1 Appendix

A single color image (1920*1080*3*255) contains more than 1.5 billions of states. A color frame (84*84*3*8) trained by DQN contains more 169k states, still very large! A single colored image converted into a 50 class segmentation map and scale down to 80 by 80 pixels contains $6400N$ (N here is the class numbers on the segment frame and N is normally less than 10) states. By this way, the segmentation not only reduced the dimensionality and compressed dataset, they also isolated those trivial details (that might be built into high level decision boundaries) from higher level learners.

With photo-realistic environment and excellent physics engines, we are able to learn and transfer to real-robots. Our synthetic dataset contains different day-time, weather condition and camera angles for collecting maximum converge of objects in the map with as diverse textures as possible. To increase the robustness, we also introduces random noise to the rgb images. Unlike UAVs which have higher oder degree of freedoms and the viewpoints count much, we only capture the viewpoints with shifted yaws. However, due to the resetting mechanism in Unreal Engine always check mesh's integration, and the map was not always flat, the car will roll over often time. We took advantage of this fact by respawn the car lower than the map to create the random rolling angle for another augumention trick which may not be able to produce in other environments.

In the data augmentation phase, we prepared 7 different weather conditions and applied random permutations (due to AirSim's model limiation, we can't add accumulated land effect like snow and dust on land surfaces).

The random combination of light and weather effects on RGB images produced large dataset for us to train the VGG16 based U-Net segmentation module. Similar to Sadeghi's method of randomizing the texture in fixed geometry environment and learn the geometry through the texture-randomized dataset which can scale into the real-world images, we can take the segmentation network off and retrain only the segmentation parts when the environments are have huge difference or geometrical layouts.

We introduced the batched sliding window strategy rather than randomly assemble samples from a stack of memories to train the recurrent model, corresponding number of LSTM units are deployed, however in the same window scope, the input samples are closely correlated which is hard to generate enough gradient. Thus, we stack of 5 windows into the replay memory and truncate through time then propagate back to the corresponding sliding window after each batch. This training operation suffers from over-fitting in the early stage. So we applied concrete-dropout along with L2 regularization. Autoregressive model may be applied to solve this problem in future extensions.

Note that in Fig 4 the high attention over the road name plate on test sequence [1] and the electric pole on test sequence [3]. All the unrealated depth information was successfully filtered out and guided the agent to focus most important parts and eventually consolidated inside LSTM cells as landmarks.

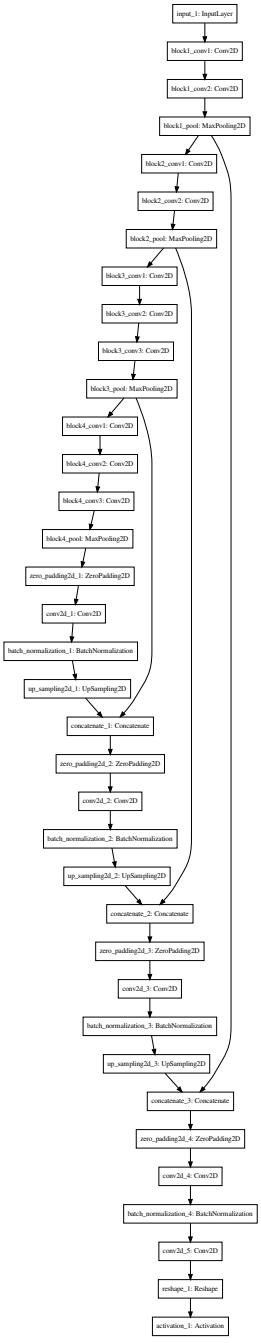

Figure 1: U-Net Model.

Figure 2: AirSim Neighborhood Training environment. This diverse map offered us essential elements for normal daily driving.

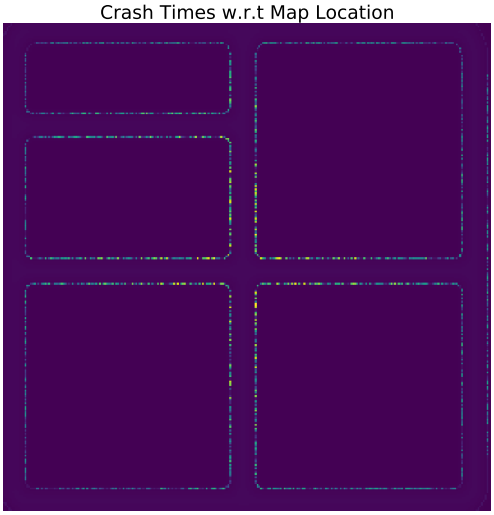

Figure 3: crash location ploted on the map (averaged over 7 runs). Note that we always initialize the vehicle in the middle of map with different orientations

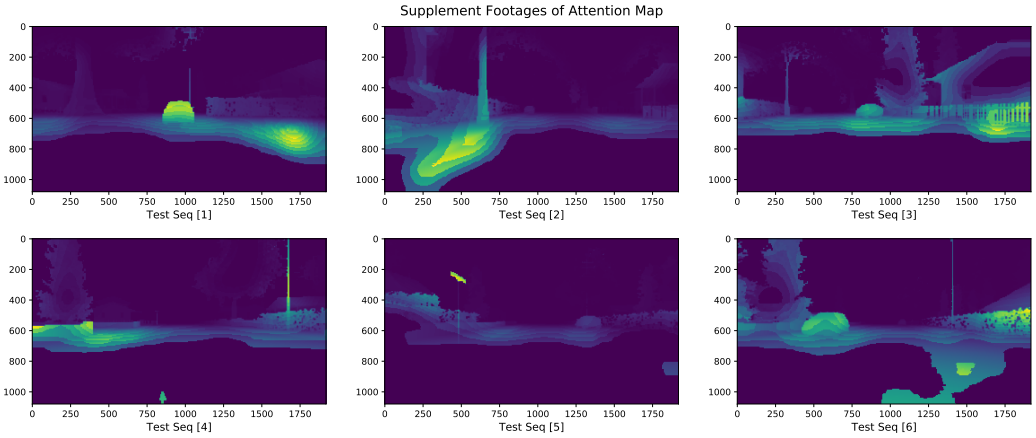

Figure 4: Additional attention maps.