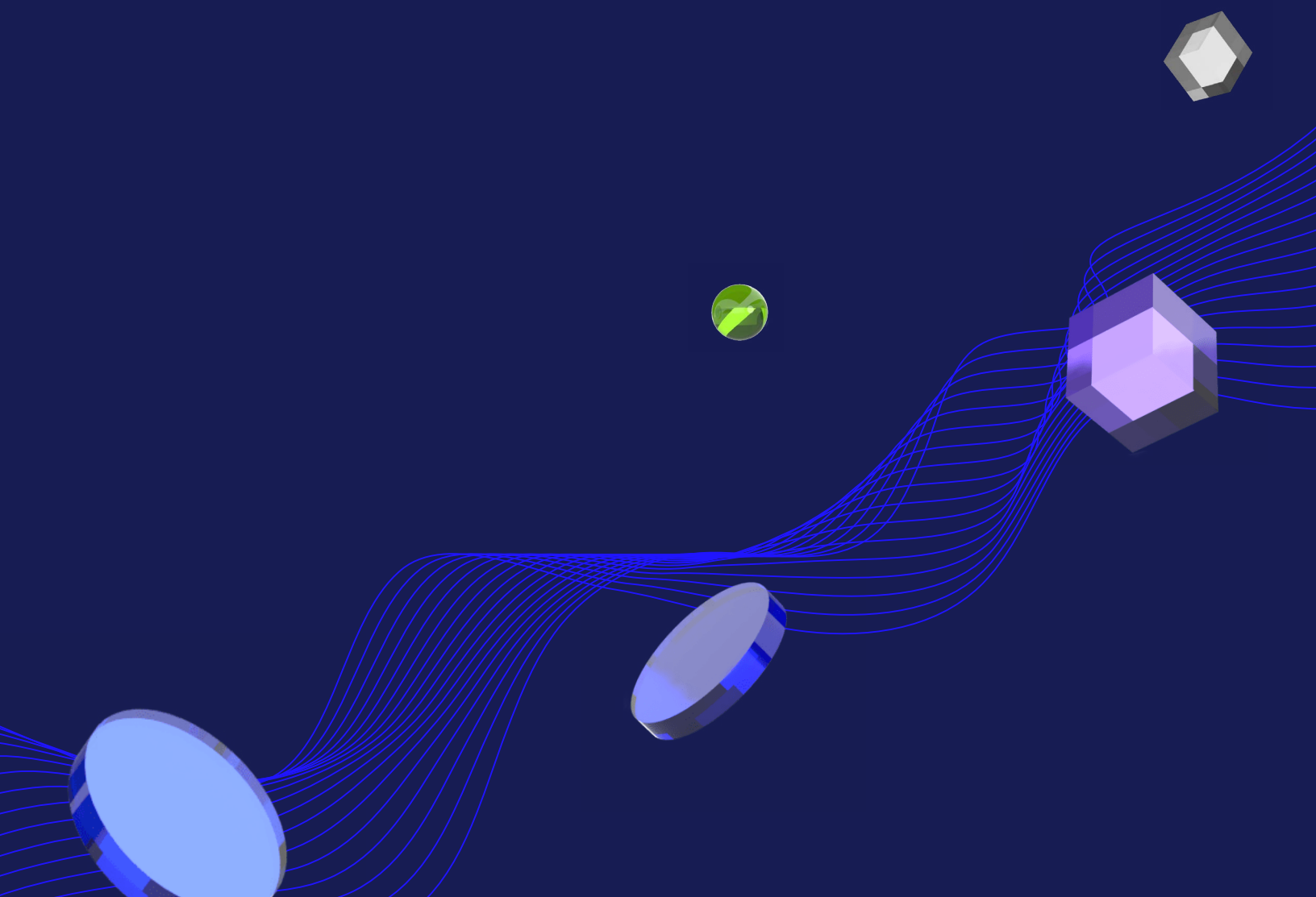
INDIVIDUAL LEARNERS

UDACITY

SCHOOL OF AUTONOMOUS SYSTEMS

Intro to Self-Driving Cars

Nanodegree Program Syllabus



Overview

In this program, learners will sharpen their Python skills, apply C++, apply matrices and calculus in code, and touch on computer vision and machine learning. These concepts will be applied to solving self-driving car problems. At the end, learners will be ready for our Self-Driving Car Engineer Nanodegree program.

Program information

Estimated Time

4 months at 10hrs/week*

Skill Level

Intermediate

Prerequisites

A well-prepared learner should have some experience with programming—writing short scripts in any programming language—and algebra is required. They should feel comfortable reading and modifying code that they are given, but solving problems in code can still be challenging.

Required Hardware/Software

None

*The length of this program is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. If you spend about 5-10 hours per week working through the program, you should finish within the time provided. Actual hours may vary.

Course 1

Bayesian Thinking

Learn a mathematical framework known as Bayesian Interference. This is the same framework that underlies a self-driving car's understanding of itself and the world around it. It's what allows a car to use unreliable sensor data to achieve surprisingly accurate estimates of its own location in the world (known as localization). It also underlies the tracking algorithms that self-driving cars use to predict what other traffic on the road will do in the future. By the end of this course learners will not only understand Bayesian Interference, but will also be able to see the world the way a self-driving car does.

Course Project

Joy Ride

Jump into writing code that controls a simulated vehicle. Send throttle and steering commands to the car to get it to navigate around a test track.

Course Project

2D Histogram Filter in Python

In this project, learners will write the sense and move functions for a 2-dimensional histogram filter in Python.

Course 2

Working with Matrices

This course will focus on two tools which are vital to self-driving car engineers: object oriented programming and linear algebra. Object oriented programming (OOP) is an approach to programming that is especially useful when the things we are modeling in our code have obvious real-world counterparts (as is often the case when writing code for something as real as a car). In this course, learners will explore the basics of OOP with a focus on how to use classes which others have created.

Linear algebra is the mathematics of matrices. For our purposes in this course we will treat it as a tool. The main goal will be to gain a basic proficiency with this powerful tool by making the notation of linear algebra approachable and understandable. With this tool in hand learners will be able to implement any of the numerous algorithms they'll encounter as they continue their self-driving car career (including the ubiquitous Kalman filter).

The goal of this course is very pragmatic: By the end learners will know how to use the tools—deeper theoretical understanding is something they will acquire gradually as they continue their journey through this Nanodegree program, and the rest of their career.

Course Project

Implement a Matrix Class

Practice using object oriented programming and matrix math skills by filling out the methods in a partially-completed Matrix class.

Course 3

C++ Basics

C++ is the language of self-driving cars, and code written in this language can run incredibly fast. This course will be the first step in a long and rewarding journey towards C++ expertise. The goal of this course is translation—learners will be able to translate a program written in Python into C++.

Course Project

Translate Python to C++

In this project learners will apply your knowledge of C++ syntax by translating the histogram filter code from the first course into C++.

Course 4

Performance Programming in C++

In C++ basics learners focused on the bare minimum required to write code that runs correctly. In this course they will start to explore how to write good code that runs correctly. We'll focus primarily on the low level language features of C++ which can make C++ fast, but we'll also discuss other best practices as well.

Course Project

Performant C++

A self-driving car can't afford to waste any cycles or memory unnecessarily. In this project learners will take some functioning (but inefficient) C++ code and optimize it.

Course 5

Navigating Complex Data Structures

What data structure should I use to model this relationship? What algorithm will accomplish that goal? These are the questions that self-driving car engineers think about on a daily basis. Algorithmic thinking is a skill learners will continue to refine throughout their programming careers. In this course learners will focus on some of the data structures and algorithms that show up most frequently in self-driving cars.

Course Project

Planning an Optimal Path

Turn on your self-driving car. Buckle up. And enter a destination. Navigating from A to B is not an easy problem. In this project learners will use their knowledge of data structures (specifically graph data structures) and search algorithms to write an algorithm which uses a map and traffic information to find the quickest route between two points.

Course 6

Visualizing Calculus & Controls

It's sometimes convenient to represent the world as a discrete grid of cells. But that isn't quite right. And when it comes time to actually issue control commands about steering, throttle, and braking we have to stop pretending the world is discrete because, in reality, the world is continuous. In this course learners will dive into the basics of calculus, the mathematics of continuity. To visualize the continuous trajectories of the real world, learners will also use some of Python's most popular visualization libraries.

Course Project

Trajectory Visualizer

As a self-driving car engineer, much of the coding involves simulation, visualization, testing, and debugging. In this project learners will write a visualization tool that will let them visualize the continuous trajectories that come from various search and control algorithms.

Course 7

Machine Learning & Computer Vision

How do you teach a computer the difference between a photo of a car and a photo of a human? As humans we can make the distinction without trying, but teaching this intuition to a computer is much more work. In this course learners will find out how a computer sees an image and how machine learning can teach a computer to identify images programmatically.

Course Project

Image Classifier from Scratch

In this project learners will build an image classifier from scratch. When they're done they'll have an algorithm that can reliably classify an image as "pedestrian" or "car."

Meet your instructors.

Sebastian Thrun

Founder & Chairman at Udacity

Sebastian Thrun is a scientist, educator, inventor, and entrepreneur. As the Founder and Chairman of Udacity, Sebastian's mission is to democratize education by providing lifelong learning to millions of students worldwide. He is also the founder of Google X, where he led projects including the Self-Driving Car, Google Glass, and more.

Andy Brown

Curriculum Lead

Andy has a bachelor's degree in physics from MIT and taught himself to program after college (mostly with Udacity courses). He has been helping Udacity make incredible educational experiences since the early days of the company.

Cezanne Camacho

Course Developer

Cezanne is an expert in computer vision with an MS 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.

Andrew Paster

Instructor

Andrew has an engineering degree from Yale, and has used his data science skills to build a jewelry business from the ground up. He has additionally created courses for Udacity's Self-Driving Car Engineer Nanodegree program.

Anthony Navarro

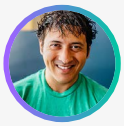
Product Lead

Anthony is a US Army combat veteran with an MS in computer science from Colorado State University. Prior to being a product lead at Udacity, he was a senior software engineer at Lockheed Martin in their Autonomous Systems R&D division.

Elecia White

Engineer, Author & Host

Elecia is an embedded software engineer at Logical Elegance, Inc, the author of O'Reilly's Making Embedded Systems, and host of the Embedded.fm podcast. She enjoys sharing her enthusiasm for engineering and devices.

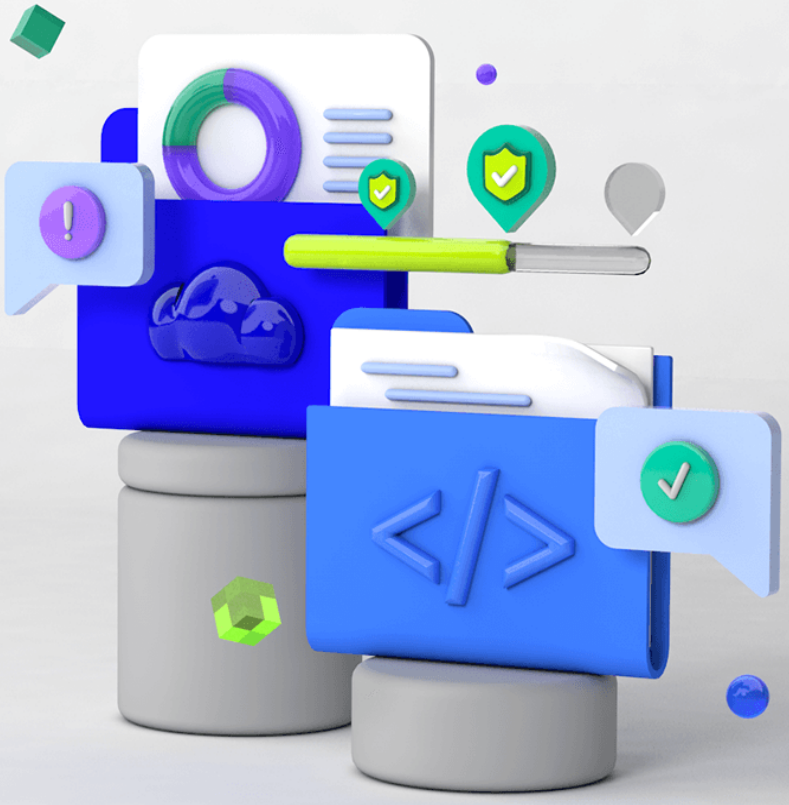


Tarin Ziyaee

Director of AI at Voyage

As the Director of Artificial Intelligence at Voyage Auto, Tarin works to deliver low-cost, self-driving taxis. He brings a total of 14 years experience in perception and deep neural networks working with companies such as Apple.

Udacity's learning experience



Hands-on Projects

Open-ended, experiential projects are designed to reflect actual workplace challenges. They aren't just multiple choice questions or step-by-step guides, but instead require critical thinking.

Quizzes

Auto-graded quizzes strengthen comprehension. Learners can return to lessons at any time during the course to refresh concepts.

Knowledge

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

Custom Study Plans

Create a personalized study plan that fits your individual needs. Utilize this plan to keep track of movement toward your overall goal.

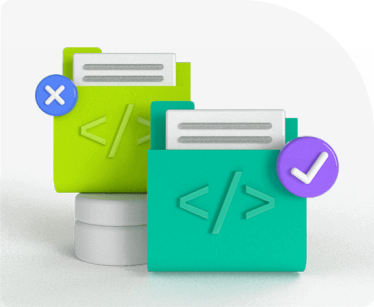
Workspaces

See your code in action. Check the output and quality of your code by running it on interactive workspaces that are integrated into the platform.

Progress Tracker

Take advantage of milestone reminders to stay on schedule and complete your program.

Our proven approach for building job-ready digital skills.



Experienced Project Reviewers

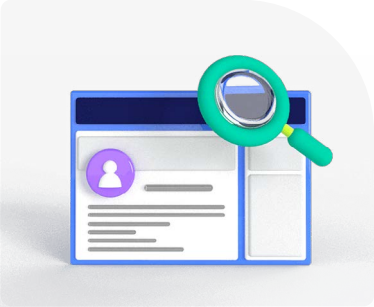
Verify skills mastery.

- Personalized project feedback and critique includes line-by-line code review from skilled practitioners with an average turnaround time of 1.1 hours.
- Project review cycle creates a feedback loop with multiple opportunities for improvement—until the concept is mastered.
- Project reviewers leverage industry best practices and provide pro tips.

Technical Mentor Support

24/7 support unblocks learning.

- Learning accelerates as skilled mentors identify areas of achievement and potential for growth.
- Unlimited access to mentors means help arrives when it's needed most.
- 2 hr or less average question response time assures that skills development stays on track.



Personal Career Services

Empower job-readiness.

- Access to a Github portfolio review that can give you an edge by highlighting your strengths, and demonstrating your value to employers.*
- Get help optimizing your LinkedIn and establishing your personal brand so your profile ranks higher in searches by recruiters and hiring managers.

Mentor Network

Highly vetted for effectiveness.

- Mentors must complete a 5-step hiring process to join Udacity's selective network.
- After passing an objective and situational assessment, mentors must demonstrate communication and behavioral fit for a mentorship role.
- Mentors work across more than 30 different industries and often complete a Nanodegree program themselves.

*Applies to select Nanodegree programs only.

UDACITY

Learn more at

www.udacity.com/online-learning-for-individuals →

12.16.22 | V1.0