Gregory Cooke

Partner: Connor Napolitano

Learning Module 8: Autonomous Vehicles/Navigation

Module 2: Follow the Gap

Week 1: Gap Analysis

Homework Question: You should roughly reiterate the activity, discuss how it was accomplished, demonstrate through images (if possible) the functionality, then briefly note any observations (including cases where it may not work so faithfully, or what is needed to work very well):

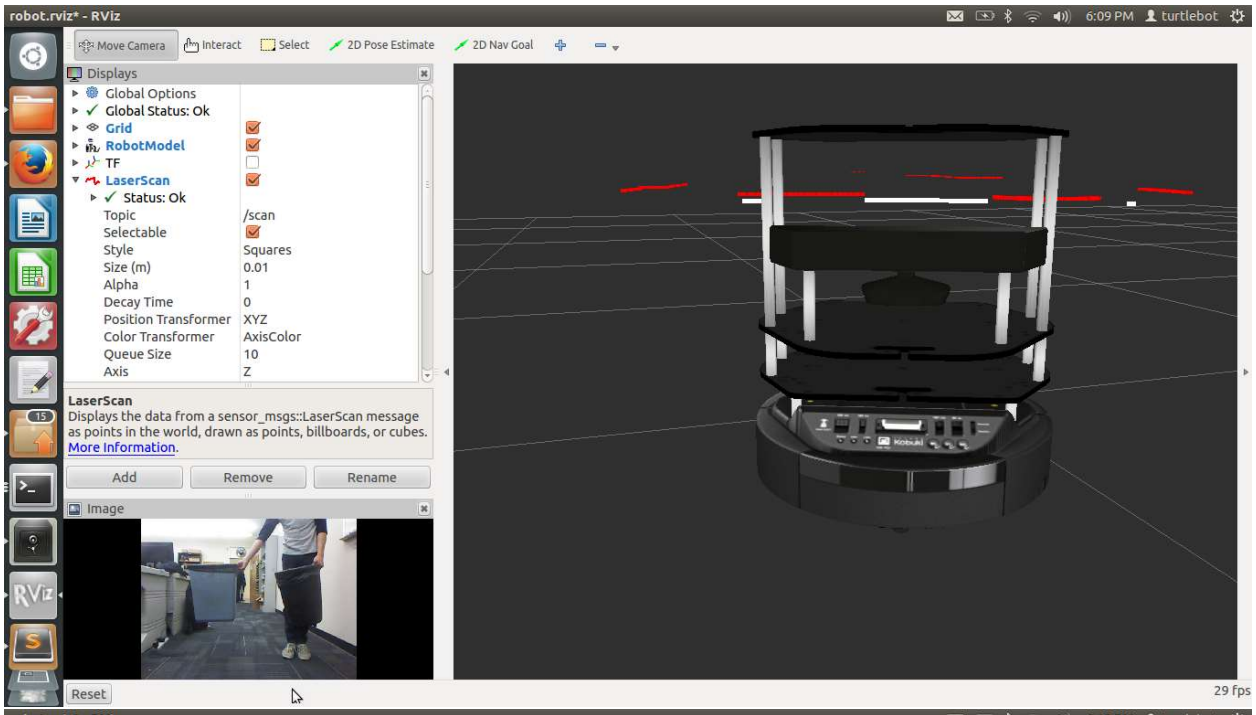
In this activity we were to implement the algorithm known as "Follow the Gap." The paper uses laserscan data to determine gaps between assumed circular obstacles. We ran into issues trying to implement the algorithm exactly as it is presented in the paper, so we simplified it. The issue was that our laserscan data was producing legitimate data even when it was looking at a good gap, because the scale of our room never really pushed the laserscan to its maximum range. This meant that, using the algorithm, we would never find a gap.

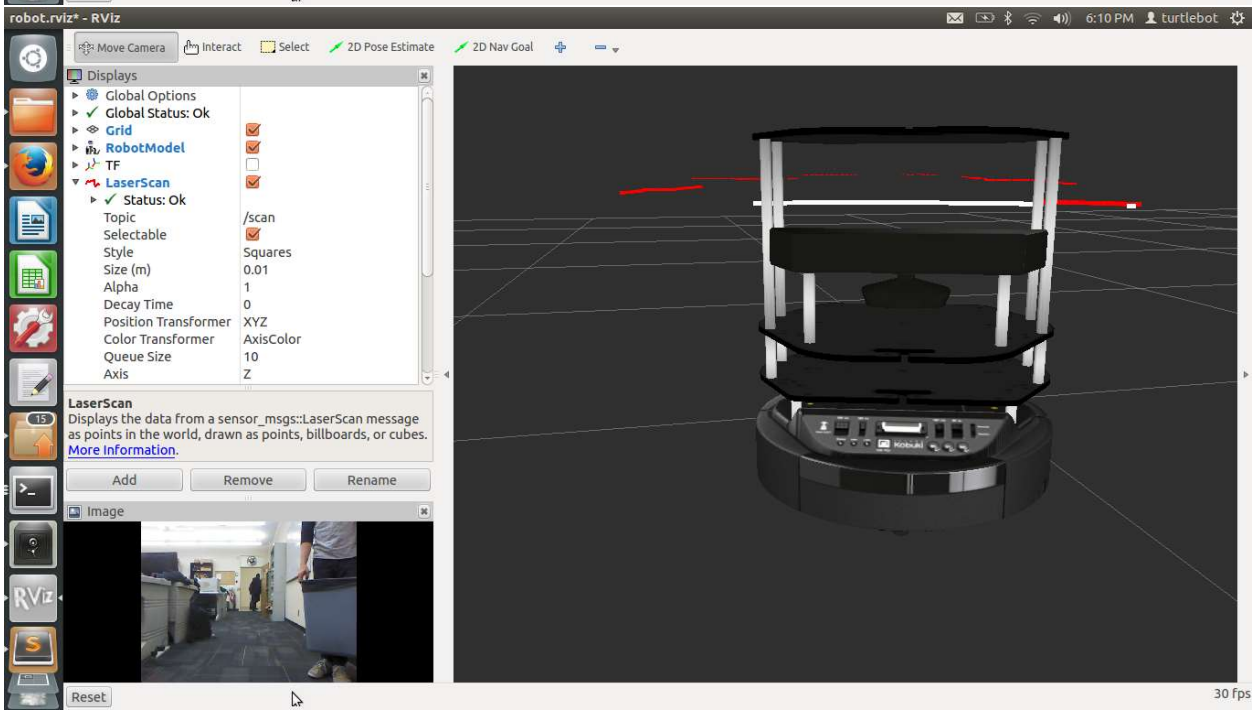
We instead began by thresholding the laserscan data at three meters out, converting any entry beyond 3 meters to a 0 to represent a gap. We did this because, realistically on our scale, if the turtlebot has an opening for three meters that is a fairly sizable amount of space for it to move into. Our algorithm then turned long strings of NaN's into 0's, but if singular NaN's were found it averages the value immediately before and after it. The algorithm then looked for the longest span of 0's and output the starting and ending index of that span. In the future, the robot will then move towards the center of that span.

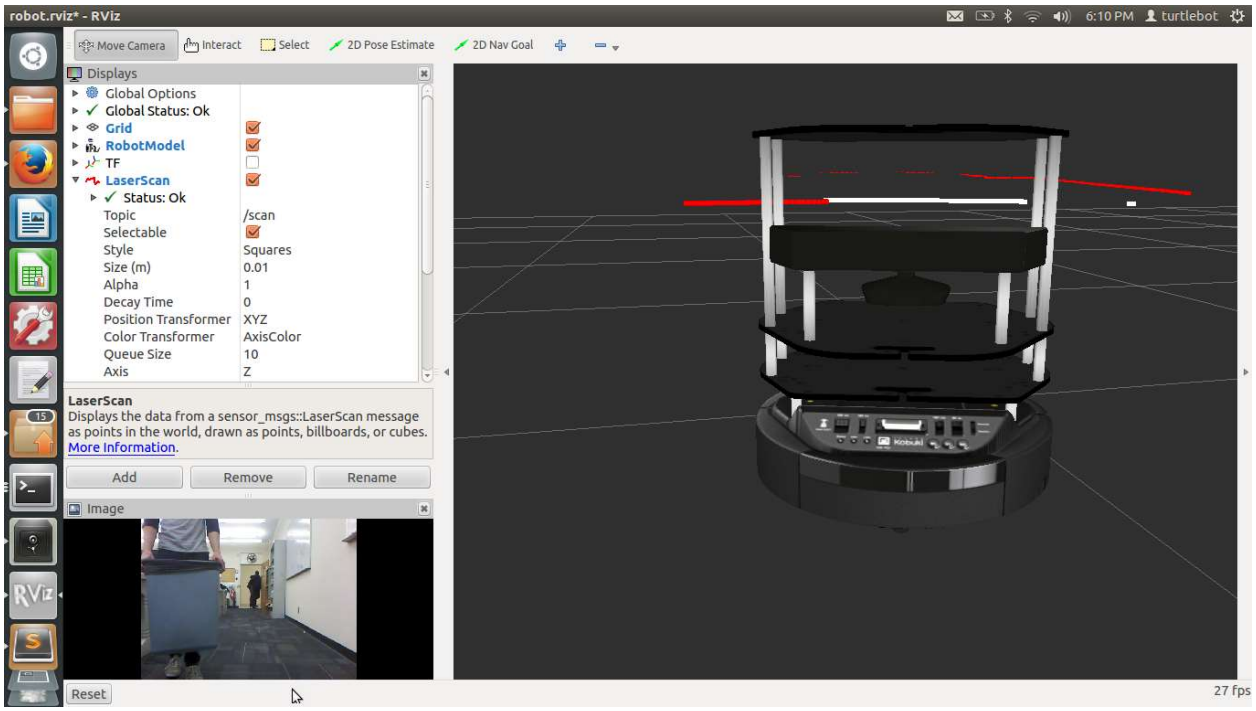
This data was then published to the /gapscan topic that we created, and that is what we used to visualize the data in rviz.

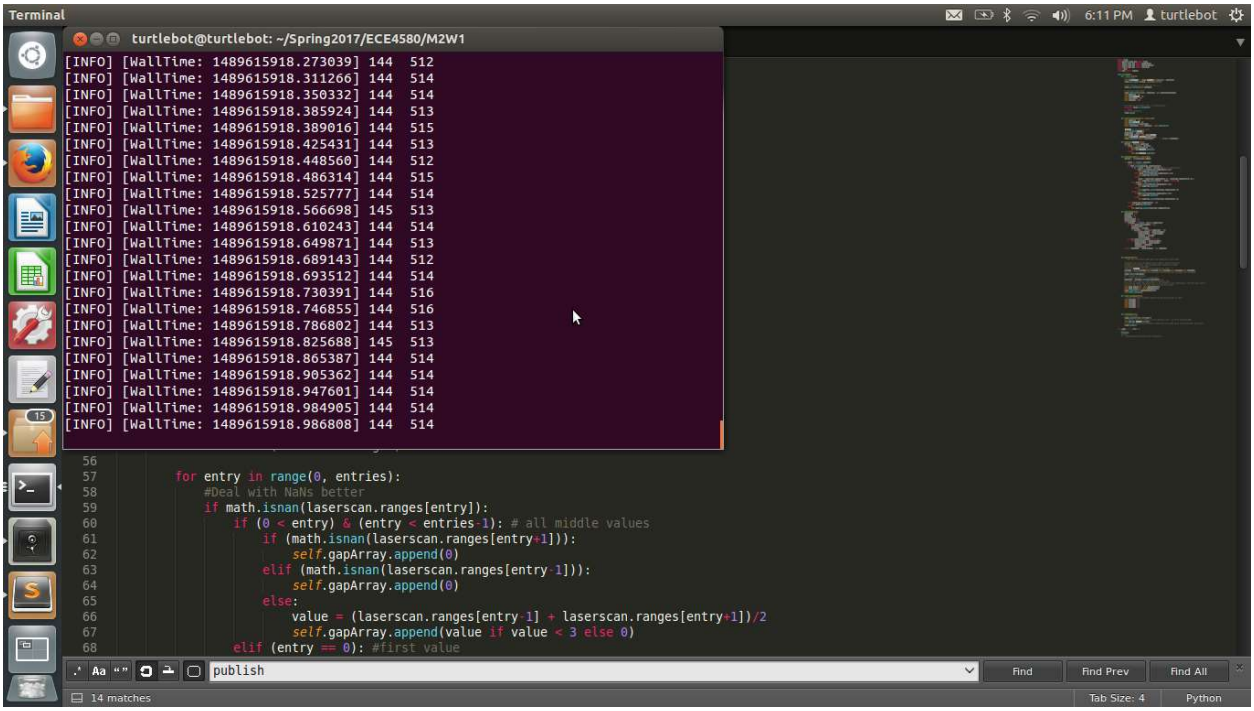
Turtlebot Adventures Answers:

1. The last image shows the output of our code finding the maximum gap. These outputs are the first and last index of the largest gap, therefore, in the case of the last image, the starting and ending indices respectively would be 144 and 514. This means the gaplength was 370 indices long.
2. Using rviz to visualize the gap array: Images below. The white represents calculated gaps whereas the red is the raw laserscan data.


robot.rviz* - RViz
6:09 PM turtlebot
Move Camera
Interact
Select
2D Pose Estimate
2D Nav Goal
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Status: Ok
Topic /scan
Selectable
Style Squares
Size (m) 0.01
Alpha 1
Decay Time 0
Position Transformer XYZ
Color Transformer AxisColor
Queue Size 10
Axis Z
LaserScan
Displays the data from a sensor_msgs::LaserScan message as points in the world, drawn as points, billboards, or cubes.
More Information.
Add
Remove
Rename
Image
Reset
29 fps


robot.rviz* - RViz
6:10 PM turtlebot
Move Camera
Interact
Select
2D Pose Estimate
2D Nav Goal
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Status: Ok
Topic /scan
Selectable
Style Squares
Size (m) 0.01
Alpha 1
Decay Time 0
Position Transformer XYZ
Color Transformer AxisColor
Queue Size 10
Axis Z
LaserScan
Displays the data from a sensor_msgs::LaserScan message as points in the world, drawn as points, billboards, or cubes.
More Information.
Add
Remove
Rename
Image
Reset
30 fps


robot.rviz* - RViz
6:10 PM turtlebot
Move Camera
Interact
Select
2D Pose Estimate
2D Nav Goal
Displays
Global Options
Global Status: Ok
Grid
RobotModel
TF
LaserScan
Status: Ok
Topic /scan
Selectable
Style Squares
Size (m) 0.01
Alpha 1
Decay Time 0
Position Transformer XYZ
Color Transformer AxisColor
Queue Size 10
Axis Z
LaserScan
Displays the data from a sensor_msgs::LaserScan message as points in the world, drawn as points, billboards, or cubes.
More Information.
Add
Remove
Rename
Image
Reset
27 fps


Terminal
6:11 PM turtlebot
turtlebot@turtlebot: ~/Spring2017/ECE4580/M2W1
[INFO] [WallTime: 1489615918.273039] 144 512
[INFO] [WallTime: 1489615918.311266] 144 514
[INFO] [WallTime: 1489615918.350332] 144 514
[INFO] [WallTime: 1489615918.385924] 144 513
[INFO] [WallTime: 1489615918.389016] 144 515
[INFO] [WallTime: 1489615918.425431] 144 513
[INFO] [WallTime: 1489615918.448560] 144 512
[INFO] [WallTime: 1489615918.486314] 144 515
[INFO] [WallTime: 1489615918.525777] 144 514
[INFO] [WallTime: 1489615918.566698] 145 513
[INFO] [WallTime: 1489615918.610243] 144 514
[INFO] [WallTime: 1489615918.649871] 144 513
[INFO] [WallTime: 1489615918.689143] 144 512
[INFO] [WallTime: 1489615918.693512] 144 514
[INFO] [WallTime: 1489615918.730391] 144 516
[INFO] [WallTime: 1489615918.746855] 144 516
[INFO] [WallTime: 1489615918.786802] 144 513
[INFO] [WallTime: 1489615918.825688] 145 513
[INFO] [WallTime: 1489615918.865387] 144 514
[INFO] [WallTime: 1489615918.905362] 144 514
[INFO] [WallTime: 1489615918.947601] 144 514
[INFO] [WallTime: 1489615918.984905] 144 514
[INFO] [WallTime: 1489615918.986808] 144 514
for entry in range(0, entries):
#Deal with NaNs better
if math.isnan(laserscan.ranges[entry]):
if (0 < entry) & (entry < entries-1): # all middle values
if (math.isnan(laserscan.ranges[entry+1])):
self.gapArray.append(0)
elif (math.isnan(laserscan.ranges[entry-1])):
self.gapArray.append(0)
else:
value = (laserscan.ranges[entry-1] + laserscan.ranges[entry+1])/2
self.gapArray.append(value if value < 3 else 0)
elif (entry == 0): #first value
publish
Find
Find Prev
Find All
14 matches
Tab Size: 4
Python