CS 309: Autonomous Intelligent Robotics
FRI I

Lecture 21:
Practical tips for HW 5
Simplified Navigation
Preparing for Final Projects

Instructor: Justin Hart

http://justinhart.net/teaching/2018_spring_cs309/

About the homework

- I realized last night 2 things
  - I was taking actually learning how to drive the robot out of the homework.
  - It was still slightly too hard.
- So I have modified it
  - HW5 will now have simpler navigation

About the homework

- Why did I do this?
  - My original version of this solved turning your modified marker position into a nav goal.
    - But I wanted you to be able to solve navigation, not me!
  - Doing so required you to learn complicated C++ functionality instead of the robotics stuff!

About the homework

- Homework 5
  - Due April 12
  - Start early, it is hard and involves the robots
  - Do everything in the HW5 pdf, but do navigation as updated in this lecture
- Homework 6
  - Also due April 12
  - 1.5 page description of your final project plan
    - Discuss with me in advance

Undergraduate Research Forum!

- Friday, April 13
- 11:00am – 2:00pm
- We will have a poster!
- I am looking for students who would like to present this poster.
- I will be judging the competition.
- This is a good opportunity to practice presenting in a professional setting, and should be easy, laid-back, and fun.
- If you’re interested, contact me.

Robotics Consortium!

- Thursday, April 12
- 12:00pm – 1:30pm – Lunch included!
- We will have the same poster!
- This is similar to the URF, except that it will be to laboratories and companies who collaborate with UT through the Robotics Consortium!

Poster Presentations

- Do either, and earn 0.5 pts of extra credit on your final grade.
- Do both for 1 full point.
- You don’t need to be there the whole time, but we need people the whole time.

A couple of quick notes

- Robotics Study
  - If you're free, we appreciate the help. See the Canvas announcement.
- RoboCup@Home
  - We're still getting ramped up and you're welcome to participate!
- Unique ID for Fall 2018
  - **PLEASE** double-check this, I don't have the official number yet, but I think it will be: **51705**
  - When you sign up, make **SURE** that you are signed up for the correct class (instructor: Hart)

Today

- Updated Navigation How-To
- Practical Tips for HW5 – From Last Time
- Previous Good Projects
- Project Brainstorming Start

Updated Navigation!

- Write a new node that is only for navigation.
- I am including a new header for you to work off of NavPR.h
  - This simplifies writing your Navigation Node.
- You can use simple_navigation_goals as a starting point!

Your main (for navigation) should

- ROS Init
- Create a NodeHandle
- Create MoveBaseClient
- Connect to the server (ac.waitForServer).
- Open your TransformListener
- Open AlvarMarker.
  - Use “nav_kinect_rgb_optical_frame” as its fromFrame parameter.

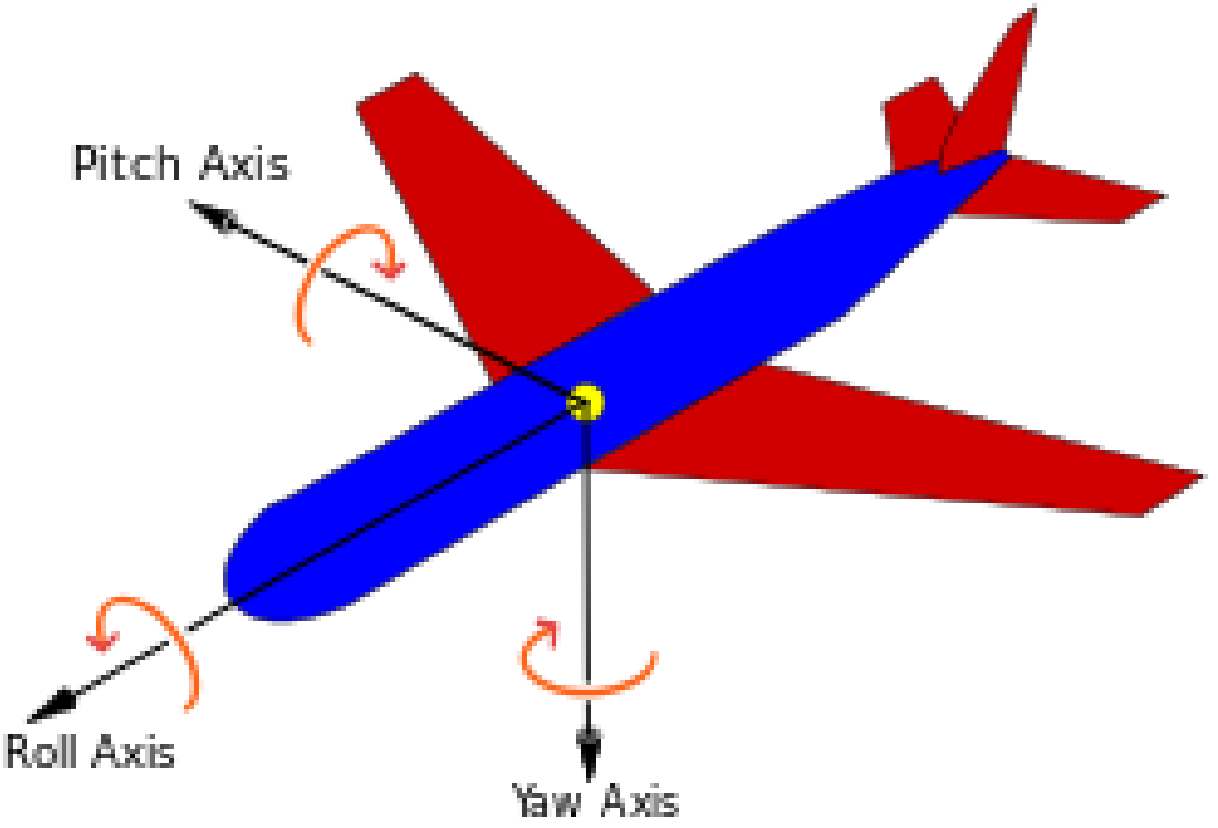
Navigating

- To navigate, you will do 2 things
  - Drive a short distance (I recommend 0.25 meters)
  - Turn towards the target

Getting the target

- Because AlvarMarker is providing the target, your target is at the incoming Pose in receivePose
- You will use a method we haven’t used before to turn the robot. tf::Quaternion in RPY format.
- RPY
  - Roll
  - Pitch
    Yaw

Roll, Pitch, Yaw



RPY

- Our robot cannot move in Roll or Pitch
  - Leaving only Yaw to control
- So, how do we determine Yaw?

A tiny bit of vector math

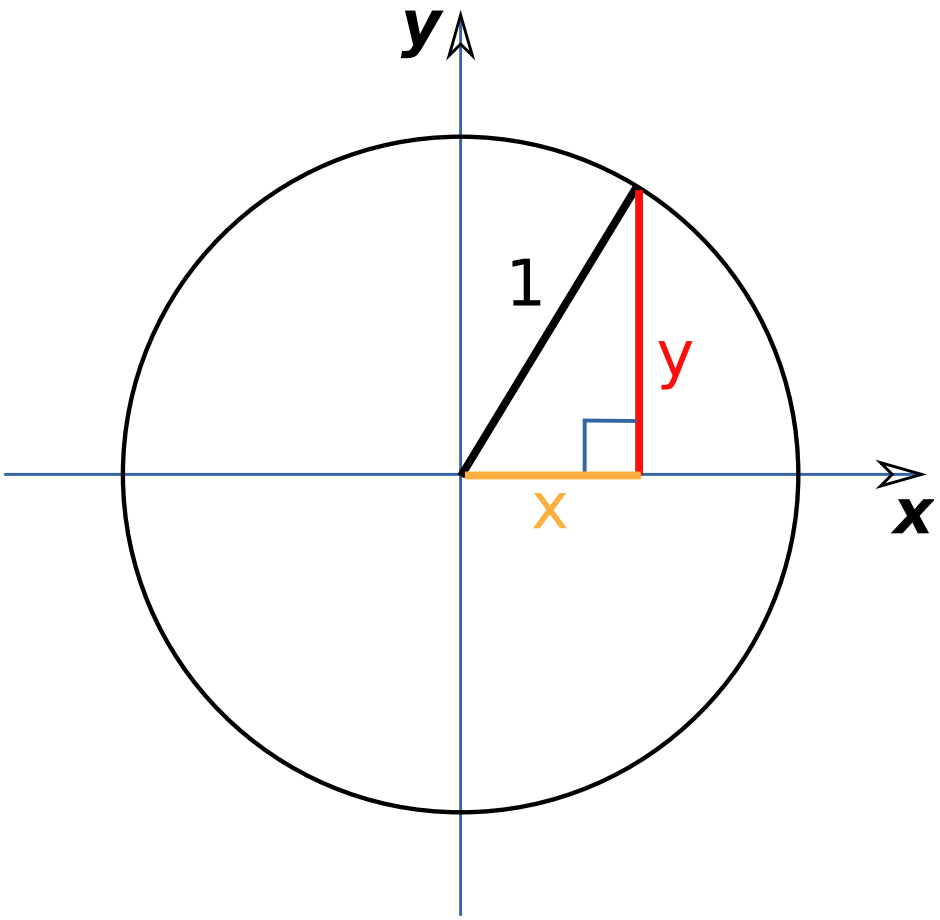
- The position of the marker (pose.position (x,y,z)) can be thought of as a vector pointing away from the camera.
  - We will whiteboard this.
- This can be put into EigenVector3d as follows
  - EigenVector3d vec(pose.position.x, pose.position.y, pose.position.z)
  - HOWEVER!!
    - Our 2nd coordinate does NOT matter. That is how high the marker is off the ground, and the robot cannot turn up and down to see it.

A tiny bit of vector math

- Knowing this, the way to point to the marker is
  - EigenVector3d vec(pose.position.x, 0.0, pose.position.z)
  - Which is parallel to the ground!
- vec.norm() is the norm of this vector
  - Which is how far away the marker is from the robot, along the ground!
- We can normalize our vector by dividing by its norm!

Remember our unit circle?

- The vector we have formed is now the hypotenuse of this triangle.
  - You can access its elements as vect(0) & vect(2)
- Knowing this, you can use atan2 to solve for the angle that the robot should be pointing.
  - Try using ROS_INFO to check your solution
  - It is expressed in RADIANS



Using that yaw!

- The tf::Quaternion class is to the rescue!
  - tf::Quaternion rpy(roll, pitch, yaw)
    - tf::Quaternion has an RPY constructor!!
  - Remember, roll and pitch should be 0!!
  - tf::Quaternion’s quaternion values can be accessed as:
    - rpy.x(); rpy.y(); rpy.z(); rpy.w();
  - Stick those into your orientation
  - What should you put into your position?

Don’t get run over!

- If you try to move to your *actual* position, you will get hit.
- If you try to move to your *actual* yaw, you will overshoot it.
- Multiply your yaw by a fraction to get the robot to face you without turning past you.
  - I used 0.5
  - tf::Quaternion quatRPY(0, 0, 0.5 * yaw);
    - Question: Should yaw be positive or negative? It depends on if we AlvarMarker transforms from base_link to the marker, or vice versa.
      - Experiment with this!
  - Since this runs in a loop, it will approach you anyway!

Don’t get run over!

- Similarly, I scaled my target position!
- Regarding your target_pose
  - X is the direction the robot is facing
  - Y is side-to-side
- This is *not* like your camera
  - In the camera, z is the direction it is facing!
- Be smart about this

Don’t get run over!

- Use an if statement to prevent the robot from getting too close.
  - The norm of the position along the floor is how far the robot is from you
    - We did this a couple of slides ago!
  - vec.norm()
  - Use an if statement to prevent the robot from coming too close. Try 0.25/0.35 meters. If the robot is closer than that, simply don’t send your nav goal to the robot.

Summary

- You figure out which direction the marker is in from the Pose passed in.
- You stick it into a Vector3d
- If you use all 3 elements (x,y,z), its norm is how far away the target is
- When you solve for the direction the target is in, use (x,0,z), because elevation from the floor is irrelevant

Summary

- Using some high school trig, you can find the angle you are trying to turn to
  - C++ has the atan2 function, which can solve this from the opposite and adjacent sides of the triangle, as in trig.
  - The unit circle can provide guidance on how to do this.
- tf::Quaternion can be instantiated with r, p, y in its parameters
  - And x(), y(), z(), w() is still the quaternion

Summary

- Using these things, you can find your orientation and position
- Put those into your target_pose in your MoveBaseGoal

Summary

- Don’t get run over or turned past!
  - Test if the distance to the target is less than 0.5m (or tighter if you like to live dangerously) before sending your nav goal.
  - Don’t move the full distance, you will overshoot your goal, instead
    - Move in smaller chunks
      - I’ve been successful with 0.25 meters
    - Halve your yaw (or make it proportional to the distance traveled, but that’s harder; and it doesn’t bother me if you don’t do this)

Summary

- It’s tempting to NOT wait call _ac.waitForResult();
- However, the Segway base makes its final TURN for orientation at the END of its motion
- So you may have problems tuning your system if you do not do this.
  - However, if you get it RIGHT, your system will follow you more smoothly.
  - This will involve proportionally turning to an angle relative to the distance traveled
    - You’ve been warned

What makes a good project?

- Have a real question or engineering goal
  - FRI II will ***require*** a good scientific question
- A question should be related to AI
  - How can a robot identify common household objects?
  - How can a robot localize itself?
  - How can a robot recognize itself?
  - Is there a faster way to learn this policy?
  - Can I teach a robot to <blank>?

What makes a good project?

- Human-Robot Interaction
  - These projects focus on the human
    - Sometimes exclusively
      - Does a human understand this motion as I think they will?
    - Sometimes through the lens of AI
      - Can a robot interpret a human’s gaze?

What makes a good project?

- Machine Learning
  - Classification
    - Tell me the names of objects in the environment.
      - Can I improve over Yolo?
  - Reinforcement Learning
    - Learn a policy based on a reward signal?
      - Can I train the robot to
        - Pick up this object?
        - Navigate to this goal?
        - Avoid a certain area?
- ML projects are a little difficult for this class

What makes a good project?

- Engineer a component into BWI!
  - One team has already decided on a related project
    - Read the CS events calendar and provide information and directions based on the events in it!
      - This is a great project idea!

Being practical about this

- Your project should demonstrate that you've learned something about robotics.
- I and the mentors will help you along the way.
- Your project should be tractable
  - Don't pick an internet-scale deep learning project, you haven't learned things like that yet!

Past Projects – Friendly Faces

- Used face recognition software to learn people's names
  - People stood in front of the robot.
  - The robot asked their name.
  - When next faced with that person, the robot said their name to them.
  - https://www.youtube.com/watch?v=FXVZ6VCu8TA

Past Projects – Make Way Please!

- Sometimes the BWIbots are unable to navigate to a goal because the path is blocked.
- For this project, the students modified the navigation code to ask for people to get out of the way when people were blocking the path.
- The default behavior is to try a different path.

Past Projects – Using Hand Gestures to Command BWI Segbots

- Used simple hand gestures to control the BWIBots
  - A library called OpenNI enables this approach.

Your goal

- Find something interesting to you
- Find something doable in 1 month
- For the rest of the class, we will discuss this.

Other great ideas!

- Person following
  - Run Yolo object recognition, try to follow a person down the hallway.
- Person leading
  - Can you lead a person to a location?
  - When you do so, are you sure the person is still with you?
- Identifying gestures, characteristics
  - Find someone with a blue shirt.
  - Find a coffee pot.
  - Identify a person’s height.
  - How many people are in a crowd?