Feature Planning

[Webcam Research (nevermind that, mission complete)](#_89qc1gqwaed8)

[2/7/18 Discussion Notes](#_8r4fxbwd04ux)

[2/10/18 Resources](#_xygc9xcxyrm0)

[2/14/18 Discussion Notes](#_4hiw73ew56gh)

[2/18/18 Meeting Notes](#_n5q0s3z8uxtq)

[2/21/18 Meeting Notes](#_aim2w0e4anxn)

[2/25/18 Meeting Notes](#_yubpsi8sgywr)

[2/28/18 Meeting Notes](#_astfixif4wv1)

[3/06/2018](#_kori4g9elgfe)

[3/07/2018](#_i22xzgf9ghvd)

# Webcam Research (nevermind that, mission complete)

* Logitech C920 (around $50)
* Logitech C930e (around $80)
  + Better than the C920 in terms of color, resolution, and manual focus
* **QUESTIONS:**
  + What is the price range we should stick with?
  + **How many cameras should we get?**
    - At least two, preferably **4**
  + **Where will the cameras be mounted?** 
    - One camera needs to be for the mini mapping area
    - At least one camera needs to be used by the robo arms, maybe one at the head and one at the body

# 2/7/18 Discussion Notes

* Depth sensing and recognition features need to use the same camera for same image frame
* **Start with image not video**
* NEXT STEPS: Download software for depth sensing camera
* GOAL: capture images from the camera

# 2/10/18 Resources

* <https://github.com/IntelRealSense/librealsense> (**software**)
* <https://github.com/toinsson/pyrealsense> (**software2**)
* <https://learndeltax.blogspot.com/2016/05/3d-object-detection-using-opencv-python.html>
* <https://www.pyimagesearch.com/2016/02/08/opencv-shape-detection/> (general shape detection)
* <https://www.pyimagesearch.com/2017/09/11/object-detection-with-deep-learning-and-opencv/> (object detection using deep learning)
* <https://medium.com/@williamkoehrsen/object-recognition-with-googles-convolutional-neural-networks-2fe65657ff90> (about Google’s CNN)
* <https://machinelearningmastery.com/object-recognition-convolutional-neural-networks-keras-deep-learning-library/> (Keras deep learning library)
* <https://becominghuman.ai/building-an-image-classifier-using-deep-learning-in-python-totally-from-a-beginners-perspective-be8dbaf22dd8> (more about the Keras library)

# 2/14/18 Discussion Notes

* Mini map has bird’s eye view
* Large map has 45 degrees view
* MVP: have program draw a boundary for the cube as it rotates around in a live web cam
* PROBLEM: OpenCV thinks a stack of cubes is one contour, so it would be hard to count the number of cubes
  + Could focus on detecting lines
  + Could use depth sensing to find the number of cubes
  + Isolate out the stack of cubes and then screen contours within the crop (do a contour and then line detection)

# 2/18/18 Meeting Notes

* Was able to detect lines of cube for multiple test pictures
* PROBLEM: using the line information to determine the number of cubes
  + Look for distance between the lines
  + Combine with depth sensing?
* QUESTION: what exactly do we need depth sensing for?
  + Live camera feed with camera in fixed position
  + MVP: find number of cubes in a stack
  + More useful for user map with bird’s eye view
* Find coordinates for as many cubes as I can find and extrapolate their coordinates (in order to take care of offset) and pair with depth map to find z coordinates from the camera’s point of view
* RESOURCES
  + <https://stackoverflow.com/questions/21522120/removal-of-vertical-houghlines-detection-of-horizontal-houghlines-only>
  + <https://docs.opencv.org/3.4.0/dd/d1a/group__imgproc__feature.html#ga8618180a5948286384e3b7ca02f6feeb>

2/20/18 Meeting Notes

* Current method of cube detection may be too rigid
* Find as many cube boundaries as we can
* Approximate center of cube and return it (return center of bounding rectangle around the cube)
  + Coordinates for mini map are more important than the real map (**top down view**)
* 3 structures that exist at a given time: minimap, real world map, codified version inside the robot script
* Re-detect all the cubes in the real map and use it to verify the camera’s moves (**45 degree view**)
  + Okay if not as precise, just used for verification
  + Finding out the relative relationships between the cubes is the key
* NEXT STEPS: make the bounding/ cube recognition feature work for live video stream

# 2/21/18 Meeting Notes

* <https://github.com/msracver/Deep-Feature-Flow>
* Live video stream
* Need to give the depth sensing team:
  + Separate cubes from a stack, detect how many cubes there are
  + Boundaries for each cube
* Took a bunch a new pictures with a more accurate angle, some pictures had better results than others.
* NEXT STEPS: need to get the line detections to work for all orientations of the stack

# 2/25/18 Meeting Notes

* Was able to adjust line detection code to detect most horizontal lines for the stack of cubes
* Ideas
  + Get all the lines
  + Differentiate between right and left lines and choose side with most
  + Then analyze the distance between the lines to determine how many cubes are in the stack
  + If the distance is too small, ignore one of the lines. If the distance is too large then there should be another line in between.
  + Use both sides to make sure you get all the lines (both sides as cross reference)
  + PROBLEM: if the distances between the lines in one dimension are around the same as the distances between the lines in the other dimension.
  + SOLUTION: ~~color the sides of the cube to make sure you compare the correct faces together~~  go by starting x coordinate of lines (the upright face should have similar x coords, the flat face should have different x coords)
  + **DON’T DRAW A LINE UNLESS YOU KNOW IT IS UNIQUE AND DRAW LINE A LITTLE LONGER THAN IT SHOULD BE, THEN FIND INTERSECTION OF THE LINES (BETWEEN LINES OR BETWEEN LINE AND AXIS) FOR ANALYSIS** 
    - Don’t draw a line unless it is unique if it isn’t extend the other line accordingly

# 2/28/18 Meeting Notes

* Same idea from last time, but now I will look for the intersections between the lines I detect and the vertical line draw from the middle of the contour
* How to differentiate between the two sides:
  + Lines with positive slope are on the right side
  + Lines with negative slope are on the left side
* What the depth sensing team wants from us (**v2**):
  + Get horizontal lines of the cubes in the stack
  + Goals: know (x, y, z) for each cube in reference to camera
* **April 1st**, first attempt at an integrated demo

# 3/06/2018

* Depth sensing team needs: bounding box of all detectable faces of a cube, equation of lines (some cubes only have front face, some only have top face, some have both front and top faces)
* Paint cubes
  + Boundary lines in red and fill in black
* April tag idea
* **Find another filter that will find bounding box around cube**
  + Contours
  + Look at the image as a matrix of numbers/ pixels
  + Hyper contrast image to make edges more pronounced
  + “Detection of everyday objects”

# 3/07/2018

* <http://ieeexplore.ieee.org/document/4795618/?reload=true>
* Harris corner detection and Shi-Tomasi Corner Detector & Good Features to Track
  + Doesn’t work because you would need to specify how many points to detect and you don’t know that unless you know the number of cubes
  + **Might work if the cubes were painted and there are not wood textures visible**
* Uses sci-kit image (another Python library): <https://peerj.com/articles/453/?utm_source=TrendMD&utm_campaign=PeerJ_TrendMD_1&utm_medium=TrendMD>
* **Start to create Haar Cascade** 
  + <https://pythonprogramming.net/haar-cascade-object-detection-python-opencv-tutorial/>
* A pure threshold and contour approximation method
  + <https://pythontips.com/2015/03/11/a-guide-to-finding-books-in-images-using-python-and-opencv/>
  + NO WORK especially with the current wood textures on the blocks.
  + Might have potential if the blocks are PAINTED