Cross-view Semantic Segmentation: Multi-View vs. Monocular Methods

Anthony Le
Stanford University
antle1@stanford.edu

Andrew Tang
Stanford University
andrewht@stanford.edu

Catherine Wang
Stanford University
cyw339@stanford.edu

Abstract

This paper surveys three different implementations for the task of cross-view semantic segmentation. We compare and contrast the implementations and results in order to conclude which state of the art model performs best on real world data, particularly the NuScenes dataset. These implementations range from using semantically segmented, multi-view images to using single monocular images with depth estimates. We conclude that using a large training set of semantically segmented simulated data on a CNN based model performs better than other state of the art methods.

1. Introduction

In a world being continually filled by autonomous technology, we must look towards further improvement of systems that use computer vision and scene reconstruction. One contemporary example is that of Autonomous Vehicles and Autonomous Robots in the household. Both technologies will become even more widespread in the near future, so it is of paramount importance that we increase state-of-the-art performance as these technologies are being deployed into the real world and are interacting with real people. Semantic segmentation from a cross-view/bird's eye view allows autonomous vehicles to better understand the objects and their relative positions in space, ultimately improving autonomous navigation in a dynamic environment. Here, we aim to survey three different implementations of cross-view/bird's eye view semantic segmentation in order to compare and contrast current state of the art models with real world datasets.

2. Problem Statement

Our problem is to convert first-view inputs to a top-down semantic segmentation of the surroundings from a set of RGB images corresponding to different angles of the scene. The motivation for this problem is to improve robot/AV navigation by giving them a better sense of their entire surroundings from multiple first-view observations.

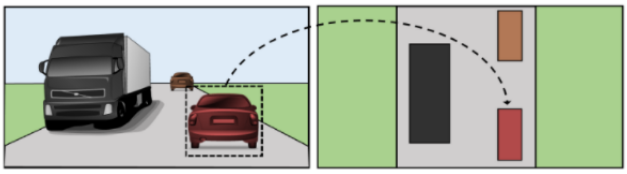


Figure 1. An example of cross-view semantic segmentation, where first-view RGB images are used to reconstruct the ground truth top-down structure of the scene.

Many state of the art models tackling this problem use simulated environments to produce the large amount of data needed to train neural networks. In the papers "Cross-view Semantic Segmentation for Sensing Surroundings" [2] and "Cam2Bev" [3], the authors first trained their network on data taken from simulated 3D environments and then transferred the network to real world data. On the other hand, some implementations train on real world data where ground truth labels have been labeled and/or acquired by Lidar/Radar sensors and constrain their inputs to a single first-view image [1]. Bird's eye view reconstruction helps AV's and other autonomous systems navigate their surroundings while interacting with an environment by perceiving both static and dynamic objects in an accurate manner. As we will see through the rest of this paper, many of the implementations will try solve this difficult problem leveraging semantic segmentation, additional input modalities like depth, and the use of neural networks.

3. Background/Related Work

3.1. View Parsing Network [2]

The View Parsing Network (VPN) uses multiple first-view observations (in a simulated environment) as inputs. For each image, it uses an encoder to extract a first-view feature map. First-view feature maps are then transformed and aggregated into one top-down-view feature map and decoded into a top-down-view semantic segmentation. The paper experimented with using different modalities of inputs (RGB, depth, semantic segmentations, combinations of these) and found that a combination of semantic segmen-

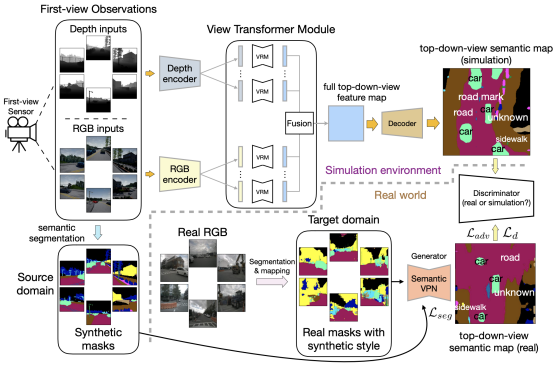


Figure 2. Occluded regions are modified in ground-truth labels (i.e. black regions)

tations and depth maps worked best for the synthetic data. When transferring to real-world data, the paper only used semantic segmentations of RGB images as inputs. This paper was among the first to handle the problem of lack of real- world labeled data by using simulated environments to produce training data and domain adaptation techniques to handle real-world data.

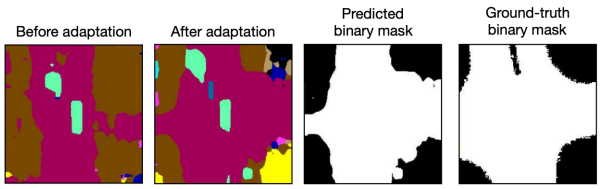
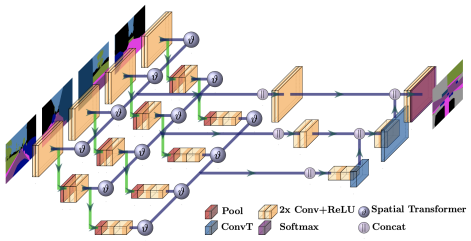


Figure 3. Caption

In order for the model to effectively work on real-world data, the authors of this paper used several domain adaptation techniques such as pixel-level adaptation to mitigate the domain shift from the simulated environment to the actual scene view RGB inputs, and output space adaptation. This resulted in overall performance gains and cleaner aesthetics of BEV segmentations.

3.2. Cam2BEV [3]



The Cam2BEV paper is concurrent to the VPN paper and proposes a solution for cross-view semantic segmentation by extending the Inverse Perspective Mapping (IPM) method. IPM uses perspective transformations to map first view images to bird's eye view but assumes that the world is flat, resulting in distortion when encountering objects with height, such as people or other cars. Cam2BEV takes in sets of multiple first-view images and uses CNNs in conjunction with IPM to correct for these distortions. Like the VPN, these models are trained on simulated data and use semantic segmentations to ensure better transfer to real-world data.

This paper also aims to tackle common problems when using simulated data such as the so-called reality gap where the models don't perform nearly as well on real-world data. This implementation removes unnecessary texture from real world data by semantically segmenting the images first. This also helps in computation because the semantic segmentations of the BEV/Cross-view scene ahve already been done as soon as the model outputs a BEV reconstruction. This is because the algorithm already has access to class information, i.e. the different colors of segmented objects, to use while correcting the flat IPM images. Not only doese this help with static objects but dynamic objects as well, overall increasing robustness.

Figure 4. Black regions represent occluded regions

Another problem that is common to cross-view reconstruction and semantic segmentation is noise in the data from occluded regions in the scene space. Cam2BEV attacks this problem by preprocessing their data in a way that all semantically segmented inputs have an additional semantic class for occluded regions or spaces. They do this by projecting rays out from the camera centers whose pixels along that ray determine their occlusion status.

3.3. MonoLayout [1]

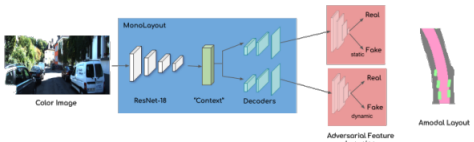


Figure 2: **Architecture**: *MonoLayout* takes in a color image of an urban driving scenario, and predicts an amodal scene layout in bird's eye view. The architecture comprises a *context encoder*, *amodal layout decoders*, and *two discriminators*.

Unlike the two previous models, which require multiple first-view inputs, this paper aims to estimate top down view geometries with a single RGB color image. Under a lot of circumstances, implementations will not have multiple first-view images to work with, and estimation under monocular terms is less data intensive. This model produces a multichannel top down semantic occupancy grid that allows AV's to estimate objects locations in relation to the camera. This paper achieves state of the art performance using single view data from KITTI and Argoverse datasets.

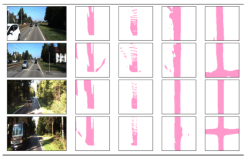


Figure 5. Impact of sensor fusion shown here, as increasing amount of Lidar/Radar sensor information being used from left to right.

This model didn't perform as well as aformentioned implementations but provided us insight into how one could produce meaningful spatial representations from a single image. This paper boosted their performance by including sensor fusion into their original approach, which led to much more defined reconstructions of BEV mappings with sharper object boundaries. This is because the inclusion of things like Lidar have been increasingly helpful as they produce accurate depth estimates. In this project we aim to discover how we can reconstruct semantically segmented cross-view scenes with RGB images alone, so this might mean monocular methods in themselves may not be the answer to robust AV/robot control needed in high risk situations such as a busy traffic corner.

4. Approach

4.1. Initial Approach

Our initial approach was to build on the VPN and add depth maps as inputs when using real-world data. With the simulated data, the authors found that the network performed best when it was given the first-view observations as both semantically segmented images and depth images. However, when transferring the model to real world data, the authors only use semantic segmentations taken from RGB images as input, losing this helpful depth data. The authors of the paper did this because acquiring depth information in the real world requires additional equipment like depth cameras, so using only RGB images is more realistic. We wanted to adhere to this constraint while still leveraging depth information to improve the cross-view semantic segmentations. Our plan was to use monocular depth estimation to obtain depth maps from RGB images without additional equipment.

However, we ran into several challenges when attempting to implement this approach. The VPN has specific data needs, where inputs must be sets of first-view images taken at the same position from different angles, and labels must be top-down semantic segmentations. To train and evaluate their model, the authors created two new datasets by taking images from navigation episodes in two simulated environments–House3D for indoor scenes and Carla for outdoor driving scenes [2]. These datasets were unavailable to us as were the 3D simulating softwares, so we looked towards approaches with other simulated datasets that were readily available.

When reviewing additional papers to explore other methods of addressing cross-view semantic segmentation and find ways we could still implement our proposed modifications, we found that the MonoLayout paper uses monocular depth estimation to supplement their inputs with depth maps and the Cam2BEV paper proposes adding depth information to inputs in their conclusion, validating our idea that depth inputs would be a useful addition to the VPN for real-world data.

4.2. Survey and Comparison

We later discovered the Cam2BEV paper, which presented work that was concurrent to the VPN paper and proposed a solution for the same task. This paper did have accessible preprocessed datasets and clear documentation, which made it a better option for running experiments. We modified our approach to instead survey the results of different cross-view semantic segmentation methods and compare their performance. Our paper serves to show the state-of-the-art methods of producing cross-view semantic segmentation from RGB images in the AV setting.

5. Experiments

5.1. Datasets

nuScenes:

The full nuScenes dataset contains 1000 outdoor driving scenes collected from Boston and Singapore. The provided training, validation, and test splits consist of 700, 150, and 150 scenes, respectively (for a total of 28350 input training samples). One thing to note, however, is that the test set does not come with scene object annotations. Nevertheless, most importantly, this dataset contains images from several directions at the same position, allowing our models to function outside of the simulated environments mentioned in the View Parsing Network paper [2]. We had planned on using the RGB inputs from this dataset both to train the semantic VPN and subsequently in our monocular depth estimation model.

CityScapes:

This dataset contains 5000 finely annotated images and 20000 coarsely annotated images, with manually selected frames and semantic segmentation (both pixel-level and instance-level). We had planned to use this dataset primarily for training an initial HRNet, as well as for creating depth information from the monocular depth estimation model.

Dataset 1_FRLR:

This synthetic dataset contains 33,000 training samples and 3,700 validation samples consisting of a set of multiple first-view input images and a top-down ground truth image [3]. The samples are obtained from the Virtual Test Drive simulation environment using a virtual vehicle with four cameras at different angles and a virtual drone [3]. The samples come in both realistic RGB and semantically segmented modalities [3].

KITTI:

This collection of datasets (containing the raw data version of KITTI, KITTI Odometry, KITTI Object, and KITTI Tracking) contains both segmentation and object detection information. These were used by the MonoLayout paper [1] to evaluate the performance of their model against existing approaches, and will be useful for us in our comparison of that paper.

5.2. Evaluation Metrics

Mean intersection-over-union (mIoU) was used across all three papers as an evaluation metric, so we will also be using this metric to compare results.

5.3. Planned Experiments

The experiments we ran consisted of a side by side comparison of the aforementioned state-of-the-art models. We were able to reproduce smaller versions of some of the models and will analyze these results alongside the original results in order to determine the most effective solution to autonomous navigation and scene reconstruction and interaction. Alongside this we will present our proposed model modifications that we believe will improve performance, but because of lack of computational resources and time, were unable to finish training.

Although we had initially planned on adding modifications to the models from the "Cross-view Semantic Segmentation for Sensing Surroundings" paper, we were hampered by issues such as a lack of accessible data and documentation that made the paper's results difficult to reproduce. As we mentioned previously, the models in the paper relied heavily on new datasets that the researchers created from simulated environments that were not readily downloadable or accessible. Furthermore, a lack of solid documentation made the code difficult to decipher. Additionally, training these models with lack of pretrained weights required several GPU's and much more compute power and time than we had. In order to expedite the process we ran the models on smaller subsets of the data and for a smaller number of epochs.

However, our initial plan definitely remained viable, even though we were ultimately unable to follow through with running the models due to the aforementioned circumstances. Once again, we were looking to use nuScenes and go through the following steps. First, we wanted to run a semantic VPN on semantic segmentations from the nuScenes RGB images to get a baseline while training a monocular depth estimation model on CityScapes. We would then feed the nuScenes RGB images to the trained depth estimation model to get depth maps. With all this complete, we would then use RGB images and the corresponding generated depth maps to get new semantic segmentations from a HRNet model, and finally combine the semantic segmentations with the depth maps as input to the semantic and depth version of the VPN, producing our top-down semantic segmentations.

5.4. Model Comparison Results

We were able to run the Cam2BEV model on the Dataset 1_FRLR for five epochs (each epoch took about four hours to run). We show the results below compared to the results reported by the VPN and MonoLayout papers. Note that the results for all three papers were obtained from different datasets, and since the MonoLayout architecture only takes in a single image as input, it is not directly comparable to the other two models.

Model	**mIoU**
Cam2BEV (5 epochs)	62.03
Cam2BEV (paper)	71.92
Semantic VPN	40.6
MonoLayout (KITTI Tracking)	53.94

Figure 6. BEV prediction of the Cam2BEV model after five epochs.

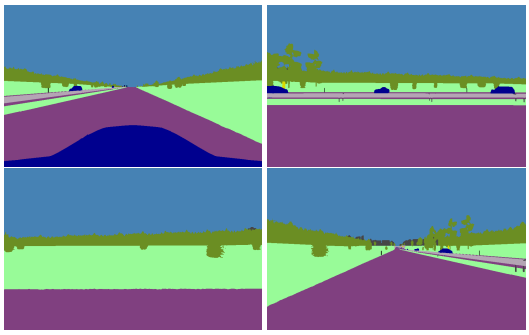


Figure 7. Front, left, right, and rear inputs used to make the prediction.

As we can see from the table, the Cam2BEV model performs much better than the semantic VPN and MonoLayout on the cross view semantic segmentation task. One reason for this may be that Cam2BEV leverages the already working IPM method and seeks to correct distortions using CNNs, whereas the VPN introduces a new architecture.

6. Conclusion

We initially proposed improving the VPN results on real world data by adding depth maps obtained using monocular depth estimation to the inputs. While we were unable to implement this due to data and documentation issues and time and computing constraints, further research confirmed that this approach would likely have led to improvements. The Cam2BEV model provided a better starting point for our experiments and we were able to run training for five epochs and replicate some of the model's results. We found comparing VPN, Cam2BEV, and MonoLayout, that Cam2BEV performed the best and provides promising results for improving autonomous driving.

Link to Github repo containing code we used: https://github.com/andrewhlt/View-Parsing-Network.git

References

[1] K. Mani, S. Daga, S. Garg, N. S. Shankar, K. M. Jatavallabhula, and K. M. Krishna. Monolayout: Amodal scene layout from a single image, 2020.

[2] B. Pan, J. Sun, H. Y. T. Leung, A. Andonian, and B. Zhou. Cross-view semantic segmentation for sensing surroundings. *IEEE Robotics and Automation Letters*, 5(3):4867–4873, 2020.

[3] L. Reiher, B. Lampe, and L. Eckstein. A sim2real deep learning approach for the transformation of images from multiple vehicle-mounted cameras to a semantically segmented image in bird's eye view. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020.