Final Report

Team

Folk Detect Innovate Bison

Members

Ian Hays

Xavier Beynon

Ashley Ng

Riley Gibson

Overview

The overarching goal of the Folk Detect Innovate Bison (FDIB) team was to identify climbing routes on a rock wall.

The final objectives for the project were:

- Stitch a series of images containing a rock wall into a single panorama
- Find the rock wall tapes on the panorama
- Determine which color (or colors) of tape are within a group of tape
- Organize the found tapes into groups by color

We accomplished the above goals and detailed examples of the process follow in the 'Results' section, along with photos.

In addition to the above objectives we created tools to aid in the usage and testing of the program:

- A GUI with slider to choose the best thresholds to use in tape detection
- A tool to choose tape regions in a photo as a means of easily establishing a ground truth to use in testing
- Automated testing of tape identification and color identification

Background

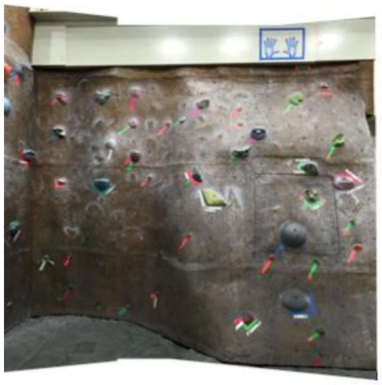
We found that very little research has been done on identifying holds on gym walls, let alone natural rock faces. Instead most research has focused on the anatomical problem of how humans climb walls, glossing over the computer vision aspect by pre-identifying holds or challenging the robot on a wall with only a single route. For instance, Robotic Rock Climbing using Computer Vision and Force Feedback by Linder, Wei, and Clay describes a small four limbed robot that identified routes by computing a series of "stable stances". However, hold identification was simplified by having a single color and shape of hold on a grid. Nevertheless their method of identifying stable stances would be a fascinating next step to our project.

We did consult some references to aid us in our tools. These included the trackbar GUI as well as an algorithm to check rectangle overlap for our tape accuracy check.

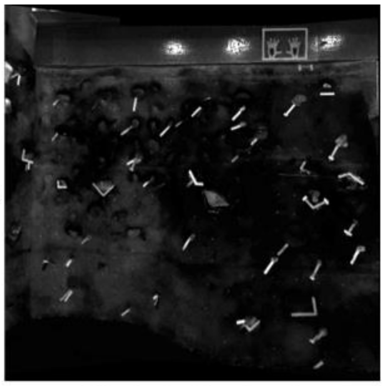
Methods

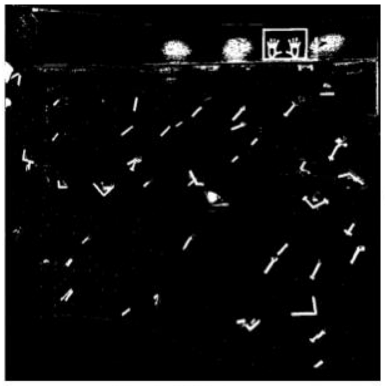
Pipeline:

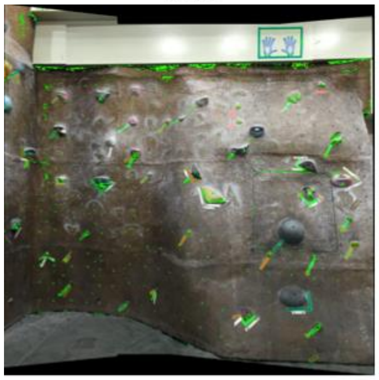
To achieve the project's goals we recognized the need for a multi-step pipeline to process a set of input images and extract routes. The current state of the pipeline is below:

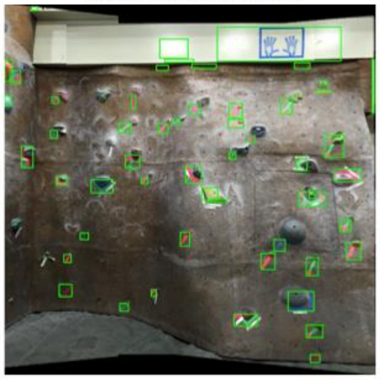


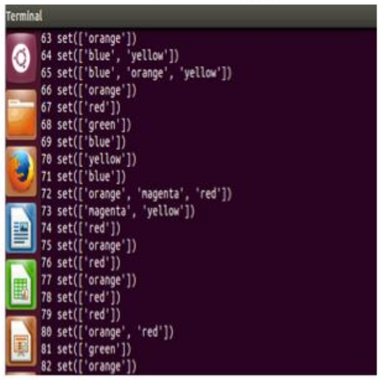
Image


Extract saturation channel


Threshold for tapes

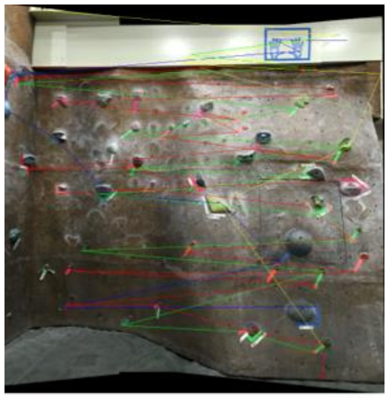

Find Contours


Filter by minimum area



```
63 set(['orange'])
64 set(['blue', 'yellow'])
65 set(['blue', 'orange', 'yellow'])
66 set(['orange'])
67 set(['red'])
68 set(['green'])
69 set(['blue'])
70 set(['yellow'])
71 set(['blue'])
72 set(['orange', 'magenta', 'red'])
73 set(['magenta', 'yellow'])
74 set(['red'])
75 set(['orange'])
76 set(['red'])
77 set(['orange'])
78 set(['red'])
79 set(['red'])
80 set(['orange', 'red'])
81 set(['green'])
82 set(['orange'])
```

Extract hue channel, classify by color


Group into paths by color

Throughout development we attempted many other pipelines that involved steps such as edge detection, rectangle detection, color averaging, and anatomical filtering. Our final pipeline is the conclusion of the approaches we found to work best for a wide range of images we took at the Gregory Rock Wall. However, there are merits to other approaches and given more time to develop, we would like to combine their strengths. For instance, a better threshold heuristic might weight the "rectangle-ness" of a given region with its "neon-ness".

Thresholding:

We begin by converting the image into the HSV/HSL color space and performing a binary threshold on it using the cv2 threshold function. The threshold value used was found by plotting the saturation values of all known tapes and finding a value that would be inclusive of 95% of all tapes (85/255) using the threshold GUI tool. We were not concerned about false positives at this stage since they would be filtered later in the contour stage. The primary interest was in filtering out most of the wall and other extraneous information to give us a smaller data set for contour detection.

Contours:

Originally Xavier wrote his own algorithm to divide the thresholded image into tape-regions. It worked by flood-filling any pixel it found and saving the region as a set of points. Upon realizing that this functionality could be achieved by calling cv2 findContours, Xavier was publically shamed and assigned to the color detection problem. Contours are further filtered by a min area requirement of 64 pixels. Similar to the thresholding, this value was decided upon by plotting the min areas of known tapes and finding the minimum that would be inclusive of the largest 95% using the minimum area thresholding GUI.

Color Detection:

Once we had a set of contours representing our tape groups, we next went about grouping the contours based on the colors of the tapes. We looked at every pixel within a given contour and recorded its color. If enough pixels were recorded of a given color range, then we determined that the contour contained that color. We set a threshold for the minimum number of pixels that must be of a color to determine that the contour contains that color. We determined that a threshold of 20 pixels was the most accurate.

This method also allowed us to find multiple colors within a contour. This was especially useful because it accounts for the situation where a piece of tape is actually comprised of two tape pieces.

To do the actual grouping of the tapes and color comparisons, we used the hue value of the contour and used its location on the hue color wheel to determine what color it was. Since we did not care to distinguish between colors like cool red, warm red, and plain red, we squashed the ranges into human appreciable segments like red, blue, or green using a manual process of trial and error. A more advanced approach would have been to generate this range automatically based on the distribution of hue ranges, but that ended up being out of scope for the project duration.

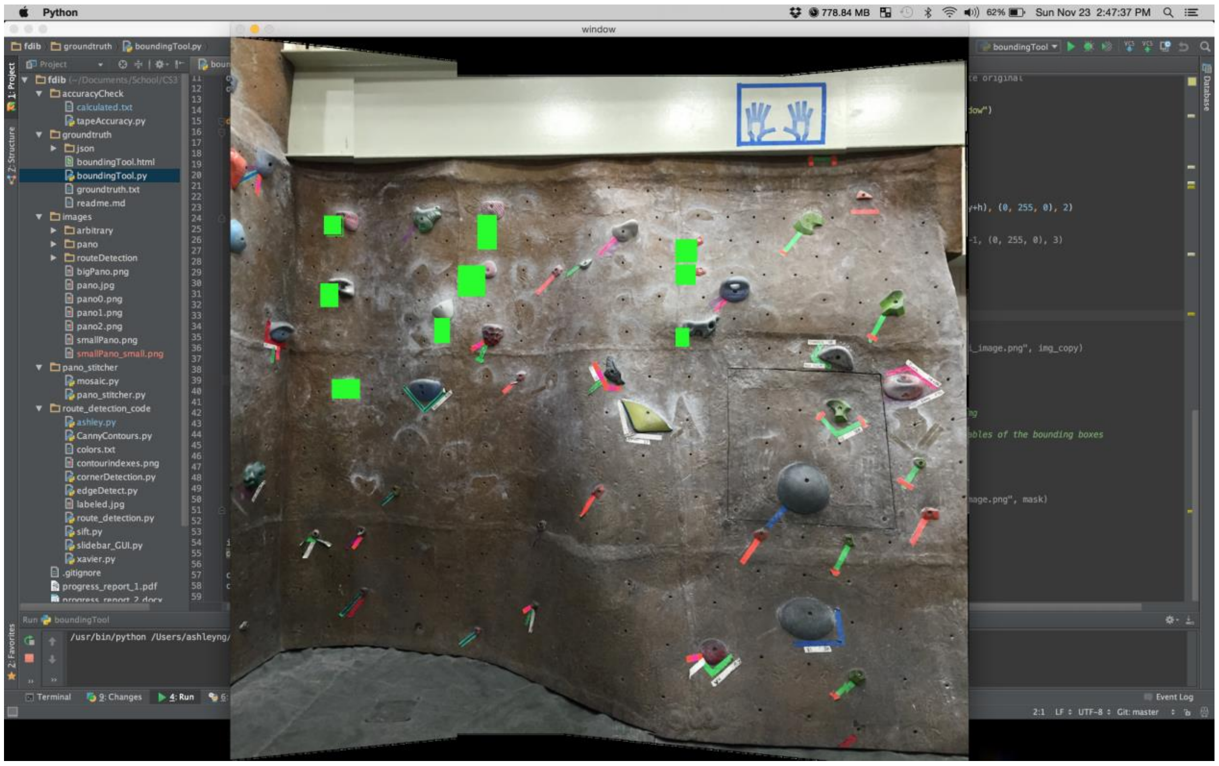
Path Grouping:

The final step in the pipeline was to group holds by color. This is done by first computing the unique colors found among the tapes. Then for each color, the relevant holds are added to the group. Therefore the mapping of tapes to groups is many->one; a given tape can belong to many groups, as it may be labeled under multiple colors.

Tools

Ground Truth Tool(s):

To test the correctness of our various goals, we developed two separate tools - one in python and one in html - that allow manual selection of square regions on a given photo. When using the tool, a file name to save can be specified for the regions selected on the photo so that we can create separate ground truths for separate tests. For each panorama we tested on, we manually selected the tapes of each path and categorized them by file name, then also created a ground truth file containing each piece of tape.
Once we have the ground truths specified for each path on an image, we can use our calculated contours and test their accuracy by comparing them to the manually defined boxes from the ground truth tool.

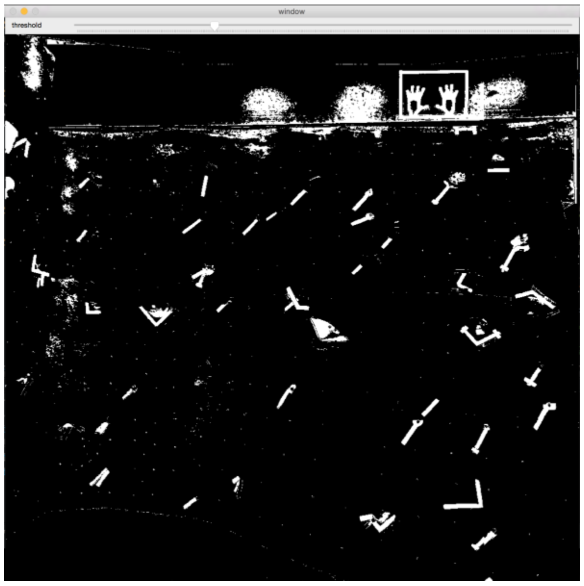


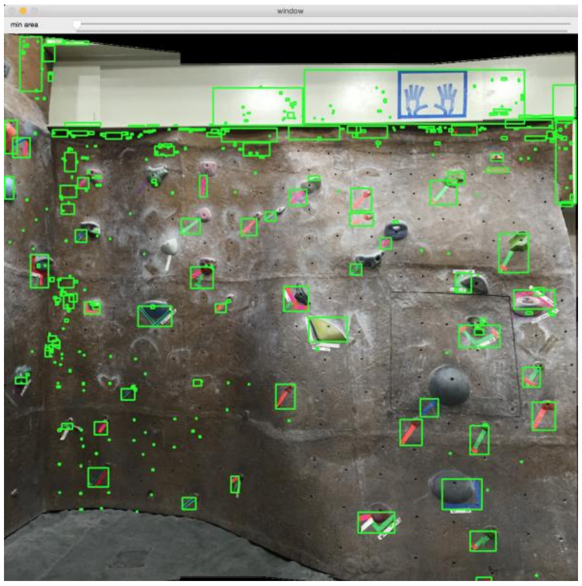
```
te original
dow")
y+h), (0, 255, 0), 2)
-1, (0, 255, 0), 3)
_image.png", img_copy)
ng
ables of the bounding boxes
mage.png", mask)
```

```
/usr/bin/python /Users/ashleyng/
```

Thresholding GUI:

We developed two GUIs to aid us and future users of the program in determining the most accurate threshold values possible: one for saturation and one for minimum area of a contour. The default program runs with threshold values that we determined to be accurate from our testing, but we decided it prudent to allow more flexibility by adding additional user options through the GUI. The first GUI has a slider to control the minimum saturation that should be considered an actual contour. The second GUI has a slider to control the minimum box area of a contour that passed the saturation thresholding.



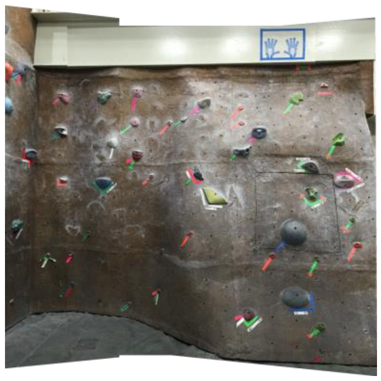


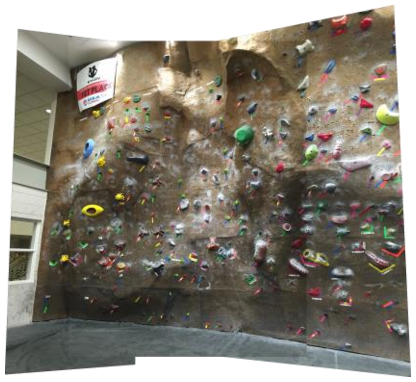
The right image shows the bounding box GUI when the slider is at the far left indicating a bounding box minimum area of 0, which clearly shows the necessity of a nonzero threshold value.

Results

As we worked on the rock wall path detection, we continuously updated our objectives to reflect what we felt were the most important goals that we hoped to meet. Based on these goals, our results met our objectives in terms of accuracy based on the ground truth values that we were able to find using the many tools we developed, discussed above.

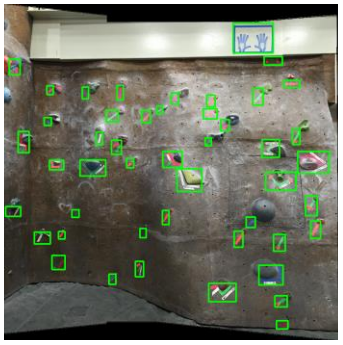
Our first goal was to accurately stitch together a panoramic view of the rock wall that we would use for our analysis. Our desire was that this stitching process would create a panoramic view with minimal color and shape distortion on both the tapes and holds in order to perform the tape and route detection. Although there is no way to empirically determine the panoramic view's accuracy, we were able to detect tape and routes based on this image, which was our goal. This is seen too in the multiple panoramic images that we were able to create from the multiple climbing wall images we obtained:

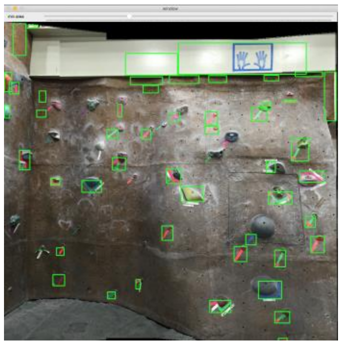


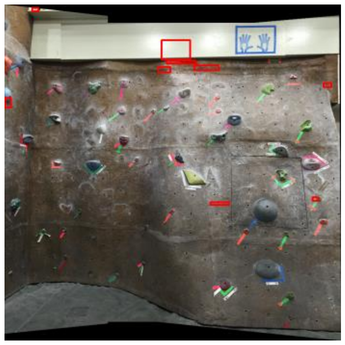


Our second goal was to accurately detect the presence of colored tape on the wall so that we could use it to detect routes with a desired accuracy of 60%: that is to say we detect 60% of existing tapes on a wall. This was done using the thresholding and contouring methods explained above. To determine the accuracy rate, we used the bounding box tool shown above to save the ground truths and calculated points as JSON files, located in the directories groundtruth/groundtruth.txt and accuracyCheck/calculated.txt directories respectively.

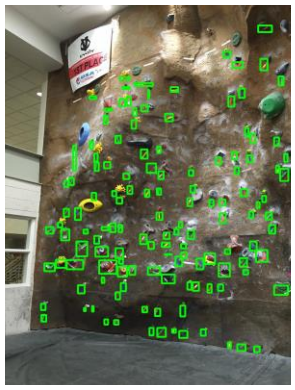
For the first test case, based on our ground truth values, we were able to detect 91.4% of existing tapes on the wall, far exceeding our initial goal of 60%. This calculation is based on the true number of tapes, determined by hand using the bounding box tool, and the number of true positives found. A result is considered a true positive if a tape is detected and matched to a tape in the ground truth data set.

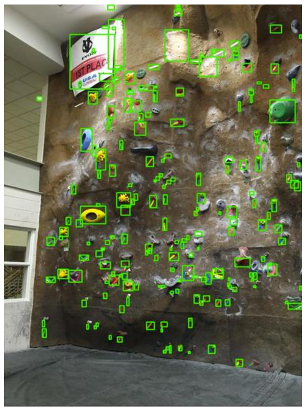

Ground Truths

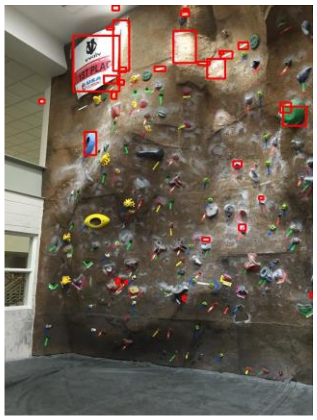

True Positives


False Positives

For the second test case, based on our ground truth values, we were able to detect 85.38% of existing tapes on the wall. Below is an image describing the tapes detected on the larger, more densely populated climbing wall. **more images to be added:** http://imgur.com/a/hhP84

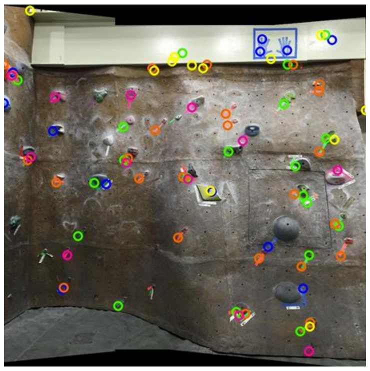

Ground Truths

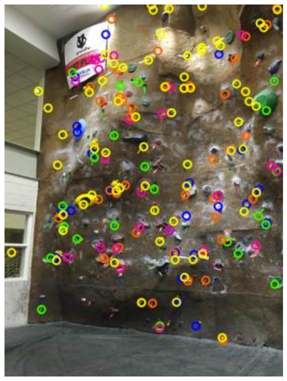

True Positives


False Positives

Our final objective that we sought to meet was, based on our tape detection, to group the tapes accurately according to hue and determine a route from ground to ceiling based on those groupings. Our goal was to include at least 50% of the known tapes of a color, determined by hand using our developed tools, in that specific color's route. Accuracy for this objective was determined by knowing the number of tapes of a given color, a ground truth determined for testing, and comparing that to the number of tapes found for a specific colored path. Tapes found were only considered if they were a true positive, meaning that it was a part of the correct color's path; this was determined by hand, having the route detection draw the found path for our inspection.

For the two panoramic wall images, based on our ground truth values, we were able to detect a tapes, and therefore routes, with an overall accuracy of 69.23% and 47.11% for the small panoramic image and the large panoramic image, respectively. The lower accuracy on the larger wall can be attributed to the much larger number of tapes present on the wall, leaving more room for misclassification and omission; full results listed below, based on color of the route.





*Route Detection Results:

Image 1 (smallPano)	Image 2 (bigWall)
Output for color: blue Total Successes: 77.7777777778 Total Failures: 22.2222222222 Total Rando: 5 Output for color: green Total Successes: 65.2173913043 Total Failures: 34.7826086957 Total Rando: 5 Output for color: orange Total Successes: 73.9130434783 Total Failures: 26.0869565217 Total Rando: 8 Output for color: magenta Total Successes: 60.0 Total Failures: 40.0 Total Rando: 6 Total output: **Total Successes: 69.2307692308** Total Failures: 30.7692307692 Total Rando: 24	Output for color: blue Total Successes: 31.5789473684 Total Failures: 68.4210526316 Total Rando: 13 Output for color: green Total Successes: 45.7142857143 Total Failures: 54.2857142857 Total Rando: 6 Output for color: orange Total Successes: 56.5217391304 Total Failures: 43.4782608696 Total Rando: 25 Output for color: magenta Total Successes: 58.3333333333 Total Failures: 41.6666666667 Total Rando: 18 Output for color: yellow Total Successes: 69.2307692308 Total Failures: 30.7692307692 Total Rando: 56 Total output: **Total Successes: 47.1074380165**

	Total Failures: 52.8925619835 Total Rando: 118

*Success = True Positive, Failure = Missed hold, Rando=False Positive