Project Report for CIS 680 - Advanced Machine Perception

Siddharth Singh [1]

Abstract

The authors in this work present a complete perception pipeline which helps in developing semantic maps in the ego-centric vehicle's coordinate frame. The architecture uses two deep learning frameworks. The first framework is yolo version 3 which is used for detection of objects in the video stream and the second deep network is a prediction network consisting of stacked convolutional LSTMs which help in generating/predicting future frames of the video sequence given a time series of context frames. The assumption is that, a time series data dependent recurrent network such as that of stacked LSTMs helps in encoding the inherent kinematics of the objects correlated with the environment in the image in order to do predictions. To simplify the problem, we do not work in the image space, but rather, we create a very simplified representation of the environment and objects detected by YOLO. This simplified representation of data is also in the form of an image but has very simple representations of the environment making it easier for the neural network to work with it.

1. Background

With the recent advances in the capabilities of using deep learning networks in developing vision based detection and prediction architectures, we are now moving towards better and more integrated capabilities in the world of robotics. Raw sensor data from cameras has always been a little difficult to work with and hence it becomes important that the data is simplified to make it easier for the deep networks to perform the predictions and ultimately be able to plan safe trajectories for traversals[1]. Moreover, the representation of the data can be extrapolated to create a model of the environment which helps in bridging the gap between simulation and reality even more. There are existing works wherein the representation of the environment is in the form of a model which can then be used to perform other tasks[2]. In this work authors propose a framework which provides prediction capabilities in the ego centric vehicle's reference frame thus getting rid of the image space and directly performing predictions for obstacles in the 3d coordinate frame of the ego centric vehicle. Also, We have simplified the representation of the environment in the reference frame of the ego centric vehicle. The representation is in the form of an occupancy grid which has semantic information about the type of the obstacle that is present and also incorporates other obstacles which are not classified confidently by the detection network. Obstacles such as walls or tracks also become important in this representation because they help in the prediction task. The network should be able to understand that if there is a turn defined by the walls or by the track, there is high probability of the detected object to take a turn.

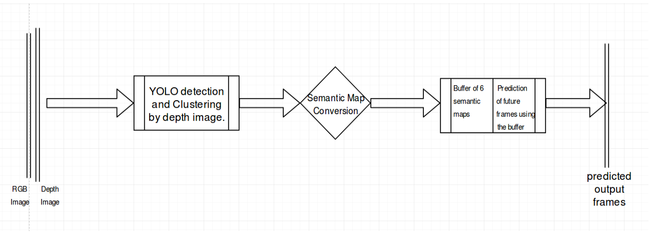

RGB Image
Depth Image
YOLO detection and Clustering by depth image.
Semantic Map Conversion
Buffer of 6 semantic maps
Prediction of future frames using the buffer
predicted output frames

Figure 1. Perception Pipeline

2. Pipeline

Our work consists of an entire perception and mapping pipeline which has the following elements:

- Detection and Classification with Yolo
- Semantic Information Representation of Detection:Mapping
- Prediction

At an abstract level, the pipeline has the flow of information represented in figure 1.

2.1. Object Detection and Classification with YOLO

The first element of the architecture is performing detection in the image space for objects and finding the pixel coordinates of the bounding boxes.

Yolo-V3[1] is a well known detection and classification

framework which can perform detection and multiclass classification and we decided to go ahead with this architecture[7].
We are using a ROS wrapper for YOLO version 3 by the Legged Robotics lab, ETH Zurich[8] and modified it in the following ways.

- It now also takes in the depth image stream from a depth camera.
- It performs clustering of the foreground and background depths and picks up only the foreground depths of the detected object.
- It publishes the coordinates of the foreground depth and the correlated image pixel coordinates.

The YOLO version 3 model we are using has been trained on the coco data set with 80 classes and pascal VOC dataset with 20 classes. Figure 3 shows some of the example representations of the detection and classification made by the model along with the semantic map created by them.

2.2. Semantic Information Representation: Mapping

After performing the detection, we need to extract the information of the classifications as well as the unclassified obstacles (such as walls) and project them on the reference frame of the camera. We project all the information we extract from the image space and depth space onto the 2d plane passing through the baseline and rectified optical center of the stereo camera. Classical formulations of geometrical vision can be used to perform this task.There have been several attempts to create such semantic maps which are simple in nature and provide a lot of information needed[1][3][4].
We use the depth from the camera and the angle between the projected ray and the ray passing through the optical center of the camera to calculate the x and y coordinates of the pixels of interest of our detected object(figure 2).
Once we have the pixel coordinates in the frame of reference of the camera, we only have to provide semantic information for those pixels. We already have that information from our detection and classification framework. We simply assign a different(pre-set) value at each of the channel to differentiate between the classes in our representation of the environment.
Figure 3 depicts the semantic maps and their associated original images. In these images the unclassified objects and under-confident classifications have been assigned the white color.
However, in order to provide channel wise object classification, we convert the color of unclassified and under classified objects in the image space to the green channel as shown in figure 4 and 5.

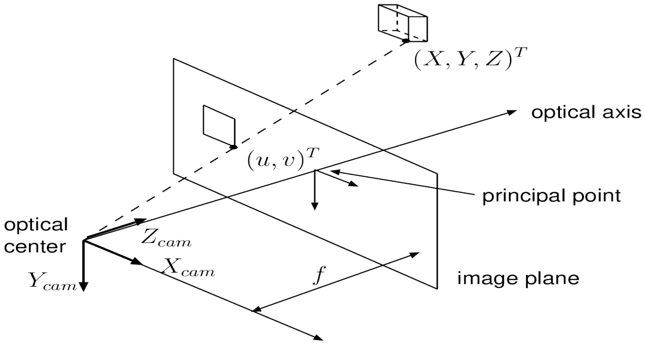
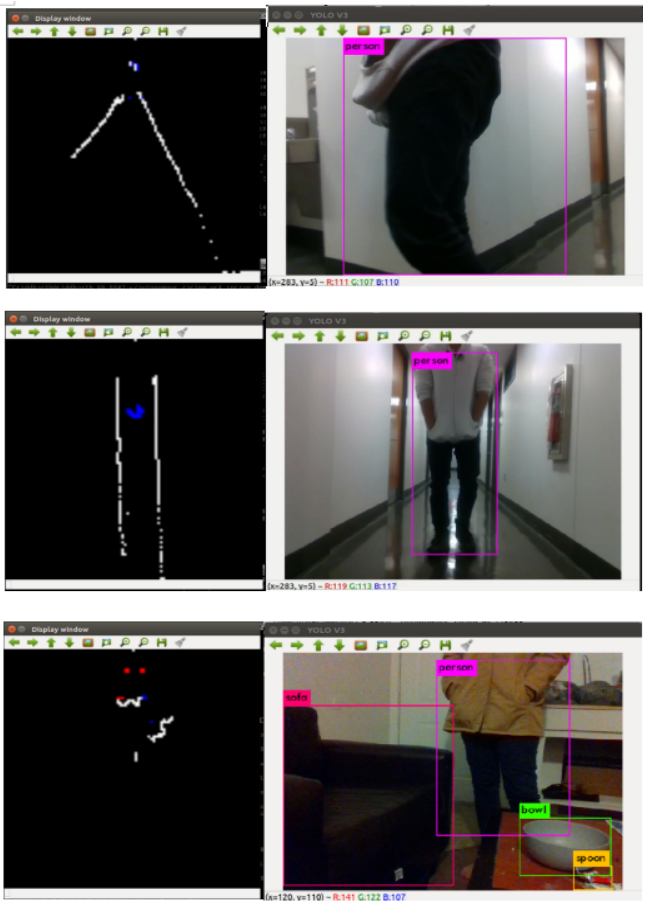


Figure 2. The pinhole camera model used along with depth to calculate camera frame coordinates



(x=283, y=5) ~ R:111 G:107 B:110

(x=283, y=5) ~ R:119 G:113 B:117

(x=120, y=110) ~ R:141 G:122 B:107

Figure 3. Detection and classification by YOLO and their symantic maps

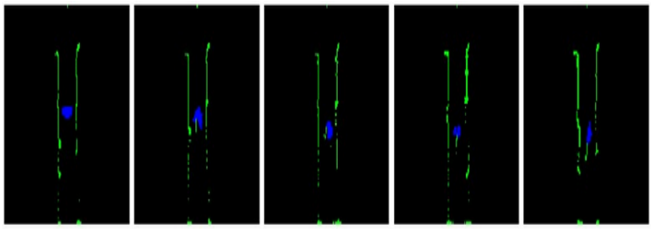


Figure 4. First five frames of the sequence of a person walking

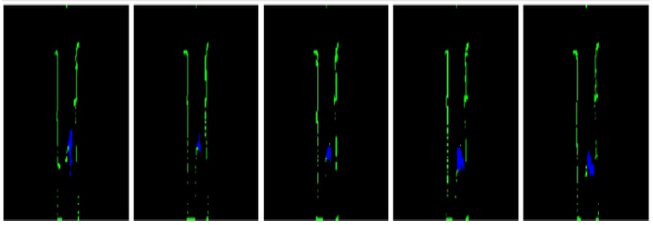


Figure 5. Second five frames of the sequence of a person walking

- blue : People
- green : unclassified objects or objects with classification confidence less than 95 percent.
- red : Any other classified object

Figure 4 and Figure 5 depict a sequence of symantic map images which include a person walking.

These semantic maps also have representations of not just the classified objects but also the free drivaeble space which is here represented as the space between the corridors.
Once we have the semantic map representation of the environment we can provide this information to the prediction network to perform the predictions.

2.3. Prediction

Now that we have the semantic map representation, we can have our prediction architecture perform predictions for the future frames of the time series data. For our prediction network we use an **architecture** of
The architecture design is represented in figure 6.
The architecture takes in a fixed number of frames as input and outputs a single frame of the future. For example, the architecture can take in 8 frames and output the 10th frame of the future.
Following are the specifications of the model that we have trained.

- Loss Functions:

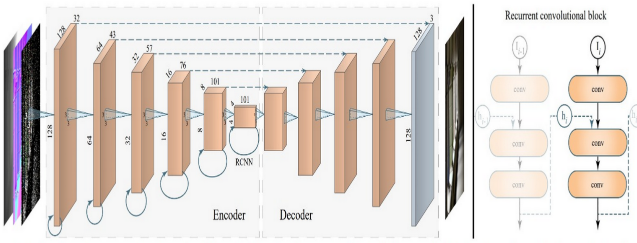


Figure 6. Architecture of Prediction Model

  - Binary Cross Entropy Loss

$$L = \frac{1}{N}\sum_{i}^{N}\left(-T_n \cdot \log P_n + (1 - T_n) \cdot \log\left(1 - P_n\right)\right)$$

  - L1 Loss

$$L = \frac{1}{N}\sum_{i}^{N}|P_i - T_i|$$

  - MSE Loss

$$L = \frac{1}{N}\sum_{i}^{N}(P_i - T_i)^2$$

  - Loss with temporal and gradient differences

$$\mathcal{L} = w_{\mathrm{s}}\mathcal{L}_{\mathrm{s}} + w_{\mathrm{g}}\mathcal{L}_{\mathrm{g}} + w_{\mathrm{t}}\mathcal{L}_{\mathrm{t}}$$

  where

$$\mathcal{L}_{\mathrm{s}} = \frac{1}{N}\sum_{i}^{N}|P_i - T_i|$$

$$\mathcal{L}_{\mathrm{g}} = \frac{1}{N}\sum_{i}^{N}|\nabla P_i - \nabla T_i|$$

$$\mathcal{L}_{\mathrm{t}} = \frac{1}{N}\sum_{i}^{N}\left(\left|\frac{\partial P_i}{\partial t} - \frac{\partial T_i}{\partial t}\right|\right)$$

$$w_{\mathrm{s}}, w_{\mathrm{g}}, w_{\mathrm{t}} = (0.8, 0.1, 0.1)$$

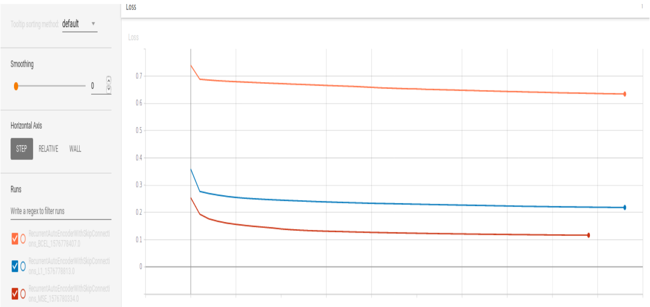
  Where P represents predicted pixel value and T represents ground truth pixel values.

- Optimizer:
  The model was trained using SGD with $1e - 3$ as the learning rate, momentum as 0.9 and weight-decay as $1e - 4$.
- Learning Rate optimization:
  Learning rate was scheduled to drop every 20 epoch by a factor of 0.8

We have also noticed that during training, since majority of our image is black (which is the space undetected) We perform weighted sampling of the background(black) pixels and the colored pixels.This helps in balancing the loss since the ratio between the black and colored pixels is very skewed. This was done for all losses with suitable weights. Also the losses for predicted sequence were weighted using a Gaussian curve with weights as (0.011, 0.044, 0.135, 0.325, 0.607, 0.882, 1).

3. Results

Our network has been able to converge and has the loss trend depicted in figure 7. The model was run for 50 epochs.


Tooltip sorting method: default

Smoothing: 0

Horizontal Axis: STEP RELATIVE WALL

Runs

Write a regex to filter runs

☑ ○ RecurrentAutoEncoderWithSkipConnections_BCEL_1576778407.0

☑ ○ RecurrentAutoEncoderWithSkipConnections_L1_1576778813.0

☑ ○ RecurrentAutoEncoderWithSkipConnections_MSE_1576780334.0

Figure 7. Loss Trend of the model ran over 50 epochs

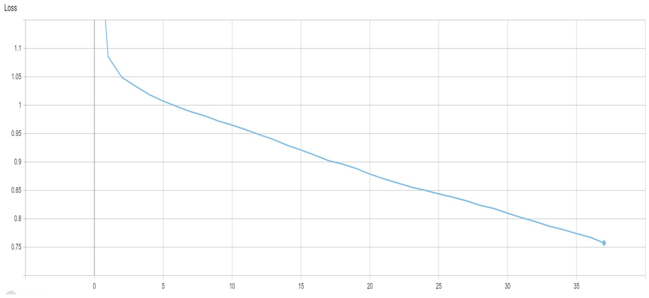


Figure 8. Loss Trend of model using the Temporal+Gradient loss function for 36 epochs

Very interestingly, we have had different results for different kinds of specified losses. The results from the different losses trends are depicted in figures 9, 10, 11 and 12. For the loss with temporal and gradient differences, a significant of tuning of weights on loss values is required, but even with un-optimized weights and less epochs for which other losses were run, we see significant improvement over L1, BCE and L2.

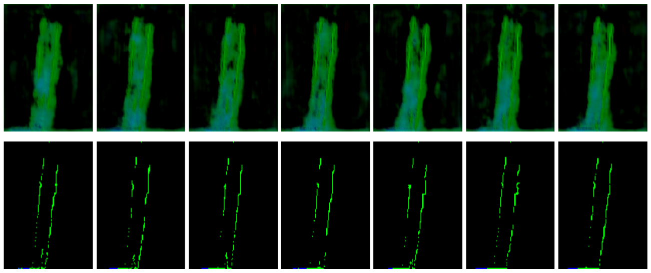


Figure 9. Predictions of frames using L2 MSE Loss

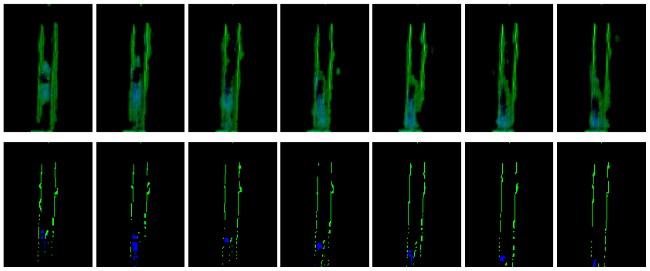


Figure 10. Predictions of frames using BCE Loss

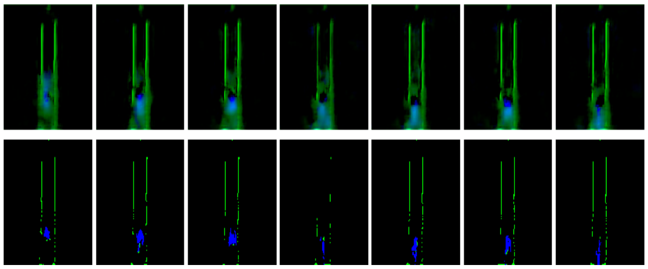


Figure 11. Prediction of frames using L1 loss

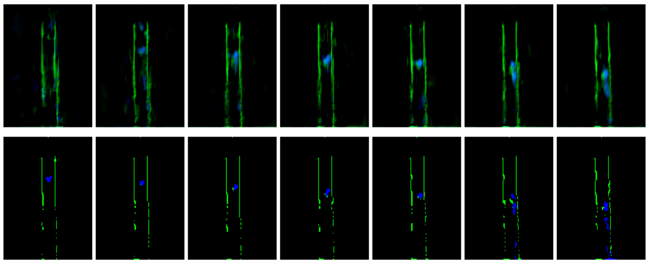


Figure 12. Predictions of frames using the Temporal+Gradient loss function

4. Conclusions

We can clearly see that the predictions of images has been the best by the loss function which uses gradient and temporal difference of pixel values as its metric. The BCE loss has also been able to perform well but it is not able to perform as well in terms of reconstructing the blue blob (the depictions of the person) as it is able to in terms of L1 Loss. It is thus important to understand that when performing predictions the temporal nature of the data and the pixel flow is an necessary factor governing the dynamics/kinetics of the object motion that we want to encode/capture.

5. Way Forward

We have been able to perform the predictions to some extent but not accurately. There is still a lot of scope that can help in improving the predictions and making the reconstructions even more accurate. Some of them are as follows:

- Using more complicated loss functions with for example adversarial losses.
- Reducing the noise in the semantic representation of the map
- Using pre-defined shapes(simple such as circles, ovals or squares) in place of blobs depicting the classifications.
- Using classification-segmentation architectures such as Mask-RCNN to create semantic maps.

Also, it is clear that we need to test this in more complex environment and understand if the network is able to generalize well and perform predictions when there are large number of objects represented in the semantic maps and their dynamics are encoded in the context frames.

6. References

1. Mayank Bansal, Alex Krizhevsky, Abhijit Ogale, **ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst**
2. David Ha, Jürgen Schmidhuber, World Model
3. Saman Kumpakeaw, Rüdiger Dillmann,Semantic Road Maps for Autonomous Vehicles
4. Daniel Maturana, Po-Wei Chou, Masashi Uenoyama and Sebastian Scherer, Real-time Semantic Mapping for AutonomousOff-Road Navigation
5. Selva Castelló, Javier. A Comprehensive survey on deep future frame video prediction. MS thesis. Universitat Politècnica de Catalunya, 2018
6. Selva Castelló, Javier. A Comprehensive survey on deep future frame video prediction. MS thesis. Universitat Politècnica de Catalunya, 2018
7. Joseph Redmon, Ali Farhadi,YOLOv3: An Incremental Improvement
8. Darknet_ros: Legged Robotics

7. Resources

A video description of the project can be found at this drive link