**PURPOSE:**

The purpose of the project is to design a ROBOT-BASED automation system using only a web camera as our sensory feedback. This project tests our abilities to deal with limited sense to control a non-linear system of mechanical components in a effort to complete the tasks given. To complete the task outlined below, a mechanical controller and vision recognition system were required to complete the many tasks involved.

**Project OUTLINE**

The tasks in the project are many, the overall task is to determine what object is on the playing field, getting to and grabbing the object and bringing it to a destination while avoiding the obstacle. To make matters more complicated, the robot design must not have differential steering. Differential steering is when each wheel has its own power, this would allow for the robot to be nearly holonomic, ie the heading angle and position of the robot are separate from each other. Because we are not allowed to have this type of steering preplanning and turning radius are important components in our path planning.

To make matters more complicated, the only sensor we have is a web camera placed above the workspace. The workspace is 4’ by at least 4’6”. The reason the second dimension is variable due to the distortion of the web camera. The objects sizes are as follows the square is 1”x1” x1” the cylinder has a 1” diameter and depth of 1” ,while the rectangle is 1”x2”x1”.

Each object has an equivalent sized destination, which is black and has no depth. To simplify the project, only having to deal with circular shapes reduce the complexity of the problem as there is no inherent orientation to such a shape.

**robot construction:**

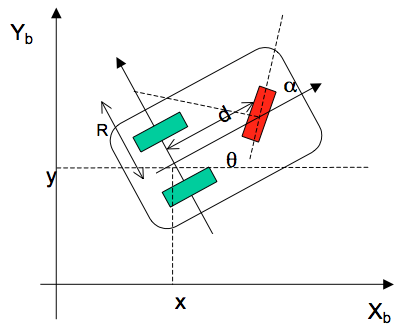
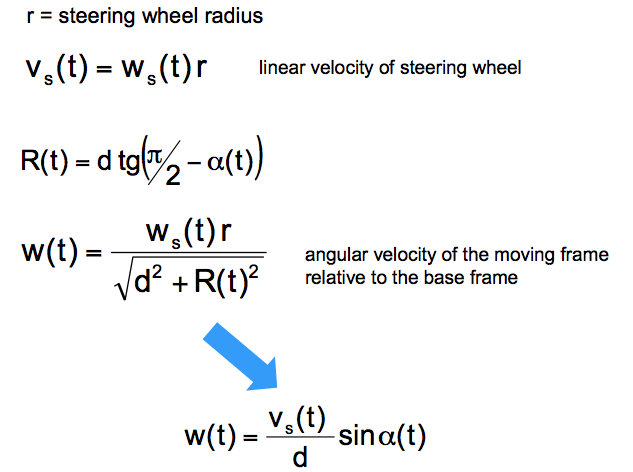
Our initial design was that of a Tricycle, this was determined to have to smallest turning area of any no differential system. Unfortunately our first iteration of this design was very unstable and wobbling substantially. This led to a redesign of our robot in which a toy car was scavenged and the steering mechanism was obtained. This design was used for a significant period of time before its turning radius was determined to be to great. In fact its turning radius was the entire workspace. To help deal with this issue we returned to our original concept but with a slightly different set up. Unfortunately at certain angles and speeds slip was produced and would cause error in our driving ability and success.

Our gripper was an extruded L shape which would pin the object to a stationary wall attached to the frame of the robot. During our demonstration it was not evident of being effective but in other runs in was extremely effective. The reason for this discrepancy was our calibration for its location and the tolerance of our Proportional controller became the issues.

If we could have rebuilt it, we would have used a prebuilt gripper which would have been more accurate, as well as make the single wheel in the back into two wheels with diameters but larger tread are to ensure greater stability. Also having some sort of light exactly above the gripper would have allowed for minor calibration for the gripper location and could have been used in the feedback of our system.

**CHALLENGES:**

One of the challenges IN this project is THE PROBLEM OF steering angle and turning radius. vehicular robots usually use differential wheels as steering mechanisms to have the advantage of turning in place. As this was prohibited for this project, steering and power needed to be separate from each other.

Where, Vs and **α** represent the speed and the steering angle.

The MOTION model implies that to achieve minimum turning radius, the steering angle should be 90 degrees off the heading angle. All thought this model was nice to have to predict if the robot could take a turn; it became difficult to implement with the controller for motion used as it was creating its path dynamically which will be described later in path planning.

Another issue was with calibrating the grabber to properly contain and release the object at the correct location, the calibration would change due to mechanical and vision noise. What is meant by this is due to shatter of the robot frame or slippage in the wheels as well as in accuracies in the vision code would require different calibrations.

The final major issue, which was ignored, was the orientation of objects and the destination this proved to difficult to obtain for our team. The reason it was so difficult was that because of the limited motions of the robot provided enough obstacles to simply just go and grab the cylinder any other object would have been caught properly simply by chance.

VISION:

As the web camera was our only sensor it was critical to make its processing critical to the success of the robot. The following are the steps taken and a simple description of what they do:

Difference: this code takes an original RGB shot, which is the background and as new images are updated the differences in values are calculated. This helped ensure that any dark areas which were within the vision where eliminated before becoming objects. The downside to this process is that it required the camera to not be moved, as it would alter the results drastically. This meant that it was not a very robust function but it allowed us to work on many different surfaces.

Scaling: this function takes the values or RGB and scales each value based on the maximum and minimum values in the image and smoothens out all the other values accordingly.

Low Pass Filter: is a way of eliminating values which are too low and there for should be ignored, this in turn uses the convolution function.

Threshold: makes the image binary, where the threshold value determines if grey scale values are to be kept or ignored.

Erode: reduces the perimeter or white space, this helps eliminate very small objects by turning them black

Dilate: to ensure that an objects size is not reduced to much or if within an object there was erosion dilation fills in the space next to white space. Unfortunately because of this objects become more rounded out making object recognition more difficult.

After the processing was done Identification was created. Each object was evaluated based on colour, number of pixels centroid, shape and orientatioin. Because we knew the dimension of the obstacle and it was the largest black object in our workspace it became our calibration for area. We did this by saying C=4\*PI/count, where count is the number of pixels in a specific object. Once C was obtained each object area was then multiplied by this constant and the process obtained fairly accurate dimensions. Colour was determined by averaging out the R,G,B values independently of a labeled object, this was more accurate then simply using the centroid but required more processing.

For the shape, the ratio of the maximum distance to the edge from the center versus the minimum distance to the edge was used to determine what type of shape it was. For example Rmax/Rmin for the circle should be close to 1 while the square it should be close to 1.41 and because the dimensions of the rectangle are 1x2 the ratio would be close to 2.23.

Finally for the orientation the second moment of area was used, but initially we used the location of the Rmin and shifted by 90 degrees to determine the value of the orientation. This was not as accurate as the second moment and so was left unused at the end.

PATH PLANNING:

The main challenge in path planning is determine how the robot will be arrive at a location with a specific orientation. For this a Proportional controller was used to get between key points in the space, while ensuring the robot does not collide with anything other than the object and destination.

The main reason why a P controller was used and not a PI or PID was due to the build up of error produced inaccuracies which were difficult to tune properly. Initially we had planned to use a Bezier curve which creates a spline using 3 control points and 2 endpoints this would have ensured that the robot would never collide with the obstacle and aid in correctly attacking the rectangular and square shapes which have orientation as well. Unfortunately following a Bezier curve proved very difficult and so we went to a simple controller which when faced with any object it would take it from any orientation, as well as use a reverse P controller for the obstacle if it enters to close into our path.

What proved most difficult was calibrating the robot gripping actuator so it would be accurate when shutting and opening. It was noticed many times that the calibration was curopted due to vision and mechanical noise. The simplest method to calibrate was to place the robot at a location in which it would have successfully released the object at the destination and calculating the distance between the destination location and a defined point on the robot. Even with this adjustments were needed to be made to ensure success.

Due to the object and destination both being lost under the robot during the specific actions associated their original position was used and not updated. This means that if the robot did not successfully grip the object it would not know whether or not it was successful. With more time a simple check by going backwards and seeing if the vision detects the object or not, would have ensured more stability in our code.

DISSCUSION

The main issues when running our program were within identification of objects are their associated name (ie robot, obstacle object, etc) proper decision making and robot control.

Ideally a more robust controller that would ensure a successful path would have been preferred. This would have incorporated a more dynamic edge detection and avoidance as well as a way to determine if from the angle and position of the robot grabbing the object was possible and if not act accordingly. Beyond these two things, adding in the code to determine the path for objects with orientation and their placement within a oriented destination would have been ideal.

Within our code there were a few bugs, which would be produced either because of distortion from being not exactly in the cameras focus or by overflow issues. Most of these were cleared up, but we suspect that there were some that were misdiagnosed.

Something else which bugged us, was that our vision system was very inefficient and not as accurate or dynamic as wanted. Looking back at our results from the presentation a more accurate vision system would have benefited us immensely.

Fortunately we did have several runs, which were successful at avoiding the obstacle as well as placing the object within a very reasonable distance from the center of the destination. In addition because of our choice to return to a tricycle geometry aided us in having an easier time turning making the controller smaller than for other geometries or larger robots.

CONCLUSION:

What we learnt from this very broad project was that a very robust decision tree is what is required for such a variable environment. In addition building a smaller robot to allow it to be more stable in such a small workspace, which was the trade off for accuracy in the vision system. Ideally having a more robust feedback system would have also aided in the process of solving this problem, in our case this means a faster and higher accuracy in how we process the images received. Finally the last thing I would have done would be to delete all the functions that became unused and clean up the code substantially ensuring that every code is being used and there is no possibility of using a less effective or altering codes, as well as simplifying the debugging process.

REFERENCES:

* W. Press, S. Teukolsky, W. Vetterling, B. Flannery “Numerical Recipes in C, The Art of Scientific Computing”, 2nd edition, Cambridge University Press, 2002.
* S. LaValle, “Planing Algorithms”, Cambridge University Press, 2006.
* http://processingjs.nihongoresources.com/bezierinfo/
* http://users.isr.ist.utl.pt/~mir/cadeiras/robmovel/Kinematics.pdf