Syllabus: Robotics Software Engineer Nanodegree Program

Prerequisites

Make sure to set aside adequate time on your calendar for focused work. In order to succeed, we recommend having experience with:

- Advanced knowledge in any object-oriented programming language, preferably C++
- Intermediate Probability
- Intermediate Calculus
- Intermediate Linear Algebra
- Basic Linux Command Lines

If you'd like to prepare for this Nanodegree program, check out our Intro to Self-Driving Cars Nanodegree program to start learning calculus, linear algebra, computer vision, and more.

Contact Info

While going through the program, if you have questions about anything, you can reach us at: robond-support@udacity.com.

For help from Udacity Mentors and your peers, join the community discussion in Student Hub (available only to Nanodegree students).

Program Overview

This program will teach you:

- the software fundamentals to work on robotics using C++, ROS, and Gazebo
- how to build autonomous robotics projects in a Gazebo simulation environment
- probabilistic robotics, including Localization, Mapping, SLAM, Navigation, and Path Planning.

This program is comprised of 6 courses and 5 projects. Each project you build will be an opportunity to demonstrate what you've learned in the lessons. **Your completed projects will become part of a career portfolio that will demonstrate to potential employers that you have skills in C++, ROS, Gazebo, Localization, Mapping, SLAM, Navigation, and Path Planning.**

Depending on how quickly you work through the material, the amount of time required is variable. We have included an hourly estimate for each section of the program. If you spend about 10 hours per week working through the program, you should finish in 14 weeks (approximately 4 months).

Length of Program: 150 Hours*

Frequency of Classes: Self-paced with project deadlines
Instructional Tools Available: Video lectures, Text instructions, Quizzes, Knowledge (forums), and Student Hub (live chat with Udacity Mentors and classmates)

* This is a self-paced program and the length is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. Actual hours may vary.

Project Overview

One of our main goals at Udacity is to help you **create a job-ready portfolio**. Building a project is one of the best ways to test the skills you've acquired and to demonstrate your newfound abilities to future employers. Throughout this Nanodegree program, you'll have the opportunity to prove your skills by building the following projects:

- **Build My World**
  - Have fun as you learn how to use Gazebo to visualize and prototype a robotics environment.
- **Go Chase It!**
  - Demonstrate your new proficiency with ROS, C++, and Gazebo by building a robot that can chase colored balls.
- **Where am I?**
  - Use the Adaptive Monte Carlo Localization algorithm in ROS to estimate your robot's position.
- **Map My World**
  - Deploy RTAB-Map on your simulated robot to localize your robot and create 2D and 3D maps of your environment.
- **Capstone: Home Service Robot**
  - Program a home service robot that will autonomously map an environment and navigate to pick up and deliver objects.
- **Optional**: **KUKA Robotic Arm Path Planning**
  - Navigate a KUKA robot through a 2D maze.

In the sections below, you'll find detailed descriptions of each project along with the course material that presents the skills required to complete the project.

Project: Build My World

Project Description:
Use the tools that you've learned in Gazebo to build your first environment.

Key Skills Demonstrated:

- Launching a Gazebo Environment
- Designing in Gazebo

Lesson Content: Gazebo World

Lesson	Learning Outcomes
Introduction to Gazebo	➔ Work with the Gazebo simulator to build new environments, and deploy assets.
Project: Build My World	➔ Step by Step - Design and build your first Gazebo environment.

Project: Go Chase It!

Project Description:
Demonstrate your proficiency with ROS, C++, and Gazebo by building a ball-chasing robot. You will first design a robot inside Gazebo, house it in the world you have built in the Build My World project, and code a C++ node in ROS to chase yellow balls.

Key Skills Demonstrated:

- Building Catkin Workspaces
- ROS node creation
- ROS node communication
- Using additional ROS packages
- Gazebo world integration
- Additional C++ practice
- RViz Integration

Lesson Content: ROS Essentials

Lesson	Learning Outcomes

Introduction to ROS	➔ Obtain an architectural overview of the Robot Operating System Framework.
Packages & Catkin Workspaces	➔ Learn the ROS workspace structure, essential command line utilities, and how to manage software packages within a project.
Write ROS Nodes	➔ Write ROS nodes in C++.
Project: Go Chase It!	➔ Step by Step - build your first robot in Gazebo. ➔ Step by Step - build a C++ service server node in ROS. ➔ Step by Step - build a C++ service client node in ROS.

Project: Where Am I?

Project Description:
You will interface your own mobile robot with the Adaptive Monte Carlo Localization algorithm in ROS to estimate your robot's position as it travels through a predefined set of waypoints. You'll also tune different parameters to increase the localization efficiency of the robot.

Key Skills Demonstrated:

- Implementation of Adaptive Monte Carlo Localization in ROS
- Understanding of tuning parameters required

Lesson Content: Localization

Lesson	Learning Outcomes
Introduction to Localization	➔ Learn what it means to localize and the challenges behind it.
Kalman Filters	➔ Learn the Kalman Filter and its importance in estimating noisy data.
Lab: Kalman Filters	➔ Implement an Extended Kalman Filter package with ROS to estimate the position of a robot.
Monte Carlo Localization	➔ Learn the MCL (Monte Carlo Localization) algorithm to localize robots.
Build MCL in C++	➔ Code the MCL algorithm in C++.
Project: Where am I?	➔ Set up and explore the steps for the Where Am I? Project using AMCL with ROS in C++.

Project: Map My World

Project Description:
Students will interface their robot with an RTAB Map ROS package to localize it and build 2D and 3D maps of their environment. Students must put all the pieces together properly to launch the robot and then teleop it to map its environment.

Key Skills Demonstrated:

- SLAM implementation with ROS/Gazebo
- ROS debugging tools: rqt, roswtf

Lesson Content: Mapping and SLAM

Lesson	Learning Outcomes
Introduction to Mapping and SLAM	➔ Learn the Mapping and SLAM concepts, as well as the algorithms.
Occupancy Grid Mapping	➔ Map an environment by coding the Occupancy Grid Mapping algorithm with C++.
Grid-based FastSLAM	➔ Simultaneously map an environment and localize a robot relative to the map with the Grid-based FastSLAM algorithm. ➔ Interface a turtlebot with a Grid-based FastSLAM package with ROS to map an environment.
GraphSLAM	➔ Simultaneously map an environment and localize a robot relative to the map with the GraphSLAM algorithm.
Project: Map My World Robot	➔ Deploy RTAB-Map on your simulated robot to localize it and create 2D and 3D maps of your environment.

Capstone Project: Home Service Robot

Project Description:
In this capstone project, you will use a SLAM package to autonomously map an environment. Then, you will interface your robot with a path planning and navigation ROS package to move objects within an environment.

Key Skills Demonstrated:

- Advanced ROS and Gazebo integration
- ROS Navigation stack

- Path planning

Lesson Content: Path Planning and Navigation

Lesson	Learning Outcomes
Intro to Path Planning and Navigation	➔ Learn what the lessons in Path Planning and Navigation will cover.
Classic Path Planning	➔ Learn a number of classic path planning approaches that can be applied to low-dimensional robotic systems.
Lab: Path Planning	➔ Code the BFS and A* algorithms in C++.
Sample-Based and Probabilistic Path Planning	➔ Learn about sample-based and probabilistic path planning, and how they can improve on the classic approach.
Capstone Project: Home Service Robot	➔ Program a home service robot that will autonomously map an environment and navigate to pick up and deliver objects.

Optional Project: KUKA Path Planning

Project Description:
Students will apply what they have learned about ROS and path planning to search for a path and navigate a KUKA robot through a 2D maze.

Key Skills Demonstrated:

- Path planning
- Using C++ and Python with external ROS API

Lesson Content: Optional KUKA Path Planning Project

Lesson	Learning Outcomes
Project Introduction	➔ Learn the requirements of the project.
Project Details	➔ Learn the project specifications and how to get started.
Project: KUKA Path Planning	➔ Search for a path and navigate a KUKA robot through a 2D maze.