Intro to Robotics Final Project

WALL-E's revenge

Max Saldi, Duke Manchester , Anand Odbayar

Presentation on youtube Presentation on Github

Base Deliverables:

- **Localization** - Lead: Duke
    - Used the Webots gps and compass to get the current robot pose

- **Create Mapping System -** Lead: Anand
    - Used Lidar to map the world and determine where obstacles were

- **Planning For Navigation -** Lead: Duke
    - Determine endpoints that need to be reached
    - Implement RRT path planning
    - Used created map and color detection to help determine where cubes were

- **Create Vision System** - Lead: Max
    - Write color detecting algorithm to differentiate cubes
    - Used color blob detection algorithm to detect yellow blocks and took picture of them to verify that it worked
    - Used to guide where the arm should grab, unfortunately it isn't accurate enough to work outright and teleoperation was used to finish the job

- **Arm Manipulation** - Lead: Anand
    - Despite poor documentation, was able to get IK library to work and control the end effector by simply providing a goal point in 3d space

- **Wheel Manipulation** - Lead: Max
    - Creates a list of waypoints and finds the angles and cartesian coordinates in the absolute space
    - Uses trigonometry in conjunction with linear math to drive between waypoints
    - Uses a ton of correction algorithms to correct for errors that were made

Reach Goals/Deliverables: (implement if time permits)

- **Improve RRT Path Planning** - Lead: Duke
    - Pruning/smoothing so that the final path is less tortuous
    - Completed and allows for a more concise and less windy path

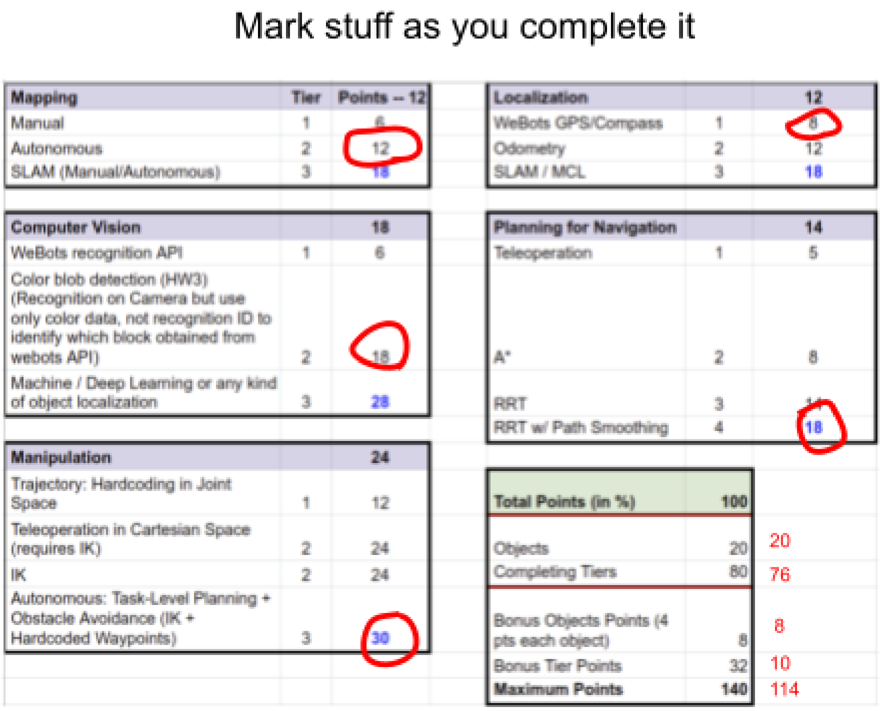
- **Implement Object Collision Avoidance to the Robot Arm -** Lead: Anand
    - Determine waypoints of objects that need to be avoided
    - We attempted to implement it but we had a hard time with finding where all the objects are using only the camera functions

Issues:

- Ik library was very difficult to implement with poor documentation pertaining to errors that you can get
- Adapting the RRT algorithm to work in a discretized environment
- With doing the angle correction as the robot was moving I had to deal with the branch cut that exists on the unit circle and had to normalize radian values to solve this
- Color vision object guesses are too inaccurate to be directly used with IK for a satisfactory result
- Robot getting stuck on smaller stabilizing wheels that existed under the chassis
- Lidar beams hit the ground, had to raise object
- We had an issue with the The cubes falling through out bucket and more physics issues

Mapping - Autonomous
Computer Vision - Color Blob Detection
Manipulation - Autonomous task-level planning and obstacle avoidance (IK + HardcodedWaypoints) for wheel movements, IK assisted teleoperation(Cartesian) for arm movement
Localization - GPS/Compass
Navigation Planning - RRT w/ Path Smoothing

Cubes = 10+2 (2 of the yellow cubes decide to phase through the basket)



Mark stuff as you complete it

Mapping	Tier	Points -- 12
Manual	1	6
Autonomous	2	12
SLAM (Manual/Autonomous)	3	18

Computer Vision		18
WeBots recognition API	1	6
Color blob detection (HW3) (Recognition on Camera but use only color data, not recognition ID to identify which block obtained from webots API)	2	18
Machine / Deep Learning or any kind of object localization	3	28

Manipulation		24
Trajectory: Hardcoding in Joint Space	1	12
Teleoperation in Cartesian Space (requires IK)	2	24
IK	2	24
Autonomous: Task-Level Planning + Obstacle Avoidance (IK + Hardcoded Waypoints)	3	30

Localization		12
WeBots GPS/Compass	1	8
Odometry	2	12
SLAM / MCL	3	18

Planning for Navigation		14
Teleoperation	1	5
A*	2	8
RRT	3	14
RRT w/ Path Smoothing	4	18

Total Points (in %)	100	
Objects	20	20
Completing Tiers	80	76
Bonus Objects Points (4 pts each object)	8	8
Bonus Tier Points	32	10
Maximum Points	**140**	114