Image Segmentation with Depth Estimation:Progress Report

Vedant Parnaik[1] and Advait Raman Samudralwar[1]

[1]Department of Engineering and Applied Science, University at Buffalo

May 11, 2023

1 Overview of the project

1.1 Description of the application being developed

The project aims to perform image segmentation along with the depths of the segments using LIDAR depth maps and camera images. Using these two inputs, we can estimate the depth of the segmented sections of the image.
Input: 2D Dense Depth map, Original Images
Output: Object Segmentation with depth

1.2 State-of-the-art

Current state-of-the-art LIDAR sensors can only provide sparse depth maps when projected back to image space and, normal camera images alone cannot provide the depth of the segmented sections of the image. Most of the models built today use neural networks, which need a lot of computation and require a larger amount of data to train. High computation is a problem for a lot of computer vision applications like object detection and tracking done on a camera mounted over a drone which has limited computational capacity. The project also has a future scope of establishing a spatial 3D metric relationship of the objects in the image using methods like Haversime distance, and Minkowski distance.

Yes, there are methods present to accomplish similar results namely,

- Depth-Based Image Segmentation
- Image Segmentation Based on Histogram of Depth and an Application in Driver Distraction Detection
- Sim2Real for Self-Supervised Monocular Depth and Segmentation
- Segmenting Unknown 3D Objects from Real Depth Images using Mask R-CNN Trained on Synthetic Data
- In Defense of Classical Image Processing: Fast Depth Completion on the CPU

2 Approach

2.1 Implementation, Details of the algorithms and their relevance

1. Depth inversion: Inverting the KITTI depth maps.
2. Custom Kernel Dilation: We start by filling empty pixels nearest to valid pixels, as these are most likely to share close depth values with valid depths. Considering both the sparsity of projected points and the structure of the LIDAR scan lines, we design a custom kernel for an initial dilation of each valid depth pixel.

3. Small Hole Closure: After the initial dilation step, many holes still exist in the depth map. Since these areas contain no depth values, we consider the structure of objects in the environment and note that nearby patches of dilated depths can be connected to form the edges of objects. This step acts to connect nearby depth values, and can be seen as a set of 5 × 5 pixel planes stacked from farthest to nearest
4. Small Hole Fill: Some small to medium-sized holes in the depth map are not filled by the first two dilation operations. To fill these holes, a mask of empty pixels is first calculated, followed by a 7 × 7 full kernel dilation operation.
5. Extension to Top of Frame: To account for tall objects such as trees, poles, and buildings that extend above the top of LIDAR points, the top value along each column is extrapolated to the top of the image, providing a denser depth map output.
6. Large Hole Fill: The final fill step takes care of larger holes in the depth map that are not fully filled from previous steps. Since these areas contain no points, and no image data is used, the depth values for these pixels are extrapolated from nearby values.
7. Median and Gaussian Blur: After applying the previous steps, we end up with a dense depth map. However, outliers exist in this depth map as a by-product of the dilation operations. To remove these outliers, we use a 5 × 5 kernel median blur.
8. Depth Inversion: The final step of our algorithm is to revert back to the original depth encoding from the inverted depth values used by the previous steps of the algorithm.
9. Original Image Segmentation: Performing Mean-shift image Segmentation on original segmentation.
10. Superimposing: Overlaying original images with depth images
11. Superimposed Segmentation: Perform segmentation on the superimposed image.

2.2 Pros and cons of the algorithm

- Uses classical OpenCV operations
- Does not require any pre-trained data
- This is faster and more efficient than any Neural Networks

2.3 Coded on our own

We have hard-coded the last 3 points (9th, 10th, 11th) of the complete algorithm.

- Original Image Segmentation
- Superimposing
- Superimposed Segmentation

2.4 Demonstrates an understanding of how the self-coded aspects of the algorithms contribute to the project

The self-coded aspects are the most important part of the project. The referenced code is only capable of creating a depth map. Our coded part of capable of performing the most crucial objectives of the project like superimposing and segmentation of both the images. The project would be incomplete without our self-coded algorithms.

2.5 Algorithms have you used from the online resources and its relevance

We have used the code from a research paper to create depth maps. We have used points 1st to 8th from a GitHub repository based on a research paper.
The Research paper: `https://arxiv.org/abs/1802.00036`
The GitHub repository: `https://github.com/kujason/ip_basic`

3 Experimental Protocol

3.1 Datasets

The KITTI Vision Benchmark Depth Completing Data-set will be used which includes :

- Annotated depth maps data set (14 GB)
- Projected raw LiDaR scans data set (5 GB)

3.2 Coding Resource Requirements

We are planning to use the following libraries:

- opencv-python
- PIL
- pandas
- scikit-image
- matplotlib
- pypng
- numpy

The GitHub repository: `https://github.com/kujason/ip_basic`

What are we going to do differently?

There is a set of algorithmic procedures given to modify the image in image-processing in the paper and code referred. Our aim is to use the completed depth image to segment objects in the scene and compare the results with segmentation without the depth estimate.

3.3 Computational Resource and Effort Requirements

No specific computational resources are required. Only the CPU is sufficient.

The GitHub repository: `https://github.com/kujason/ip_basic`

What are we going to do differently?

There is a set of algorithmic procedures given to modify the image in image-processing in the paper and code referred. Our aim is to use the completed depth image to segment objects in the scene and compare the results with segmentation without the depth estimate.

4 Results

Processing: 999 / 999, **Avg Fill Time:** 0.01896s, **Avg Total Time:** 0.09479s

4.1 Visualization results

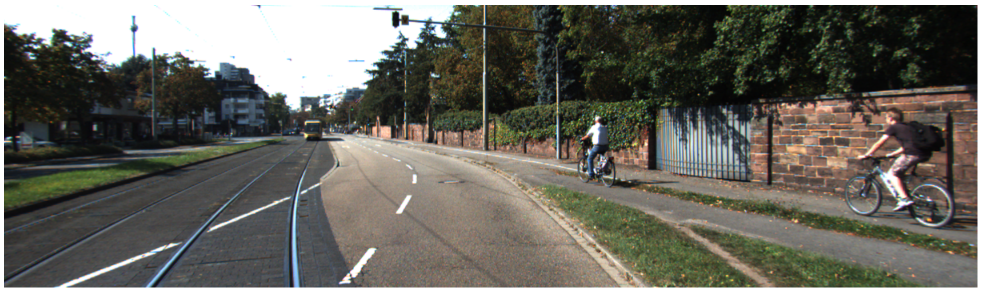

Figure 1: Original Image:1

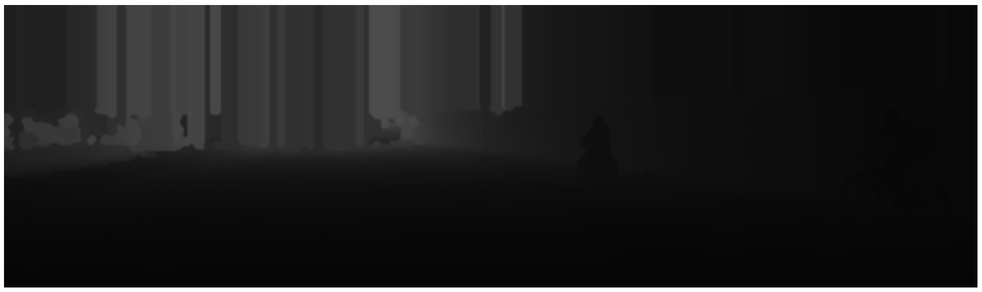

Figure 2: Depth Image:1

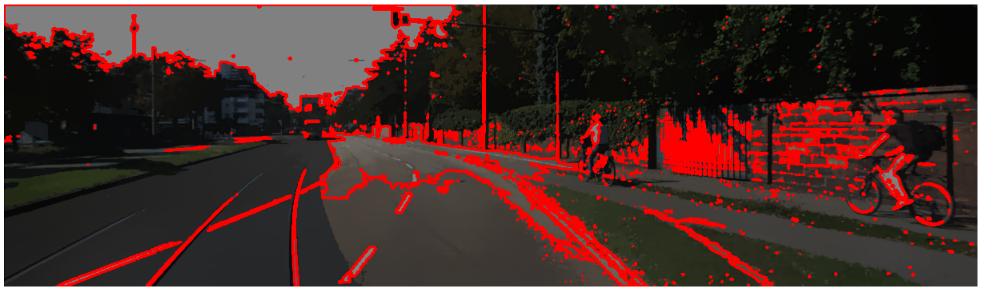

Figure 3: Segmentation on Blended Images:1

Figure 4: Original Image:2

4.2 Comparing SOTA

The State of the Art algorithms cannot segment images with depths. They are capable of performing depth estimation on images with very good accuracy. Our project is capable of superimposing original

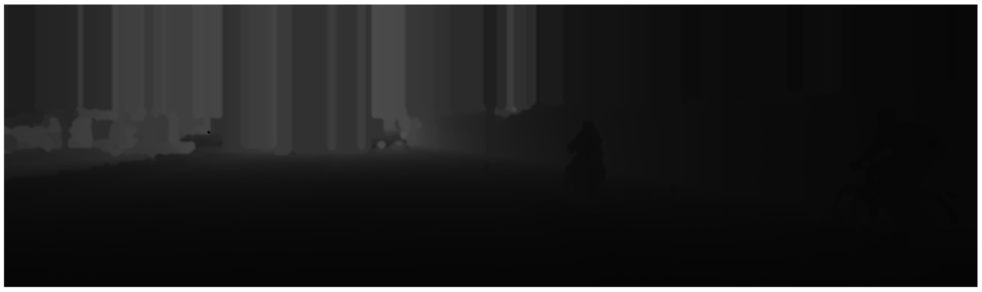
Figure 5: Depth Image:2

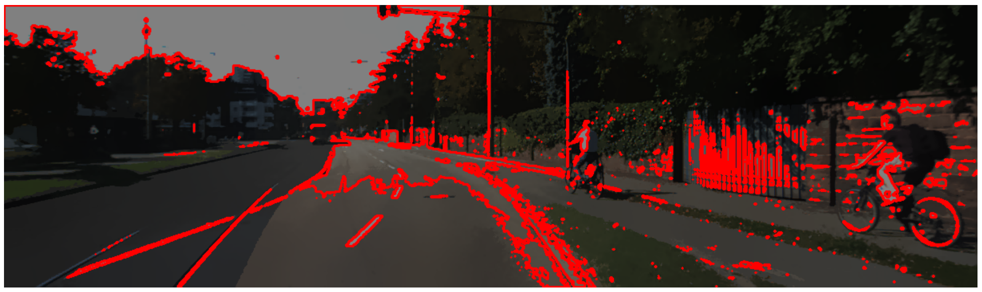
Figure 6: Segmentation on Blended Images:2

and depth images and segmenting them. Hence, it can find the depths of the segmented objects. We are also able to find the spatial relationship of the depth points in the image, which is not performed by SOTA algorithms.

4.3 Conclusions from results

We can conclude that we're successfully able to conceive depth for any points in the images. The superimposed images are much smoother with less noise.

5 Analysis

5.1 Limitations/ Advantages of the algorithm used

- It is extremely fast compared to modern techniques like neural networks
- No separate implementation would be required for image segmentation with depth
- The superimposed image is automatically smoothened and noise is reduced.

6 Discussion and Lessons Learned

6.1 Summarize what you learned from this project and how you hope it will help you in the future.

From the project, we got a deeper understanding of depth completion and image segmentation. We understood and implemented 3 different types of image segmentation techniques. Apart from we found ways to superimpose dense depth maps and original images.
This has encouraged us to work in-depth on the depth completion and superimposition of images. This summer, we are planning to work on this project and extend it to a wider scope.

7 Bibliography

[1] Zhai, Mingliang Xuezhi, Xiang. (2021). Geometry understanding from autonomous driving scenarios based on feature refinement. Neural Computing and Applications. 33. 10.1007/s00521-020-05192-z.

[2]N. Loewke, Depth-Based Image Segmentation, n.d.

[3]T. H. Dinh, M. T. Pham, M. D. Phung, D. M. Nguyen, V. M. Hoang and Q. V. Tran, "Image segmentation based on histogram of depth and an application in driver distraction detection," 2014 13th International Conference on Control Automation Robotics Vision (ICARCV), Singapore, 2014, pp. 969-974, doi: 10.1109/ICARCV.2014.7064437.

[4]N. Raghavan, P. Chakravarty, S. Shrivastava, Sim2real for self-supervised monocular depth and segmentation, ArXiv Preprint ArXiv:2012.00238. (2020).

[5]M. Danielczuk et al., "Segmenting Unknown 3D Objects from Real Depth Images using Mask R-CNN Trained on Synthetic Data," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 7283-7290, doi: 10.1109/ICRA.2019.8793744.

[6]E. Xie, W. Wang, Z. Yu, A. Anandkumar, J.M. Alvarez, P. Luo, SegFormer: Simple and efficient design for semantic segmentation with transformers, Advances in Neural Information Processing Systems. 34 (2021) 12077–12090.

[7]J. Ku, A. Harakeh, S.L. Waslander, In defense of classical image processing: Fast depth completion on the cpu, in: 2018 15th Conference on Computer and Robot Vision (CRV), IEEE, 2018: pp. 16–22.