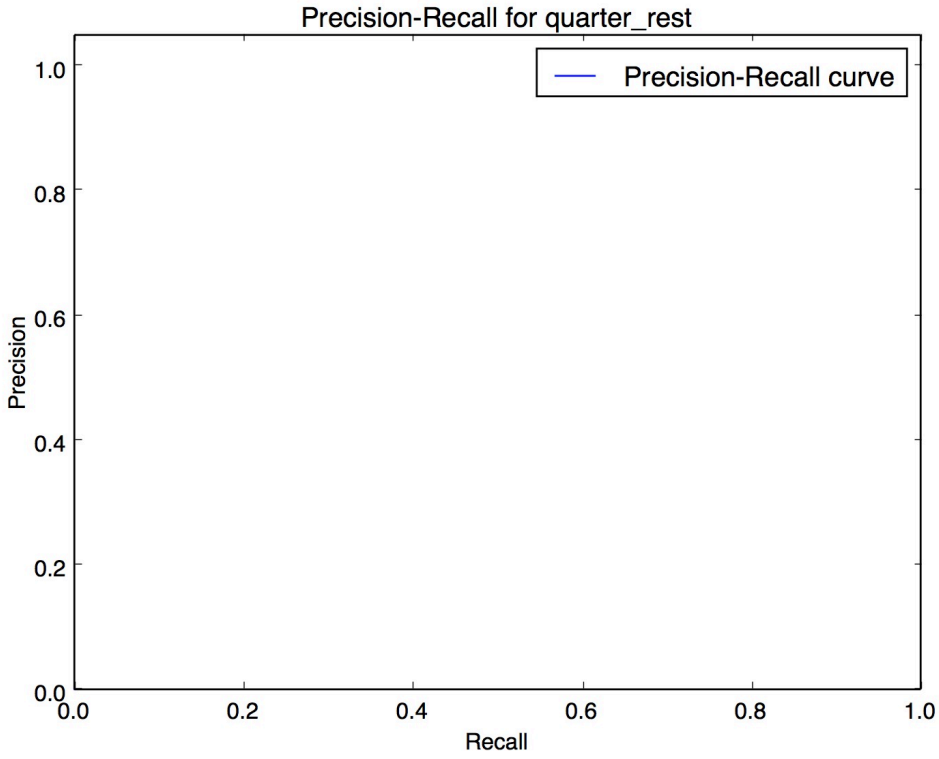
IV. Results of evaluation files from staff provided python test script and ground truth files

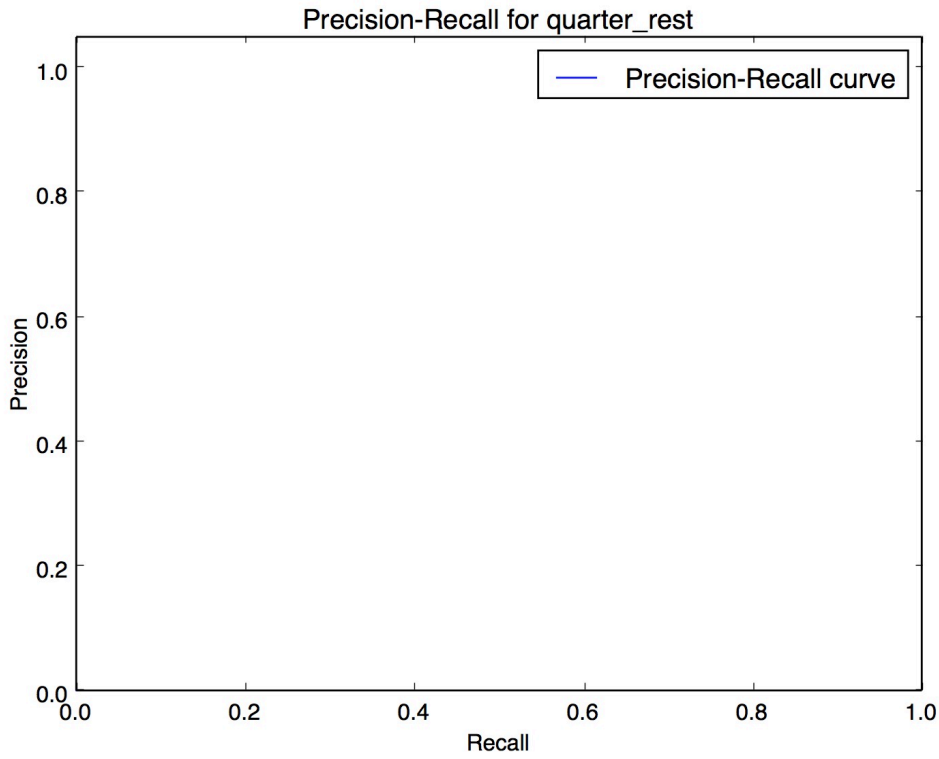





Precision-Recall for filled_note
Precision-Recall curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


Precision-Recall for filled_note
Precision-Recall curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


Precision-Recall for quarter_rest
Precision-Recall curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


Precision-Recall for quarter_rest
Precision-Recall curve
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

V. References

Aside from help from awesome AI's and looking up online references for C++ syntax on cplusplus.com and stackflow.com (none of us really knows C++ well and each has had quite a steep learning curve), there's not really many other references we used except the idea behind the normalization function found on the web – this is identified in the code next to the function "standardization()". This makes all the votes normalized to (0, 255). As already stated above, we used Mac's built-in resizing tool to create our library of templates.

VI. How to improve in the future

Despite our best effort to set the edge detector threshold, a "thick" edge gets developed when two edges in close proximity get "detected." This effect throws off the scoring function. Using a non-maximum suppression method in the future will improve performance.

Also, as edge detection is a critical step, a more robust approach using Canny will also likely improve results.

Lastly, an improved re-scaled image template that is of higher resolution leading to finer edges will help our detection approach.