AER 1515 - Perception for Robotics

Assignment 3

Submission guidelines: This assignment is due on Friday, December 8, 2023 at 11:59 pm. Include the following items in a `.zip` file and upload to Quercus:

- PDF of assignment solution write-up
- Python code that generated the images in your write-up
- Estimated depth maps and segmentation images for the test set (in separate folders named `est_depth` and `est_segmentation`). **Ensure your images are in .png format and saved as `<image number>.png` (i.e. 000011.png)**
- `README` detailing how to run the code and any dependencies the code has

Do not include the network weights in your final submission

3D Object Detection and Instance Segmentation

In this assignment, you are given rectified stereo image pairs from the KITTI dataset to estimate 2D bounding boxes and instance segmentations of cars.

- Calibration information for the left and right cameras is provided with the data files. The left camera is labeled as camera **p2** and the right camera is labeled as **p3**.
- Data is split into a training set and a test set. You are provided with ground truth for depth maps, 2D bounding boxes, and instance segmentation to verify your algorithm. Please present the results for the **test** set in your submission.
- You are given functions to read KITTI labels and calibration files in `kitti_dataHandler.py`. See https://github.com/kujason/avod/wiki/Data-Formats for the format of KITTI labels.

1 Depth Estimation

Similar to Assignment 2, you will estimate the depth map for the given image pairs. However, note that the given maps are **disparity** maps this time. Use these maps and the calibration image to estimate the depth maps for the left test images. Pixels without a disparity have a value of 0 in the disparity map. Assign these pixels a depth of 0 in the final depth image. Additionally, assign a depth of 0 if the depth at the pixel is greater than 80 m or less than 10 cm. We will evaluate your attached depth images for marks. You can use the ground truth provided for the train images to debug. Again, the ground truth depth map is generated using LiDAR-based depth completion so it does not cover the entire image. Include a short discussion about the quality of the produced depth images, including specific environmental factors that lead to better or worse performance.

2 2D Bounding Box Estimation

Run the provided 2D object detector detector YOLOv3 or a detector of your choice to find 2D bounding boxes for all the cars in the left images. A 2D bounding box is represented with two corner points, the bottom left corner $(x_{\text{left}}, y_{\text{bottom}})$ and the top right corner $(x_{\text{right}}, y_{\text{top}})$. For YOLOv3, tune the confidence and non-maximum suppression thresholds to obtain best results on the training set, then generate the bounding boxes for the test set. Please state the thresholds used in your write-up. You do not need to submit your estimated labels for the test set. Instead, include visualizations in the report where the detected 2D bounding boxes are overlaid on the test set images in your write-up, similar to Fig. 1. Note that you will need these estimated labels for Part 3.

Note: this portion of the assignment can be extremely straightforward as the provided YOLOv3 detector is pre-trained. You may choose to experiment with other detectors, run on a larger portion of the KITTI dataset, or train a 2D detector on the dataset. However, these will require significant GPU resources and is not required. Any experimentation or training beyond running YOLOv3 is **entirely optional** and will not be marked.

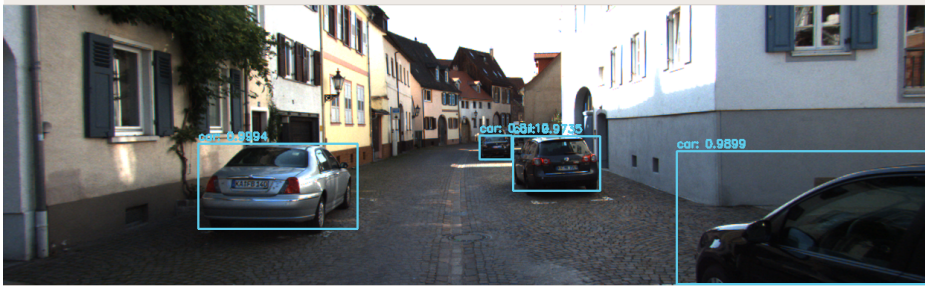

Figure 1: 2D detections

3 Instance Segmentation

In this section, you will perform instance segmentation on the left image for every detected car from the previous section. Instance segmentation means you will attempt to find all pixels inside a detected bounding box that belongs to the object. A typical method to perform instance segmentation is:

1. For every detected bounding box, compute the average depth of the object
2. Find all pixels inside each bounding box that are within a certain distance (can be adjusted) from the computed average depth

This basic approach should be able to achieve roughly 80% precision and recall on the test set, and can be fine-tuned. Full marks will be given for exceeding 75% precision and recall on the test set and scaled based on performance relative to 75%. Use the ground truth provided for the training set to tune your algorithm. We will evaluate your attached segmentation masks for marks.

If you have access to GPU and want to try out deep methods, you are welcome to use a deep learning algorithm. You do not have to show your average depth estimation results for any approach that you choose. If you follow the suggested algorithm above, please state the threshold(s) that you used. Otherwise, please provide an explanation of whatever algorithm you used to perform this task. We will evaluate your results for this part based on the segmentation results for the full image and using precision and recall. Please submit a mask for each image in the test set with a value of 0 for any car pixels and a value of 255 for any non-car pixels. The ground truth segmentation mask is labeled instance by instance where any pixels with a value less than 255 belonging to a car.

Marking guidelines:

- Code Clarity [10 pts]
- Depth Estimation [20 pts]
  - Depth images for the 5 test images uploaded in corresponding folder [15 pts]
  - Short discussion on the quality of depth images [5 pts]
- 2D Bounding Box Detection [20 pts]
  - RGB images with overlaid bounding boxes for the 5 test images in write-up [20 pts]
- Instance Segmentation [50 pts]
  - Segmentation images uploaded in corresponding folder [10 pts]
  - Explanation of the segmentation algorithm (see Section 3 for details) [10 pts]
  - Precision/recall scores [30 pts]