UDACITY

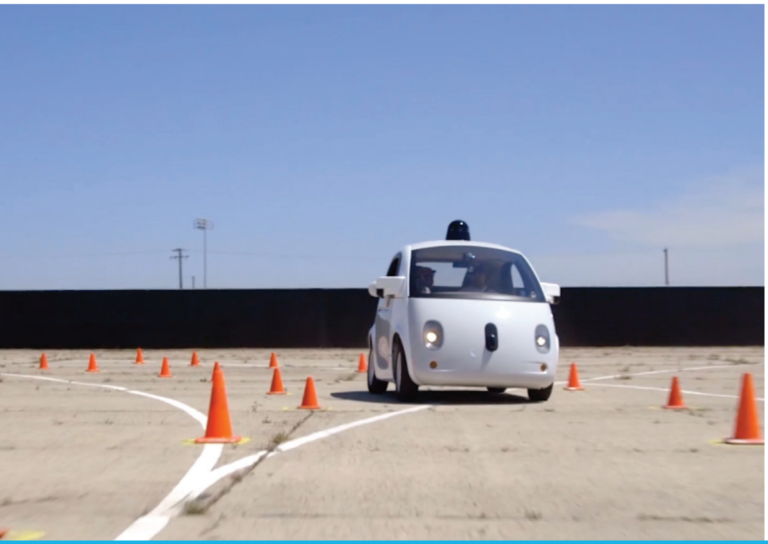
NANODEGREE PROGRAM SYLLABUS

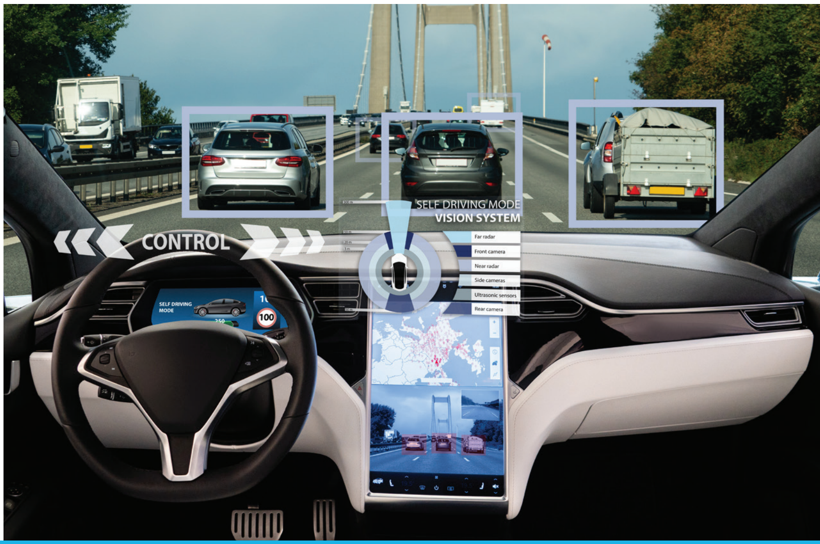
# Self-Driving Car Engineer

# 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. |





UDACITY

# 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

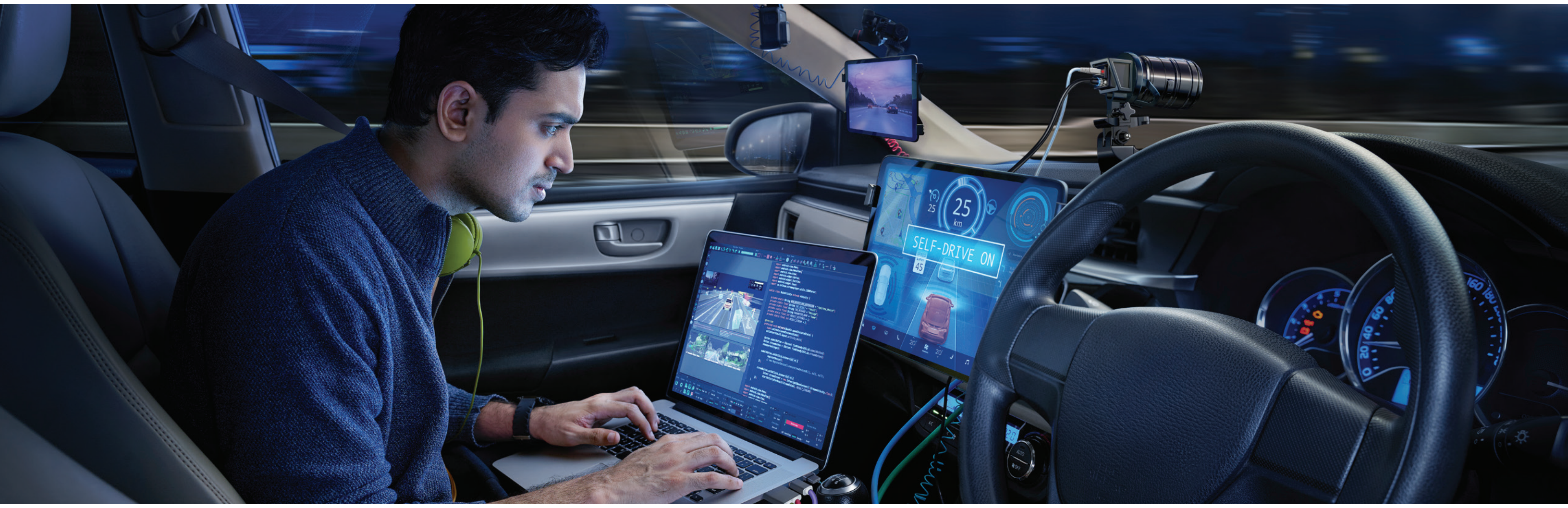
You just finished getting your feet wet with deep learning. Now put your skills to the test by using deep learning to classify different traffic signs! In this project, you will use what you've learned about deep neural networks and convolutional neural networks to classify traffic signs.

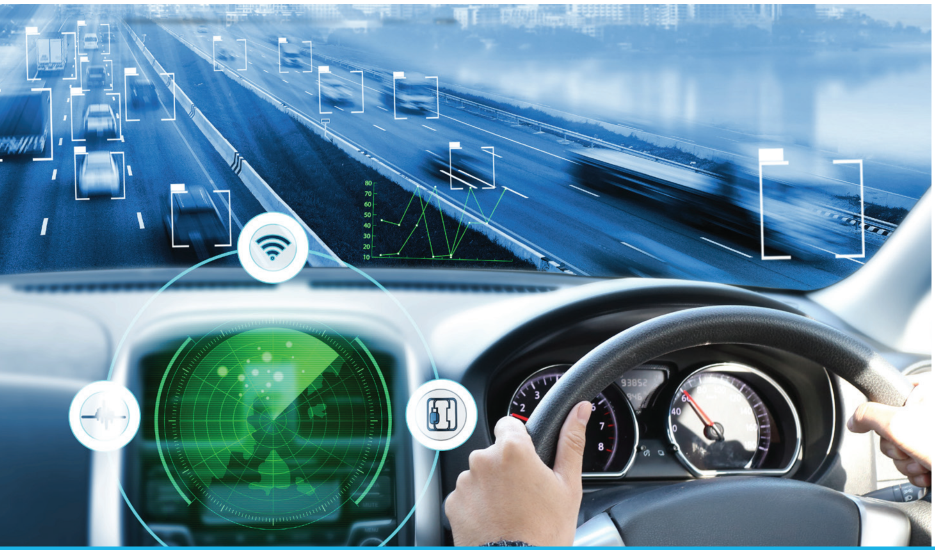
**Course Project**
Behavioral Cloning

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

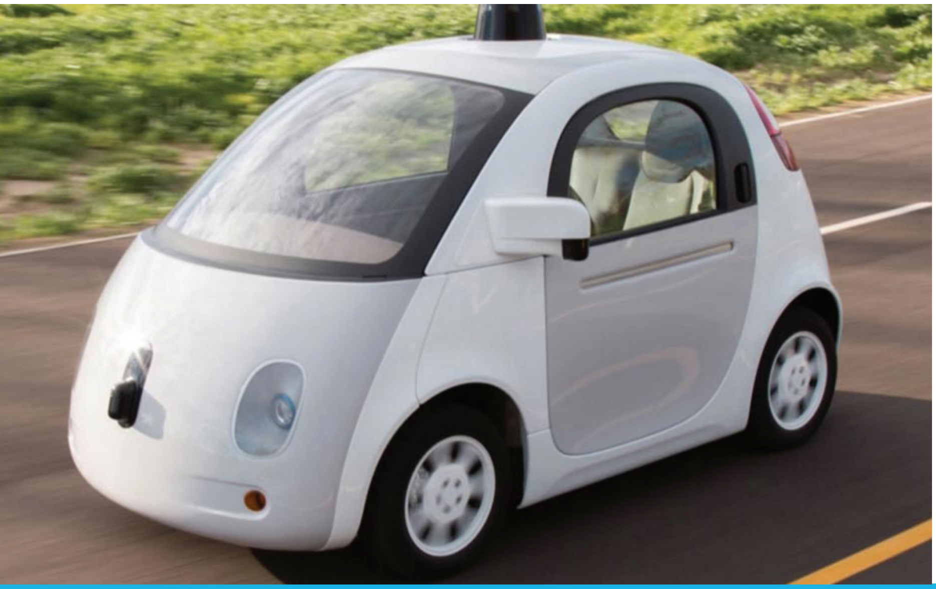
Localization is how we determine where our vehicle is in the world. GPS is great, but it's only accurate to within a few meters. We need single-digit centimeter-level accuracy! To achieve this, Mercedes-Benz engineers will demonstrate the principles of Markov localization to program a particle filter,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