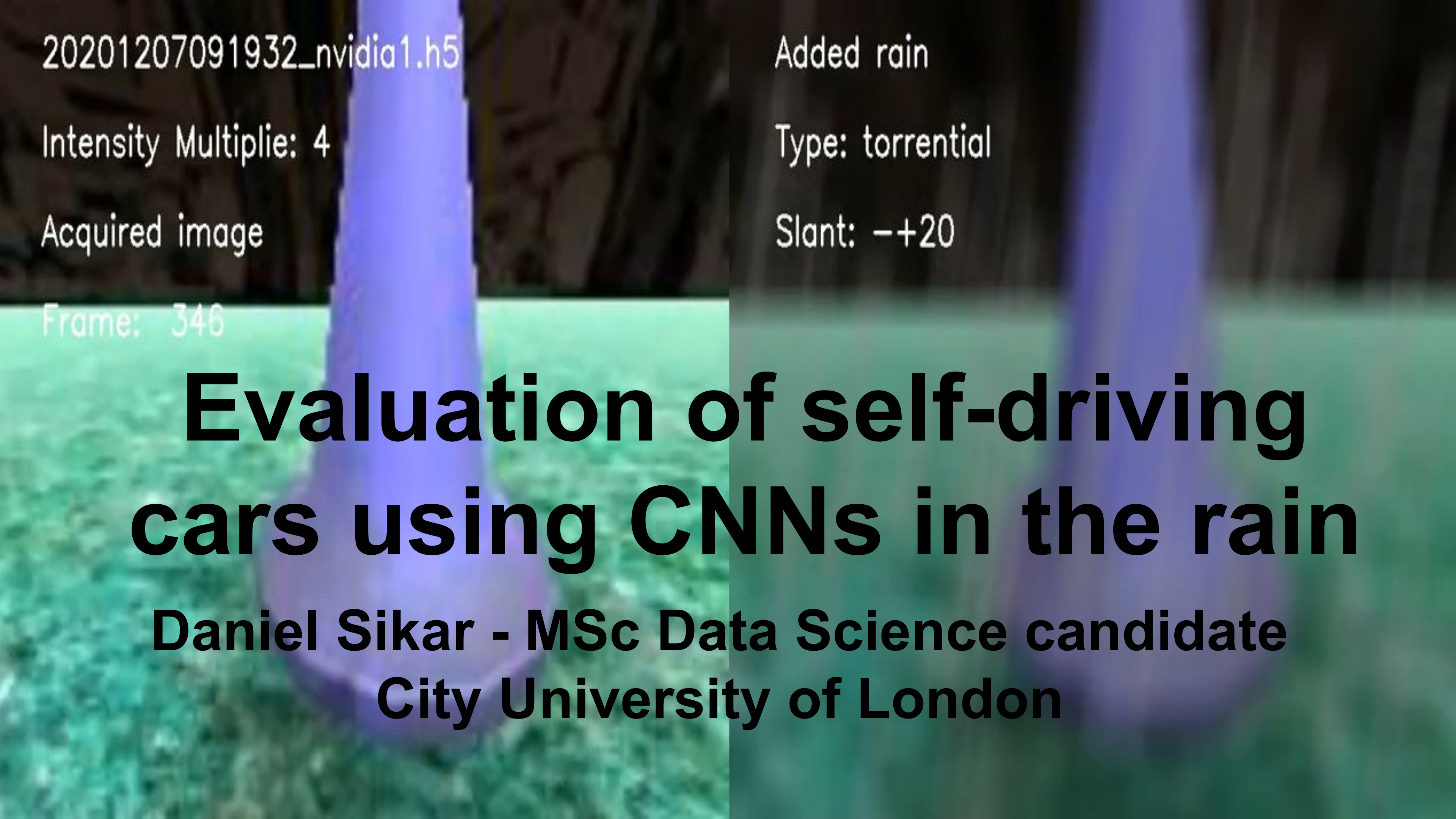


Evaluation of self-driving cars using CNNs in the rain

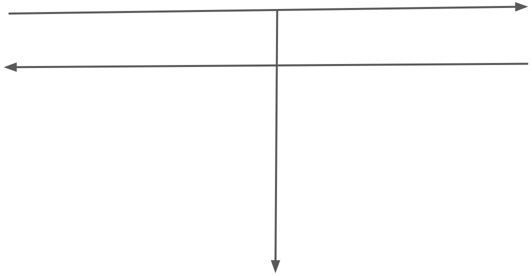
Daniel Sikar - MSc Data Science candidate
City University of London

Prediction Engine and Simulator Setup

Prediction Engine

https://youtu.be/mDjtnnVZdic

TCP Network



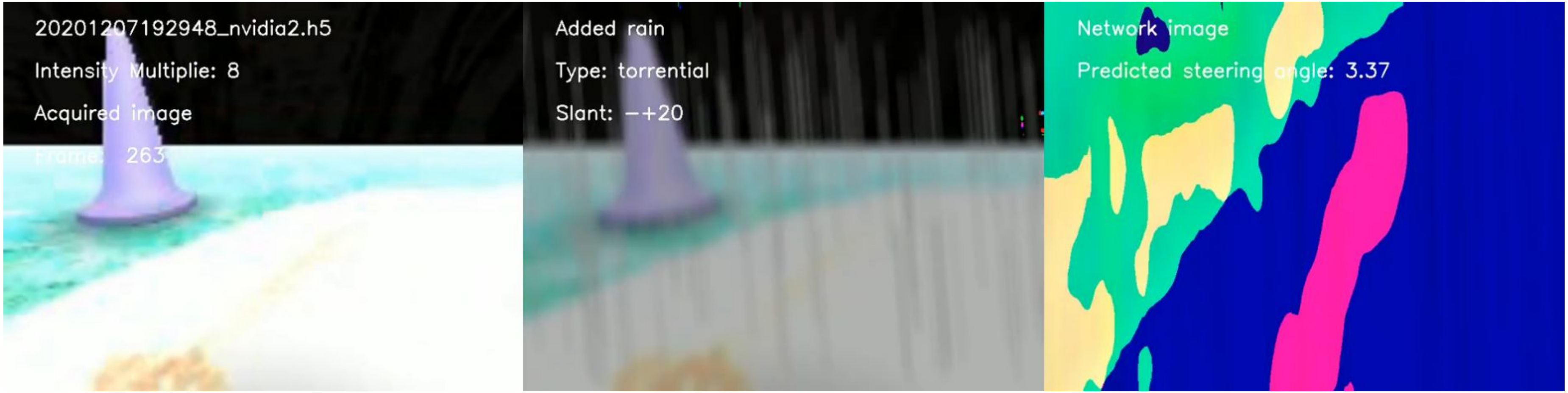
tcpflow network monitor

Unity SDSandbox Simulator

{"msg_type": "control",
"steering": "-0.0020742654",
"throttle":
"0.04071302339434624", (...)

Evaluation of self-driving cars using CNNs in the rain

Prediction engine self-driving in the rain



https://youtu.be/W1eRN5DWPXw

$ python **predict_client.py**
--model=../trained_models/nvidia2/20201207192948_nvidia2.h5
--modelname=nvidia2 --record=True --rain='torrential' --slant=20

https://github.com/dsikar/sdsandbox/blob/master/src/utils/**RecordVideo.py**

Evaluation of self-driving cars using CNNs in the rain

Motivation 1

- CNNs trained end to end are SOTA for Computer Vision, used by AVs (Autonomous Vehicles)
- Investigate safety of AVs in the rain
- Comparing different deep convolutional network architectures
- Trained with public AV datasets

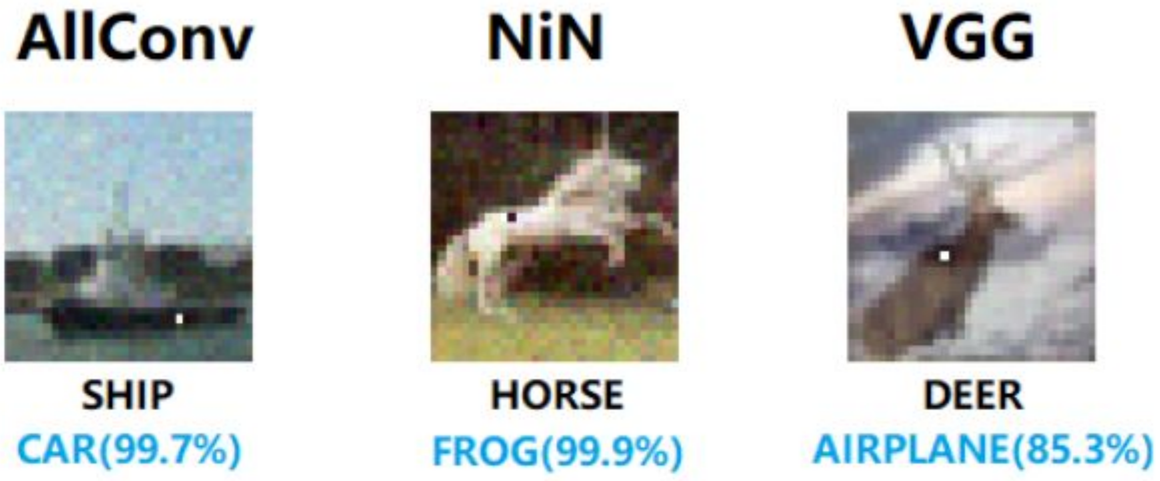
DAVE, LeCun et al. 2004

End to End Learning for Self-Driving Cars, Bojarski et al. 2016

Evaluation of self-driving cars using CNNs in the rain

Motivation 2

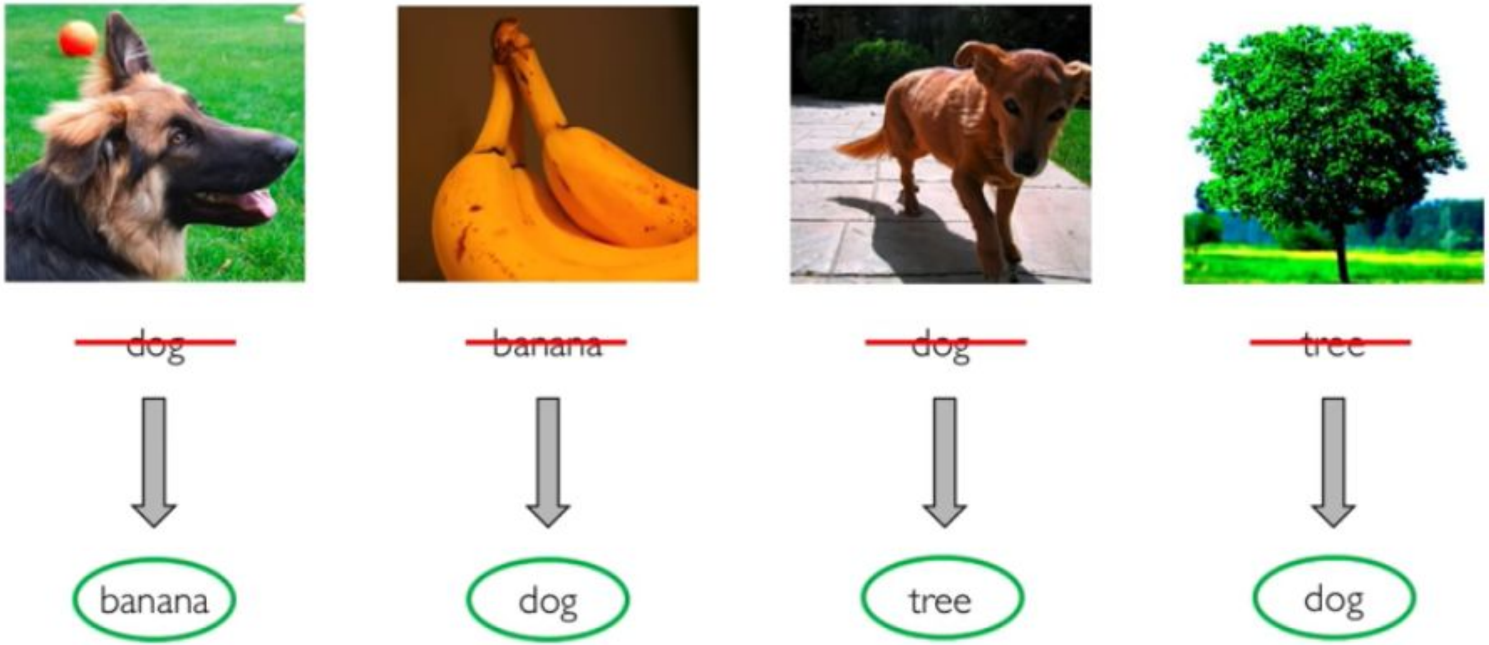
- CNNs have been shown to lack robustness - Su et al. 2019



Evaluation of self-driving cars using CNNs in the rain

Motivation 3

- CNNs have been shown to overfit - Zhang et al. 2017

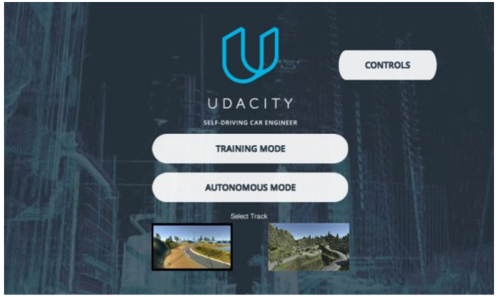


Evaluation of self-driving cars using CNNs in the rain

Self-driving testing environments considered

NVIDIA DRIVE AGX Developer Kit

Udacity Carla Simulator



Udacity Legacy Simulator

Donkey Car

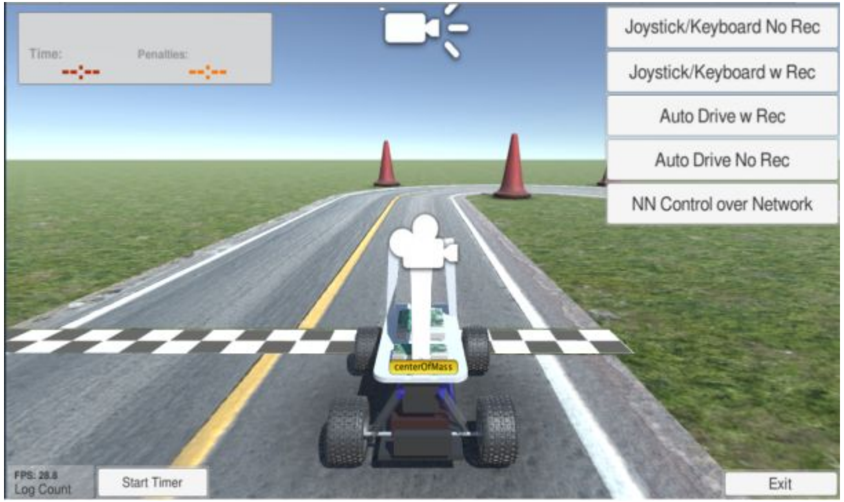
DIY Robocars maker/racers

SDSandbox Simulator

SDSandbox Simulator Setup 1

FIGURE B.2: Left to right: Unity Hub, SDSandbox home screen and simulation ready to run

SDSandbox Simulator Setup 1

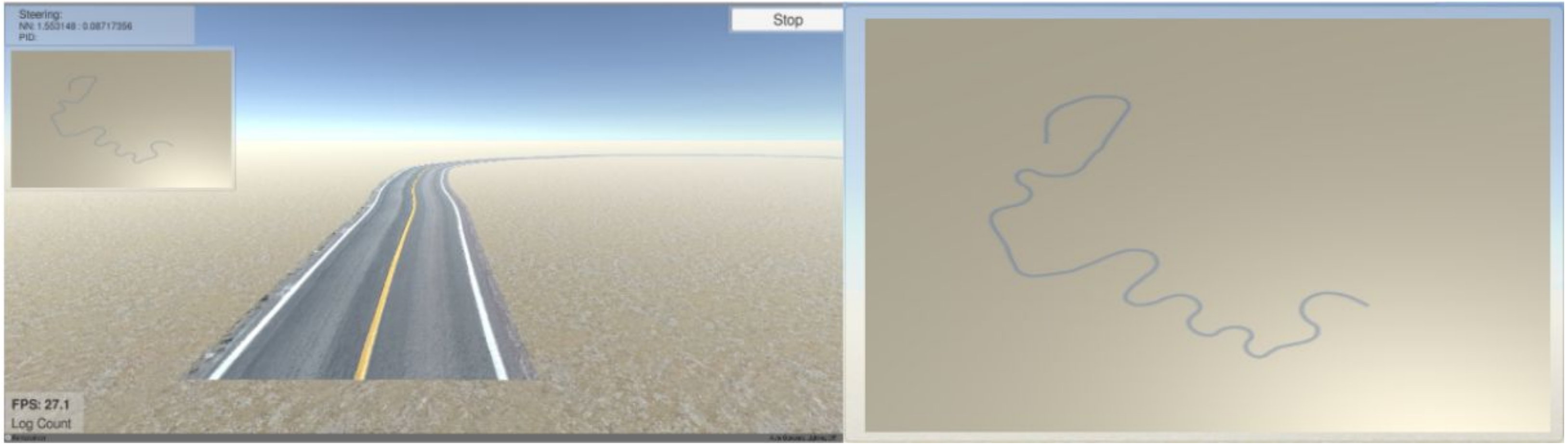


Unity SDSandbox Simulator

small_looping_circuit track

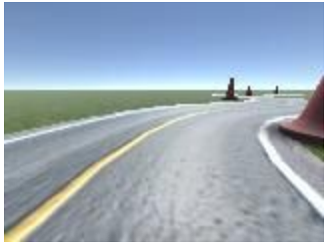
Evaluation of self-driving cars using CNNs in the rain

SDSandbox Simulator Setup 2



“Generated Road” simulation running in “NN Control over Network” mode

Unity SDSandbox Simulator Labelled Dataset 1



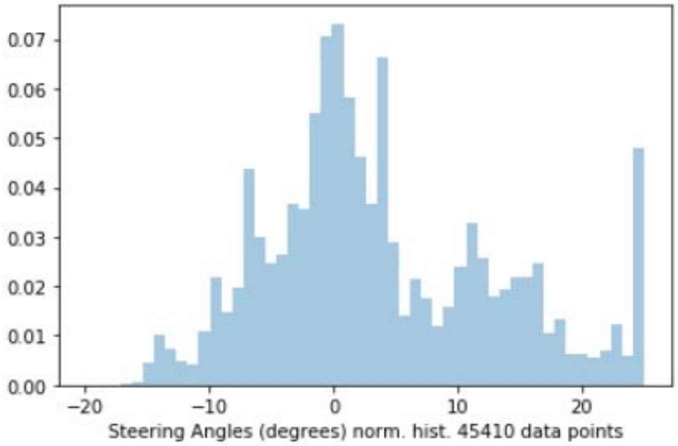
logs_Wed_Nov_25_23_39_22_2020/**400_cam-image_array_.jpg**
logs_Wed_Nov_25_23_39_22_2020/**record_400.json**
{"cam/image_array":"400_cam-image_array_.jpg","user/throttle":0.0,**"user/angle":-0.0017995497910305858**, (...)
}

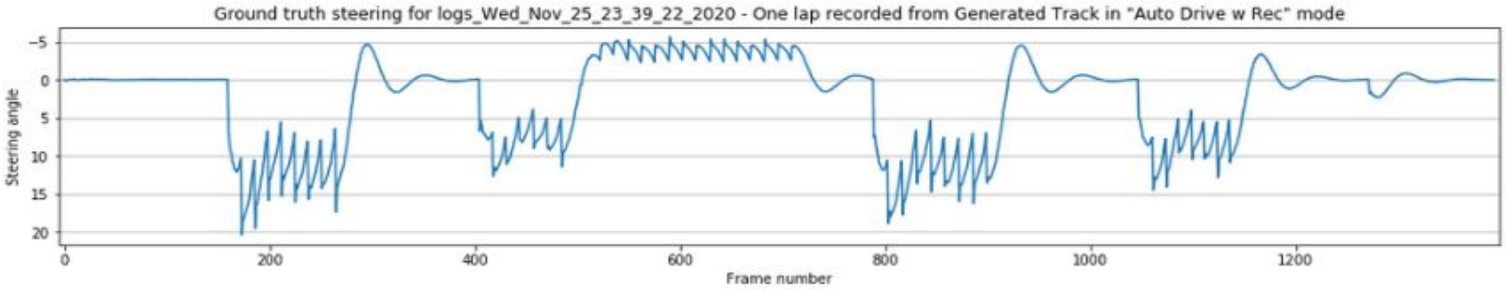
logs_Wed_Nov_25_23_39_22_2020/**1000_cam-image_array_.jpg**
logs_Wed_Nov_25_23_39_22_2020/**record_1000.json**
{"cam/image_array":"1000_cam-image_array_.jpg","user/throttle":0.30000001192092898,**"user/angle":-0.016777465119957925**, (...)
}

Unity SDSandbox Simulator Labelled Dataset 2

```
$ . count.sh
871403 files in directory .
25562 files in directory ./roboRacingLeague
25795 files in directory ./log_sample
82245 files in directory ./warehouse
44496 files in directory ./quarantine
90842 files in directory ./smallLoop
516997 files in directory ./genRoad
16570 files in directory ./genTrack
68886 files in directory ./smallLoopingCourse
```

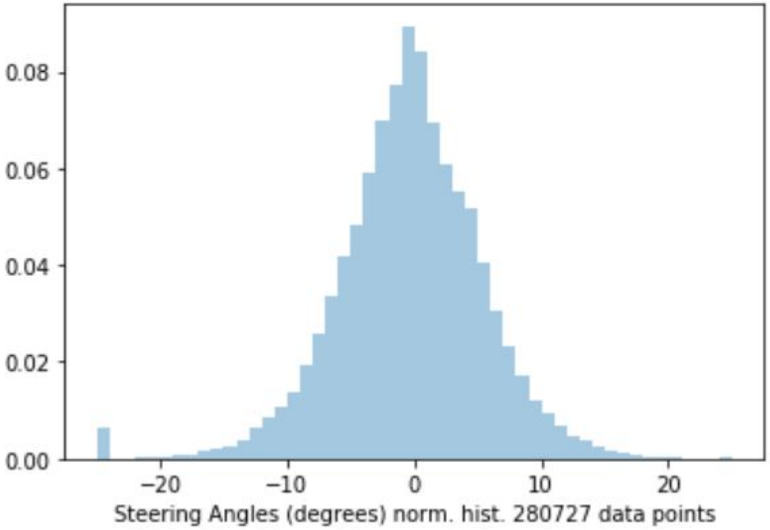
Unity SDSandbox Simulator Labelled Dataset 3





Evaluation of self-driving cars using CNNs in the rain

Unity SDSandbox Simulator Labelled Dataset 4



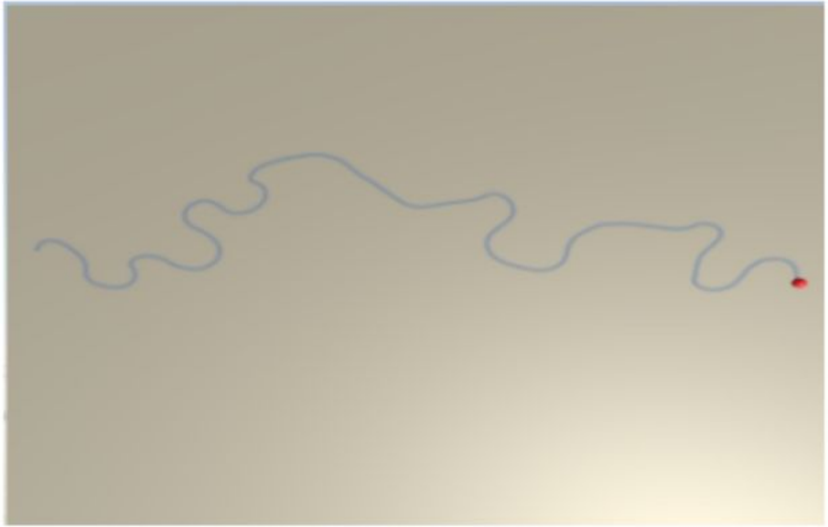
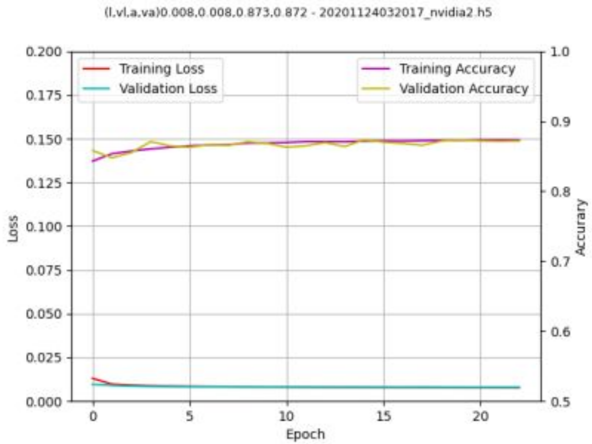
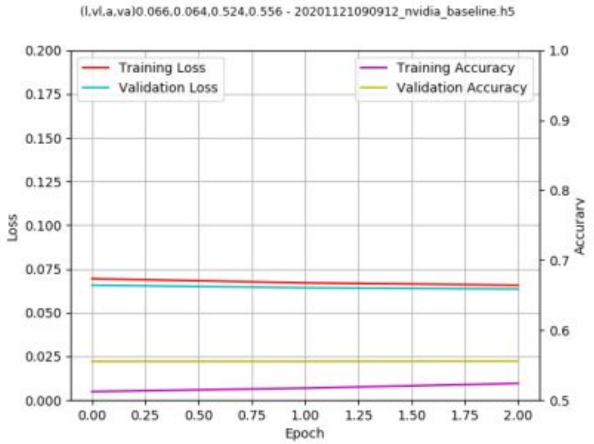
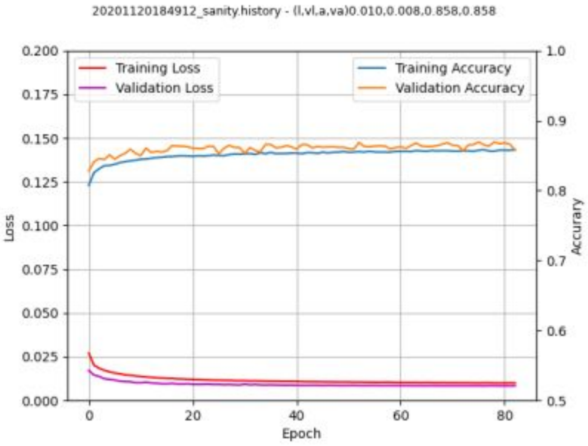


FIGURE 4.5: Normalized histogram of Unity 3D SDSandbox generated road, steering angles for 280727 image frames. A sample randomly generated road is shown on the right. Outliers in negative range are due to oversteering when vehicle reached the end of the road and simulator was left recording.

Training environments



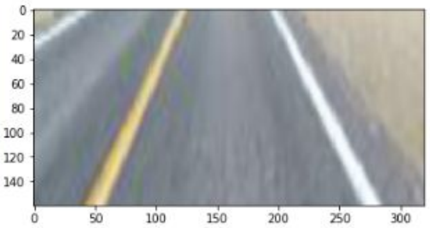


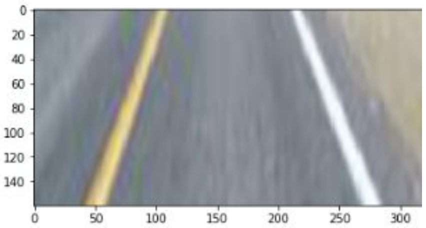


- Intel DevCloud - shared cloud
- Camber - City shared cloud
- Dell 12-core Xeon CPU 32MB RAM - local machine, also testing environment

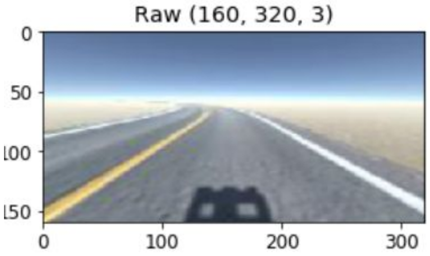
Image Pre-Processing

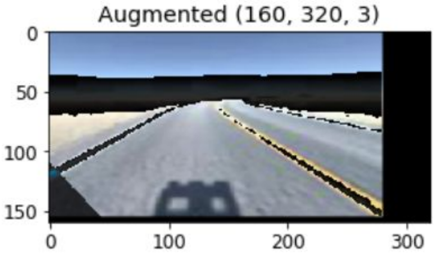
nvidia1 crop

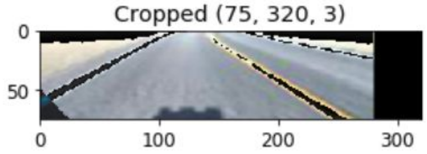


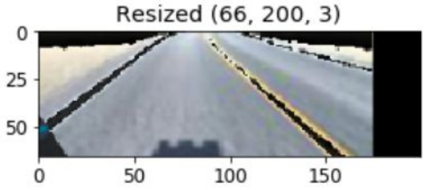


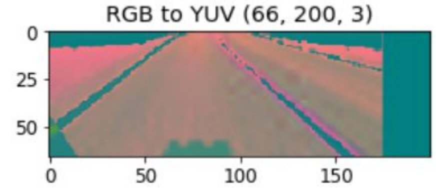
nvidia2 crop











Reference image
(160w x 120h)

nvidia2 image pre-processing pipeline

Evaluation of self-driving cars using CNNs in the rain

Image pre-processing at inference time

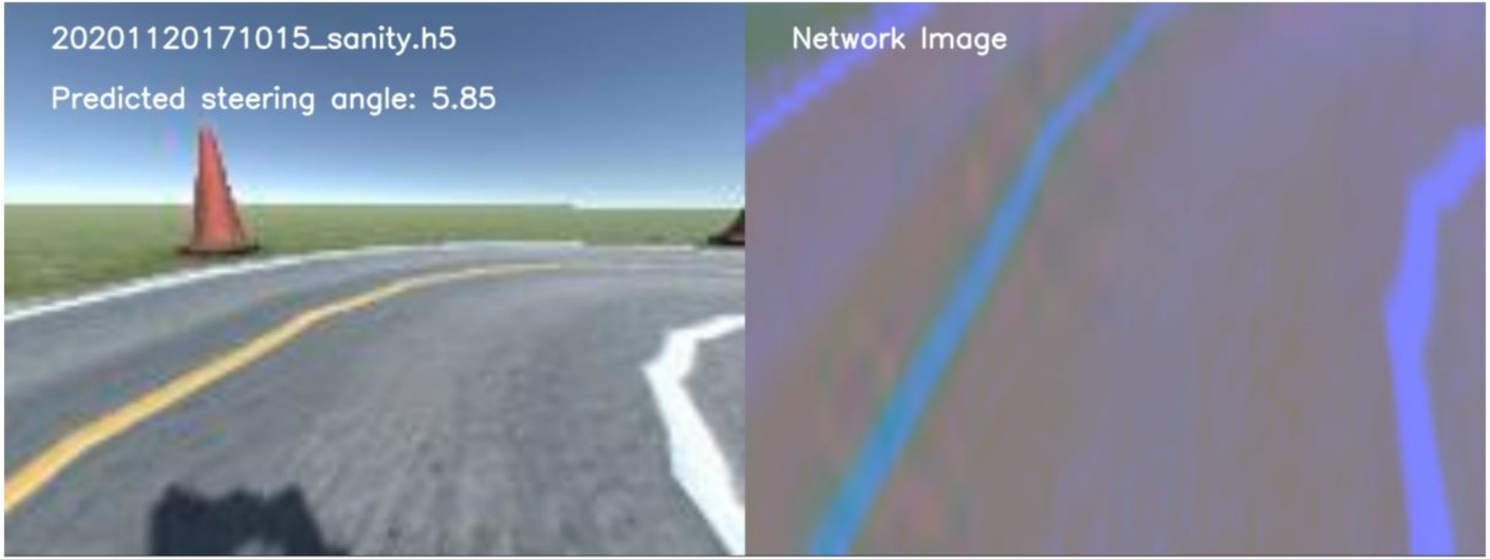
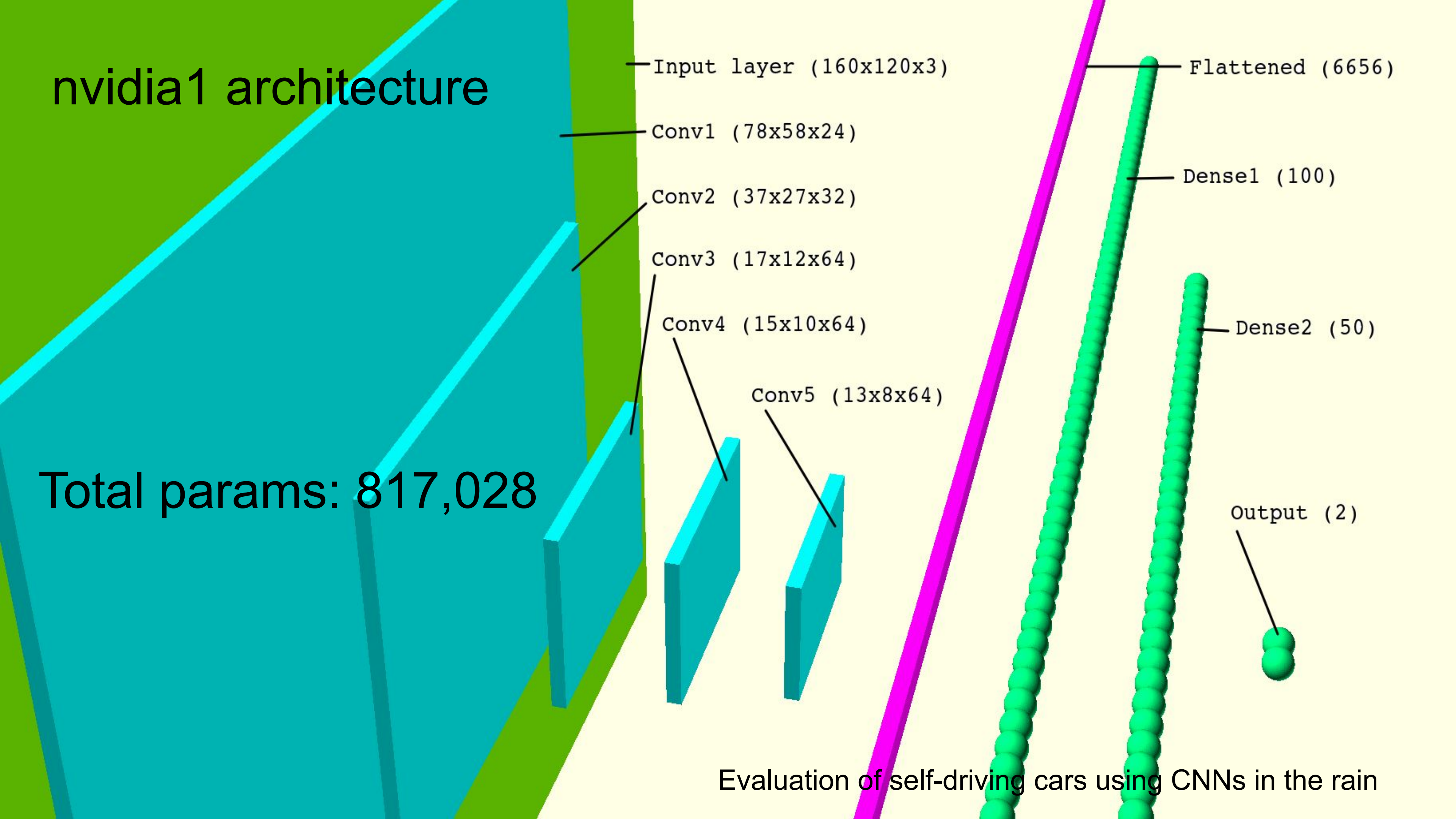
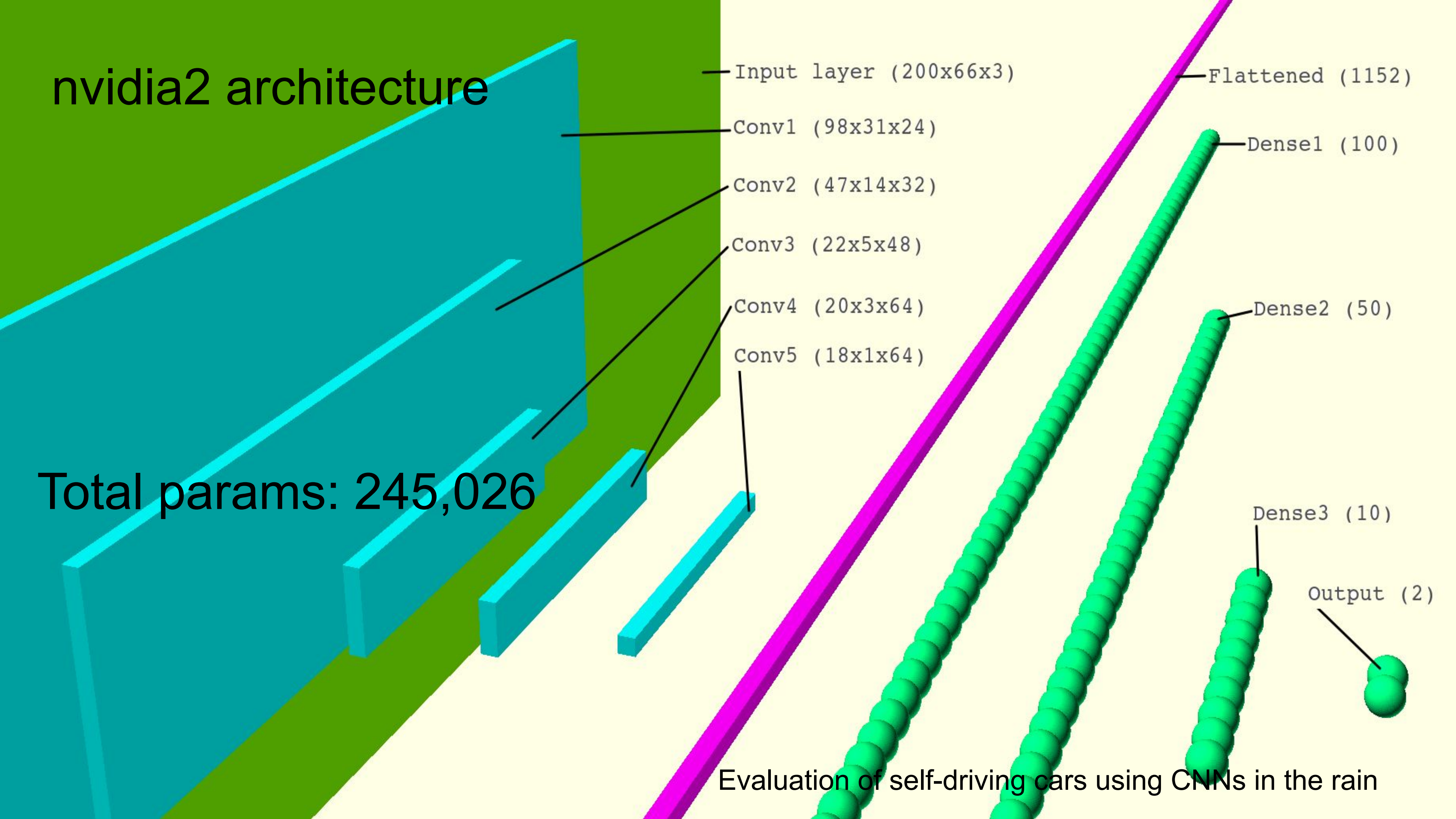


FIGURE 4.8: Video still from run 41 video https://youtu.be/LEmZJJzJkEE showing simulator image as sent over TCP network on the left with added CNN (20201123162643_ sanity.h5 model) steering angle prediction and processed image (as presented to CNN) on the right

Evaluation of self-driving cars using CNNs in the rain

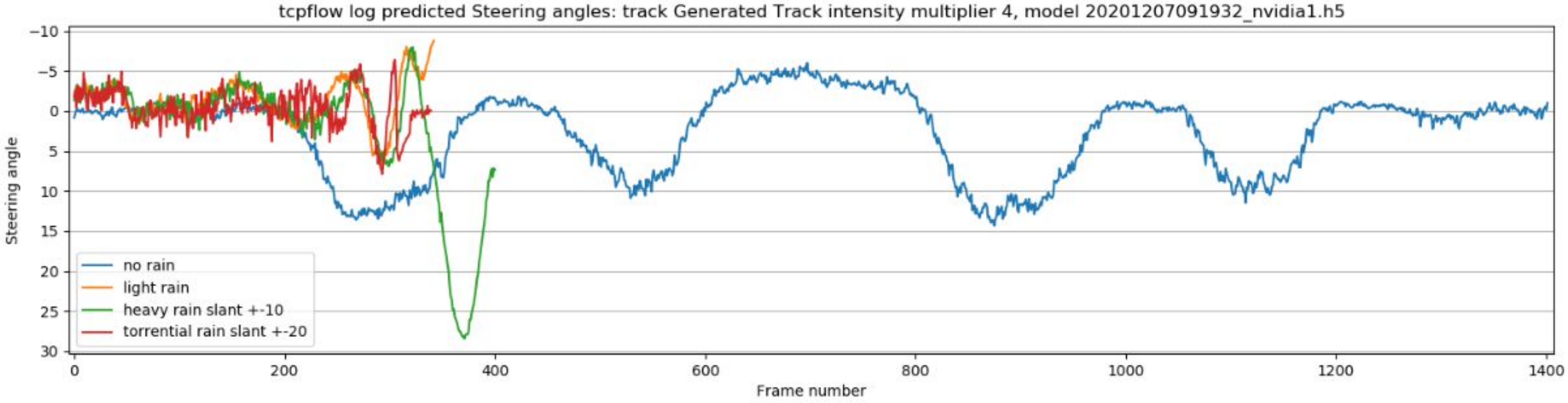

nvidia1 architecture
Input layer (160x120x3)
Conv1 (78x58x24)
Conv2 (37x27x32)
Conv3 (17x12x64)
Conv4 (15x10x64)
Conv5 (13x8x64)
Flattened (6656)
Dense1 (100)
Dense2 (50)
Output (2)
Total params: 817,028

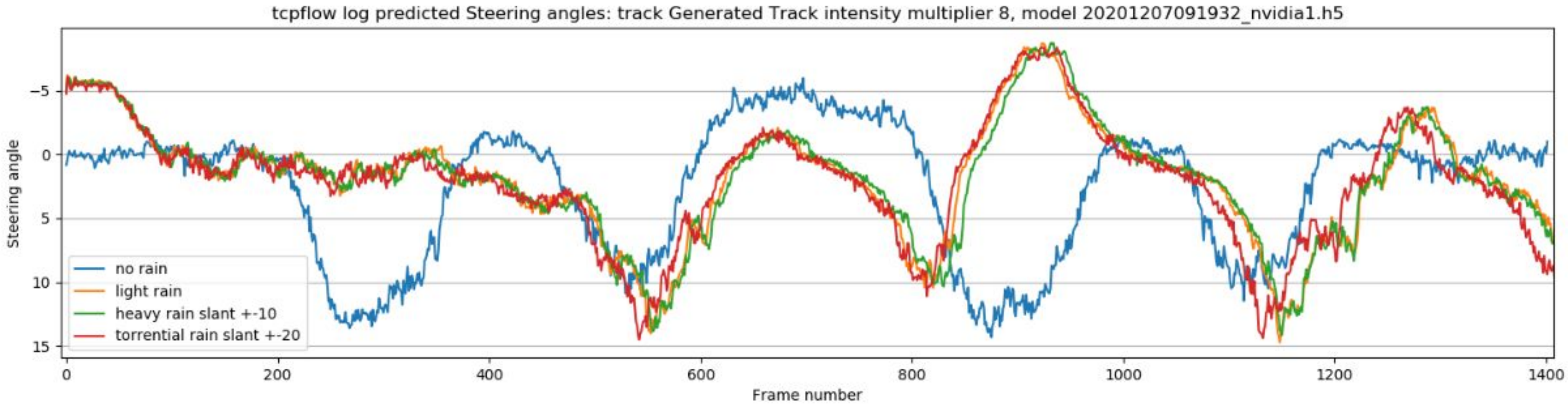
nvidia2 architecture



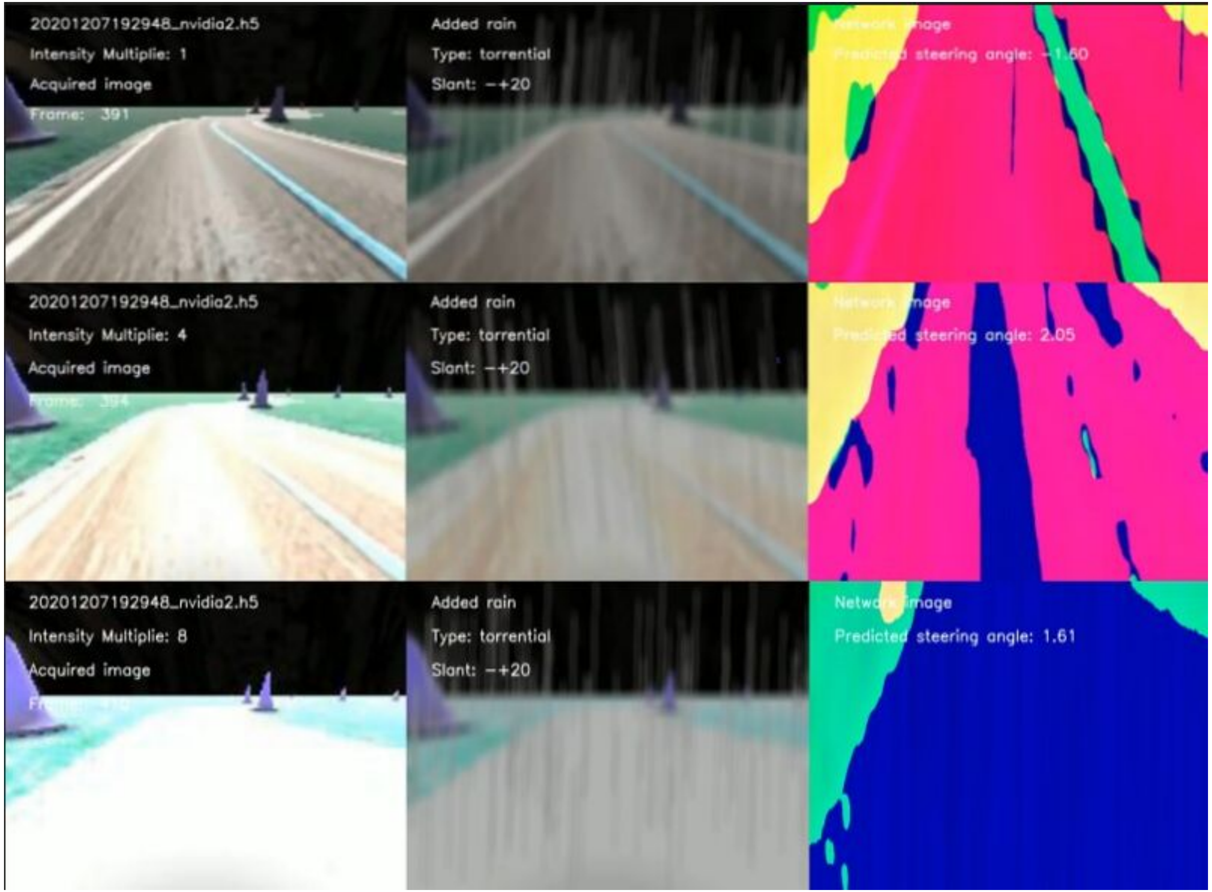
Total params: 245,026

Qualitative Analysis 1, nvidia1 model





Qualitative Analysis 2, nvidia1 model



Quantitative Analysis

Goodness-of-steer results - Generated Track and Generated Road SDSandbox logs					
Generated Track log: logs_ Wed_ Nov_ 25_ 23_ 39_ 22_ 2020 (1394 images)					
ID	Keras model file name	Model	Rain Type	Slant	g_s
1	20201207192948_ nvidia2.h5	nvidia2 :)		0	1.68 *
2	20201207192948_ nvidia2.h5	nvidia2	light	0	2.12
3	20201207192948_ nvidia2.h5	nvidia2	heavy	10	2.17
4	20201207192948_ nvidia2.h5	nvidia2	torrential	20	2.30
5	20201207091932_ nvidia1.h5	nvidia1		0	1.82 *
6	20201207091932_ nvidia1.h5	nvidia1	light	0	2.11
7	20201207091932_ nvidia1.h5	nvidia1	heavy	10	2.13
8	20201207091932_ nvidia1.h5	nvidia1	torrential	20	2.28

(...)

$$g_s(p,g) = \frac{\sum_i^N |p(i) - g(i)|}{N} \times n_c \qquad G_s(p,g) = \left(\frac{\sum_i^N |p(i) - g(i)|}{N} \times n_c + C_c \right)^{-1}$$

Conclusions

- Geometry matters - input size, cropping, feature maps
- Image pre-processing, augmentation, RGB to YUV matters
- Same image pre-processing pipeline (minus flipping, random shadows and shifting) must be applied to image at inference time
- Small datasets work
- Outliers ruin training
- Models train quickly once cropping and geometry are right
- Models generalise! Trained on one circuit, drives on the other, trained in dry weather, drives in wet weather
-

Future work

- Try alternative network architectures (Inception, VGGNet, Alexnet, ResNet, etc)
- Try turning regression into classification, if binned quantised value is acceptable minimal steering value
- Better understanding of camera characteristics
- Explore possibilities with “sliver” feature-map geometries, horizontal and vertical
- Given small datasets, short training times and small model sizes, explore ensembles
-

Thanks for watching!

- Questions?
- Feedback?

daniel.sikar@city.ac.uk

Source code: https://github.com/dsikar/sdsandbox/tree/master/src

SDSandbox: https://github.com/tawnkramer/sdsandbox/tree/master/sdsim