Gregory Cooke

Partner: Connor Napolitano

Learning Module 8: Autonomous Vehicles/Navigation

Module 1: Wandering

Week 1: Drive Commands and Visual Sensing

Homework 5 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 explored using python's rospy package to write basic ROS nodes for the turtlebot. The first was simply looking over nodes that moved the robot forward continuously, and another that moved the robot in a square. The robot moving in a square was rather unreliable with regards to actually returning to its starting point, so that didn't work very well.

Using that knowledge, we then created a ROS node that subscribed to the rgb camera topic and the depth camera topic and used those data streams to display the images. This required knowledge of python's cv2 and cv_bridge library and was substantially more difficult than the first task. After many bugs were worked out, we were able to achieve the goal.

readSensor.py has been included to show how we displayed the image streams.

Turtlebot Adventures Answers:

Turtlebot: Moving Part 1:

1. How to command the turtlebot to move its wheels at predefined velocities
    a. To go forward at a specific velocity, you send a Twist message with a given linear x velocity.
2. How to command them to move the robot a certain distance
    a. To go a certain distance, you set the rospy rate to a given frequency and the robot's linear velocity to a given speed then simply send the message a certain number of times at that frequency (distance = velocity * time).
3. How to command them to rotate for a certain amount (rather than at a certain speed)
    a. This is done the exact way as moving a certain distance, but instead of a linear x velocity you give it an angular z velocity.

Explore:

1. How well does the robot execute the commands?
    a. The robot executes the commands somewhat effectively. As discussed above, the move-in-a-square function doesn't actually function effectively.
2. If you work out the math for them, is it following the theoretical movements?
    a. The turtlebot did follow the theoretical movements, but not extremely accurately. For example, when trying to make a square, it would sometimes overshoot and sometimes

undershoot the turning in particular. The moving forward tended to be more accurate than the turning. When run continuously, the robot tended to always get somewhat close to the starting point, which means the errors are more random in nature. Otherwise it would slowly drift further and further away from the start.

3. Try out different surfaces. How do they impact the robot movement?
    a. The turtlebot did function more reliably on the tile floor in the hallway of Van Leer versus the carpet floor in the lab.
4. What are the topics published to in order to send the commands?
    a. The robot publishes to the cmd_vel topic, really the cmd_vel_mux/input/navi topic. This takes a Twist message.
5. Do they agree with what you found?
    a. I'm not exactly sure what this question is asking, but we suspected this to be the topic we that it was publishing to after running "rostopic list".
6. Did you find any extra? If so, how do they relate to the ones here?
    a. There are many topics running even with minimal.launch. I'd imagine as we attempt to explore more complicated navigation options we will start to use navigation topics other than just the cmd_vel.

Turtlebot: Sensing Part 2:

1. What are the ROS topics that must be subscribed to in order to get these sensor measurements? Identify the main two that provide the proper raw data associated to the image stream and to the depth stream for the kinect camera.
    a. For the Color Image /camera/rgb/image_raw/ (or /camera/rgb/image_color/)
    b. For the Depth Image /camera/depth/image_raw/ (or just /camera/depth/image/) These both send Image messages, which using the cv_bridge.imgmsg_to_cv2() method can be converted to cv2 images which can then be displayed using the cv2.imshow() method.
2. Name a few of the different imagine sensor topics and explain how they differ from the raw versions.
    a. There are various image_compressed, image_raw, etc, topics that are also available. These just send slightly different ROS messages (CompressedImage, for example) for which cv_bridge has a different function.

To summarize, today we learned how to write our own rosnodes using python, and how to link OpenCV into ROS and view both the depth camera and rgb camera feeds coming from the Kinect.