Ajaay Chandrasekaran

U.S. Citizen • (248) 719-2472 • ajaay@umich.edu • Novi, MI 48375 • ajaayc.github.io

Education

University of Michigan Sept 2013 - Apr 2018
MSE in Electrical and Computer Engineering — GPA: 3.730/4.000 Ann Arbor, MI
BSE in Computer Science — GPA: 3.759/4.000
Relevant Coursework: Autonomous Robotics Laboratory, Self-Driving Cars: Perception & Control, Machine Learning, Motion Planning, Mobile Robotics Algorithms, Robot Kinematics & Dynamics, Software Engineering, Control Systems Design, Data Structures & Algorithms, Operating Systems, Computer Architecture

Experience

Stellantis (formerly Fiat Chrysler Automobiles) Dec 2019 - Present
Automated Driving Software Engineer Auburn Hills, MI

- Contribute to development of L3 autonomous driving platform by integrating software built by BMW into Stellantis vehicles
- Conduct frequent in-vehicle tests to diagnose and resolve problems in L3 system's C++ ROS codebase and Simulink models

Epic Jul 2018 - Aug 2019
Software Developer Madison, WI

- Migrated several features of electronic medical record software to JavaScript and C# given existing Visual Basic codebase
- Fixed security vulnerabilities and various bugs in SVN version-controlled codebase touched by hundreds of developers

University of Michigan Transportation Research Institute May 2017 - Jun 2018
Graduate Student Research Assistant Ann Arbor, MI

- Constructed vehicle perception system to monitor the safety and quality of construction materials along the US-23 highway over a 2 year period, merging data from a Novatel GPS, Quanergy M-8 Lidar, and a Mobileye unit
- Developed multi-threaded C++ code utilizing VTK and PCL libraries to extract, interpret, and visualize Lidar readings
- Interfaced with a Mobileye unit via vehicle CAN message network to measure lane marking quality over time

Projects

Collision Estimation for Safe Planning • Motion Planning Course Project Mar 2018 - Apr 2018

- Devised and implemented a Gaussian Mixture Model-based algorithm in C++ to expand upon existing Monte Carlo simulation methods to estimate the probability that a robot following a motion plan would collide with an obstacle
- Simulated a linear feedback controller and a Kalman Filter estimate for robot position via OpenRAVE and Armadillo

Deep Neural Network Vehicle Detection System • Self-Driving Cars Course Project Nov 2017 - Dec 2017

- Collaborated with a team to develop a classifier to detect cars in various images captured from GTA simulation
- Utilized TensorFlow Python Object Detector API and NVIDIA CUDA library on AWS EC2 GPU instances to train convolutional neural network (faster RCNN) on ~6,000 training images with known bounding box locations of cars
- Ranked 5th place among 35 teams in class competition by achieving 63% mean absolute error on instructor test set

TrashBot • Autonomous Robotics Laboratory Course Project Mar 2017 - Apr 2017

- Collaborated with a team to build a robot that would navigate and detect and remove trash in urban environments
- Implemented forward and inverse kinematics as well as trajectory planning algorithms in Python for movement of a robotic arm to efficiently pick up and dispose of objects detected via 3D point cloud data from a Microsoft Kinect

KinEval Robotics Framework • Robot Kinematics and Dynamics Course Project Sept 2016 - Dec 2016

- Built a JavaScript-based robot simulation framework to better understand modeling and control of autonomous agents
- Implemented various robotics algorithms, such as path planning via A*, forward kinematics via homogeneous transforms and quaternions, inverse kinematics via the Jacobian matrix, and motion planning via rapidly-exploring random trees

Tools

Languages: C++/C, Python, Matlab, Java, SQL, JavaScript, C#, CSS
Software: Linux terminal, ROS, Gazebo, Git, SVN, CMake/Make, Visual Studio, Simulink, Scikit-Learn, OpenRAVE
Other Skills: Multi-threaded programming, Sockets, CAN networks, REST API design, Linear algebra, Kalman filtering