Project proposal TDT4265:

Autonomous self-driving car (task 3)

Goal of project:

The main goal of the project will be to implement and train a neural net architecture capable of learning how to control a car's steering angle on the basis of raw image input. The car will during testing be able to drive satisfactorily along some previously unseen test course, without suffering a crash or driving outside of the course. If succeeded with this, if possible, we will try to expand the problem to control of speed aswell and optimizing lap times, still without crashing or driving outside of the course.

Relevant literature:

- https://arxiv.org/pdf/1704.07911.pdf
- https://arxiv.org/pdf/1604.07316.pdf
- Chen et al: *End-to-end learning for lane keeping of self-driving cars*
- https://arxiv.org/pdf/1710.03804.pdf
- https://selfdrivingcars.mit.edu/
- Zitzewitz: *Survey of neural networks in autonomous driving*
- http://cs231n.stanford.edu/reports/2017/pdfs/626.pdf
- https://arxiv.org/pdf/1608.01230.pdf + https://github.com/commaai/research (possibly relevant datasets)
- https://devblogs.nvidia.com/explaining-deep-learning-self-driving-car/
- https://devblogs.nvidia.com/deep-learning-self-driving-cars/

Model architecture:

We will most likely use a deep CNN architecture. We will also explore the possibilities of using the CNN as a visual feature extractor and combine it with C-LSTM architechture. These are the architectures used in most of the relevant literature we've looked at.

Code:

We will most likely utilize the starter code provided as a part of the project proposal that this task is linked too, found here:

https://drive.google.com/file/d/1hKVc4METKj2aQy4yC3xnP8Dwc4zEd-Cn/view?usp=sharing

We are however open to the idea to change our framework if we discover an architecture that is more easily implemented in another framework/language. The handed out code seems to have a straightforward interface for testing the trained model as long as it is an exported Keras-model, another incentive to use Keras over other frameworks. If we change our framework we have to do some research on the Udacity simulator to see if the new framework is easily supported.

Generating dataset, model training and testing:

Since implementing our solution on an actual car is both expensive and somewhat dangerous we do as recommended and use the Udacity simulator for dataset generation, training and testing of our network.

To generate the dataset for training we have to drive around in the Udacity simulator we record our driving session. These images will then be the basis that our network will train on. Our network will only drive as good as the dataset it trains on, meaning that care must be taken to provide it with a good dataset. Most likely we will have to spend quite some time generating the dataset.

When we have a trained model we can export it from Keras and provide it to the Udacity simulator by running it in autonomous mode.