1. We are going to be using the camera on the GoPiGo to do lane following on a test-track roadway. The test-track is in Room 611 CEPSR, and you can also create your own test track if needed, but final testing will be on the 611 CEPSR track.

2. Follow the yellow line: Your GoPiGo will be placed on the track, with a yellow lane marker in view. Treat this yellow marker as the centerline of the lane, and move the robot around the track following this yellow line. Your program should also stop when you see an orange marker (stop sign) on the track. If you see the orange marker, then your robot will rotate 180 degrees and follow the yellow line back to another orange stop marker. Your program should do the following:
   a. Precalculate a homography from your image camera plane to the planar roadway. Your roadway grid will have a known metric distance that you can define (e.g. 100 pixels = 2 inches). This will allow you to take any camera pixel coordinates and transform them to roadway coordinates. You can calculate the homography by clicking on 4 known points in the image and mapping them to the roadway grid. The code below takes 4 points (pts_source array) you mouse click on the input camera source image (im_source) and maps them to 4 known metric points (pts_roadway array) that you have pre-defined on the roadway grid image im_roadway (such as the 4 corners of the roadway image). This gives you a 3x3 transform matrix to warp any point in the source image to the roadway image.

   ```python
   transform, status = cv2.findHomography(pts_source, pts_roadway)
   ```

   You can transform the entire input image to the roadway grid image (of size width, height):

   ```python
   im_roadway = cv2.warpPerspective(im_source, transform,(width,height))
   ```

   You can also transform single points in the source image to the roadway image.

   ```python
   source_point = np.array([source_x, source_y, 1])
   roadway_point = transform.dot(source_point)
   roadway_x = int(roadway_point[0]/roadway_point[2])
   roadway_y = int(roadway_point[1]/roadway_point[2])
   ```
b. Now start driving! Take a pic and warp it (homography) to the roadway grid - think of this as the image projected onto the flat 2D roadway.

c. Do color filtering/thresholding for yellow center lane marker. You may want to use erosion/dilation here also to remove small color artifacts.

d. Find the edges in the filtered image from step c above (e.g. you may use the cv2.canny() function). Note there are a number of parameters in the edge detector (edge thresholds etc.) that will affect the edge finder’s performance.

e. You may use cv2.hough() function to find center line from the edge image in step d above.

f. The center line has already been projected onto the roadway image, so your robot’s location with respect to the rotation angle (want to stay parallel to the centerline) and horizontal distance of the robot from centerline can be computed and used to move the robot accordingly to stay in the center of the road and move forward.

g. Repeat steps b-f above.

3. Notes:
   a. You should mount your camera up higher where the sonar sensor is. You can remove the sonar sensor and servo (leave in blue tool box) and mount the camera there.
   b. The camera needs to point downward for best performance. You can use some tape or other method to point the camera slightly downward to properly image the track.

4. Below are some images showing the sequence of processing leading to a centerline detection in the roadway grid, and computing the robot’s position relative to that centerline.

5. Video of yellow line following (not orange stop though)
   https://www.youtube.com/watch?v=naFzmkthOuQ&feature=youtu.be
Original image

Original image with mouse clicks (white dots) for 4 point homography. Known points are the location of the 4 blue thumbtacks in the previous image.

Original image warped onto roadway grid after homography. White circles at Corners of image are the warped mouse clicks of the 4 points needed for the homography
Thresholded and masked yellow image

Canny edge detector applied to thresholded and masked yellow image
Hough transform line detector (red lines) overlaid on roadway image. The magenta line is the orientation (direction of travel) of the robot relative to the grid when the picture was taken. If we extend this line back from the bottom of the image we can also find out the robot’s position on the roadway.

Left: Here is another image taken from an angle. Right: the homography and the mapping of the robots direction of travel onto the grid. The angle between the magenta line and the red hough line on the yellow centerline can be used to re-orient your robot to move parallel to the centerline. The magenta line is calculated by taking the vertical line in the middle of the original image and transforming it with the homography matrix into a line whose coordinates are now in the roadway grid.