Gate Detection

Philipp Duernay

February 1, 2018

1 Recap

In the last meeting from 09.01.2018 several next steps were defined:

a. **Data Generation**. In the first experiments the data generation was limited to one gate, placed on randomly selected backgrounds from the Pascal VOC dataset. As these also contained a lot of people, the background was often not very realistic. In the new dataset more realistic backgrounds should be chosen. The new data should also contain a various amount of gates placed at different positions.

b. **Evaluation**. In the first experiment the performance was evaluated in terms of mean average precision/ precision-recall. A detection was marked "correct" when the intersection-over-union between the true and the predicted bounding box exceeded 0.4. This allows limited interpretability as the false positive rate is not covered. An intersection-over-union of 0.4 might be to inaccurate if we want to use the bounding box for navigation. In the this step further evaluation should be done.

c. **Effect of Pose** In the last evaluation no real effect of the camera pose was measurable. This is quite unlikely and should be investigated.

d. **New Methods** So far only the Yolo-Object[2] detector was evaluated. We also want to compare the results to other methods.

2 New Datasets

To place multiple gates in one image 3D-models of the gates are placed randomly in a 3D environment. That way it is made sure the gates don't overlap in a physically impossible way. Then shots are taken from various positions in the scene. An example can be seen in Figure 1.
For the new dataset data was gathered from different datasets that contained images of cities, roads, traffic signs and the like. Also some pictures of fences, gates and in industrial environments were added. Thus more human like structures with shapes similar to gates should make the set more realistic/challenging. An example can be seen in Figure 1.

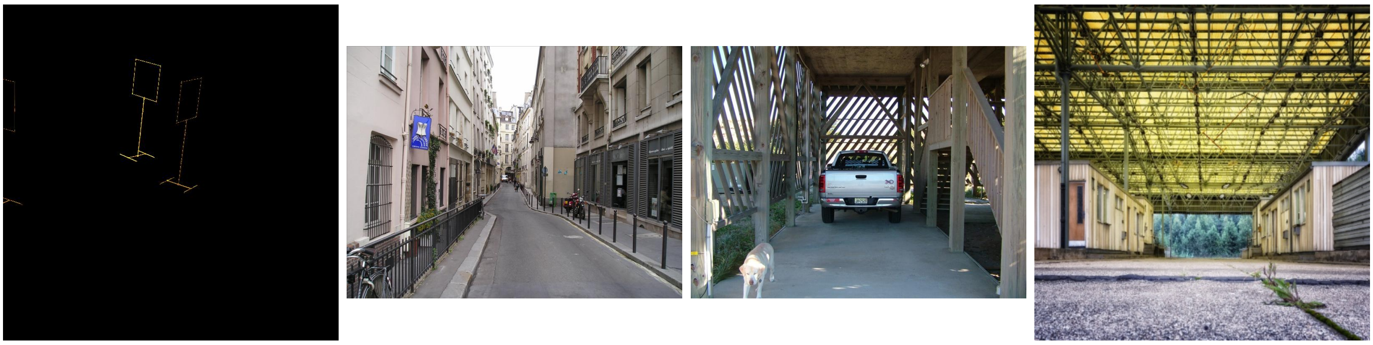

Figure 1: New Data

3 New Methods

For comparison the smaller version of Yolo a network with 9 layers was tested. Unfortunately, the results are just random. The Single Shot Multibox Detector [1] is in the making but not fully implemented yet.

4 Experiments

Several experiments were conducted. Some example outputs can be seen in Figure 2 and Figure 4.

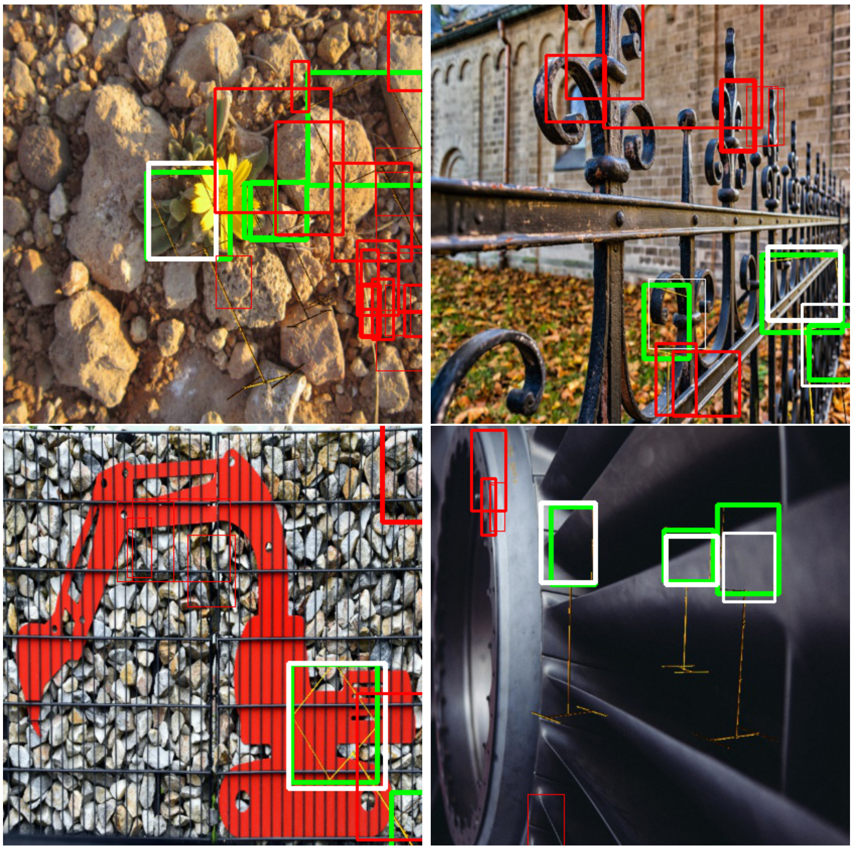

Figure 2: Various outputs of Yolo-Object-Detector. True labels in green, correct (IoU ≥ 0.4) detections in white, wrong predictions in red. The thickness of the lines is based on the detectors confidence (4 levels). The highest confidence between 0.75 and 1 corresponds to the thickness of the ground truth boxes (green)

4.1 Bounding Box Detection

In this experiment the detectors are tested on various backgrounds, with various amount of gates. Performance is evaluated in terms of "bounding box overlaps". If the intersection-over-union between true and predicted box exceeds a certain threshold, the detection is marked as "correct". The precision-recall plot can be seen in Figure 3. Localization Error is calculated for all correct boxes. That is the absolute difference in pixels between the predicted and true box center, width and height. The mean localization error is:

$$c_x = 4.56 +/- 4.89 \quad c_y = 6.19 +/- 6.41 \quad w = 10.35 +/- 10.93 \quad h = 12.23 +/- 15.58$$
$$\epsilon_L = 8.33 +/- 10.78$$

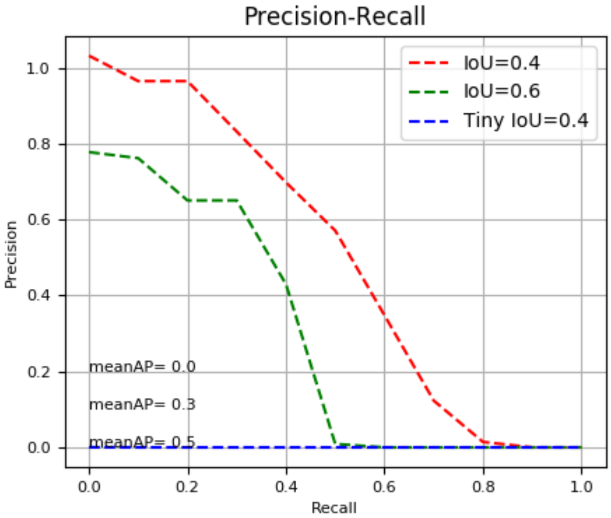

Figure 3: Precision-Recall over 1000 samples

4.2 Single Gate Detection

In this experiment only one gate is placed in the image and only the predicted box with the highest confidence is taken into account. The camera ”flies” around the object. The trajectory is kept pretty simple since the first experiments did not motivate to make it more challenging.

Examples are shown in Figure 4 or in the attached videos. Boxes with a confidence value of more than 30% are displayed. The box with the highest confidence is displayed in blue the true box in green. Hence, we want the blue box to be where the green one is. Its best to watch them first to get some impression.

In Figure 5 the localization error is displayed to see how wrong the predicted box is. Interesting is that sometimes the gate is detected at a high distance but when it comes closer it gets not detected anymore. Sometimes also a step towards the gate can cause the detector to produce a bounding box at a random position. Right in front of the gate the boxes are pretty inaccurate. In the video one can also see how more and more (random) boxes are predicted the closer we get to the gate. The model also seems to like the upper left corner. This also needs further investigation.

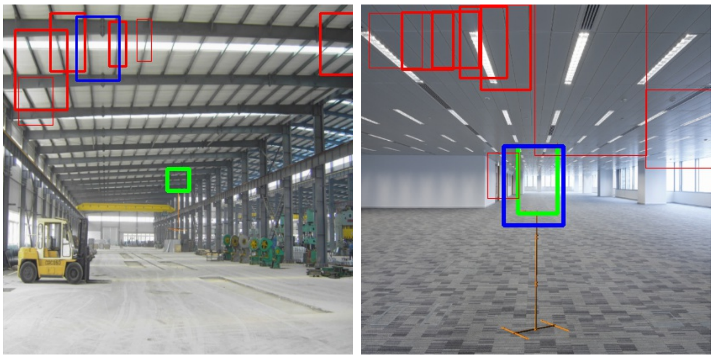

Figure 4: The two backgrounds for the ”Single Gate Detection”. Boxes with a confidence value of more than 30% are displayed. The box with the highest confidence is displayed in blue.

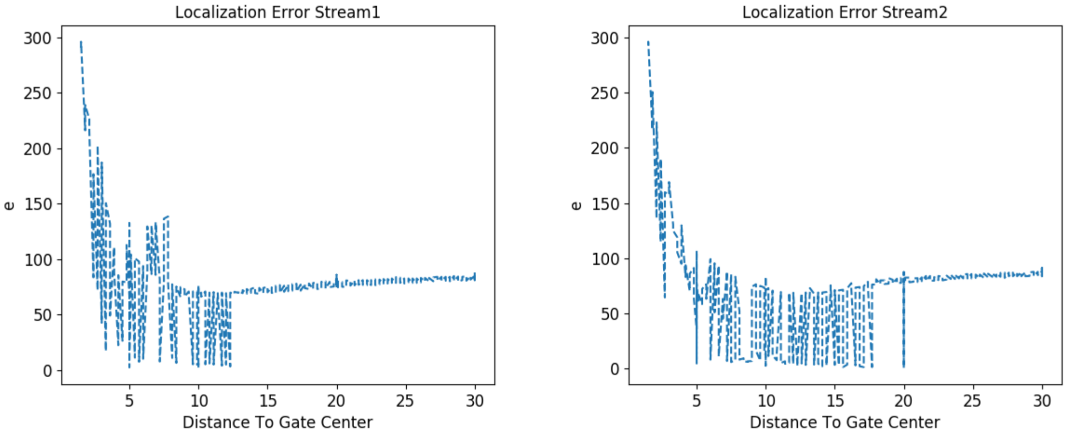

Figure 5: The mean localization error of the box with the highest confidence for both streams left and right respectively to Figure 4. The localization error is the mean distance between all 4 coordinates (center x, center y, width and height). A correct bounding box usually has a localization error between 0 and 20. Hence, everything above is pretty bad. Note that the localization error is only calculated when a box is actually predicted.

5 Thoughts & Conclusions

a. **Performance Drop.** The performance dropped quite a bit compared to the last evaluation. There are several potential reasons for that:

   - In the last set there was only one gate which was always close to the center of the image. The model ”just” needed to predict a bounding box somewhere in that region and it was pretty likely to be correct. This was quite independent of direction or distance.
   - In the last set there were only a few images without a gate. Hence, the learning problem basically simplified to finding **the** gate in the image instead of actually detecting various amounts of gates.
   - In the last set the backgrounds were easier.

b. **Results on Video.** Assuming we would use the box with the highest confidence for navigation we would crash pretty quickly. This happens even for the quite simple background in the second video. The model seems to predict on strong contrasts rather than color. For training Yolo data augmentation is used. That includes randomly changing the color of the images. Although we would loose generalizability we could reduce the augmentation and thus potentially improve detection performance.

c. **Results at Low Distance.** At low distances the model performs not good. A reason could be the anchor boxes used by Yolo. The model seems to perform best at distances where the anchor boxes ratios fall into the ratios of the gate bounding boxes. SSD uses additional convolutional layers at higher scales to tackle this problem. This could also work for the given setting.

6 Next Steps

a. **Get Real(er) Data.**

b. **Visualize what model has learned.**

c. **Implement SSD**

d. **Reading.**

References

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single Shot MultiBox Detector.

[2] C. Szegedy, S. Reed, P. Sermanet, V. Vanhoucke, A. Rabinovich, M. Simon, E. Rodner, J. Denzler, J. Redmon, A. Farhadi, S. Ioffe, C. Szegedy, W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-y. Fu, A. C. Berg, S. Ioffe, V. Vanhoucke, A. Alemi, S. Reed, P. Sermanet, V. Vanhoucke, A. Rabinovich, J. Shlens, Z. Wojna, F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer, K. He, X. Zhang, S. Ren, J. Sun, T. Chen, and C. Guestrin. YOLO9000: Better, Faster, Stronger. *Data Mining with Decision Trees*, 7(3):352350, 2016.