Udacity Self-Driving Car Engineer Nanodegree Highlights

Term 1

(Dec 12, 2016 - Apr 4, 2017)

Lesson	Brief	Notes
Introduction		

1	Welcome	• Curriculum: https://medium.com/self-driving-cars/term-1-in-depth-on-udacitys-self-driving-car-curriculum-ffcf46af0 8#.api26st07
		• Two approaches to self-driving (both actively pursued):
		  ◦ robotics: model based sensor fusion and navigation
		  ◦ deep learning: learn by mimicking human driving behaviour
		• History: 'The Great Robot Race' (below) on how teams adopted different approaches to 2003/2004 DARPA Gran Challenges . Basically, a few focused on hardware as the main challenge while others focused on software. Software won, and by a big margin! The video is long but worth watching.

2	Finding Lane Lines (project due 22 Dec)	• Review of Canny Edge Detection and Hough Transforms • Lane detection project using OpenCV and Python (Application of my lane detection pipeline on the 'challenge' video)
Deep Learning		
3	Introduction to Deep Learning Module	
4	Regression and Classification	• Regression by least-squares curve fitting • Cross-validation to detect underfitting or overfitting model to data

5	Neural Networks	• Perceptrons
		  ◦ compute operations on boolean data (AND, OR, NOT, XOR)
		  ◦ Linear and nonlinear separability
		  ◦ training: perceptron rule, gradient descent rule
		• Back-propagation
		• Types of inductive bias - preference bias and restriction bias

6	Miniflow	• Hands-on exercise - implementation of a TensorFlow-like system in Python
7	Introduction to Tensorflow	• Multinomial logistic classifiers: Softmax function, 1-hot encoding, cross-entropy • Initialisation and numerical stability of loss function calculations • Stochastic Gradient Descent: Momentum and Learning Rate Decay
8	Deep Neural Networks	• Rectified Linear Units (RELU) • Chain rule to setup back-propagation • Deep vs wide NNs - advantages • Regularisation to prevent overfitting and to increase robustness: L2 regularisation, 'Drop-Out' • Implementing deep NNs in TensorFlow
9	Convolutional Neural Networks	• Weight sharing • How CNNs transform input from spatial to semantic information • Dimensionality equation (number of neurons per layer) • Improving convnets: Pooling, 1x1 convolutions, Inception module • Exercise: Implement Lenet-5 deep neural network model (lecun-98.pdf) in TensorFlow

10	Traffic Sign Classifier (Project Due 23 Jan)	• Traffic sign recognition using multi-scale convolutional networks (sermanet-ijcnn-11.pdf) • Developed a network with 197415 parameters, trained on German traffic signs and tested on UK traffic signs • Report: https://github.com/cvilas/CarND/blob/master/P2-TrafficSigns/Traffic_Sign_Classifier.ipynb
11	Keras	Re-implemented the traffic sign classifier using both sequential and functional APIs provided by Keras
12	Transfer Learning	• The idea of re-purposing an existing trained network for a new similar task • Feature Extraction and Fine Tuning • Repurposed AlexNet (60 million parameters, 650000 neurons) for traffic sign classification • Bottleneck features • Benchmark VGGNet, GoogLenet, ResNet on cifar10 and traffic signs datasets • The importance of experimenting with network architectures (practise is ahead of theory at the moment)
13	Behavioral Cloning (Project Due 13 Feb)	• End-to-end learning (i.e. no modeling or rules) and its advantages (nvidia paper: nvidia_end2end-16.pdf, nvidia_e 2end2.pdf) • Simulator: https://techcrunch.com/2017/02/08/udacity-open-sources-its-self-driving-car-simulator-for-anyone-to-us • Report: https://github.com/cvilas/CarND/blob/master/P3-BehavioralCloning/submission/report/report.md • My architecture has 546619 parameters to tune. Others have achieved the same using SqueezeNet with 52 parameters: https://github.com/mez/carnd/blob/master/P3_behavioral_cloning/writeup_report.md • Batch-normalisation in improving learning ability of a network: batch_normalisation.pdf • Project Video:
Computer Vision		
14	Advanced Lane Finding (Project Due 27 Feb)	• Reversing lens distortions • Perspective transforms • Project Video:
15	Vehicle Detection and Tracking (Project Due 13 Mar)	• Supervised classification algorithms - naive bayes, support vector machines, decision trees • Entropy and information gain • Bias-variance dilemma • Histogram of Gradients: dalal-cvpr05.pdf • Creating feature vectors with HoG • Multiscale HoG windows • Heat maps to combine multiple detections • Others have used SSD classifier instead for faster detection (see https://arxiv.org/abs/1512.02325) • Project Video:

Term 2

(Apr 6, 2017 - Jul 17, 2017)

Lesson	Brief	Notes
1	Welcome	• Curriculum: https://medium.com/udacity/term-2-in-depth-on-udacitys-self-driving-car-curriculum-775130aae502 #.lra2w8s8m • Focus: Sensor fusion, localisation, control
Sensor Fusion		
2	Introduction to Sensors	• Types of sensors on a car: stereo camera, traffic signal detection camera, radars, lidars • Comparison between Lidar and Radar
3	Kalman Filters	• Multivariate Gaussians • Intuitive understanding of how KF works - interaction between observable and unobservable states. • Linear filter equations
4	C++ Checkpoint	• Series of simple exercises. No biggie!

5	Lidar and Radar Fision with KF in C++	• Constant velocity motion model for pedestrian tracking • Lidar sensor measurement model • Radar sensor measurement model • Derivation of Jacobian for radar measurement model • Extended Kalman Filter equations • Estimating filter performance using RMSE
6	EKF Project (Project Due 1 May)	• EKF-based sensor fusion to track objects using lidar and radar • https://github.com/cvilas/CarND/tree/master/P6-EKF
7	UKF	• Constant Turn Rate and Velocity Magnitude (CTRV) process model • Generating the sigma-points, applying the process model and computing mean/covariance of the predicted sigma-points. • Consistency check using normalised innovation squared • Process noise model tuning based on consistency check
8	UKF Project (Due 15 May)	• UKF-based sensor fusion for bicycle tracking • https://github.com/cvilas/CarND/tree/master/P7-UKF
Localisation		
9	Introduction to Localisation	
10	Localisation Overview	• Bayes Rule • Theorem of total probability

11	Markov Localisation	• Probabilistic representations
		    • Localisation (known map) vs SLAM
		    • Likelihood - Observation Model
		    • Prior - Motion Model
		• Markov assumptions and simplification of above probabilistic models
		• Derivation of recursive Baye's filter for localisation

12	Motion Models	• The bicycle model for a car
13	Particle Filters (PF)	• Particle filter vs Kalman and histogram filters • Conceptual description of steps and their implementation in Python • Algorithm for resampling with replacement
14	Implementation of a PF	• Implementation of various steps in pseudo-code: initialisation, prediction, data association, weights update, error calculation. • Introduction to kidnapped vehicle code
15	Kidnapped Vehicle Project (Due 5 Jun)	• Particle filter based localisation • https://github.com/cvilas/CarND/tree/master/P8-PF
Control		
16	PID Control	• The individual effects of P, I, and D terms • Twiddle algorithm for automatic control gain tuning
17	PID Controller (Project Due 19 Jun)	• PID control of steering angle in a simulated car • https://github.com/cvilas/CarND/tree/master/P9-PID
18	Vehicle Models	• Difference between kinematic and dynamic models • Review kinematic model • Polynomial fitting for smooth desired vehicle trajectories • Cross track and orientation error computations • Brief intro to dynamic models. No details discussed.
19	Model Predictive Control	• Development of a cost function • Concept of prediction horizon • MPC algorithm development and optimisation (Ipopt and cppad) • Incorporating actuation latency into the control algorithm

20	Model Predictive Control Project (Project Due 3 Jul)	• Model predictive control of a simulated vehicle around a track, with actuation latency • https://github.com/cvilas/CarND/tree/master/P10-MPC

Term 3

(July 28, 2017 - Nov 6, 2017)

Lesson	Brief	Notes
1	Welcome	Curriculum: https://medium.com/udacity/term-3-in-depth-on-udacitys-self-driving-car-curriculum-15d03e45d7ea
2	Search	• Basic search • A* search on grid map • Dynamic programming
3	Prediction	• Multi-modal nature of driver behaviour prediction • Model-based vs data-driven approaches • Trajectory clustering and prototype trajectories • Frenet coordinates • Different types of process models for prediction and their comparison. • Hybrid approach to prediction (process model + ML classifier) • Naive Bayes classifier example

4	Behaviour Planning	• Finite State Machines as an approach to implement behaviour planner
		    • Strengths and weaknesses of FSM approach
		    • Example: highway driving
		    • State transition functions
		• Cost functions and difficulties in designing them properly
		• Scheduling compute time for various modules (behaviour planning, prediction, trajectory planning, localisation, sensor fusion)

5	Trajectory Generation	• Definition of motion planning (MP) problem • MP algorithm properties - completeness and optimality • Classes of MP algorithms - combinatorial, potential fields, optimal control, sampling-based • Focus on sampling-based methods - difference between discrete and probabilisitic methods • Hybrid A* implementation • Paper review: Junior: The Stanford entry in the Urban Challenge
6	Project: Path Planning (Due Nov 27)	
7	Intro to electives	
8	**Elective: Advanced Deep Learning**	
9	Fully Convolutional Networks	
10	Scene Understanding	
11	Inference Performance	
12	Project: Semantic Segmentation (Due Dec 25)	
13	**Elective: Functional Safety**	
14	Introduction to Functional Safety	
15	Functional Safety: Safety Plan	

16	Functional Safety: Hazard Analysis and Risk Assessment	
17	Functional Safety: Functional Safety Concept	
18	Functional Safety: Technical Safety Concept	
19	Functional Safety at the Software and Hardware Levels	
20	Elective Project: Functional Safety (Due: Dec 25)	
21	**Autonomous Vehicle Architecture**	
22	Introduction to ROS	
23	Packages and Catkin Workspaces	
24	Writing ROS Nodes	
25	Project: Systems Integration (Due Jan 22,'18)	

References

- Vilas' GitHub repo: https://github.com/cvilas/CarND
- A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles, https://arxiv.org/pdf/1604.07446v1.pdf
- Visualizing and Understanding Convolutional Networks: https://www.youtube.com/watch?v=ghEmQSxT6tw
- How the future of autonomous cars will unhold. 16 questions: https://vimeo.com/198256576
- Dropout: A Simple Way to Prevent Neural Networks from Overfitting: JMLRdropout.pdf
- Gradient-based learning applied to document recognition (Lenet): lecun-98.pdf
- Traffic sign recognition with multi-scale convolutional networks: sermanet-ijcnn-11.pdf
- ImageNet classification with deep convolutional neural networks (AlexNet): alexnet-12.pdf
- Going deeper with convolutions (GoogLenet): GoogLeNet-15.pdf
- Deep residential learning for image recognition (Microsoft Resnet): resnet-15.pdf
- Very deep convolutional networks for large-scale image recognition (vggnet): vggnet-15.pdf
- End to end learning for self-driving cars (Nvidia). Original paper: nvidia_end2end-16.pdf. Paper explaining how it works: nvidia_end2end2.pdf
- A summary of key recent deep learning architectures: https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html
- SqueezeNet: AlexNet level accuracy with 50x fewer parameters and < 0.5 MB model size: squeezenet-17.pdf
- Batch Normalization: Accelerating deep network training by reducing internal covariate shift: batch_normalisation.pdf
- Histograms of Oriented Gradients for Human Detection: dalal-cvpr05.pdf
- Nvidia Drive PX2 and what it does: https://www.youtube.com/watch?v=URmxzxYlmtg
- Chris Urmson (Google), How a driverless car sees the road: https://www.ted.com/talks/chris_urmson_how_a_driverless_car_sees_the_road
- A comparative study of multiple-model algorithms for maneuvering target tracking: a-comparative-study-of-multiple-model-algorithms-for-maneuvering-target-tracking.pdf
- Junior: The Stanford entry in the Urban Challenge: junior_stanford-2008.pdf