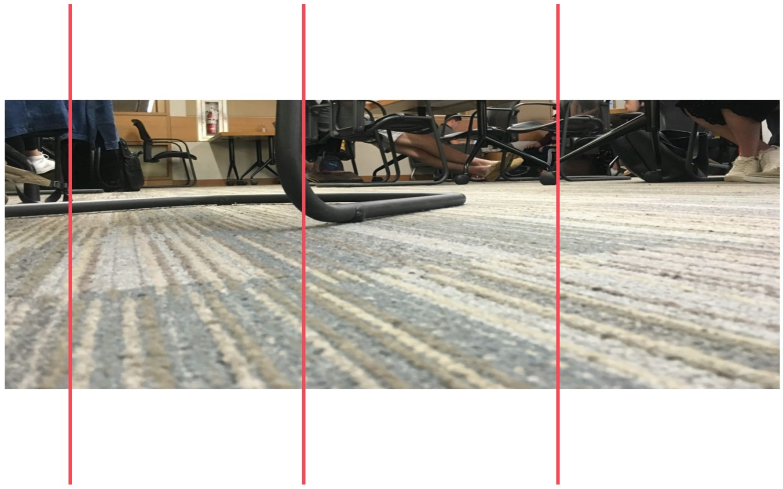
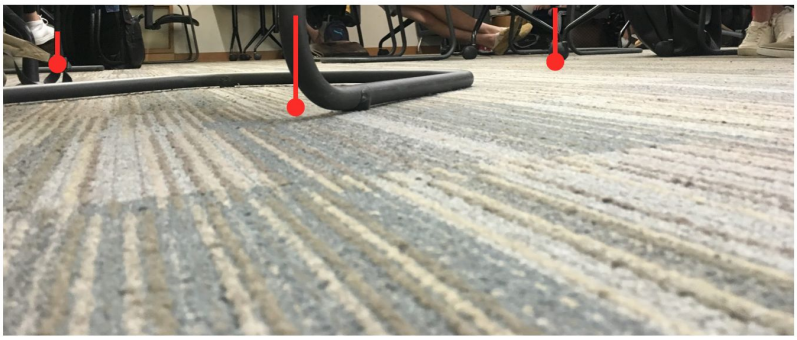
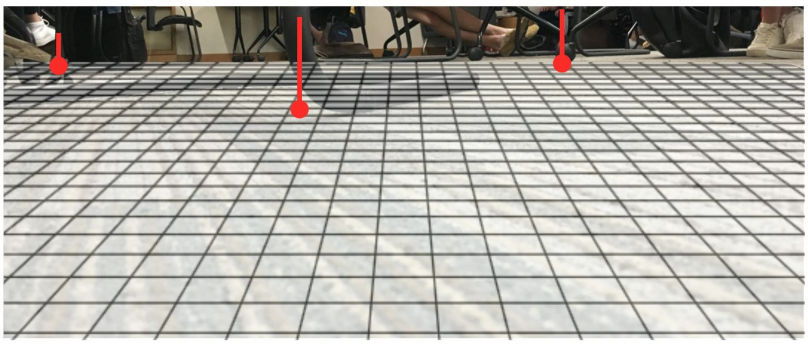
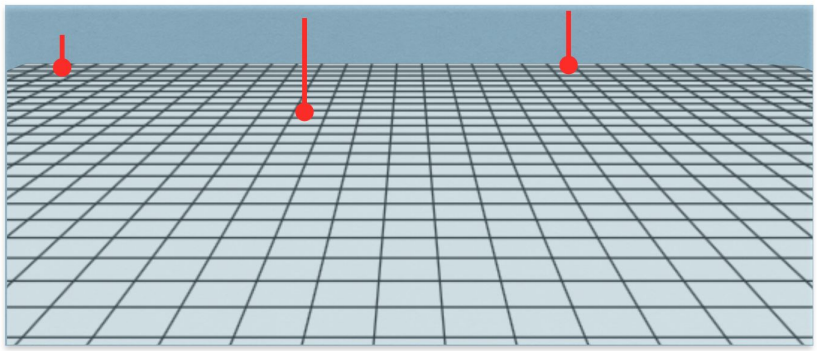
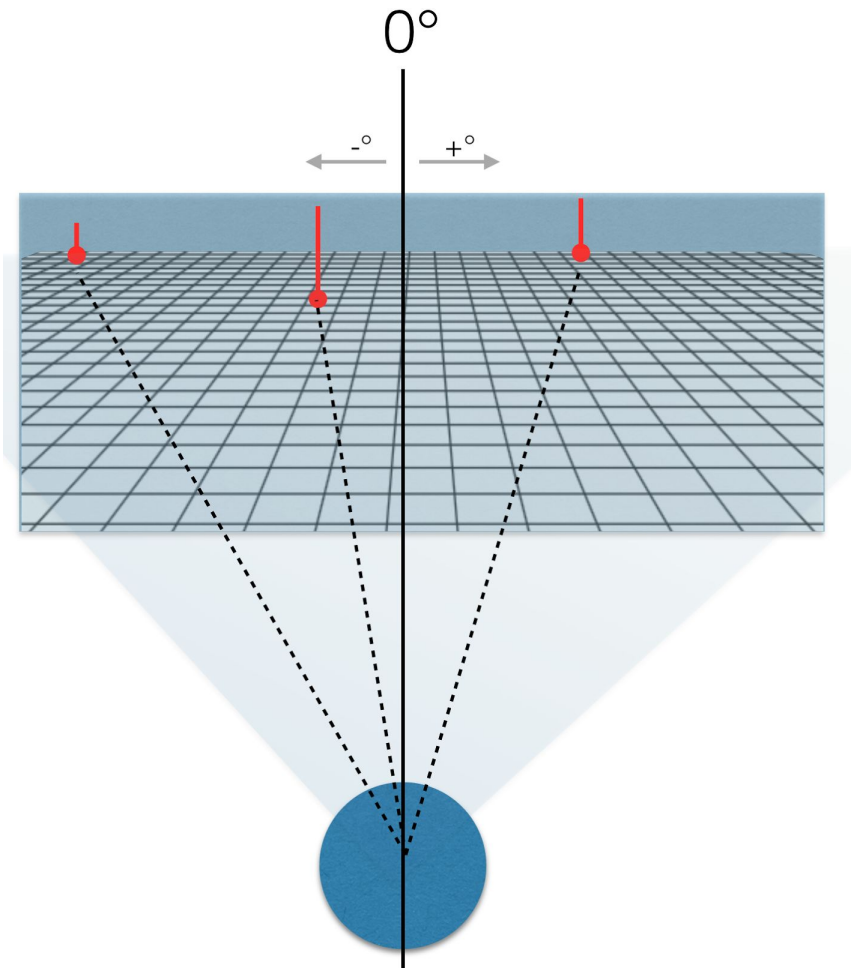
1. Proposal Title and Group Members
    a. **An iRobot Create that moves around, mapping the chair legs, and avoiding collision as well.**
    b. *Group 3*
        i. Yating Zhan (yz1339)
        ii. Alexandra Serralta (avs327)
        iii. Ricardo Spinola (rs3826)
2. Background
    a. In previous homework, our group made a line detection method that can detect lines in an image. We want to extend that functionality and apply it by solving problems like detecting chair legs in a classroom setting. This project is based on the line-detection method and dynamic map building problem we learnt from class. We will also incorporate resources such as the iRobot Create and Raspberry Pi API.
3. Problem Identification
    a. This project involves the line detection and depth calculation problem in Computer Vision and the SLAM path planning problem in Robotics.
4. Project Aim
    a. Program the robot to analyze its surroundings (chair legs) in order to avoid collision.
    b. Gradually build a map of containing the position data of chair legs in a room
5. Research Questions
    a. How can one calculate depth of objects in an image.
    b. How can one calculate an object's distance from source using its position in an image?
    c. How can one extract data from an image and verify this information to be correct/accurate?
6. Significance of Questions
    a. In order, to create a two-dimensional map, we will need to calculate the object depth of the many obstacles in the classroom in order to be able to find their separation relative to each other. Without this information, a reliable map is unobtainable.
    b. It is necessary to more or less have an idea of the pivot of the camera, and the distance from the robot to the obstacles in the picture, in order to further create a reliable map.
    c. We need to consider how we can use Computer Vision tools such has convolutions and filters in order to make more readily visible the vertical items in an image and the plane they stand on.
7. Literature Review
    a. Slam For Dummies
    b. Tutorials about Raspberry Pi
    c. Tutorials about iRobot Create
8. Theoretical Framework

a. Our setup will consist of
    i. One Arduino/Raspberry Pi with a camera module
    ii. One iRobot Create
    iii. One computer tethered to both the microcomputer and the iRobot Create, to do all the processing.
b. The robot will move around obstacles by analyzing its surroundings using the camera mounted on the robot. In theory, the way the computer will process the images that are taken from the robot will go as follows:
c.

SAMPLE IMAGE  	STEP 1 - Take a snapshot (this is a sample, not from the class and not using the chairs that will be used in the project). Most of the upper half of the image is not going to be needed or analyzed.
SAMPLE IMAGE  	STEP 2 - Since the camera is going to be at a predetermined angle, we can expect certain elements of it to remain constant, in this case the horizon line.

	STEP 3 - From this, analyze the image and look for vertical and almost vertical lines.
	STEP 4 - Determine where these lines start and end (below the expected horizon) - Gather information about these line starts.
	STEP 5 - Transfer the information to another matrix, one that maps the images' approximate horizon to a theoretical matrix (created before) in which each location (where a leg meets the floor) contains the distance of that point to where the camera is (or the center of the robot).

<table>
<tr>
<td>  </td>
<td></td>
</tr>
<tr>
<td> 
</td>
<td>STEP 5 (CONT.)

- Using this information we can now calculate the angle at which the leg starts from the ground, then we map that data to a matrix that corresponds to the MAP of the room the robot is traversing.
- As the robot moves, the robot is now in a position known in the same map that it is creating.
- Once it moves and takes another image, our algorithm will compare the “new” points in the image to the MAP, at which point it will know if the leg being mapped has already been seen and added to the map or not.</td>
</tr>
</table>

9. Research Methods
    a. Build Methodology: we will twist and improve our methods as we experimenting our functions
10. Resources
    a. OpenCV

b. Matlab built-in functions
c. API Tutorials
d. Previous Homeworks
e. Literature Introduced In Class

11. Timeline
    a. Week 1 (April 18th - April 25th):
        i. Research iRobot's API and projects:
            1. Question1: How to control the robot
            2. Question2: How to get its configuration space
            3. Question3: What are the resources we can use
        ii. Write a method to detect chair legs (black poles):
            1. Name: findLegs(image)
            2. Input: static image may or may not contain poles
            3. Output: separation array from the image of the pole to the robot's configuration space, if there is no pole, output -1
        iii. Write another method to calculate distance between two poles
            1. Name: separation(p1,p2,c1,c2)
            2. Input: d1: distance between one pole; d2: distance between another pole; c1: configuration space of the robot when it took the picture of the first pole; c2 the configuration space of the robot when it took the picture of the second pole
            3. Output: separation between the two poles
    b. Week 2 (April 25th - May 2nd):
        i. Research the camera module
        ii. Write the method to calculate the separations between all the poles
            1. Name: getSeparations(SepToRobot)
            2. Input: An array contains all the pole's relative position to the robot (separation and angle)
            3. Output: A two dimensional array contains the separation between each pair of poles
        iii. Write a method to create a path depends on the environment:
            1. Name: createPath(Environment)
            2. Input: vector of environment contains separations between poles and robot
            3. Output: Short path (<= 2 feet)
        iv. Write a method to control the robot's movement:
            1. Name: executePath(path)
            2. Input: A short path
            3. Output: new configuration space
    c. Week 3 (May 2nd - May 9th):
        i. Write a function to take picture from the camera module
            1. Name: takePicture()
            2. Output: One image the camera took at this configuration space

ii. A main function to start the program:
    1. Name: main()
    2. Important variables:
        a. robotConfs[] ---- an array of the space configurations of the robot
        b. sepToRobot[] ---- an array of the separation between all the poles to the robot's current configuration space
    3. Pseudo:
       robotConfs[0] = the configuration space where the robot starts (hardcoded in for now).
       while(true)
           image = takePicture()
           sepToRobot = findLegs(image)
           sepBetPoles = getSeparations(sepToRobot)
           path = createPath(sepToRobot, sepBetPoles)
           newConf = executePath(path)
           robotConfs.add(newConf)

iii. Clean things up, write final report, and prepare for demo and presentation

12. References

Raspberry Pi Tutorial:
http://www.irobotweb.com/~/media/MainSite/PDFs/About/STEM/Create/RaspberryPi_Tutorial.pdf?la=en
iRobot Create projects:
http://www.irobot.com/About-iRobot/STEM/Create-2/Projects.aspx
Slam For Dummies
http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-412j-cognitive-robotics-spring-2005/projects/1aslam_blas_repo.pdf