Object Detection and Orientation Estimation for Autonomous Driving

Jinyi Lu
Information Networking Institute
jinyil@andrew.cmu.edu

Xiaoqing Tao
Information Networking Institute
xtao@andrew.cmu.edu

Keywords: [computer vision, object detection, object orientation estimation, autonomous driving]

Instructions

Your midway report should be 4 pages long (excluding references) and include the following:

- Project title and list of group members (please follow the given template).
- **Introduction (1 – 1½ pages):** Problem and literature overview; how your idea fits into the existing work. If you already had an elaborate proposal, you may borrow this part from there, but try to make it as focused and concise as possible. If your proposal did not include enough background information, please make sure this section fills that gap (this will also save you time on preparing the final report).
- **Methods (1½ – 2 pages):** If you are developing a new method / model / architecture, this section should contain all the math, derivations, formal details, illustrations (if necessary). If you are proving theorems, assumptions and theorem formulations should be here. If your project is application driven, this section should highlight important aspects of the application and details on how your methodology addresses those.
- **Preliminary Results (½ – 1 page):** Please include experimental results you have obtained by the deadline. We expect you to have at least implemented and tested a major part of your models (maybe on some toy data) and run a few baselines. Describe how the current results align with your expectations. You may include the details on how you mined / prepared the data, if it is an important aspect of the project.
- **Final plan (½ page):** The adjusted plan for the next steps (in terms of methodology, experiments, data collection, etc.)

Additional notes

If you have changed or adjusted your project plan significantly (compared to what you have proposed), please provide a brief note on this in a subsection of the introduction.

Grading rubric

Total weight: 25% of the project grade (i.e., 10% of the course grade).

- Introduction & literature survey (10%)
- Methods (40%)
- Preliminary results (20%)
- Plan of the next steps (20%)
- Quality of writing (10%)

1 Introduction

1.1 Problem Overview

Nowadays cameras are available onboard of of almost every new car produced in the last few years. Computer vision provides a very cost effective solution not only to improve safety, but also to one of the holy grails of AI, fully autonomous self-driving cars. In this project we are planning to use deep neural networks to solve the object detection and object orientation estimation problems for autonomous driving.

There are lots of potential challenges that we need to solve for example, due to the weather, road conditons and car location, images from the car cameras will have a high-variety, which requires high robustness for our model. And besides the classical object detection task, we want to further estimate the 3D orientation from the 2D images. Last, but not least our system need to produce a good result within a limited runtime in order to be used in practise.

1.2 Dataset

The dataset that we are planning to use is the KITTI Vision Benchmark Suite [1]. It's developed for use in mobile robotics and autonomous driving research. So it contains several novel challenging benchmarks for the tasks of stereo, optical flow, visual odometry/SLAM and 3D object detection. In our project, we mainly focus on the object detection and orientation estimation task. The corresponding benchmark [1] consists of 7481 training images and 7518 test images, comprising a total of 80,256 labeled objects (up to 15 cars and 30 pedestrians are visible per image). All images are color and saved as png.

Dataset statistics

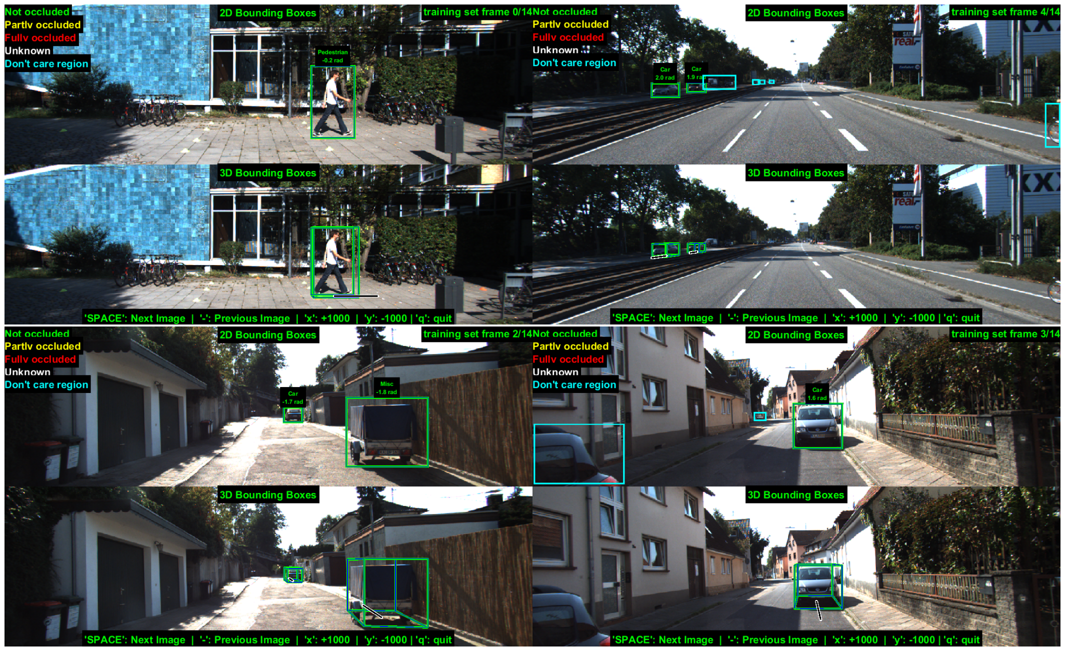

Figure 1: Some Ground Truth Examples

1.3 Evaluation

For evaluation, the benchmark is split into three parts: First, we need to evaluate the classical 2D object detection by measuring performance using the well established average precision (AP) metric as described in [2]. Detections are iteratively assigned to ground truth labels starting with the largest

[1]`http://www.cvlibs.net/datasets/kitti/eval_object.php`

overlap, measured by bounding box intersection over union. True positives are required to overlap by more than 50% and multiple detections of the same object are counted as false positives.

Second, we assess the performance of jointly detecting objects and estimating their 3D orientation using a novel measure which is called the average orientation similarity (AOS) [1] and is defined as:

$$AOC = \frac{1}{11} \sum_{r \in \{0,0.1,..,1\}} \max_{\tilde{r}:\tilde{r} \geq r} s(\tilde{r}) \tag{1}$$

Here, $r = \frac{TP}{TP+FN}$ is the PASCAL object detection recall, where detected 2D bounding boxes are correct if they overlap by at least 50% with a ground truth bounding box. The orientation similarity $s \in [0,1]$ at recall r is a normalized ([0..1]) variant of the cosine similarity defined as

$$s(r) = \frac{1}{|D(r)|} \sum_{i \in D(r)} \frac{1 + cos\Delta_{theta}^{(i)}}{2} \delta_i \tag{2}$$

where $D(r)$ denotes the set of all object detections at recall rate r and $\Delta_{theta}^{(i)}$ is the difference in angle between estimated and ground truth orientation of detection i. To penalize multiple detections which explain a single object, we set $\delta_i = 1$ if detection i has been assigned to a ground truth bounding box (overlaps by at least 50%) and $\delta_i = 0$ if it has not been assigned.

Finally, we will also evaluate pure classification (16 bins for cars) and regression (continuous orientation) performance on the task of 3D object orientation estimation in terms of orientation similarity.

1.4 Related work

Traditional methods for object detection usually utilize image features,such as SIFT and HOG. We investigated three methods utilizing deep convolutional network in object detection.

1.5 R-CNN: Rich feature hierarchies for accurate object detection and semantic segmentation

The first challenge for object detection is how to implement localizing within an image. The new method proposed in this paper[3] combines CNN with region proposals. So it is called R-CNN. The general detection process is to extract about 2000 region proposals for each input image. Then utilize CNN to compute a fixed length feature for each region proposal. Finally utilize linear SVM to classify each region.

The second challenge is how to train a large CNN using limited labeled data. This paper proposed to pre-train CNN on a large auxiliary data set, then continually train on a small data set. This method significantly improved the accuracy of objection detection compared with feature learning models. But training is expensive in space and time, and detection is also slow at test time.

1.6 Fast R-CNN

This paper[4] talked about how to train a detection network faster. R-CNN is very slow because it extracts feature for each region proposal and there are many duplicate computations. The process could be faster if we share computation.

This paper proposed to modify R-CNN's architecture by taking an image and multiple regions of interests as input. Region proposal method usually depends on Selective Search. Each region of interest is pooled into a fixed-size feature map and fully connected layers are used to extract features. There two output vectors: softmax probabilities and per-class bounding-box regression offsets.The second one is to reduce mislocalization.

1.7 Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

Fast-CNN achieves near real-time rates using very deep networks, when ignoring the time spent on region proposals. Select Search takes about 2 seconds per image on the CPU. Faster R-CNN[5] adds a fully convolutional network to compute region proposals directly. Then use the same detector on the proposed regions as what Fast R-CNN does.

1.8 Yolo

2 Methods

3 Preliminary Results

In this section, we will discuss some of our preliminary results. We first randomly split all 7841 labeled images into two parts: training set (4953 images) and validation set (2538 images). For now, we have conducted three experiments on object detection, including using a pre-trained yolo model, using a re-trained tiny yolo model based on our training set and using a pre-trained Faster-RCNN model.

In terms of evaluating the object detection results, KITTI benchmark requires a minimum overlap of 70% for cars and 50% for pedestrians. It also has three kinds of difficulties: easy, moderate and hard. Difficulties are defined as follows:

Difficulty	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40 Px	Fully visible	15%
Moderate	25 Px	Partly occluded	30%
Hard	25 Px	Difficult to see	50%

Table 1: Different Difficulties Requirements

The pre-trained yolo model is trained on Pascal VOC 2012 and 2007 dataset [2]. It supports about 20 object categories including car and people (corresponding to pedestrian). However it don't support cyclist category. So in all the following experiments, we only evaluate the performance for cars and pedestrians.

re-trained tiny yolo

The pre-trained Faster-RCNN model is trained on Pascal VOC 2007 dataset [3]. It is able to detect cars and pedestrians and also lacks of supports for cyclists.

[2] `http://pjreddie.com/darknet/yolo/`

[3] `https://github.com/rbgirshick/py-faster-rcnn`

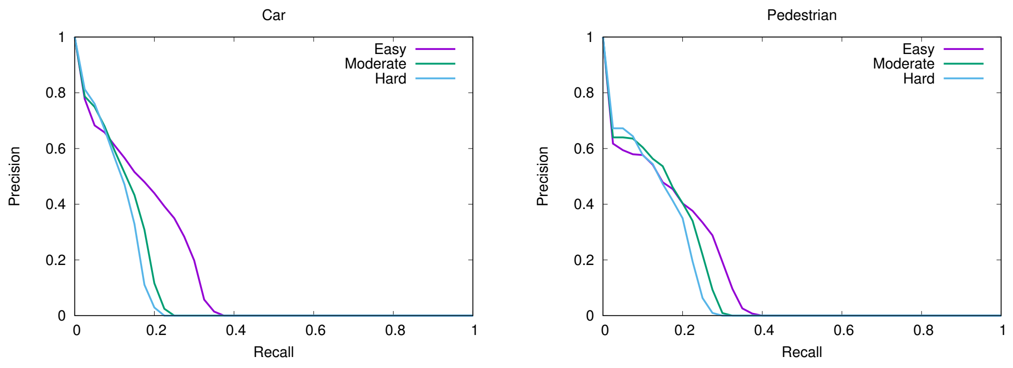

Figure 2: Using a Pre-Trained Yolo Model

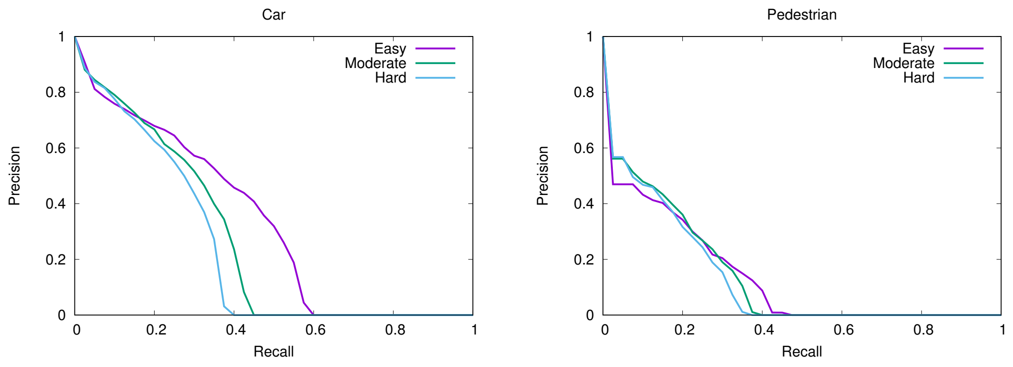

Figure 3: Using a Tiny Yolo Model Trained on Our Dataset

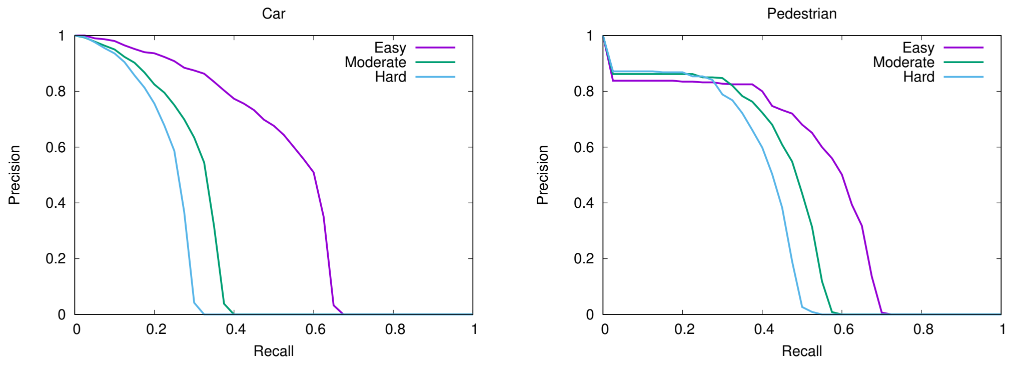

Figure 4: Using a Pre-Trained Faster-RCNN Model

Method	Easy	Moderate	Hard
Pre-Trained Yolo	0.204438	0.155358	0.145101
Tiny Yolo	0.344293	0.291823	**0.257863**
Pre-Trained Faster-RCNN	**0.522754**	**0.309875**	0.248554

Table 2: Average Precision on Car Detection

Method	Easy	Moderate	Hard
Pre-Trained Yolo	0.197457	0.183323	0.175022
Tiny Yolo	0.187898	0.184558	0.176187
Pre-Trained Faster-RCNN	**0.498992**	**0.429862**	**0.377569**

Table 3: Average Precision on Pedestrian Detection

As we can see from the precision-recall curve and the average precision results. For pedestrians, Faster-RCNN outperforms yolo models a lot in all three difficulity levels. And a re-trained tiny yolo model on our dataset shows no improvement compare to the pre-trained yolo model. For cars, Faster-RCNN outperforms yolo models in easy level. And our re-trained tiny yolo model shows great improvement in all three difficulity levels compare with the pre-traiend yolo model.

4 Project Plan

We plan to implement 2D object detection using faster R-CNN by the second milestone. After that, we will focus on 3D object detection. We may also try YOLO[6] method and see if there are any improvement. Detailed plan depends on time and compute power.

References

[1] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[2] M Everingham, L Van Gool, CKI Williams, J Winn, and A Zisserman. The pascal visual object classes challenge 2011 (voc 2011) results (2011). In *URL http://www. pascal-network. org/challenges/VOC/voc2011/workshop/index. html*, 2010.

[3] Trevor Darrell Jitendra Malik Ross Girshick, Jeff Donahue. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Conference on Computer Vision and Pattern Recognition(CVPR)*, 2014.

[4] Ross Girshick. Fast r-cnn. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[5] Ross Girshick Jian Su Shaoqing Ren, Kaiming He. Faster r-cnn: Towards real-time object detection with region proposal networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[6] Ross Girshick Ali Farhadi Joseph Redmon, Santosh Divvala. You only look once: Unified, real-time object detection. In *arXiv preprint arXiv:1506.02640*, 2015.