David
Esteso

Critique of the Ground Truth

Ground Truth

The ground truth provided for the pedestrian crossing detection system consists of two lines, one above and one below the crossing. While it works well for defining the upper and lower boundaries, a minor critique is that these lines do not address the lateral edges. Consequently, while the area within those lines generally represents the crossings accurately, it may also include adjacent areas like sidewalks. The only instance I find questionable is in image 27, where the rectangular area to the left could be considered part of the solution. Overall, the ground truth is good for its purpose and effectively captures the pedestrian crossings.

System flow

Our system receives an input image in RGB format with a resolution of 504 x 378 pixels. It starts by converting the images to **grayscale** and then applies **histogram normalization**. This first step simplifies the data and is necessary for the next step. Histogram normalization enhances image contrast by stretching the intensity values to cover the full scale from 0 to 255. ensures consistency in images, reducing the influence of varying lighting conditions and improving the reliability of pedestrian crossing detection across different environments. This yields a grayscale image.

Next, we applied a filter to eliminate noise from the image while prioritizing the preservation of distinct and sharp edges. For this reason, **median filtering** was the most suitable choice for our task. Median filtering replaces each pixel's value with the median of the pixel values in its neighborhood, determined by the kernel size. We opted for a kernel size of 5 to maintain the main features of the image and avoid excessive blurring, as a value of 5 is typical, leaning lower. Additionally, median filtering is particularly effective at removing salt-and-pepper noise, which can resemble the irregularities often found in asphalt. We keep working with a grayscale image.

Input Grayscale Hist. normalization Median blur

The next step in the system is to **binarize the image**. In our case, we set the binary threshold to 230, which effectively highlights the crosswalks while minimizing other irrelevant details. Since we are specifically looking for pedestrian crossings, which are white and have a high contrast against the road, we can afford this high increase in the threshold.

The last step yields a binary image. We then used a **contour segmentation** technique that effectively identifies the boundaries of distinct shapes. It's unnecessary to apply an edge detection technique because the edges in a binary image are trivial and would yield nearly identical results with or without this technique. This process has produced a vector containing each identified contour. For these contours, we applied a **polygonal approximation technique**, which is based on the Ramer–Douglas–Peucker algorithm, to search for those that have **four sides**. This technique approximates the shape of a contour by reducing the number of points, making it ideal for detecting rectangles.

With the yielded contours, we proceed with an **analysis of rectangularity and elongatedness** to ensure we are exclusively detecting the rectangles that comprise them. A perfect rectangle would ideally have a rectangularity of 1, calculated as the ratio of the contour area to the area of the **minimum bounding rectangle** that encloses it, and an a certain degree of elongation. It is captured by the larger eigenvalue representing the "main direction" where most of the shape's area is spread (the long side), while the smaller eigenvalue captures the width (the short side). When we take the ratio of these two values, we get a higher number if the shape is very elongated (a long side compared to a short side) and a number closer to 1 if the shape is more balanced (sides are closer in length, like a square).

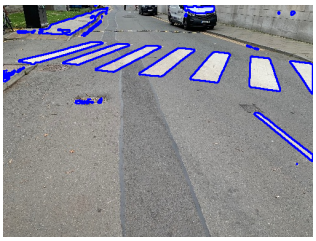
In our analysis, we have set thresholds for rectangularity above 0.5 and elongation values greater than 1.5. The choice to lower the rectangularity threshold to 0.5 reflects our understanding that the detected rectangles may not be perfect; they are often tilted and may not fully occupy their minimum bounding rectangle, especially given the noise surrounding each rectangle that can distort its exact rectangular shape. We have chosen an elongation threshold that is not overly strict, as there is no standard way to view crosswalks. Perspective can vary considerably, leading to substantial differences in elongation. In addition, we apply a filter to remove contours with an **area** smaller than 200. This size is large enough to eliminate noise but not so large that we lose important information. We keep working with contours.

At this stage, the process involves segmenting the image into a **grid** to organize and analyze detected contours based on their spatial locations. The image is divided into **six** horizontal sections. This horizontal division is intentional, as from a car's perspective, a pedestrian crossing is typically viewed horizontally or partially horizontally. The decision to divide the image into six sections balances having enough lines to filter out misaligned contours without over-segmenting, which could risk discarding potential candidates.

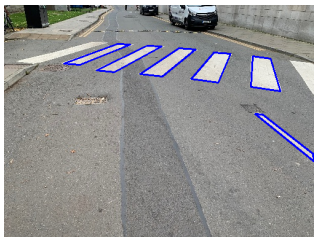
Once the contours are organized, the focus shifts to **identifying groups** that meet certain criteria. Specifically, only those **clusters containing a significant number of contours (at least three)** are considered for further analysis. This threshold of three is reasonable, as fewer contours may indicate noise, while a larger number could be overly strict and may lead

to excluding valid detections. The result is a refined collection of contours that are more likely to represent meaningful structures in the image.

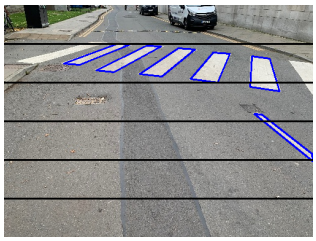
Binary



Contours



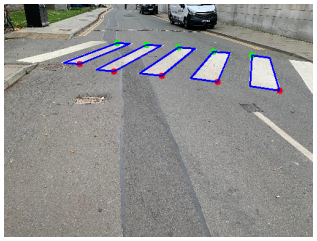
Filtered contours



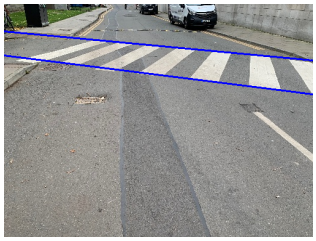
Grid

At this stage, the final step involves determining the solution by analyzing the contours identified as pedestrian crossings. For each contour, we search for the highest and lowest points, which correspond to the upper and lower edges of the crosswalk. To formalize this, we employ a **regression** technique based on the **least squares method**. This approach allows us to derive two lines that effectively define the upper and lower margins of the pedestrian crossing.

Once these lines are calculated, they are drawn onto the final processed image, providing a clear visual representation of the identified pedestrian crossing. **The result is an image with defined boundaries for the pedestrian crossing**

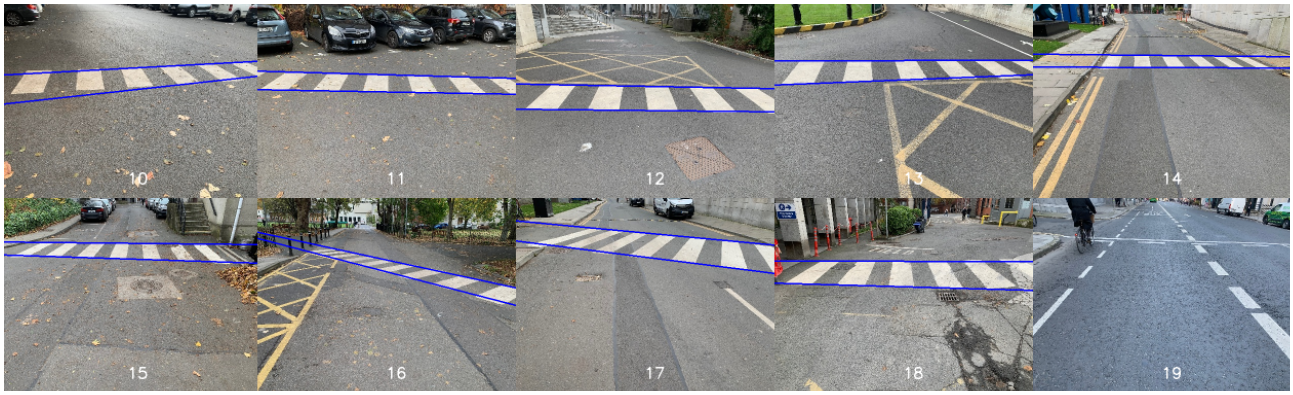


Regression points



Output

Evaluation First set



Results for images 10 to 19

Now, the problem is to know how good the identification of the lines, which are responsible for the setting of the upper and lower bounds of an area of crosswalk. Let's note that the lines

themselves do not consider the lateral boundaries. One of the things that could be measured would be IoU, which indicates how well the area our system generates overlaps with the ideal area.

We considered further pursuing the calculation of the average line distance, but this strategy has two disadvantages: it would mean measuring the distances in areas that do not have a crosswalk even though we may have the ground truth data present. This is the same problem we have with IoU. Second, determining a good threshold for such distance calculations can be very tricky. IoU, on the other hand, is a much more standardized metric that is also easier to interpret. A score of 0.7 in IoU clearly conveys how much overlap exists between the predicted and ground truth areas. In contrast, a distance measurement between the lines of 50 pixels may not provide as intuitive an understanding of the actual performance of our system. We thus find it more readable and hence better adapted to our needs within the evaluation.

To face the problem of the lateral boundaries, another possible concept could be to give more weight to the IoU in the central areas of the crosswalk, not at the lateral edges from where we might not view it. Again, this is a one-dimensional approach.

For instance, as one may well consider from the ground truth, in most of the images analyzed, the lines seem to edge of the image as a bounding box; this weighing is thus a lot less relevant. Essentially, **this issue stems from the very nature of the ground truth itself**, and there's no entirely flawless way to address it. For instance, in image 26, the most effective approach to performance evaluation would involve manually editing the ground truth to concentrate on the areas that truly matter. Without making arbitrary changes to the ground truth, we cannot accurately measure the lines in locations devoid of a crosswalk. Furthermore, without this modification, there is no method available that could avoid measuring areas that are not genuinely relevant to the evaluation, which could lead to misleading conclusions about the system's performance. It's clear that with IoU, we will get high values if we've done well, and those values will decrease if we haven't aligned the lines with the ground truth, so it's a metric that cannot be considered totally incorrect.

This is why, to evaluate the validity of the results presented above and determine if they are considered valid, **we will use Intersection over Union**. As mentioned, it is calculated by determining the intersection area between the ground truth and the obtained solutions, and then computing the union area to perform the ratio of these values. The intersection is identified by counting the overlapping pixels, while the union area is derived by combining the individual areas and subtracting the intersection area.

Here is a worked example. Substituting the values obtained from image 10, wich are also shown in the images below:

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} = \frac{18,644}{20,664 + 22,176 - 18,644} \approx 0.77$$

This approach allows us to quantitatively assess the accuracy of the obtained results relative to the ground truth. We'll use an IoU threshold of 0.5 for a result to be considered valid, following the standard practice. However, we could increase this threshold if we wanted to make the system more stringent.

Image	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**Average**
IoU	0.77	0.96	0.93	0.93	0.88	0.91	0.83	0.93	0.91	-	0.89

Table 1: IoU for images 10 to 19 with average value

In Table 1, we present the IoU for each image set. As we can see, we have obtained a value of more than 0.5 for all 10 images. The value for image 19 has not been calculated because

Area of the ground truth, result and intersection of Image 10

there is no area to detect, and we have not detected anything either. With this, we calculate the confusion matrix and the metrics of precision, recall, F1, and accuracy in Table 2.

Accuracy is the ratio of correctly predicted instances among all predictions. Precision is important for minimizing false positives, indicating the proportion of positive predictions that are correct. Recall measures the proportion of true positives among actual positives and is crucial for avoiding false negatives. The F1 Score represents the harmonic average of Precision and Recall, making it useful when both metrics are relevant. Finally, Specificity is the ratio of true negatives to the sum of true negatives and false positives, measuring how well the model avoids false alarms.

	Pred Pos	**Pred Neg**
Real Pos	9	0
Real Neg	0	1

Metric	**Value**
Precision	1.00
Recall	1.00
F1-Score	1.00
Accuracy	1.00
Specifity	1.00

Table 2: Confusion Matrix and Metrics Calculation

In conclusion, the system demonstrates excellent performance with the initial set of images, meeting high accuracy and reliability standards for the evaluated threshold.

Evaluation Second set

Results for images 20 to 29

We used the same evaluation methodology to assess performance on the validation set, Set 2. Here, the results were not as strong, as we observe a false negative in image 25, indicating

Image	20	21	22	23	24	25	26	27	28	29	Average
IoU	0.91	0.90	0.80	0.95	0.92	0.00	0.79	0.97	-	-	0.78

Table 3: IoU for images 20 to 29

that we failed to capture the relevant rectangles. In future sections, we will analyze possible causes for this discrepancy.

With these data of Table 3, we calculate the confusion matrix and metrics, shown in Table 4. We have not calculated the IoU for images 28 and 29 for the same reason as with image 19: there is nothing to detect, and we have not detected anything. As previously explained, the recall metric is crucial for avoiding false negatives, and this, along with accuracy, has been the most impacted.

	Pred Pos	Pred Neg
Real Pos	7	1
Real Neg	0	2

Metric	Value
Precision	1.00
Recall	0.875
F1-Score	0.93
Accuracy	0.90
Specifity	1.00

Table 4: Confusion Matrix and Metrics Calculation

Global Evaluation

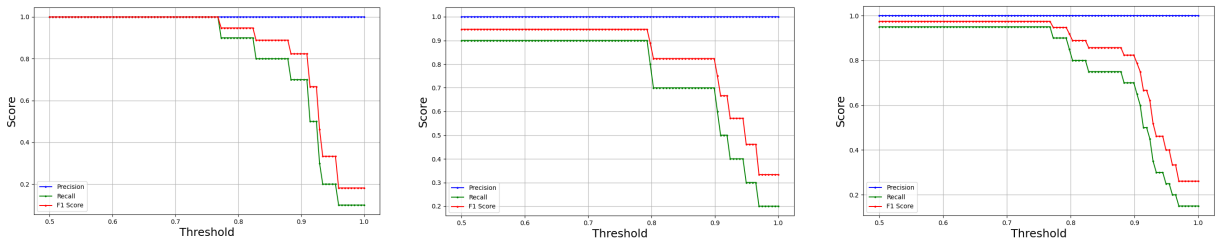
	Pred Pos	Pred Neg
Real Pos	16	1
Real Neg	0	3

Metric	Value
Precision	1.00
Recall	0.94
F1-Score	0.97
Accuracy	0.95
Specifity	1
Average IoU	0.84

Table 5: Global confusion Matrix and Metrics Calculation

The system demonstrates excellent performance with a precision of 100%, indicating that all predicted positive cases are true positives. The recall of approximately 94.1% shows that the system effectively identifies the majority of actual positive cases, with only one false negative. The F1-score of about 96.8% reflects a strong balance between precision and recall, while the accuracy of 95% confirms the overall reliability of the system. Overall, these metrics indicate that the system is highly effective in its tasks.

The metrics versus threshold graph shows a clear trend that is in line with our previous analysis of the metrics. In this case, precision remains at 1, which means the system always makes correct positive predictions at lower thresholds. However, beyond a threshold of about 75%, both recall and F1 scores start to drop together. Such a tendency might mean that,



Precision, recall, and F1 scores against the threshold for sets 1, 2, and both

although very selective, the system starts to omit relevant instances and thus true positives decrease.

This pattern is not only evident in the training set but also in the validation set, reinforcing the notion that the model may have difficulties identifying all relevant cases as the threshold increases. In this case, we do not see any trade-off between recall and precision: there are never any false positives, so the precision remains at 1, which really points out the difficulty of balancing these metrics in our specific model setup. We can conclude that we have a robust system that avoids false positives.

What fails and how to improve it

The results suggest that our system can be improved in its ability to identify rectangles. The false negative indicates that we may be too strict in how we filter the rectangles, leading to the loss of relevant information. Therefore, after analysis, we have found that a significant improvement for the system lies in refining the filtering process related to the number of edges by employing contour approximation techniques. Requiring exactly four sides, although theoretically ideal, does not reflect reality accurately. We can now accept shapes with three to five sides, which allows for greater flexibility in our filtering criteria. Thus, expanding the margin both upwards and downwards enhances the results, all while maintaining the high precision of our system in avoiding false positives.

Results of the improved system

As shown above, we have successfully developed a system that achieves a precision of 100%,

with 17 true positives and 3 true negatives, resulting in all metrics being at 1 and an average IoU of 0.9. This reflects excellent performance and represents a significant improvement over the initial design. However, it is essential to test the system under a wider variety of scenarios to draw conclusive results. A thorough evaluation across diverse conditions will ensure the robustness and reliability of our system, enabling us to confidently assert its effectiveness in real-world applications.

It is noteworthy that the performance for images like image 27 has decreased, likely due to our approach now considering more rectangles, leading to the identification of one that doesn't belong to the ground truth. While broadening our detection criteria can capture more valid shapes, it also risks including false positives, which may affect overall accuracy.

Weaknesses and Strengths

The program's main weakness lies in its assumption that it will only identify rectangles associated with pedestrian crossings. If it fails to distinguish between actual crossings and noise, it may incorrectly include irrelevant rectangles, causing inaccuracies. On the other hand, excessive filtering in the second set might result in failing to detect any valid crossings.

Additionally, highly diagonal crossings may lead to misclassification, as the strict grouping criteria could separate parts of the same crossing. The presence of a pedestrian on the crosswalk also likely hinders detection.

Lastly, the absence of color filters in our method is a drawback. While the binarization process benefits from the white color, incorporating the understanding that the desired rectangles are generally white against a gray asphalt background could enhance detection.

In fact, we tested our system in low-light situations with and without pedestrians. The presence of pedestrians and their shadows often hindered detection, resulting in false negatives. Without it, the system initially struggled in low-light conditions, but after adjusting the binary threshold to a slightly lower value of 210, we observed significant improvement, as shown in the image. This suggests we may have overfitted to the initial set of images and relied too heavily on the binary threshold. This experience highlights the importance of adaptability in our strategy and confirms that histogram normalization is a beneficial practice, due to the fact that we have done a slight change.

Results in new conditions

Our approach has several strengths, such as the absence of false positives and false rectangles, which reinforces the system's reliability under tested conditions. Additionally, it demonstrates computational efficiency, processing the image five times—during grayscale conversion, histogram normalization, median filtering, binarization, and contour segmentation—each requiring a complete image pass. In contrast, contour analysis and grouping are computationally negligible, operating only on detected contours. Furthermore, by avoiding techniques like Mean Shift, we maintain efficient processing times, preventing delays that could extend to seconds across the image set.