Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Master's Thesis

Multi-View Temporal Fusion in Semantic Segmentation

Manoj Kolpe Lingappa

Submitted to Hochschule Bonn-Rhein-Sieg,
Department of Computer Science
in partial fullfilment of the requirements for the degree
of Master of Science in Autonomous Systems

Supervised by

Prof. Dr. Nico Hochgeschwender
Prof. Dr. Sebastian Houben
M.Sc. Deebul Sivarajan Nair

Sprint 15.05.2022
July 17, 2022

1

Sprint review

1.1 Goals for 15.07.2022 sprint

Below are the goals set for the period of 15.06.2022 to 15.07.2022

Q1 Find one dataset

Q2 Focus on the first research question

Q3 Find a list of datasets for semantic segmentation with the camera dataset (Look for Synthetic dataset)

Q4 Evaluation of the model with and without the Gaussian process

Q5 Create a table of results

1.1.1 Q1 and Q3

During the literature review three datasets are shortlisted and described below.

a. Scannet Dataset

b. Virtual Kitti 2 dataset

c. VIODE dataset

---

a. Scannet Dataset

Scannet is a RGB-D based video dataset with 2.5 million views obtained from 1500 scans along with camera poses. Each image has a ground truth of instance level semantic segmentation, surface reconstruction. Entire dataset is 1.3TB is size. Either specific scan can be downloaded or entire data can be downloaded at a time. The data need to be preprocessed to get the color, label and pose data. Procedure to preprocess the data is presented in the following link: Click This. Currently 86 scenes are downloaded and preprocessed. The color and label images are in ".png" format. The labels present in the are listed below

Labels of classes in the images - 1 wall, 2 floor, 3 cabinet, 4 bed, 5 chair, 6 sofa, 7 table, 8 door, 9 window, 10 bookshelf 11 picture, 12 counter, 13 blinds, 14 desk, 15 shelves, 16 curtain, 17 dresser, 18 pillow, 19 mirror, 20 floor mat, 21 clothes, 22 ceiling, 23 books 24 refridgerator, 25 television, 26 paper, 27 towel, 28 shower curtain, 29 box, 30 whiteboard, 31 person, 32 nightstand, 33 toilet, 34 sink, 35 lamp 36 bathtub, 37 bag, 38 otherstructure, 39 otherfurniture, 40 otherprop

Table 1.1: Camera Pose

-0.869565	0.231948	-0.435955	2.750575
0.492522	0.471291	-0.731647	3.154689
0.035758	-0.850932	-0.524058	1.290553
0.000000	0.000000	0.000000	1.000000

(a) RGB image

(b) Segmentation mask

Figure 1.1: 2 RGB image and Segmentation mask

b. Virtual KITTI 2 Dataset

Virtual KITTI is the first synthetic datasets created to autonomous driving application. The dataset represent the real world environment and created with the Unity game engine.

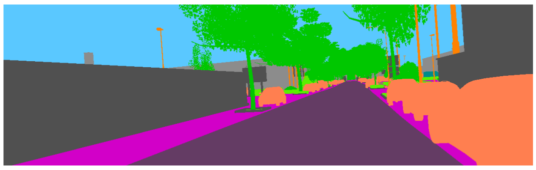
- Color images are in RGB format with 8-bit representation per channel
- Segmentation images are encoded in 8-bit per channel
- The pose of the camera is represented with rotation and translation vector with x-axis pointing on the right hand side, y-axis on down and z- axis is going forward
- The images are captured at different position of camera and different environment. They are 15-deg-left, 15-deg-right, 30-deg-left, 30-deg-right, clone, fog, morning, overcast, rain, sunset.
- The dataset has 15 classes in it and they are Terrain, Sky, Tree, Vegetation, Building, Road, GuardRail, TrafficSign, TrafficLight, Pole, Misc, Truck, Car, Van, Undefined

Camera Pose

rotation_world_space_x rotation_world_space_z camera_space_X camera_space_Y camera_space_Z rotation_camera_space_y rotation_camera_space_x rotation_camera_space_z

0 0 0 -1.994751 1.85 1.50992 4.930564 6.371316 -111 -5.044228 0.2694305 0 0 4.462387 1.322803 5.465487 -1.31005 0.02017088 -0.06939133

(a) RGB image



(b) Segmentation mask

Figure 1.2: 2 RGB image and Segmentation mask

c. VIODE Dataset

A dataset generated from simulated challenging environment with the help of UAV data

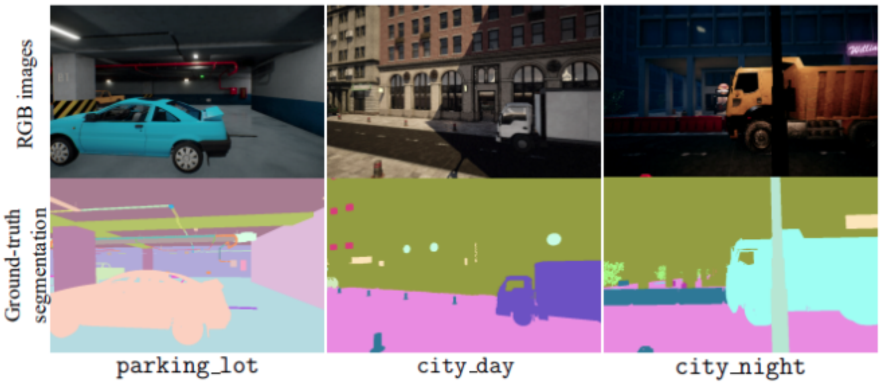


Figure 1.3: VIODE RGB and ground truth

1.1.2 Q2 and Q4

The research question deals with the study of temporal fusion on the semantic segmentation. Unet is a convolutional neural network based deep learning model developed for image segmentation. The architecture of the Unet model is presented below.

Initially the model is trained without temporal fusion i.e vanilla model and results are presented below.

The vanilla Unet model is trained with 2 sequences (['scene0000_00', 'scene0000_01']) from scannet. For evaluation third sequence (scene0000_02) has been taken. Number of samples in scene0000_00 = 279, scene0000_01 = 296, scene0000_02 = 309

Training parameter: batch_size = 4, epochs = 201, lr = 0.001, criterion = nn.CrossEntropyLoss(), optimizer = optim.Adam(model.parameters(), lr=lr)

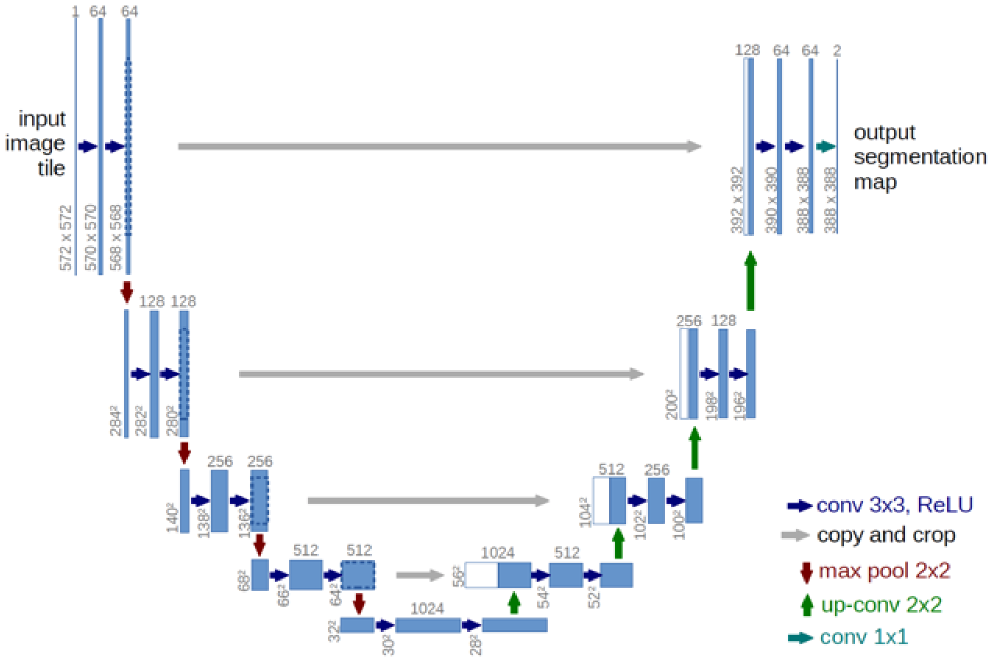


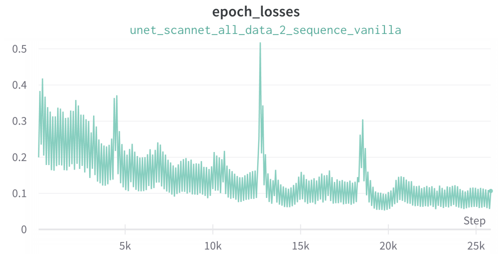
Figure 1.4: Vanilla UNet model

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   43C    P0    29W / 250W |      0MiB / 16280MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

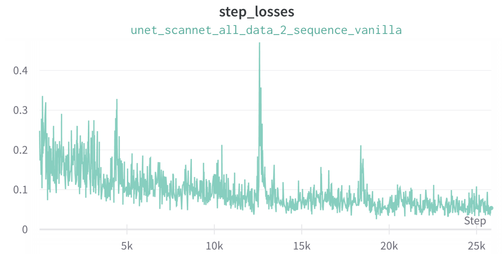
Figure 1.5: GPU configuration

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 256, 256]	1,792
ReLU-2	[-1, 64, 256, 256]	0
BatchNorm2d-3	[-1, 64, 256, 256]	128
Conv2d-4	[-1, 64, 256, 256]	36,928
ReLU-5	[-1, 64, 256, 256]	0
BatchNorm2d-6	[-1, 64, 256, 256]	128
MaxPool2d-7	[-1, 64, 128, 128]	0
Conv2d-8	[-1, 128, 128, 128]	73,856
ReLU-9	[-1, 128, 128, 128]	0
BatchNorm2d-10	[-1, 128, 128, 128]	256
Conv2d-11	[-1, 128, 128, 128]	147,584
ReLU-12	[-1, 128, 128, 128]	0
BatchNorm2d-13	[-1, 128, 128, 128]	256
MaxPool2d-14	[-1, 128, 64, 64]	0
Conv2d-15	[-1, 256, 64, 64]	295,168
ReLU-16	[-1, 256, 64, 64]	0
BatchNorm2d-17	[-1, 256, 64, 64]	512
Conv2d-18	[-1, 256, 64, 64]	590,080
ReLU-19	[-1, 256, 64, 64]	0
BatchNorm2d-20	[-1, 256, 64, 64]	512
MaxPool2d-21	[-1, 256, 32, 32]	0
Conv2d-22	[-1, 512, 32, 32]	1,180,160
ReLU-23	[-1, 512, 32, 32]	0
BatchNorm2d-24	[-1, 512, 32, 32]	1,024
Conv2d-25	[-1, 512, 32, 32]	2,359,808
ReLU-26	[-1, 512, 32, 32]	0
BatchNorm2d-27	[-1, 512, 32, 32]	1,024
MaxPool2d-28	[-1, 512, 16, 16]	0
Conv2d-29	[-1, 1024, 16, 16]	4,719,616
ReLU-30	[-1, 1024, 16, 16]	0
BatchNorm2d-31	[-1, 1024, 16, 16]	2,048
Conv2d-32	[-1, 1024, 16, 16]	9,438,208
ReLU-33	[-1, 1024, 16, 16]	0
BatchNorm2d-34	[-1, 1024, 16, 16]	2,048
ConvTranspose2d-35	[-1, 512, 32, 32]	4,719,104
Conv2d-36	[-1, 512, 32, 32]	4,719,104
ReLU-37	[-1, 512, 32, 32]	0
BatchNorm2d-38	[-1, 512, 32, 32]	1,024
Conv2d-39	[-1, 512, 32, 32]	2,359,808
ReLU-40	[-1, 512, 32, 32]	0
BatchNorm2d-41	[-1, 512, 32, 32]	1,024
ConvTranspose2d-42	[-1, 256, 64, 64]	1,179,904
Conv2d-43	[-1, 256, 64, 64]	1,179,904
ReLU-44	[-1, 256, 64, 64]	0
BatchNorm2d-45	[-1, 256, 64, 64]	512
Conv2d-46	[-1, 256, 64, 64]	590,080
ReLU-47	[-1, 256, 64, 64]	0
BatchNorm2d-48	[-1, 256, 64, 64]	512
ConvTranspose2d-49	[-1, 128, 128, 128]	295,040
Conv2d-50	[-1, 128, 128, 128]	295,040
ReLU-51	[-1, 128, 128, 128]	0
BatchNorm2d-52	[-1, 128, 128, 128]	256
Conv2d-53	[-1, 128, 128, 128]	147,584
ReLU-54	[-1, 128, 128, 128]	0
BatchNorm2d-55	[-1, 128, 128, 128]	256

Figure 1.6: Network architecture



(a) epoch losses

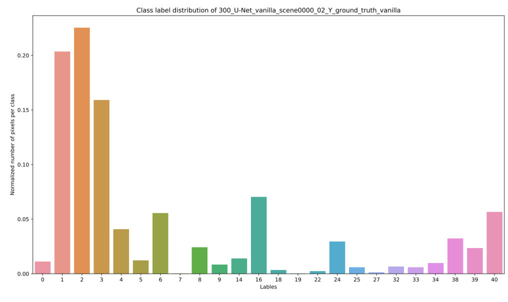


(b) step loses

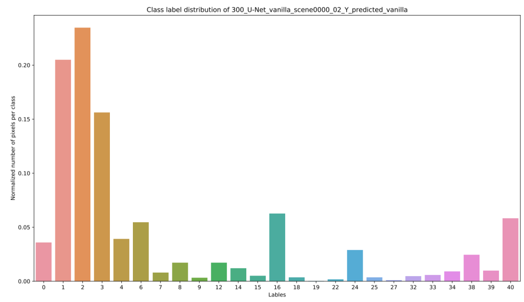
Figure 1.7: 2 Epoch losses and step losses

```
Experiment: 300_U-Net_vanilla_scene0000_02_validation_data


iou:[0.04437906 0.73293136 0.87293239 0.71476792 0.83797387 0.
 0.79456671 0.         0.54382845 0.33428701        nan        nan
 0.                nan 0.67717294 0.         0.72800201        nan
 0.70462709 0.                nan        nan 0.45290468        nan
 0.69677545 0.45357421        nan 0.63785677        nan        nan
        nan        nan 0.37872412 0.63086775 0.65806111        nan
        nan        nan 0.43221691 0.34005826 0.47502526]


meaniou:0.4669820514093615
-------------------------------
Experiment:300_U-Net_vanilla_scene0000_02_Y_ground_truth_vanilla


Counted:{3: 3777909, 1: 4827271, 16: 1665538, 14: 333439, 40: 1339738, 0: 266606, 2: 5343004, 5: 289097, 22: 55467, 32: 155687, 4: 961328, 7: 1138, 39: 552443, 18: 78042, 38:
762545, 8: 573372, 34: 228948, 19: 1747, 33: 139176, 9: 194530, 24: 695429, 25: 140006, 6: 1316397, 27: 32343}


normalised_d:{3: 0.15919586872977345, 1: 0.20341453445253505, 16: 0.07018347154800432, 14: 0.014050659048004314, 40: 0.056454709412081985, 0: 0.011234408710895361, 2:
0.2251468109492988, 5: 0.012182148395361381, 22: 0.002337302791262136, 32: 0.0065604352076591155, 4: 0.04050903451995685, 7: 4.795374865156418e-05, 39: 0.023279185207659116,
18: 0.003288582197411002, 38: 0.032132593379180154, 8: 0.024161104368932037, 34: 0.009647552588996763, 19: 7.361616774541532e-05, 33: 0.005864684466019418, 9:
0.008197225593311758, 24: 0.02930441781283711, 25: 0.00589965951995685, 6: 0.0554711519012945, 27: 0.001362889360841424}
-------------------------------
Experiment:300_U-Net_vanilla_scene0000_02_Y_predicted_vanilla


Counted:{3: 3706662, 1: 4860148, 16: 1483896, 2: 5563705, 14: 286159, 40: 1380036, 18: 82360, 34: 213237, 32: 106425, 7: 191024, 12: 406877, 0: 852254, 38: 580916, 39: 233800,
24: 686335, 4: 928383, 8: 407538, 6: 1291977, 22: 42421, 9: 73631, 25: 78363, 33: 133828, 15: 113953, 19: 2575, 27: 24697}


normalised_d:{3: 0.1561936185275081, 1: 0.2047999258360302, 16: 0.0625293284789644, 2: 0.23444684634573895, 14: 0.01205834513214671, 40: 0.058152811488673135, 18:
0.0034705366774541533, 34: 0.008985512742718447, 32: 0.0044846025485436895, 7: 0.008049487594390507, 12: 0.017145234964940668, 0: 0.035912806769147786, 38:
0.02447899811218986, 39: 0.009852009169363539, 24: 0.028921209209816612, 4: 0.039120777710355985, 8: 0.01717308859223301, 6: 0.05444212682038835, 22: 0.0017875623651564185, 9:
0.003102708670442287, 25: 0.0033021086165048543, 33: 0.005639327130528587, 15: 0.004801822073894283, 19: 0.00010850694444444444, 27: 0.001040697478425027}
-------------------------------
```

Figure 1.8: Validation results, IOU and meanIOU



(a) Pixel distribution of the validation data



(b) Pixel distribution of the predicted label

Figure 1.9: Pixel distribution of the validation and predicted label

In the second experiment temporal fusion is done at the latent space. The output from latent space is taken and subjected to Gaussian process regression. The pictorial representation of the same is presented below.

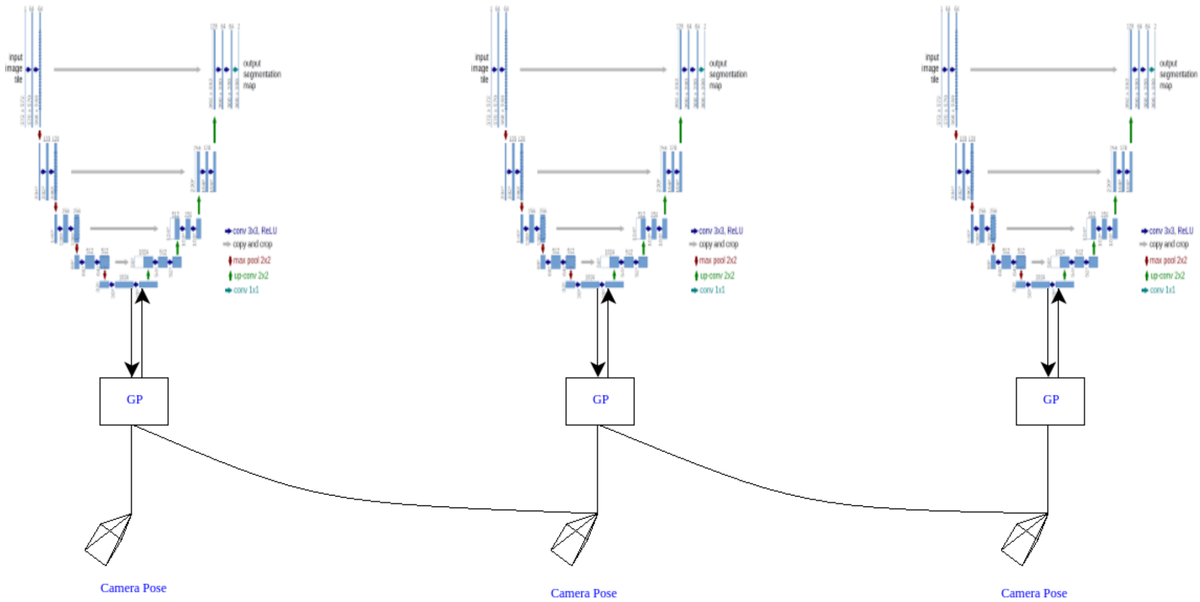


Figure 1.10: VIODE RGB and ground truth

Training is completed, however when I run the script for validation data the system is throwing error that needs to be fixed.

num_classes = 41

batch_size = 4

epochs = 201

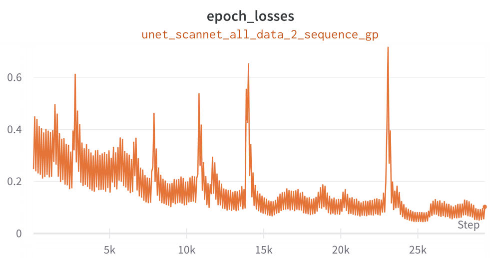
lr = 0.001

criterion = nn.CrossEntropyLoss()

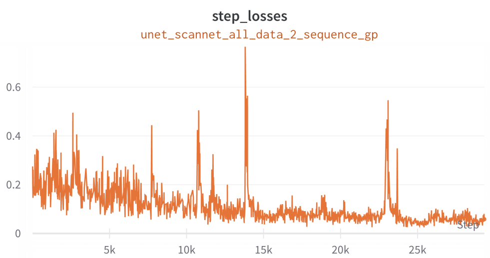
optimizer = optim.Adam(model.parameters(), lr=lr)

1.2 Future work

- Work on the Temporal fusion
- Train the model with more sequences
- Evaluation strategy
- Table of results



(a) Epoch losses



(b) Step losses

Figure 1.11: Epoch and Step losses