UDACITY

NANODEGREE PROGRAM SYLLABUS

# Self-Driving Car Engineer

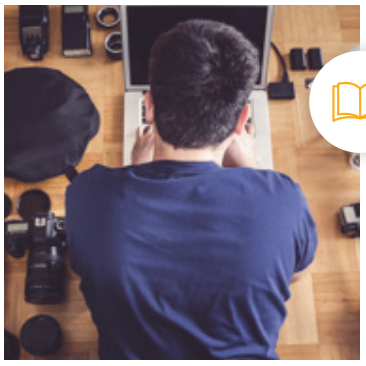
# Overview

You'll first apply computer vision and deep learning to automotive problems, including detecting lane lines, predicting steering angles, and more. Next, you'll learn sensor fusion, which you'll use to filter data from an array of sensors in order to perceive the environment. Then, you'll work with a team to program Carla, Udacity's real self-driving car.

IN COLLABORATION WITH

Mercedes-Benz NVIDIA UBER ATG DiDi BMW McLaren APPLIED TECHNOLOGIES

**Estimated Time**:
6 Months at
15 hrs/week



**Prerequisites**:
Python, C++,
Mathematics

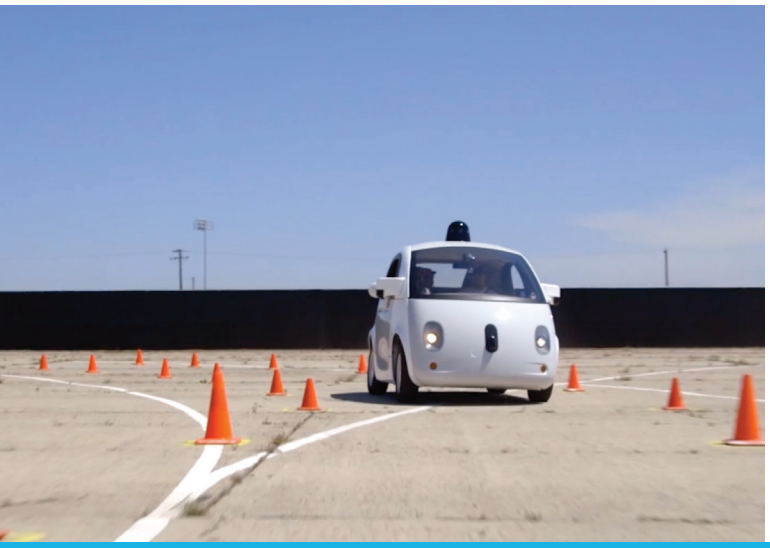
**Flexible Learning**:
Self-paced, so
you can learn on
the schedule that
works best for you

**Need Help?**
udacity.com/advisor
Discuss this program
with an enrollment
advisor.

# Course 1: Introduction

In this course, you will learn about how self-driving cars work, and you'll take a crack at your very first autonomous vehicle project - finding lane lines on the road! We'll also introduce the Nanodegree program and the services we provide over the course of the journey.

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Welcome** | Take your first steps towards becoming a Self-Driving Car Engineer! In this lesson, we'll introduce you to the program, help you discover the services we provide, and show you all the incredible projects you'll build. Get ready for an incredible 6-month journey! |
| LESSON TWO | **Workspaces** | Many projects and some quizzes will be accessed via Workspaces. These workspaces streamline environment setup, simplify project submission, and can be enabled with GPU support. All workspaces are Linux-based and can be interfaced via a shell (BASH). Some workspace interfaces are direct from the shell, others run a JUPYTER Notebook server and interaction is mainly through the JUPYTER notebook interface. |



# Course 2: Computer Vision

You'll use a combination of cameras, software, and machine learning to find lane lines on difficult roads and to track vehicles. You'll start with calibrating cameras and manipulating images, and end by applying support vector machines and decision trees to extract information from video.

**Course Project**
Finding Lane Lines on a Road

In this project, you will be writing code to identify lane lines on the road, first in an image, and later in a video stream (really just a series of images). To complete this project, you will use the tools you learned about in the lesson and build upon them.

**Course Project**
Advanced Lane Finding

In this project, your goal is to write a software pipeline to identify the lane boundaries in a video from a front-facing camera on a car.

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Computer Vision Fundamentals** | In this first lesson, you'll get taste of some basic computer vision techniques to find lane markings on the road. We will be diving much deeper into computer vision in later lessons, so just relax and have some fun in this first week! |
| LESSON TWO | **Advanced Computer Vision** | Discover more advanced computer vision techniques, like distortion correction and gradient thresholding, to improve upon your lane lines algorithm! |

# Course 3: Deep Learning

Deep learning has become the most important frontier in both machine learning and autonomous vehicle development. Experts from NVIDIA and Uber ATG will teach you to build deep neural networks and train them with data from the real world and from the Udacity simulator. By the end of this course, you'll be able to train convolutional neural networks to classify traffic signs, and to drive a vehicle in the simulator the same way you drive it yourself!

**Course Project**
Traffic Sign Classifier

You just finished getting your feet wet with deep learning. Now put your skills to the test by using deep learning to classify different traffic signs! In this project, you will use what you've learned about deep neural networks and convolutional neural networks to classify traffic signs.

**Course Project**
Behavioral Cloning

Put your deep learning skills to the test with this project! Train a deep neural network to drive a car like you!

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Neural Networks** | Learn to build and train neural networks, starting with the foundations in linear and logistic regression, and culminating in backpropagation and multilayer perceptron networks. |
| LESSON TWO | **TensorFlow** | Vincent Vanhoucke, Principal Scientist at Google Brain, introduces you to deep learning and Tensorflow, Google's deep learning framework. |
| LESSON THREE | **Deep Neural Networks** | Vincent walks you through how to go from a simple neural network to a deep neural network. You'll learn about why additional layers can help and how to prevent overfitting. |

| | | |
|---|---|---|
| **LESSON FOUR** | **Convolutional Neural Networks** | Vincent explains the theory behind Convolutional Neural Networks and how they help us dramatically improve performance in image classification. |
| **LESSON FIVE** | **Keras** | Take on the neural network framework, Keras. You'll be amazed how few lines of code you'll need to build and train deep neural networks! |
| **LESSON SIX** | **Transfer Learning** | Learn about some of the most famous neural network architectures and how you can use them. By the end of this lesson, you'll know how to create new models by leveraging existing canonical networks. |


# Course 4: Sensor Fusion

Tracking objects over time is a major challenge for understanding the environment surrounding a vehicle. Sensor fusion engineers from Mercedes-Benz will show you how to program fundamental mathematical tools called Kalman filters. These filters predict and determine with certainty the location of other vehicles on the road. You'll even learn to do this with difficult-to-follow objects, by using an advanced technique: the extended Kalman filter.

**Course Project**
Extended Kalman Filte

In this project, you'll apply everything you've learned so far about Sensor Fusion by implementing an Extended Kalman Filter in C++!

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Sensors** | Meet the team at Mercedes who will help you track objects in real-time with Sensor Fusion. |
| LESSON TWO | **Kalman Filters** | Learn from the best! Sebastian Thrun will walk you through the usage and concepts of a Kalman Filter using Python. |
| LESSON THREE | **C++ Checkpoint** | Are you ready to build Kalman Filters with C++? Take these quizzes to find out. |
| LESSON FOUR | **Extended Kalman Filters** | In this lesson, you'll build a Kalman Filter in C++ that's capable of handling data from multiple sources. Why C++? Its performance enables the application of object tracking with a Kalman Filter in real-time. |

# Course 5: Localization

Localization is how we determine where our vehicle is in the world. GPS is great, but it's only accurate to within a few meters. We need single-digit centimeter-level accuracy! To achieve this, Mercedes-Benz engineers will demonstrate the principles of Markov localization to program a particle filter, which uses data and a map to determine the precise location of a vehicle.

**Course Project**
Kidnapped Vehicle

In this project, you'll build a particle filter and combine it with a real map to localize a vehicle!

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Introduction to Localization** | Meet the team that will guide you through the localization lessons! |
| LESSON TWO | **Markov Localization** | In this lesson, you'll learn the math behind localization as well as how to implement Markov localization in C++. |
| LESSON THREE | **Motion Models** | Here you'll learn about vehicle movement and motion models to predict where your car will be at a future time. |
| LESSON FOUR | **Particle Filters** | Sebastian will teach you what a particle filter is as well as the theory and math behind the filter. |

**LESSON FIVE**

**Implementation of a Particle Filter**

Now that you understand how a particle filter works, you'll learn how to code a particle filter.

# Course 6: Path Planning

Path planning routes a vehicle from one point to another, and it handles how to react when emergencies arise. The Mercedes-Benz Vehicle Intelligence team will take you through the three stages of path planning. First, you'll apply model-driven and data-driven approaches to predict how other vehicles on the road will behave. Then you'll construct a finite state machine to decide which of several maneuvers your own vehicle should undertake. Finally, you'll generate a safe and comfortable trajectory to execute that maneuver.

**Course Project**
Highway Driving

Drive a car down a highway with other cars using your own path planner.

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Search** | In this lesson you will learn about discrete path planning and algorithms for solving the path planning problem. |
| LESSON TWO | **Prediction** | In this lesson you'll learn how to use data from sensor fusion to generate predictions about the likely behavior of moving objects. |
| LESSON THREE | **Behavior Planning** | In this lesson you'll learn how to think about high level behavior planning in a self-driving car. |
| LESSON FOUR | **Trajectory Generation** | In this lesson, you'll use C++ and the Eigen linear algebra library to build candidate trajectories for the vehicle to follow. |

# Course 7: Control

Ultimately, a self-driving car is still a car, and we need to send steering, throttle, and brake commands to move the car through the world. Uber ATG will walk you through building both proportional-integral-derivative (PID) controllers and model predictive controllers. Between these control algorithms, you'll become familiar with both basic and advanced techniques for actuating a vehicle.

**Course Project**
PID Controller

In this project you'll revisit the lake race track from the Behavioral Cloning Project. This time, however, you'll implement a PID controller in C++ to maneuver the vehicle around the track!

**Course Project**
Build Your Online Presence Using LinkedIn

Showcase your portfolio on LinkedIn and receive valuable feedback.

**Course Project**
Optimize Your GitHub Profile

Build a GitHub Profile on par with senior software engineers.

| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **PID Control** | In this lesson you'll learn about and how to use PID controllers with Sebastian! |

# Course 8: System Integration

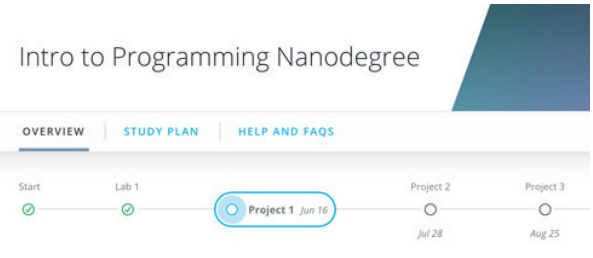
This is capstone of the entire Self-Driving Car Engineer Nanodegree Program! We'll introduce Carla, the Udacity self-driving car, and the Robot Operating System that controls her. You'll work with a team of other Nanodegree students to combine what you've learned over the course of the entire Nanodegree Program to drive Carla, a real self-driving car, around the Udacity test track!

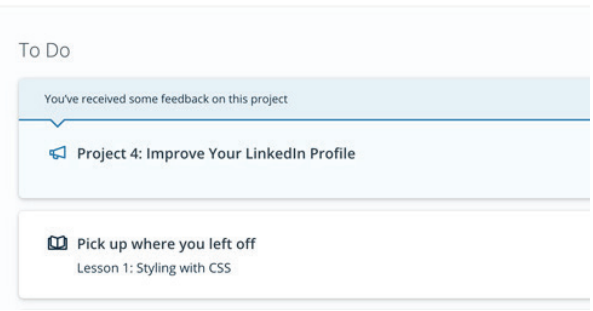
**Course Project**
Programming a Real Self-Driving Car

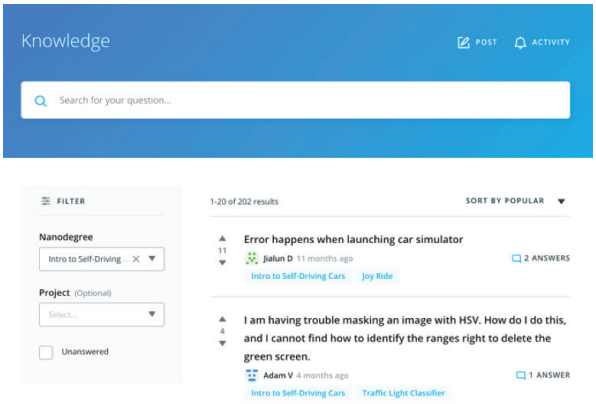
Run your code on Carla, Udacity's autonomous vehicle!

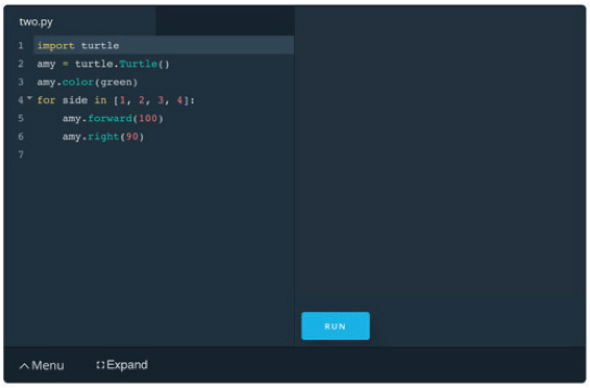
| | LEARNING OUTCOMES | |
|---|---|---|
| LESSON ONE | **Autonomous Vehicle Architecture** | Learn about the system architecture for Carla, Udacity's autonomous vehicle. |
| LESSON TWO | **Introduction to ROS** | Obtain an architectural overview of the Robot Operating System Framework and setup your own ROS environment on your computer. |
| LESSON THREE | **Packages and Catkin Workspaces** | Learn about ROS workspace structure, essential command line utilities, and how to manage software packages within a project. Harnessing these will be key to building shippable software using ROS. |
| LESSON FOUR | **Writing ROS Nodes** | ROS Nodes are a key abstraction that allows a robot system to be built modularly. In this lesson, you'll learn how to write them using Python. |

# Our Classroom Experience




### REAL-WORLD PROJECTS

Build your skills through industry-relevant projects. Get personalized feedback from our network of 900+ project reviewers. Our simple interface makes it easy to submit your projects as often as you need and receive unlimited feedback on your work.

### KNOWLEDGE

Find answers to your questions with Knowledge, our proprietary wiki. Search questions asked by other students, connect with technical mentors, and discover in real-time how to solve the challenges that you encounter.

### STUDENT HUB

Leverage the power of community through a simple, yet powerful chat interface built within the classroom. Use Student Hub to connect with your fellow students in your Executive Program.

### WORKSPACES

See your code in action. Check the output and quality of your code by running them on workspaces that are a part of our classroom.

### QUIZZES

Check your understanding of concepts learned in the program by answering simple and auto-graded quizzes. Easily go back to the lessons to brush up on concepts anytime you get an answer wrong.

### CUSTOM STUDY PLANS

Preschedule your study times and save them to your personal calendar to create a custom study plan. Program regular reminders to keep track of your progress toward your goals and completion of your program.

### PROGRESS TRACKER

Stay on track to complete your Nanodegree program with useful milestone reminders.

# Learn with the Best

## Sebastian Thrun

UDACITY PRESIDENT

Scientist, educator, inventor, and entrepreneur, Sebastian led the self-driving car project at Google X and founded Udacity, whose mission is to democratize education by providing lifelong, on-demand learning to millions of students around the world.

## David Silver

CURRICULUM LEAD

David Silver leads the School of Autonomous Systems at Udacity. Before Udacity, David was a research engineer on the autonomous vehicle team at Ford. He has an MBA from Stanford, and a BSE in computer science from Princeton.

## Ryan Keenan

COURSE DEVELOPER

Ryan has a PhD in Astrophysics and a passion for teaching and learning. He is also a lead instructor in the Robotics Nanodegree program. When he's not building Udacious learning content you'll find him up in the mountains or out in the surf.

## Cezanne Camacho

COURSE DEVELOPER

Cezanne is an expert in computer vision with an M.S. in Electrical Engineering from Stanford University.Inspired by anyone with the drive and imagination to learn something new, she aims to create more inclusive and effective STEM education.

# Learn with the Best

## Mercedes-Benz

MERCEDES-BENZ TEAM

Mercedes-Benz R&D North America develops the world's most advanced automotive technology and vehicle design with luxury and style. The team from Mercedes built our Sensor Fusion, Localization, and Path Planning content.


## NVIDIA

NVIDIA TEAM

NVIDIA is a company built upon great minds and groundbreaking research. GPU deep learning has ignited modern AI - the next era of computing - with the GPU acting as the brain of computers, robots, and self-driving cars that can perceive and understand the world.


## Uber ATG

UBER ATG TEAM

The Advanced Technologies Group is comprised of Uber's self-driving engineering team dedicated to self-driving technologies, mapping, and vehicle safety.

# All Our Nanodegree Programs Include:

## EXPERIENCED PROJECT REVIEWERS

REVIEWER SERVICES

- Personalized feedback & line by line code reviews
- 1600+ Reviewers with a 4.85/5 average rating
- 3 hour average project review turnaround time
- Unlimited submissions and feedback loops
- Practical tips and industry best practices
- Additional suggested resources to improve

## TECHNICAL MENTOR SUPPORT

MENTORSHIP SERVICES

- Questions answered quickly by our team of technical mentors
- 1000+ Mentors with a 4.7/5 average rating
- Support for all your technical questions

## PERSONAL CAREER SERVICES

CAREER COACHING

- Personal assistance in your job search
- Monthly 1-on-1 calls
- Personalized feedback and career guidance
- Interview preparation
- Resume services
- Github portfolio review
- LinkedIn profile optimization

# Frequently Asked Questions

## PROGRAM OVERVIEW

### WHY SHOULD I ENROLL?

The Self-Driving Car Engineer Nanodegree program is one of the only programs in the world to both teach students how to become a self-driving car engineer, and support students in obtaining a job within the field of autonomous systems. The program's nine projects equip students with invaluable skills across a wide array of critical topics, including deep learning, computer vision, sensor fusion, localization, controllers, vehicle kinematics, automotive hardware, and more. As part of their capstone project, students have the rare opportunity to run their code on an actual autonomous vehicle owned by Udacity.

### WHAT JOBS WILL THIS PROGRAM PREPARE ME FOR?

Our wide-ranging curriculum will prepare you for a variety of roles in the autonomous vehicle industry, including: System Software Engineer, Deep Learning Engineer, Vehicle Software Engineer, Localization and Mapping Engineer and many others. If you elect to work outside of automotive engineering, your foundation in deep learning and robotics will enable you to fill any number of related roles in artificial intelligence, computer vision, machine learning, and more.

### HOW DO I KNOW IF THIS PROGRAM IS RIGHT FOR ME?

This advanced Nanodegree program is ideal for anyone with a programming, technical, or quantitative background who is interested in obtaining a job within the field of autonomous systems, or refreshing or developing their skills within the realm of machine and deep learning, systems integration, sensor fusion, and many others.

### WHAT IS THE DIFFERENCE BETWEEN THE INTRO TO SELF-DRIVING CARS NANODEGREE PROGRAM AND THE SELF-DRIVING CAR ENGINEER NANODEGREE PROGRAM?

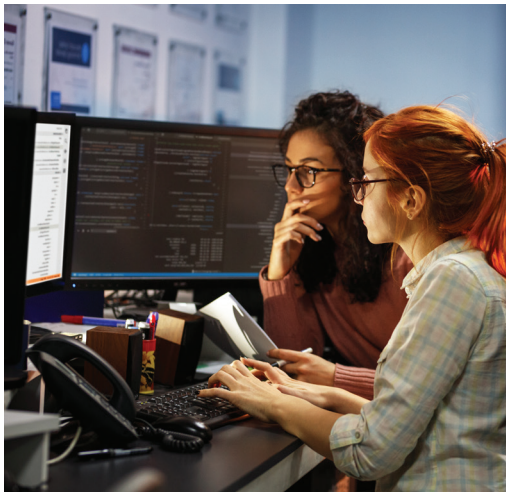
The **Intro to Self-Driving Cars Nanodegree** program is an intermediate program open to anyone with an interest in autonomous systems, who has some programming experience, and/or a quantitative background. The **Self-Driving Car Engineer Nanodegree** program is an advanced program focusing on in-depth knowledge of autonomous systems. The program is designed for those with moderate to high programming, technical, and/or quantitative skills.

## ENROLLMENT AND ADMISSION

### DO I NEED TO APPLY? WHAT ARE THE ADMISSION CRITERIA?

# FAQs Continued

There is no application. This Nanodegree program accepts everyone, regardless of experience and specific background.



### WHAT ARE THE PREREQUISITES FOR ENROLLMENT?

Students should have prior experience with the following:

- Intermediate Python or C++
- Basic Linear Algebra
- Basic Calculus
- Basic Statistics
- Basic Physics

You will also need to be able to communicate fluently and professionally in written and spoken English.

### IF I DO NOT MEET THE REQUIREMENTS TO ENROLL, WHAT SHOULD I DO?

We have a number of Nanodegree programs and free courses that can help you prepare, including: **Intro to Self-Driving Cars Nanodegree** program, **Robotics Software Engineer Nanodegree program**, and **AI for Robotics Course**.

## TUITION AND TERM OF PROGRAM

### HOW IS THIS NANODEGREE PROGRAM STRUCTURED?

The Self-Driving Car Engineer Nanodegree program is comprised of content and curriculum to support nine (9) projects. We estimate that students can complete the program in six (6) months, working 15 hours per week.

Each project will be reviewed by the Udacity reviewer network. Feedback will be provided and if you do not pass the project, you will be asked to resubmit the project until it passes.

### CAN I SWITCH MY START DATE? CAN I GET A REFUND?

Please see the Udacity Nanodegree program **FAQs** for policies on enrollment in our programs.

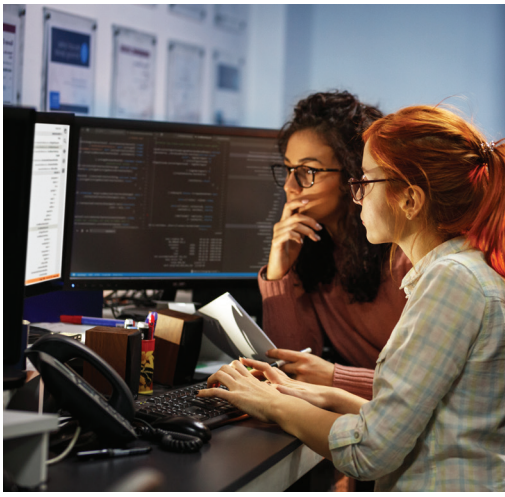
### HOW LONG IS THIS NANODEGREE PROGRAM?

Access to this Nanodegree program runs for the length of time specified in your subscription plan. See the **Terms of Use** for other policies around the terms of access to our Nanodegree programs.

# FAQs Continued

### I HAVE GRADUATED FROM THE SELF-DRIVING CAR ENGINEER NANODEGREE PROGRAM BUT I WANT TO KEEP LEARNING. WHERE SHOULD I GO FROM HERE?

Once you have completed the Self-Driving Car Engineer Nanodegree program, the **Robotics Software Engineer Nanodegree** program and the **Flying Car Nanodegree** program are ideal for continuing your learning.



### WHAT SOFTWARE AND VERSIONS WILL I NEED IN THIS PROGRAM?

For this Nanodegree program, you will need to the minimum equipment requirements outlined here: **https://www.udacity.com/tech-requirements**.

### WHAT SOFTWARE AND VERSIONS WILL I NEED IN THIS PROGRAM?

The following versions are taught in this program (subject to update):

- TensorFlow Version 1.3
- Keras version 2
- ROS Kinetic
- Python Version 3
- C++ Version 11