POLITECNICO DI TORINO

Master degree course in Mechatronic Engineering

Master Degree Thesis

Analysis of environmental conditions on the performances of autonomous driving algorithms

Advisor
prof. Massimo VIOLANTE

Candidate
Giacomo BARBI

October 2019

Ringraziamenti

Ai miei genitori, per avermi sostenuto in ogni momento, per l'educazione che mi hanno dato e per avermi permesso di inseguire sempre le mie ambizioni. A mia sorella, per essere stata da sempre un esempio ed una fonte di ispirazione. Alla mia famiglia, per tutto il sostegno e l'amore che mi ha sempre dato. Agli amici, a quelli di sempre ma anche a quelli che ho conosciuto durante questo percorso, per esserci sempre stati nei momenti di gioia così come in quelli di difficoltà. Al mio relatore, per avermi guidato durante questo lavoro conclusivo. In ultimo, ma non per ordine di importanza, ad Eleonora, per tutto l'aiuto che mi ha dato ma soprattutto per essere stata una compagna meravigliosa sin dal primo momento in cui è entrata nella mia vita.

Contents

List of Figures

Part I

Introduction

In the last years autonomous driving has been, and continues to be, one of the most trending topics in the automotive environment. The progresses performed in this field are continuous and the main car manufacturers are pushing to overcome all the limits of the current technologies through a huge amount of investments. Vision systems play a key role inside this process of driving automation, for this reason during this work will be performed an accurate analysis about the effects of the environmental conditions on the performances of two of them. The two systems that will be analyzed are a lane detection system and an object detection system; this choice is mainly due to the fact that these two systems allow to implement some crucial functions for autonomous driving. For example the lane detection system can be used to implement the automatic lane keeping, while the object detection system can be used to implement a lot of other functions like the emergency braking or the recognition of road signs and pedestrians. Moreover, since the two systems are based on two different techniques, i.e. image processing and neural networks, this choice allowed to analyze a wider range of technologies. The lane detection system has been implemented in Matlab® using the same working principle of the GOLD system developed by Massimo Bertozzi and Alberto Broggi. The object detection system, instead, is YOLOv3, which is the most powerful algorithm for object detection in real time currently available, it uses neural networks to detect objects and has been developed by Joseph Redmond et al. .

The validation procedure has been done through a series of tests performed using specific datasets, which have been developed taking into account the following parameters: lighting condition, presence of defects and weather conditions. The lighting conditions were evaluated through a set of images taken in different moments of the day, allowing to consider different orientations and amounts of light. Then a synthetic dataset has been obtained by modifying appropriately these images in order to evaluate also the effects of micro and macro defects and of different weather conditions. Moreover, in addition to these tests, some corner cases have been considered in order to evaluate the performances in some infrequent but particularly critical conditions.

The main aim of this work is to evaluate how and if these parameters influence the performances of the systems under test, in order to understand what must be improved and what, instead, has reached a sufficient level of maturity.

Part II

Bibliographic analysis and state of the art

Chapter 1

Detection of road features through image processing

1.1 Introduction

The ability of understanding the characteristics of the road we are driving in is a fundamental feature to achieve autonomous driving. Computer vision represents a key tool for this purpose since it allows to extract many road features through the execution of specific algorithms. Inside this chapter the basic concepts of computer vision will be explained together with the analysis of some interesting ADAS implementations.

1.2 Image processing and edge detection

Artificial vision is performed through a hierarchical organization which is composed by the following steps: perception, pre-processing, segmentation, description, recognition and interpretation. Perception is the process where the input image is acquired by means of specific sensors, pre-processing is the phase where the input image is prepared for the following steps, it deals with noise reduction and detail enhancement. After the pre-processing, the image is divided into objects of interest through the segmentation phase, then the features of each object are extracted during the description phase. Depending on these features objects identification is performed by means of the recognition phase, the results of this operation are then evaluated during the interpretation phase, where a meaning is given to the recognized objects.

Each level of the hierarchy described previously provides the input for the following one. Between these levels probably one of the most important is the pre-processing,

in fact this step is crucial to achieve good performances. Pre-processing is typically performed through histogram manipulations (e.g. equalization) and filters application, the aim of these procedures is to improve image quality highlighting specific features of the image in order to achieve better results in the following steps.

An important operation that can be performed into the pre-processing phase is the edge detection, which has a key role for many algorithms since it allows to highlight the shape of the elements contained inside the input image. There are several algorithms which implement edge detection, most of them use gradient and laplacian operators to detect intensity variations which often corresponds to an edge. In most of the cases, the application of these operators is not sufficient since input images are affected by noise and other disturbs, e.g. non-uniform illumination. In order to obtain better results, several techniques have been developed. For example, a possible solution is represented by the canny method, where the image gradient is computed after the application of a Gaussian filter. Depending on the values of the gradient, two thresholds are defined in order to have two possible kind of edges: strong edges and weak edges, a weak edge will be taken into account only if it is connected to a strong one.

In general there are two possible approaches: a local analysis or a global analysis. The local analysis is performed dividing the image into regions which are processed one by one connecting the points with similar features, e.g. direction and magnitude of the gradient vector. Global analysis instead takes into account the entire image to understand how to connect the edges, in particular we have that contour points are connected only if they belong to a predefined curve. An implementation which uses this kind of approach is the Hough transform [12], where only a set of parameterized curves is considered to reduce the computational cost.

1.3 Image processing for autonomous driving purposes: The GOLD system

1.3.1 Introduction

Now that a brief introduction to image processing has been given, the following section will explain how it can be used for ADAS (Advanced Driver Assistance Systems) implementations, in particular the GOLD lane and object detection system will be analyzed. The decision of focusing on a lane detection system is mainly due to the fact that it is one of the systems where image processing is most involved. The choice of GOLD, instead, is related to the fact that the working principle of

this algorithm is simpler with respect to others lane detection systems (e.g. LOIS, ARCADE, LANA, RALPH), making it easier to re-implement. So, even if there are more sophisticated systems based, for example, on shape hypothesis and frequency domain features [2], GOLD can be analyzed in a more intuitive way since the causes of eventual fails during the validation procedure are easier to be understood.

1.3.2 System description

GOLD stands for Generic Obstacle and Lane Detection [1], it has been developed by Massimo Bertozzi and Alberto Broggi and it allows to perform both lane and obstacle detection (on flat roads with visible markings) by using only visual data acquired through standard cameras directly installed on the vehicle. In particular, since a stereoscopic view is necessary to implement both lane and obstacle detection, two cameras are used.

The first issue that GOLD has to deal with is that, because of perspective effects, it is not possible to perform efficiently low-level processing with SIMD (Single Instruction Multiple Data) systems on the images acquired by the cameras. In fact, these kind of systems perform the same operation on each pixel of the image, while the perspective effect gives a different meaning to each image pixel depending on its position. In order to overcome this limit, Bertozzi et al. developed a system able to remove the perspective effect allowing to process input images in an easier way. In order to do that each pixel of the input image must be remapped into a new 2D image which represents a top view of the region in front of the vehicle. As reported in [1], two Euclidean spaces $W = \{(x, y, z)\}$ and $I = \{(x, y)\}$ are defined, the I space is the space that contains the images acquired by the cameras, while the W space is the 3D space that represents the real world, the remapped image will belong to the plane of W such that $z = 0$. The mathematical relationship between the two spaces depends on several variables: the camera position $C(l, d, h) \in W$, the viewing direction, the camera angular aperture and the camera resolution $n \times n$. By applying this relationship it's possible to move from one Euclidean space to another, in particular it is used to move from I to W, with the resolution of the remapped image that is chosen as a trade-off between information loss and processing time.

Once that the perspective effects have been removed, it is possible to process the remapped image in order to perform lane detection. Assuming that a road marking can be considered as an almost vertical bright line of constant width surrounded by darker regions, we can consider that pixels belonging to a road marking will have an higher brightness with respect to their right and left neighbours at a given horizontal distance. So Bertozzi et al. propose the following process which allows to detect dark-bright-dark transitions: considering the brightness value $b(x, y)$ of a generic

pixel belonging to the remapped image, each pixel is compared with its right and left neighbours at a specified distance m with $m \geq 1$. The result of this comparison is a new image that encodes the presence of a road marking, as explained in [1], each value $r(x, y)$ of that image is computed according to the following expression:

$$r(x,y) = \begin{cases} d_{+\mathrm{m}}(x,y) + d_{\text{-m}}(x,y), & \text{if } (d_{+\mathrm{m}}(x,y) > 0) \wedge (d_{\text{-m}}(x,y) > 0) \\ 0, & \text{otherwise} \end{cases}$$

with

$$d_{+\mathrm{m}}(x,y) = b(x,y) - b(x,y+m)$$
$$d_{\text{-m}}(x,y) = b(x,y) - b(x,y-m)$$

where $b(x, y + m)$ and $b(x, y - m)$ are the values of the right and left neighbours respectively.

In order to reduce the effects of a non uniform illumination, for example because of shadows, a geodesic morphological dilatation is performed on the filtered image. The effect of this process is to make the levels of brightness more uniform for all the elements of the image with a non-null value. At this point the enhanced image is binarized using an adaptive threshold and then scanned row by row in order to detect the road features, in particular all the pixels with a value different from zero are considered. Each of these pixels can represent the right/left edge of the street or the center line, so the road can be identified by a set of three non-zero pixels. Every set of pixels will represent a road hypothesis that specifies which is the width w_{i} and where is the center c_{i}; in order to understand which set of pixels must be considered and which one must be discarded some constraints based on the image horizontal size are taken into account. Moreover a histogram containing the values of w_{i} for each line is generated and it is used to determine which is the most frequent value W, then only the set of pixels with $W - W/4 < w_{\mathrm{i}} < W + W/4$ are considered to reconstruct the shape of the road.

For what concerns obstacle detection, the starting point is the same described for lane detection, i.e. the remapped image. In fact it can be shown that, computing the difference between the remapped images obtained from the left and the right cameras, a square obstacle in front of the vehicle is transformed in two triangles. In order to detect these triangles a polar histogram is computed from the "difference image", the presence of a triangle will be indicated by a peak inside this histogram. Since an obstacle corresponds to two triangles, two consecutive peaks will correspond to an obstacle. This approach is based on the hypothesis that we are looking

for obstacles with quasi-vertical edges, so it wont work as expected for other kind of objects, e.g. objects with a triangular or pyramidal shape.

1.3.3 System performances

The GOLD system runs on a specific hardware which uses a SIMD (Single Instruction Multiple Data) [1] computing architecture to achieve real time performances with low power consumption. This kind of architecture uses small processing elements in parallel to reduce the power consumption without sacrificing the computational speed. For what concerns the performances in terms of precision, we have that, if the system is used on a road that is compliant with the initial assumptions, i.e. flat road with visible markings and square obstacles, satisfactory results can be achieved (at least for lane detection). Once these assumptions are not valid anymore, performances degrade and the quality of the results becomes unacceptable for automotive standards.

1.4 Conclusion

What has been seen inside this chapter allows to understand the key concepts of image processing showing how it can be used for autonomous driving purposes through the analysis of a possible approach for lane and object detection. These informations represent the starting point for what will be done later on, that is the development, and above all the validation, of a lane detection algorithm based on the GOLD approach.

Chapter 2

Object detection based on machine learning

2.1 Introduction

Object detection represent one of the most interesting challenges for what concerns computer vision. The main problem of object detection is that it is infeasible to develop an algorithm for this task made with specific instructions. In order to overcome this limit, machine learning can be used as a solution to build a mathematical model through ad-hoc training data sets. The aim of this chapter is to report the most effective approaches developed so far for object detection through machine learning, all of them will be briefly described in order to understand how they work and what their limits are.

2.2 Deformable Parts Models

2.2.1 Introduction

One of the simplest approaches to object detection is the sliding window approach. In this kind of technique we basically "slide" a box inside an input image analyzing the content of each box to determine if it contains an object and, if yes, to recognize it. Typically the object detection inside each box is performed through filters applied to HOGs (Histogram of Oriented Gradient). These kind of histograms represents a very good method to extract salient features about the morphology of an image, so they are perfect to represent object categories. One of the biggest problems of using this approach to represent an object category is that often objects belonging to the same category can differs in a lot of aspects. For example: SUV, coupé and station wagons belong to the category "car" even if they have a very different aspect. In order to overcome this problem, deformable part models use a

Figure 2.1. Example of an HOG overlapped to the original image

collection of parts arranged in a deformable configuration to represent objects.

2.2.2 Models description

In DPM models [3] we have that linear filters are applied to dense feature maps, these maps are an array whose entries are d-dimensional feature vectors computed starting from the locations of the original image. Each linear filter has a rectangular shape and is defined by an array of n-dimensional weight vectors. As reported in [3], considering a filter F, we have that its score at a position (x, y) in a feature map G is given by the dot product of the filter and a sub-window of the feature map with top-left corner at (x, y):

$$\sum_{x',y'} F\left[x', y'\right] \cdot G\left[x + x', y + y'\right]$$

Since during the detection there is the need of defining a score at different positions and scales inside the image, a feature pyramid is used. Feature pyramids

specifies a finite number of feature maps with different scales, each level of the pyramid will represent the same image but with a different resolution. The more we go deep into the pyramid levels, the higher resolution we have.

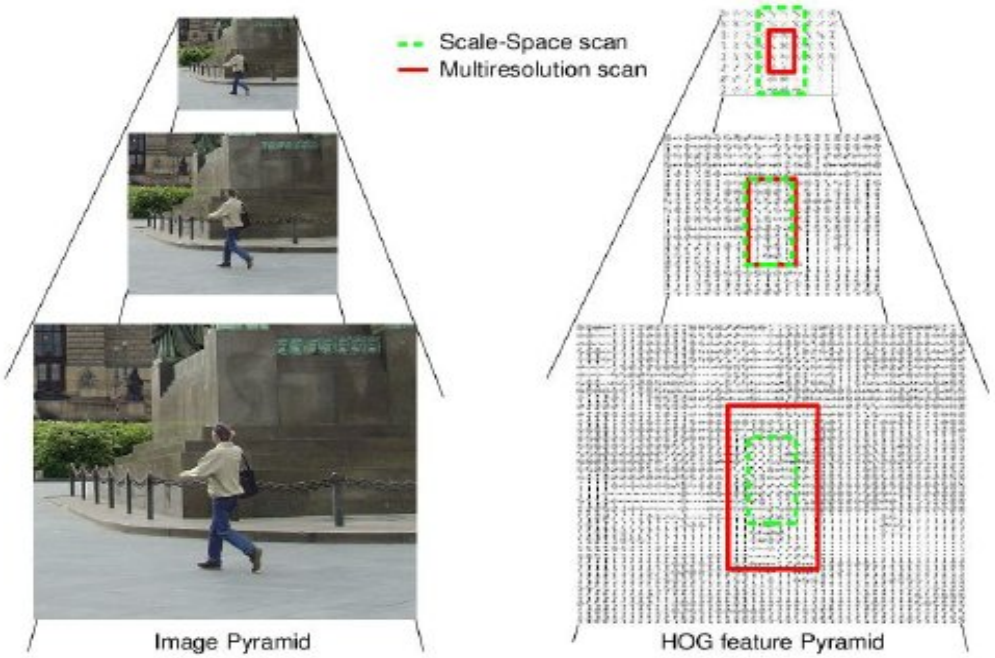

Figure 2.2. Sample image of a feature pyramid

For what concerns the objects, they are represented through a star-structured part based model, these kind of models are defined by a "root" filter accompanied by a set of "part" filters with an associated deformation model. The root filter is used to approximatively identify the entire object, while part filters cover smaller parts of the object with a higher resolution. In order to apply the filters to the input image with different resolutions, we just have to apply them to a different level of the feature pyramid. Root filters will be applied to upper levels of the pyramid, while part filters will be applied to deeper levels. The fact that part filters are defined with a higher resolution is fundamental for deformable parts models to achieve good recognition performances.

Given a certain position, the score of each star model is given by the score of the root filter plus the scores of the part filters minus a deformation cost. The value of the deformation cost depends on where parts are placed with respect to the root filter, the more they will be far from the "ideal" position, the higher will be the deformation cost value.

Going a bit more into details, in [3] is explained that in deformable parts models each object with n parts is modelled through a (n+2)-tuple (F_0, P_1, ... , P_n, b): F_0 is the root filter, P_i is the i-th part filter while b is a bias. Each part is modelled with a 3-tuple (F_i,v_i,d_i): F_i is the part filter, v_i is a two-dimensional vector which specify an "anchor" for the part and d_i is a four-dimensional vector that contains the coefficients of a quadratic function used to compute the deformation cost. Every time we have an object hypothesis, the location $p_i = (x_i, y_i, l_i)$ of each filter in the model is indicated. For each location x_i and y_i specify the position while l_i specifies the level inside the feature pyramid. So the object hypothesis will be given by an array $z = (p_0, ... , p_n)$ where p_0 is the location of the root filter and p_i is the location of the i-th part filter. An important aspect that must be highlighted is that the level of part filters is chosen such that the resolution is the double with respect to the root filter.

At this point, considering a filter F with size $w \times h$, a feature pyramid H and a position p = (x, y, l), in [3] we have that the score of the filter will be:

$$F' \cdot \phi\left(H, p, w, h\right)$$

Where F' is the vector obtained by concatenating the weight vectors of the filter F and $\phi\left(H, p, w, h\right)$ is the vector obtained by concatenating the feature vectors in the $w \times h$ sub-window of the feature pyramid H with top-left corner at the p position.

The score of each object hypothesis, according to what is reported in [3], will be given by:

$$score\left(p_0, ..., p_n\right) = \sum_{i=0}^{n} F' \cdot \phi\left(H, p, w, h\right) - \sum_{i=1}^{n} d_i \cdot \phi_d\left(dx_i, dy_i\right) + b$$

Where the first summation is the score of the filter F, the second one is the deformation cost and b is a bias term. The term ϕ_d represents the deformation features of the i-th part for a distance (dx_i, dy_i) with respect to the anchor.

Depending on the score of each object hypothesis, bounding boxes are placed into the image to highlight the presence of a certain class of objects. Multiple detections of the same object are rejected through non-maximum suppression.

2.2.3 Method performances

Deformable parts models represent a valid method for object detection for what concerns the performances in terms of average precision, but, like all the methods based on the sliding window approach, they require a very long time to extract

features from an input image. Even if there are several implementations of this method (e.g. Branch-Bound, Cascade, FFT, Coarse-to-fine ecc... ecc..), none of them is fast enough to provide real time performances. In any case, there are some new proposed implementations that shown to have promising results in terms of speed but poor performances in terms of precision.

2.3 R-CNN

2.3.1 Introduction

One of the most effective way to extract features from an image is the use of a CNN (Convolutional Neural Network). CNNs are basically used to perform operations on an input image through a series of multilayer perceptrons, these networks are composed by an input layer and an output layer with several "hidden" layers between them. Each layer performs a different kind of operation on the output of previous layers, in the end we obtain a processed image which can also have a different resolution with respect to the original one. The key concept of R-CNN is to use region proposals as input for a CNN, performing object detection depending on the features extracted from each proposed region.

2.3.2 Method description

As reported in [5], R-CNN object detection is basically based on three fundamental steps: 1) generation of region proposals starting from the input image, 2) features extraction from each proposed region through CNN, 3) analysis of the extracted features for each class through SVM (Support-Vector Machines).

The generation of region proposals is very important to obtain a more efficient detection, in fact, performing object detection through an exhaustive search is infeasible from the computational point of view. Region proposals can be obtained in several ways, in R-CNN a "selective search" algorithm is used.

As explained in [4], the main goal of selective search is to reduce the number of analyzed locations allowing to save time which can be spent to perform a more accurate detection during next steps. In order to do that, a graph-based segmentation of the image is performed to find a set of small initial regions in a fast way. Starting from these regions a grouping procedure based on similarities is performed. In few words, the similarities between all neighbouring regions are computed, then the two most similar regions are grouped and the process is repeated until a single region is obtained. In the end of the selective search a set of about 2000 regions is generated.

After the generation of region proposals, the feature extraction is performed on each proposed region. In particular a 4096-dimensional feature vector is extracted using a CNN composed by five convolutional layers and two fully connected layers which takes as input a 227×227 RGB image. Since proposed regions can have a resolution that is not compatible with the CNN, regardless of the region size, input images are warped in order to fit into the network.

The extracted features are then scored for each object class using a Support-Vector Machine (SVM) specifically trained for that class. SVMs are learning models used to perform classification and regression analysis [16]. In SVMs data are represented as points belonging to n-dimensional spaces, the aim of this machine learning method is to find a set of hyperplanes that identify all the considered objects classes. Once the score has been computed for each proposed region, non maximum suppression is performed to avoid multiple detections of the same object, then, depending on the score, each region is associated to an object class.

To perform this association an IoU (Intersection over Union) overlap threshold is chosen, which means that regions are labeled only if their score overcomes a certain value. It's worth noting that the choice of the threshold value has a significant impact for what concerns the performances in terms of mean Average Precision (mAP). Choosing an appropriate value can make this method much more precise while a wrong one can lead to poor performances.

2.3.3 Fast R-CNN

R-CNN represents an effective method for object detection, it allows to achieve a good level of precision in a reasonable amount of time, even when a big amount of classes is considered. As reported by Ross Girshick et al. [5], the developers of this method, considering 10.000 classes, object detection on VOC 2007 can be performed in about one minute with a mAP of 59%.

The limit of R-CNN is that it is too slow to perform real-time object detection, moreover the training phase requires a long amount of time and a lot of storage to be performed. In order to improve the performances of this method, the same author of R-CNN has developed a reviewed version called Fast R-CNN [6].

The main reason that makes R-CNN slow, is that it requires to pass each object proposal through a convolutional network to extract features from it. To solve this problem Fast R-CNN uses a more efficient network architecture which allows to achieve better performances. Basically the entire image is passed through a series of convolutional and max pooling layers to produce a feature map, for each

object proposal a region of interest (RoI) pooling layer is then used to extract a fixed-lenght feature vector. At this point, each feature vector is fed into a series of fully connected layers that finally split in two branches: one pass through a softmax layer which estimates the predicted class while the other produces as output the offset values for the bounding box.

As explained in [6], RoI pooling layers are very important in this method since they allows to express the features of each region of interest with small sized feature vectors. Each region of interest is defined by a four-tuple (r, c, h, w), where (r, c) specifies the top-left corner and (h, w) the height and width. RoI pooling layers take as input these regions and divide them into a $H \times W$ grid of $h/H \times w/W$ sub-windows, then max-pooling is performed for each sub-window obtaining as output the feature vector for the considered region of interest.

Using this network architecture, the entire image pass through a convolutional network only one time instead of repeating the procedure for all the 2000 proposed regions. These regions, instead, are analyzed through non-convolutional layers, which are less expensive from the computational point of view. The tests performed by Ross Girshick et al. and reported on its paper shown that Fast R-CNN requires only about 0.3 seconds to process an image (excluding region proposals generation), against the 47 seconds required by "slow" R-CNN. Moreover Fast R-CNN shown significant improvements also for what concerns training time and mAP.

2.3.4 Faster R-CNN

Even if Fast R-CNN is much faster with respect to R-CNN, if we take into account also the region proposals generation, it requires about 2.3 seconds to process an image, so it cannot be used for real time object detection. Both R-CNN and Fast R-CNN, in fact, use selective search to generate region proposals, which is a slow process where no kind of machine learning is performed. In order to overcome the limits introduced by selective search, Shaoqing Ren et al. developed another alternative implementation of R-CNN called "Faster R-CNN" [7].

The key idea of Faster R-CNN is to obtain proposals through a deep convolutional network instead of a time consuming algorithm, these kind of convolutional networks are called Region Proposal Networks (RPNs). One of the advantages of using RPNs is that they can share convolutional layers with other networks, e.g. the object detection network, allowing to reduce the overall computational cost for region proposals generation.

Faster R-CNN is composed of two modules: a deep fully convolutional RPN and the Fast R-CNN detector. These two modules are merged together into a single

network where convolutional layers are shared to reduce the computational cost. So, a common set of convolutional layers is used to produce a feature map from the input image, this feature map is then used by the RPN to generate a set of region proposals which are finally used to feed the Fast R-CNN network.

Thanks to its efficient architecture, which allows to get rid of selective search, Faster R-CNN is able to perform object detection on an image in about 0.2 seconds, fast enough to use this method for real time purposes.

2.4 OverFeat

Another interesting method that uses CNNs to perform object detection is OverFeat. This method, developed by Pierre Sermanet et al. [8], uses three networks specifically trained to perform object classification, localization and detection respectively. Each of these three networks performs a specific task which can be considered as a sub-task for the next network, so they can be merged into a single CNN, which is applied to the input image in a sliding window fashion.

Even if OverFeat introduces a lot of interesting solutions for object detection, it is not very efficient and its performances are not so brilliant. With a mAP of about 24%, it is way less precise with respect to the other methods reported previously, so it won't be explained in detail. In any case, since it is often used as a term of comparison, it has been reported for the sake of completeness.

2.5 Single Shot MultiBox Detector (SSD)

2.5.1 Introduction

The methods reported so far allow to perform object detection with a good level of precision, however, they have poor performances in terms of speed (most of them are way far from real time object detection, with the only exception of Faster R-CNN, which is able to run at most at 7 FPS). In order to speed up the detection without sacrificing the accuracy, "single shot" methods have been developed. These kind of methods are able to perform object localization and classification through a neural network with a single forward pass, allowing to achieve significant improvements in terms of speed. One of the most famous methods belonging to this category is Single Shot MultiBox Detector, better known as SSD [10].

2.5.2 Method description

SSD uses a feed-forward convolutional network to produce a set of fixed-size bounding boxes from a given input image, non-maximum suppression is then performed depending on the score of each bounding box to obtain the final detections. The VGG-16 network, truncated before classification layers, is used by SSD as "base network", other auxiliary layers are then added to improve the detection features. To obtain predictions of detections at multiple scales, a series of convolutional feature layers with a progressively decreasing size is added to the base network. Each of these layers uses a set of convolutional filters to produce a fixed set of detection predictions, so, every feature map cell of each feature map is associated to a default set of bounding boxes. Since each bounding box is defined by 4 offset values and c class scores, considering a set of k bounding boxes for each feature map cell into a $m \times n$ map, every feature map will return a set of $(c + 4)kmn$ values. The offset values are used to express the position and the dimension of each bounding box, these values are measured with respect to "default" bounding boxes similar to the anchor boxes used in R-CNN.

2.5.3 Method performances

SSD represents a significant step forward in terms of performances. As reported in [10], it is able to perform object detection at 59 FPS achieving 74.3% of mAP with a 300×300 input image, which becomes 76.9% when the resolution grows to 512×512. With these features SSD outperforms all the methods seen previously, introducing a valid object detection system for real time purposes.

2.6 YOLO (You Only Look Once)

2.6.1 Introduction

The meaning of the acronym YOLO is "You Only Look Once", as suggested by the name itself, YOLO is a single shot object detection system that uses a single convolutional network to obtain simultaneous bounding boxes predictions and class probabilities for each of them.

2.6.2 Method description

As mentioned before and reported in [9], YOLO uses a single convolutional network that processes the entire image at the same time to perform object detection. In order to do that, the input image is divided into a $S \times S$ grid and B bounding

boxes are predicted for each cell of this grid. Each bounding box is composed by 5 predictions: x, y, w, h and a confidence score, x and y are the coordinates of the center while w and h specify the width and the height respectively. The confidence score is defined as $Pr(Object) * IoU$, where $Pr(Object)$ is the probability estimated by the model of having an object inside the box, while IoU is the intersection over union between the predicted box and the ground truth.

Regardless of the number of bounding boxes B, a set of C conditional class probabilities is predicted for each grid cell. These probabilities are computed as $Pr(Class_{\text{i}}|Object)$, which is the probability of having an object belonging to the i-th class inside a cell, conditioned by the probability of actually having an object inside it. Taking into account all the parameters mentioned so far, we have that with this model predictions are encoded as a $S \times S \times (B * 5 + C)$ tensor.

As explained in [9], at test time the conditional class probabilities and the confidence score are multiplied, so we have:

$$Pr(Class_{\text{i}}|Object) * Pr(Object) * IoU = Pr(Class_{\text{i}}) * IoU$$

The result of this product gives the class-specific confidence score for each bounding box. At training time, since YOLO predicts a set of bounding boxes for each grid cell, in order to avoid having more than one bounding box to be responsible for the prediction of the same object, a loss function is used to select only one bounding box for each object prediction.

For what concerns the network architecture, YOLO is implemented through 24 convolutional layers followed by two fully connected layers: the convolutional layers are used to extract features from the input image while the fully connected layers are used to generate predictions. To obtain an even faster implementation, called Fast YOLO, a smaller architecture with only 9 convolutional layers can be used. The main drawback of this implementation is that it is less precise compared to the "slow" version.

Requiring only a single network evaluation, YOLO is extremely fast. The results reported in [9] show that it is able to perform object detection at 45 FPS with a mAP of 63.4%. Fast YOLO pushes the speed up to 155 FPS without sacrificing too much the mAP score, which decreases to 52.7%. The main limit of YOLO is that, because of its strong spatial constraints which impose a fixed number of bounding box and classes for each grid cell, it struggles with groups of small objects, e.g. a flock of birds.

2.6.3 YOLOv3

Several "improved" versions of YOLO have been developed after the original version, the latest one is called YOLOv3 [11]. The first difference from previous versions is about bounding box predictions. While YOLO uses two fully connected layers to predict bounding boxes from the features map, in YOLOv3 these layers are removed and predictions are made using dimension clusters as anchor boxes. According to [11], in YOLOv3 the network predicts 4 coordinates for each bounding box: $t_{\mathrm{x}}, t_{\mathrm{y}}, t_{\mathrm{w}}, t_{\mathrm{h}}$. If the offset of a cell with respect to the top left corner is $(c_{\mathrm{x}}, c_{\mathrm{y}})$ and previous bounding box has a width p_{w} and a height p_{h}, the prediction is given by:

$$
\begin{aligned}
b_{\mathrm{x}} &= \sigma(t_{\mathrm{x}}) + c_{\mathrm{x}} \\
b_{\mathrm{y}} &= \sigma(t_{\mathrm{y}}) + c_{\mathrm{y}} \\
b_{\mathrm{w}} &= p_{\mathrm{w}} e^{t_{\mathrm{w}}} \\
b_{\mathrm{h}} &= p_{\mathrm{h}} e^{t_{\mathrm{h}}}
\end{aligned}
$$

where $\sigma()$ is a sigmoid function used to predict the center coordinates of the box.

During training, a sum of squared error loss is used, where the error is given by the difference between the prediction $\hat{t}_*$ and the ground truth t_*. For each bounding box an "objectness" score is predicted using logistic regression, the value of this score is 1 if the considered bounding box overlaps a ground truth object more than any other [11].

The convolutional feature extractor used in YOLOv3 is more complex compared to the one used in YOLO. It is composed by 53 convolutional layers, instead of 24, and its name is Darknet-53 [11]. Starting from the feature map generated by this network, class predictions are performed. Rather than using a softmax layer, each bounding box performs a multilabel classification done with independent logistic classifiers. These predictions are performed for 3 different scales, so the last layer of the classifier generates a 3-d tensor that encodes bounding boxes, objectness and class predictions.

While YOLO struggled with small objects, thanks to multi-scales predictions, YOLOv3 shown to have better performances with small objects with respect to large and medium ones. In general YOLOv3 represents a significant improvement, it is three times faster than SSD with the same level of precision, making YOLO one of the best methods for object detection.

2.7 Conclusion

All the methods seen so far represent approximatively the state of the art in object detection. In the following chapters we will focus on YOLOv3, which is the system that shown to have the best trade off between real time performances and precision. A validation process will be performed to measure the robustness of this system discovering its limits with a particular focus on automotive environments.

Part III

Development and validation of a lane detection algorithm based on image processing

Chapter 3

Algorithm description and implementation

3.1 Introduction

There are several approaches that can be followed to implement a lane detection system. During the bibliographic analysis the focus has been set on GOLD because of its intuitive working principle and of its relatively good performances, but many other possible solutions can be adopted. Some of these solutions are very effective to perform lane detection, however they can involve the use of complex techniques which often are not very easy to be interpreted in case of malfunctioning (e.g. image processing in the frequency domain). So, since the main goal of this work is to highlight the limits and the main issues of vision systems for autonomous driving introduced by the environmental conditions, the validation process will be performed on an algorithm for lane detection based on the working principle of GOLD, which represents a good tradeoff between simplicity of the algorithm and quality of the performances.

3.2 Algorithm description

Since GOLD has been developed on a custom hardware architecture, it is almost impossible to find an implementation that can run on a standard PC. To overcome this problem a new algorithm has been implemented in Matlab® following the working principle of GOLD as a guide line. This "GOLD-based" algorithm performs the following steps:

1. Production of a greyscale "Bird's eye view" of the road;

2. Filtering and thresholding of the produced image to highlight road lanes;
3. Histogram-based generation of road lanes;

The first step has been performed using a specific function belonging to the Matlab® Automated Driving Toolbox called *transformImage()* [13]. This function is able to obtain a bird's eye view using inverse perspective mapping starting from: a 2D image, a *birdsEyeView* object containing the camera properties (incapsulated into a *monoCamera* object), the portion of the camera view that will be transformed into a bird's eye view (provided in vehicle coordinates) and the size in pixel of the output image.

For what concerns the second step, it includes many processes performed to detect vertical lanes inside the top view image of the road following the same approach used in GOLD, that is the search of horizontal dark-bright-dark transitions. This search is performed scanning the bird's eye view pixel by pixel and applying a rule similar to the one proposed by Bertozzi et al. for GOLD [1], which is based on the comparison between the level of brightness of the i-th pixel $b(x, y)$ and of its right and left neighbours $b(x, y + m)$ and $b(x, y - m)$. The variable m can be modified in order to change the width of the lines that must be detected by the algorithm, so a bigger value of m will lead the system to look for thicker lines while a smaller one will have the opposite effect. The result of this comparison allows to produce a remapped image r whose values are obtained as follow:

$$r(x,y) = \begin{cases} d_{+\mathrm{m}}(x,y) + d_{-\mathrm{m}}(x,y), & \text{if } ((d_{+\mathrm{m}}(x,y) > threshold) \wedge (d_{-\mathrm{m}}(x,y) > threshold)) \\ 0, & \text{otherwise} \end{cases}$$

with

$$d_{+\mathrm{m}}(x,y) = b(x,y) - b(x,y+m)$$
$$d_{-\mathrm{m}}(x,y) = b(x,y) - b(x,y-m)$$

The *threshold* variable is used to specify the sensitivity of the detection system, a smaller threshold value will lead to consider also "softer" dark-bright-dark transitions, while a higher one will lead the system to detect only "strong" transitions. A similar expression has been already explained inside the bibliographic analysis, in this case the main difference with respect to GOLD is that the threshold value is not always 0 but can be set by the user in order to choose a certain level of sensitivity.

After this first process, in order to remove spurious detections, a median filter with a 3×3 structuring element is applied, then a thresholding procedure is performed to obtain a binary image where all the pixels with a low value are cutted off.

At this point a last process is performed to detect and remove the pixels that does not belong to a road marking with a "vertical" analysis instead of a "horizontal" one. For each pixel of the remapped image $r(x, y)$, a check is performed on the pixels belonging to the same column applying the following rule:

$$r_{\text{filtered}}(x, y) = \begin{cases} 255, & \text{if } (r(x, y) = 255) \wedge ((r(x + m, y) = 255) \vee (r(x - m, y)) = 255)) \\ 0, & \text{otherwise} \end{cases}$$

So, in few words, considering a pixel belonging to the i-th row and the j-th column of r, if it has a value equal to 255 (logic 1) and the pixel above (same column and $(j - m)$-th row) or below (same column and $(j + m)$-th row) has a value equal to 255, the pixel keeps its value, otherwise it is considered null. Naturally the value of each pixel is stored into a new remapped image called r_{filtered}. The main aim of this sort of custom filter is to detect, and eventually remove, all the pixels with a positive value that do not belong to a line. The process is iterated twice with two different values of m in order to perform a check that consider two different distances. The first iteration is performed using a longer "step" with respect to the second one, in this way the pixels are checked with an increasing level of "rigidity".

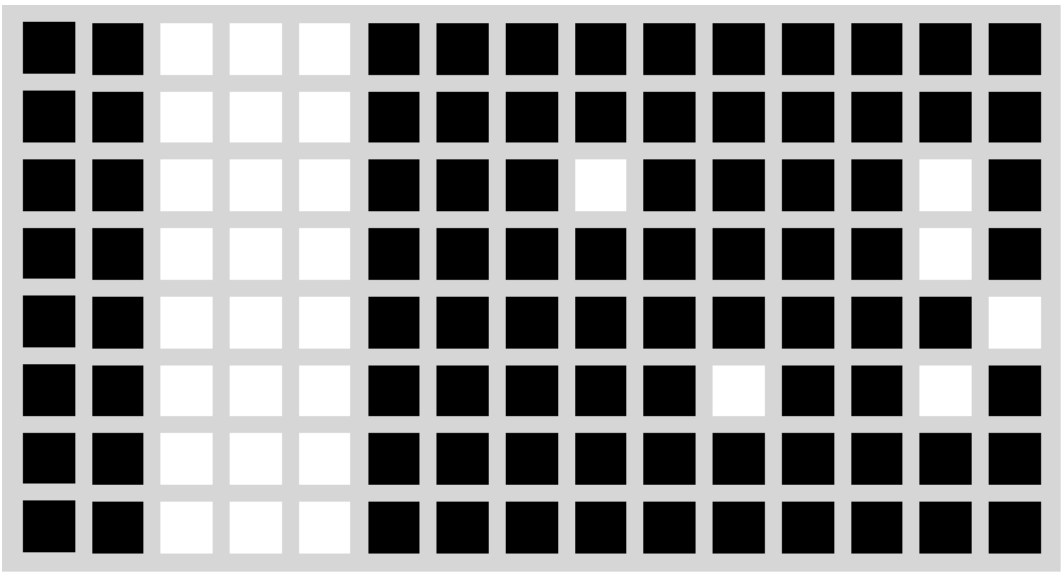

Figure 3.1. Sample of an area inside the remapped image r

Just to make an explanatory example, Figure 3.1 shows a sample of an area inside r where every square represents a pixel. It is possible to see that on the left there is a well defined line while on the right there are some white pixels due to spurious

detections. Considering $m = 4$, the resulting image r_{filtered}, which is shown in Figure 3.2, will keep the pixels belonging to the line discarding all the others.

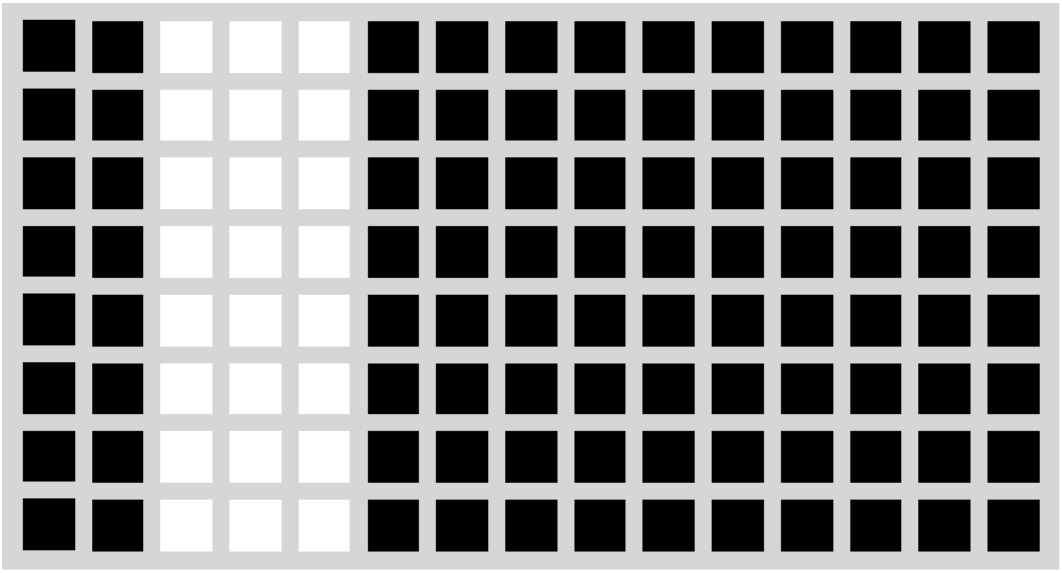

Figure 3.2. Output image considering the one reported in Figure 3.1 as input

In the end, in order to have an image with lines composed only by segments of a certain length, the third step is performed. During this last process the image is subdivided in horizontal slices with a height of h pixels, for each of these slices a histogram containing the number of pixels with a value equal to 255 (i.e. the logic "1") inside each column is computed. If the number of "positive" pixels overcomes a threshold value, all the pixels inside the column are set to 255, otherwise they are set to 0. The result of this process is a final image r_{final} that does not contain any isolated spurious pixel but only little segments with a length equal to h composing the road markings detected by the system. This final image can then be used to compute the position of the lanes with respect to the vehicle reference frame through the *imageToVehicle()* Matlab® function, which allows to convert the bird's-eye-view image coordinates to vehicle coordinates [13].

3.3 Camera calibration

As already explained in the previous section, the bird's eye view of the road is obtained using the *transformImage()* Matlab® function. This function requires

multiple inputs, one of them is a *monoCamera* object that contains all the properties of the camera used to take the input pictures, including the intrinsic camera parameters [14]. These parameters are: focal length, principal point and image size. The focal length is specified as a two-element vector $[f_x, f_y]$ with $f_x = F \times sx$ and $f_y = F \times sy$, where F is the focal length in world units (usually millimeters) while sx and sy are the number of pixels per world unit in the x and y direction respectively, so both f_x and f_y are expressed in pixels. The principal point, instead, is the optical center of the camera expressed in pixels and specified as a two-element vector $[cx, cy]$. These parameters are fundamental to perform the inverse perspective mapping that allows to obtain the bird's eye view image. The main problem is that they are different for each camera and that they are not provided by camera manufacturers, so the only way to compute them is by performing a camera calibration. Luckily Matlab® includes a Camera Calibration App [15] that allows to perform this operation, obtaining all the parameters needed to remove the perspective effects in an easy and fast way.

The calibration procedure is performed by taking a series of pictures of a checkerboard with a square pattern, like the one reported in Figure 3.3, from different positions and with different angulations. In this way, providing the exact size of the squares, the calibration app is able to compute all the extrinsic and intrinsic camera parameters. Once that the calibration session is completed, all the computed parameters are stored in the workspace and some plots are displayed. One of these plots shows the reprojection errors for each input image, which is the error (expressed in pixels) between the real and the estimated projection of a world point on the image. The mean error between all the input images should not overcome 1 pixel to achieve acceptable results. Two other interesting images are the "pattern-centric" and the "camera-centric" views, which show respectively the estimated positions of the camera with respect to the checkerboard and the estimated positions of the checkerboard with respect to the camera for each input picture. Figure 3.4 reports the pattern-centric view, which shows the checkerboard in a fixed position and the camera in all the positions used to take each of the pictures needed for the calibration.

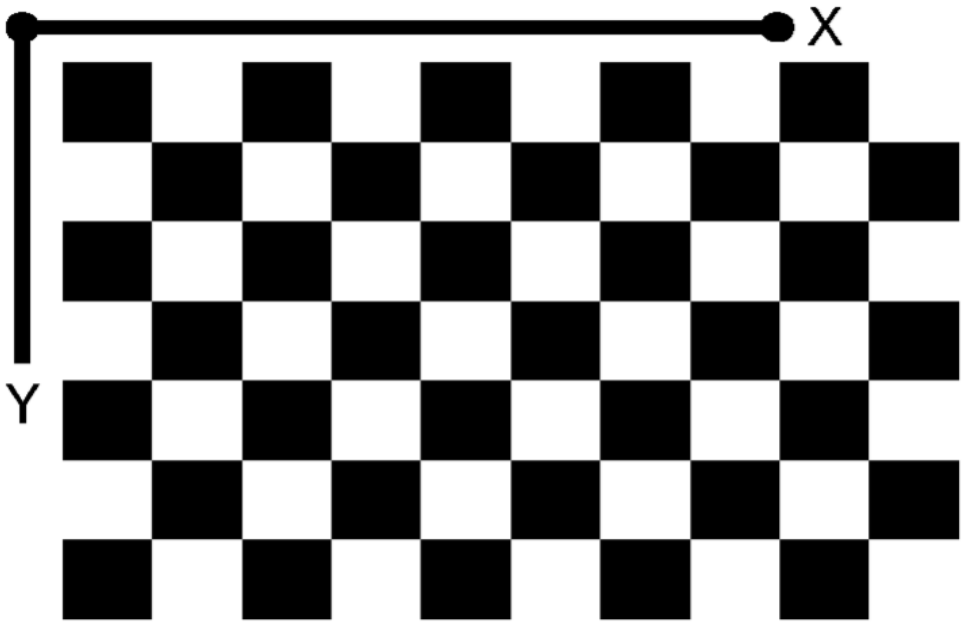

Figure 3.3. Checkerboard provided by Matlab®

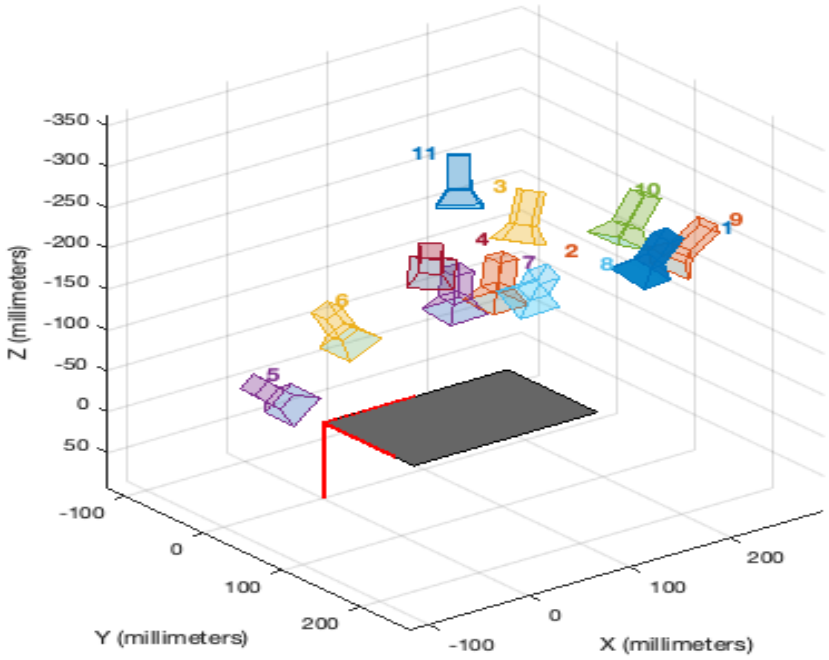

Figure 3.4. Pattern-centric view

Chapter 4

Algorithm validation

4.1 Introduction

Now that the working principle of the lane detection algorithm has been explained, the following part will focus on its validation. The validation process is crucial to understand the reliability and the robustness of any system, especially for a system designed for safety purposes, where there is the need to guarantee an almost null fail probability. During this procedure many different scenarios will be evaluated to understand how system performances change depending on several factors like, for example, weather conditions and illumination. The data set needed to perform the validation procedure has been entirely developed in Turin using a GoPro® Hero 7 Black edition camera mounted on an easel with a height of about 1,5 meters and a pitch of about 8. The height and the pitch have been set in a way that reproduces the typical position used for cameras inside vehicles, i.e. near to the rearview mirror. The main street that has been used as a "benchmark" is Corso Castelfidardo in Turin (Italy), this choice is mainly due to the fact that the characteristics of this road are compliant with the requirements of the implemented lane detection system (it is almost straight with clear road markings), moreover the traffic conditions allowed to take pictures in a safe way. In order to evaluate the performances with different light conditions, a picture has been taken for every different moment of the day: morning, afternoon, nightfall and night. Starting from this initial data set other synthetic images have been developed to evaluate different weather conditions and to add micro and macro defects. For the sake of completeness also a more "critical" environment with bad asphalt conditions and colored road markings has been considered, so some pictures have been taken in Via Sant'Antonio da Padova (always in Turin, Italy), which is a street that has the aforementioned characteristics. As anticipated previously, the following sections will analyze the behaviour of the lane detection algorithm in different scenarios. First of all the original images will be used, subsequently the synthetic images derived from the original ones will be evaluated.

4.2 Light conditions

One of the main variables for every vision process is the environmental illumination. Especially when working outdoor, these conditions change hour-by-hour because of the cyclic alternation of day and night. So, in order to evaluate the level of sensitivity (with respect to lighting variations) of the implemented lane detection algorithm, a picture has been taken for each moment of the day: morning, afternoon, nightfall and night.

4.2.1 Morning

Just to follow a sort of chronological order, the first part of the day that will be considered is the morning. Theoretically this is the moment of the day with the best possible illumination, i.e. it is uniform and mostly free from any kind of shadow due to the inclination of the sunshine. Figure 4.1 shows Corso Castelfidardo during the morning, as expected the quality of the input image is pretty good: the road markings are clear and shadows are almost absent. After a first calibration of the parameters used by the algorithm, i.e. the "step" m and the *threshold* value, the image reported in Figure 4.1 has been used as input obtaining the results reported in Figure 4.2. The Matlab® script has been developed in a way such that it shows in the same image all the intermediate passages, which are: the bird's eye view, the binary image *r filtered* (obtained through the initial filtering procedures), the final result contained in the image *r final* (obtained through an histogram-based procedure starting from *r filtered*) and the overlap of the bird's eye view and the final output image to check the goodness of the detected lanes (highlighted in green). Since all the road marking have been correctly detected, the final output can be considered acceptable. The only imperfection is represented by some spurious detections due to the fact that the sidewalks are very close to the borders of the lane, in fact, since the edges of the sidewalks are brighter than their surroundings, they can be mistakenly detected as part of a lane by the algorithm. Another thing that it is worth noting is that, when the system tries to reconstruct the bird's eye view of the most distant parts of the road, the corresponding part of the reconstructed image is a bit blurred and the shape of the road markings becomes distorted. This effect is mainly due to the fact that, as the distance increase, the algorithm has less available pixels to reconstruct the road, so the level of definition is clearly reduced.

Figure 4.1. Input picture of Corso Castelfidardo during the morning

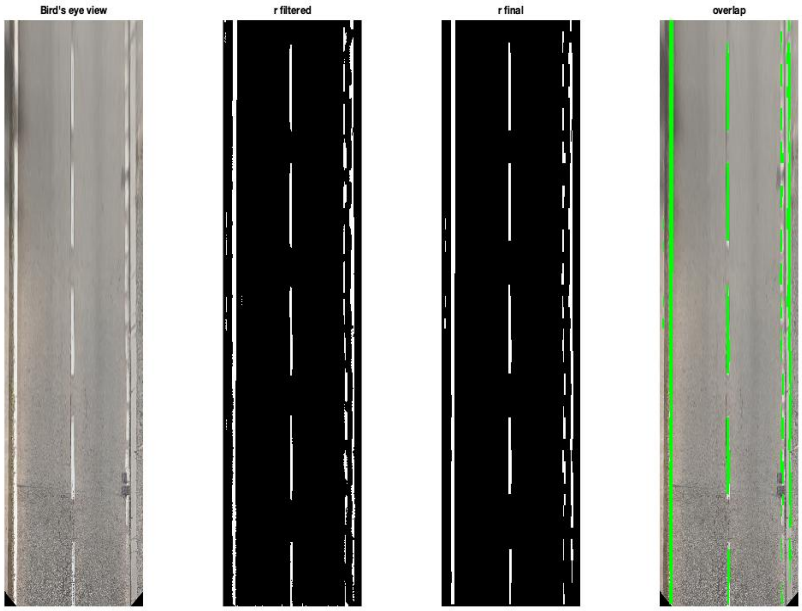

Figure 4.2. Output images obtained using as input the image reported in Figure 4.1

4.2.2 Afternoon

While the morning was an "ideal" condition because of the reasons mentioned previously, during the afternoon, even if the luminosity is still high, the position of the sun is such that the road is not lighted uniformly, producing shadows that can somehow disturb the lane detection process. Figure 4.3 shows a pictures of Corso Castelfidardo taken from the same position of Figure 4.1 but during the afternoon instead of the morning.

As it is possible to see, because of the shadows, some areas are darker with respect to the ones directly lighted by the sunlight. Despite these darker areas, the results reported in Figure 4.4 are still good but not as good as the ones obtained in the morning. For example, some of the dashed lines on the right were not detected, in particular the ones that belong to the brighter road areas. Moreover, probably because of the different sunlight angulation, the amount of spurious detections due to the sidewalks is higher with respect to the morning. In addiction to this, there is a spurious detection inside the left part of the lane, which is due to the fact that the sunlight highlights a clearer area of the asphalt that looks like a horizontal stripe. The last thing that is worth reporting can be noticed in Figure 4.4, on the top-left corner of the overlap between the bird's eye view and the detected lanes. Inside this area, because of a tree with a particularly "intense" shadow, the thickness of the detected lane results reduced. It is reasonable to think that, in worse light conditions, the system will not be able to detect the road marking inside this area producing an interruption.

To conclude, it is possible to say that, since the borders of the lane have been detected properly despite some little imprecisions, the overall quality of the results can be considered satisfactory.

Figure 4.3. Input picture of Corso Castelfidardo during the afternoon

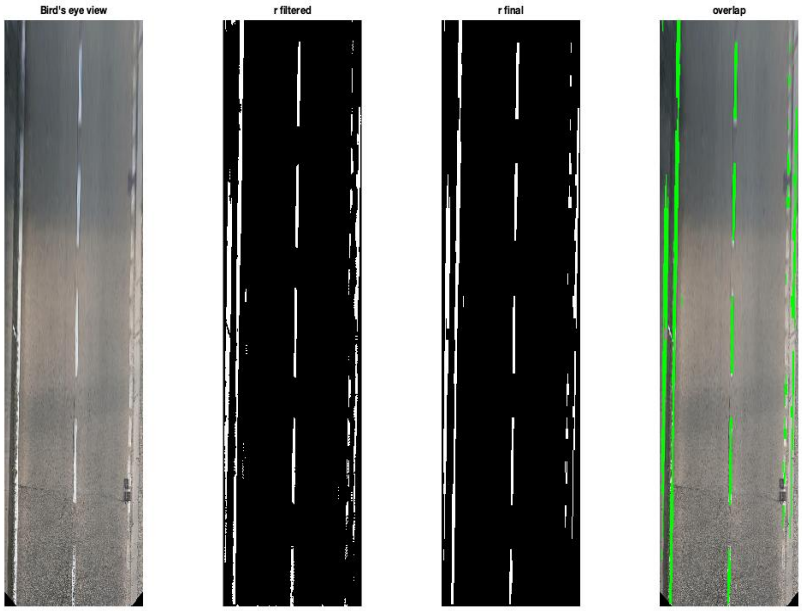

Figure 4.4. Output images obtained using as input the image reported in Figure 4.3

4.2.3 Nightfall

As the night approaches, the light conditions becomes more critical hour after hour. One of the worst moments from this point of view is the nightfall, where the road lights are still off and there is not direct sunlight. A positive aspect of this condition is that shadows are almost absent with respect to the afternoon, unfortunately the price to pay is a much lower illumination. Figure 4.6 shows a picture of Corso Castelfidardo during the nightfall, the conditions of the road are coherent with respect to what has been previously mentioned, i.e. low illumination and only light shadows. The outputs provided by the lane detection system are shown in Figure 4.7. As it is possible to see, also in this case, the quality of the results is not bad and the system detected correctly the borders of the lane. Nevertheless, the lower illumination led to an overall worsening of the performances and, even if the spurious detections due to the sidewalks are reduced with respect to the previous cases, most of the dashed lines on the right were not detected. This is mainly due to the fact that these lines are not as clear as the other road markings, so, with a poor illumination, the system is not able to detect them. Moreover, as supposed during the analysis of the results obtained in the afternoon, at first the continuous line on the left was not detected in the proximity of the shadow produced by a tree (Figure 4.5). So, in order to improve the performances, the sensitivity of the system has been increased by reducing the threshold parameter from 8 to 6. In this way it has been possible to compensate the worsening of the light conditions keeping an acceptable level of performance.

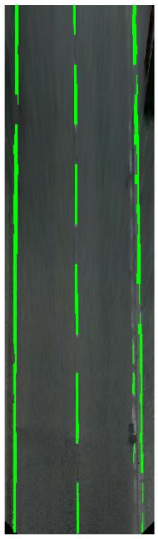

Figure 4.5. Final result during the nightfall with threshold = 8, (lower sensitivity)

Figure 4.6. Input picture of Corso Castelfidardo during the nightfall

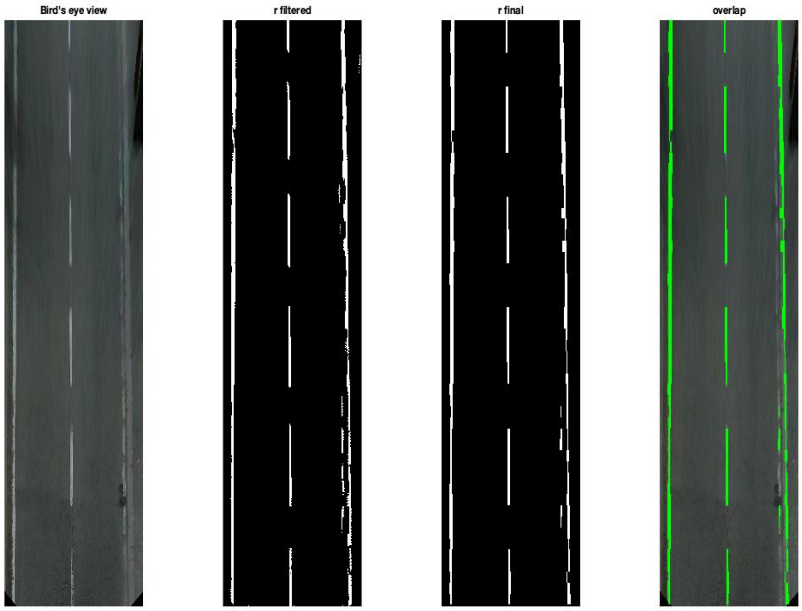

Figure 4.7. Output images obtained using as input the image reported in Figure 4.6

4.2.4 Night

Let's now consider the worst possible scenario in terms of light conditions: the night. The main problems linked to this part of the day are the low lighting and the non homogeneous illumination generated by the artificial lights. Figure 4.8 shows a picture of Corso Castelfidardo taken during the nighttime, in order to evaluate the worst possible condition in terms of illumination, the car lights have been considered turned off.

The final results obtained using Figure 4.8 as input are reported in Figure 4.9. As done for the nightfall, in order to achieve better performances, the sensitivity has been increased by reducing the *threshold* value to 4. With this level of sensitivity, despite the worse lighting, the final results are even better with respect to the ones obtained during the nightfall. Anyway, also in this case some of the dashed lines on the right were not detected, moreover the high sensitivity increased the amount of spurious detections due to the sidewalks. In any case, even if the quality of the results is certainly inferior with respect to the morning and the afternoon, the system detected very well the continuous line on the left and struggled only with the dashed line on the right, so the performances can be considered satisfactory.

Figure 4.8. Input picture of Corso Castelfidardo during the night

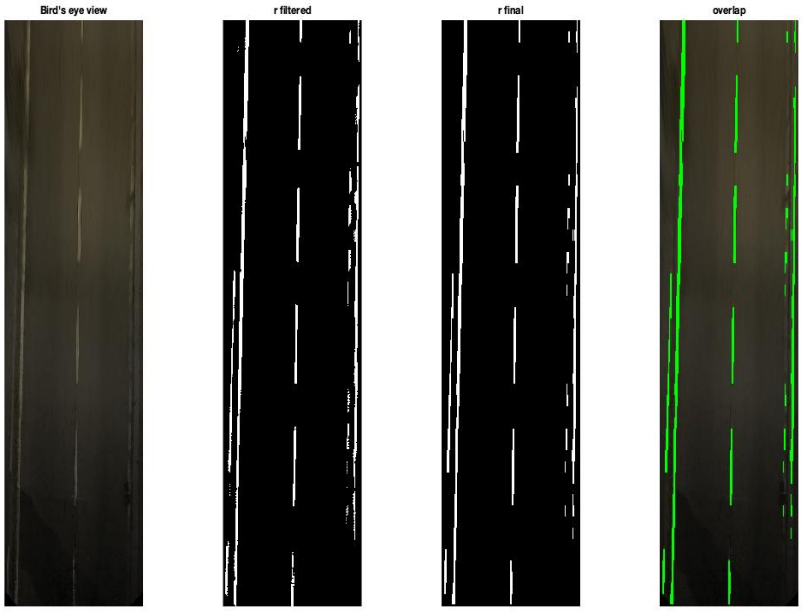

Figure 4.9. Output images obtained using as input the image reported in Figure 4.8

4.3 Image defects

While the previous sections considered the robustness of the lane detection system with respect to different lighting conditions, now the focus is moved to the effect of image defects. In fact, when an outdoor environment is considered, the hypothesis of having an image affected by defects due to dust particles or any other kind of dirt cannot be neglected. The two following sections will evaluate the robustness of the system when there are micro and macro defects on the input image, trying to evaluate their impact on the performances taking also into account the quality of the illumination.

4.3.1 Micro defects

Micro defects means a small imperfection with an order of magnitude of few pixels, these defects can be "hot pixels", i.e. damaged pixels, or, for example, defects generated by dust or other small particles. In order to simulate these defects a "salt & pepper" noise has been added to the original images using the *imnoise()* Matlab® function.

The following figures report the results obtained using the images created as previously described, different tests have been performed considering different moments of the day. The results obtained during the morning (Figure 4.10, 4.11) shown that, in a scenario with a strong illumination, the level of performance is almost unchanged. A similar result has been obtained during the nightfall (Figure 4.14, 4.15) but only with a higher level of sensitivity, achieved by reducing the *threshold* value from 6 to 4. A light performance degradation occurred in the afternoon (Figure 4.12, 4.13) and in the night (Figure 4.16, 4.17) with an increase of the spurious detections. In general, the main effect of this kind of defect is the generation of small spots inside the bird's eye view, which become more elongated in the vertical direction as one consider the reconstruction of a farther part of the road. At a certain point, these "elongated spots" are so long that they can be considered like short vertical stripes, which luckily are too thin to be detected as road markings by the system.

Figure 4.10. Input picture of Corso Castelfidardo during the morning with micro defects

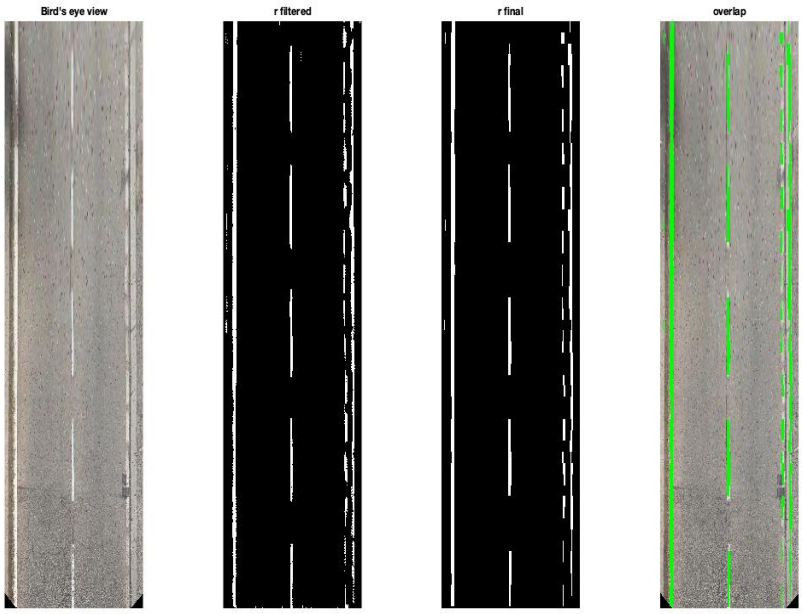

Figure 4.11. Output images obtained using as input the image reported in Figure 4.10

Figure 4.12. Input picture of Corso Castelfidardo during the afternoon with micro defects

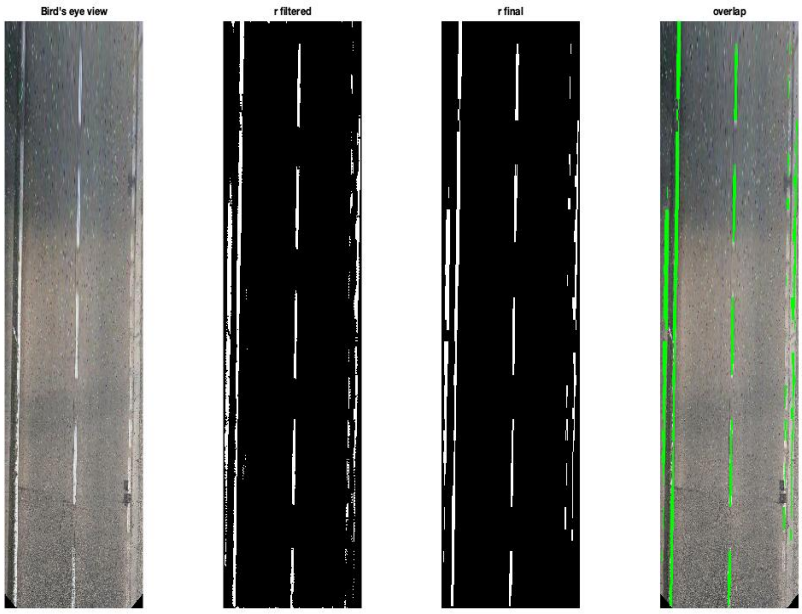

Figure 4.13. Output images obtained using as input the image reported in Figure 4.12

Figure 4.14. Input picture of Corso Castelfidardo during the nightfall with micro defects

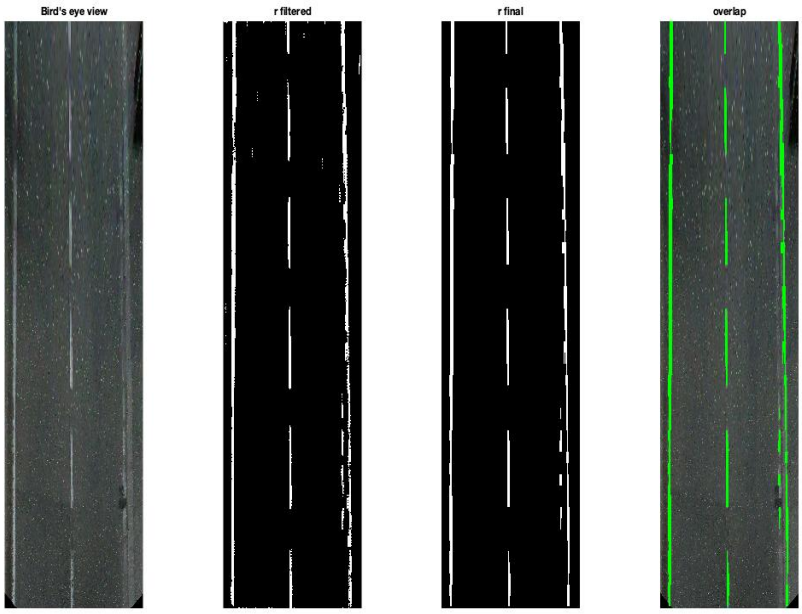

Figure 4.15. Output images obtained using as input the image reported in Figure 4.14

Figure 4.16. Input picture of Corso Castelfidardo during the night with micro defects

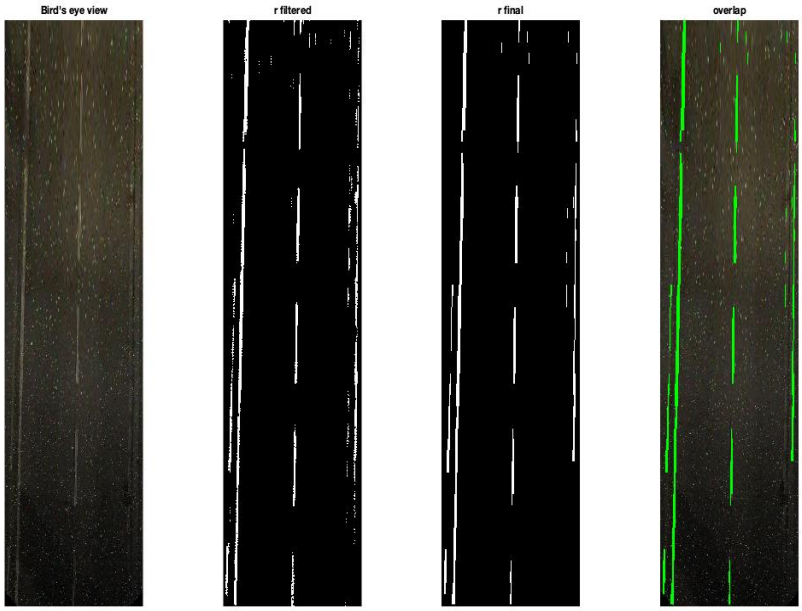

Figure 4.17. Output images obtained using as input the image reported in Figure 4.16

4.3.2 Macro defects

Now that the effects of micro defects have been evaluated, it is time to perform the same procedure for macro defects. Macro defects means any kind of image defect with considerable size, e.g. a spot with a size in the order of magnitude of the squared centimeters. This kind of defect can be produced by several causes like, for example, mud or any kind of dirt on the glass of the camera sensor. In order to evaluate how macro defects affect the performances of the developed lane detection system, a series of synthetic images has been developed using Adobe® Photoshop, in particular, as done for micro defects, all the different moment of the day have been taken into account. A sample of one of these synthetic images is reported in Figure 4.18, where four dark spots with a random shape are placed in different areas of the image.

The performed tests shown that the quality of the illumination does not affect in any way the outcome of the detection, in fact in every light condition the macro defects produced always the same effect. Figure 4.19 shows the output produced using as input the image reported in Figure 4.18, as expected, only the spots that cover a part of the road affected the lane detection process, preventing the detection of the underlying road markings.

Figure 4.18. Input picture of Corso Castelfidardo during the morning with macro defects

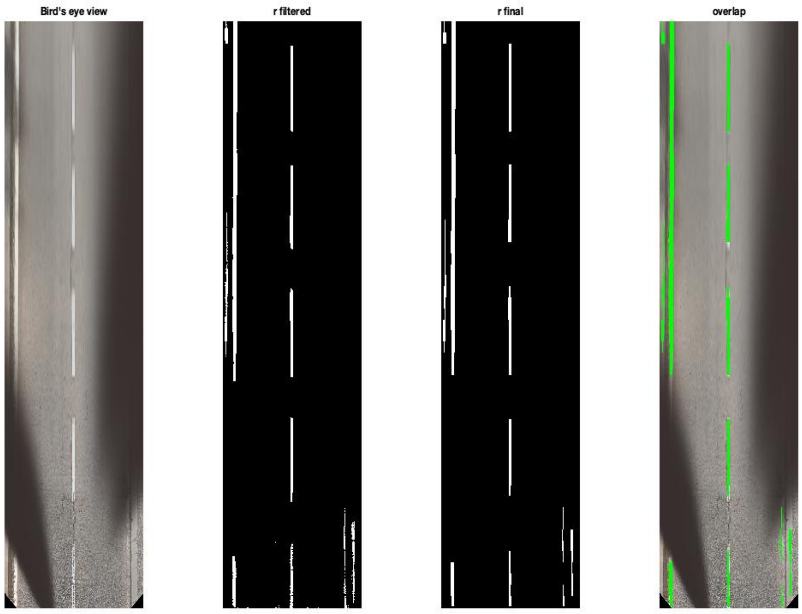

Figure 4.19. Output images obtained using as input the image reported in Figure 4.18

4.4 Weather conditions

At this point of the validation procedure the focus is set on the weather conditions, for reasons of safety and feasibility it was not possible to directly take pictures of the street in critical weather conditions. In order to overcome this problem a set of synthetic images has been created using Adobe® Photoshop, in particular fog and heavy rain have been added to the images of Corso Castelfidardo considering different moments of the day.

4.4.1 Fog

One of the worst weather conditions for what concerns the visibility is the presence of fog. Depending on its intensity, the fog can considerably reduce the field of view, leading in the worst cases to an almost total loss of information. During the tests four conditions have been taken into account: daytime with light fog, daytime with thick fog, nighttime with light fog and nighttime with thick fog.

Figure 4.20 shows Corso Castelfidardo during the daytime with light fog, in this case the field of view is not very reduced but the sharpness of the image is worse. As can be seen from Figure 4.21, using a very high sensitivity (i.e. with the *threshold* value set to 4), the level of performance in this weather conditions is not reduced that much and is comparable with the "standard" conditions. Moreover, in this particular case, the fog has brought a beneficial effect reducing the amount of spurious detections due to the sidewalks. Considering a more critical situation, Figure 4.22 shows Corso Castelfidardo during the daytime with a thick fog that considerably reduces the field of view. In this case, as reported in Figure 4.23, the performance degradation noticed during the tests has been very high, in fact, even with a high level of sensitivity, after a certain point the lines were not detected. Once again, the detection of the dashed lines on the right has been the most problematic and the one where the system shown the worst performances. Passing to the nighttime, in general, the tests shown a significant drop in performance. With light fog, as shown in Figure 4.25, the continuous line on the left was only partially detected while the dashed lines were not detected at all. However, the worst results came out during the tests with thick fog (Figures 4.26 and 4.27), where the system, even with the highest sensitivity available, did not detect nothing. The only improvement brought by the night is that, with light fog, the spurious detections were almost absent.

Figure 4.20. Input picture of Corso Castelfidardo during the daytime with light fog

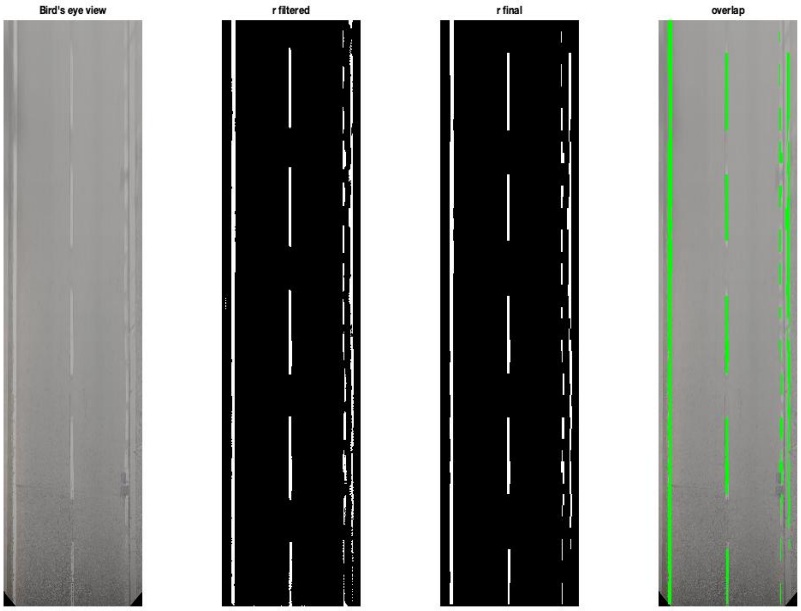

Figure 4.21. Output images obtained using as input the image reported in Figure 4.20

Figure 4.22. Input picture of Corso Castelfidardo during the daytime with thick fog

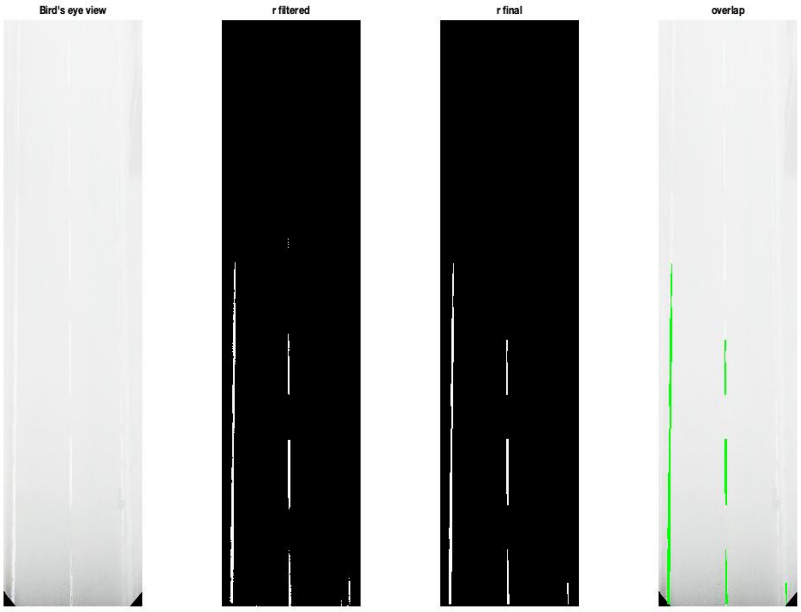

Figure 4.23. Output images obtained using as input the image reported in Figure 4.22

Figure 4.24. Input picture of Corso Castelfidardo during the nighttime with light fog

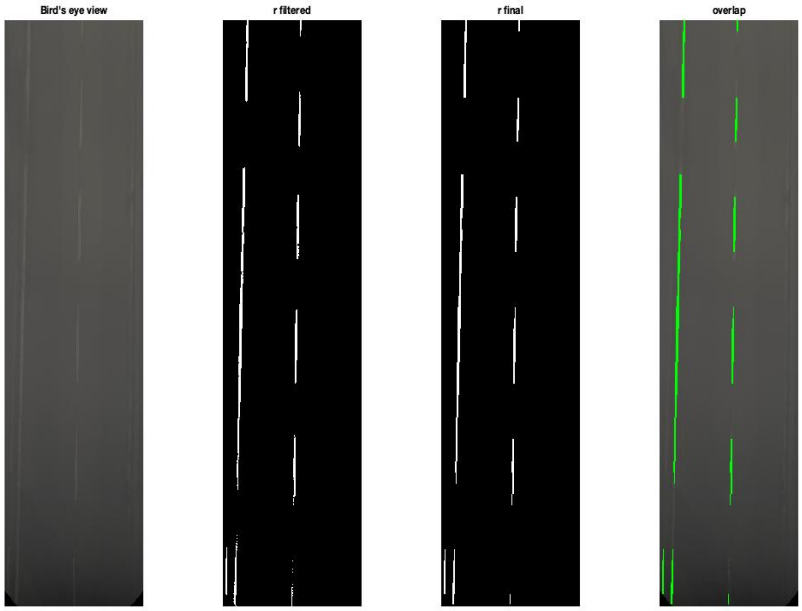

Figure 4.25. Output images obtained using as input the image reported in Figure 4.24

Figure 4.26. Input picture of Corso Castelfidardo during the nighttime with thick fog

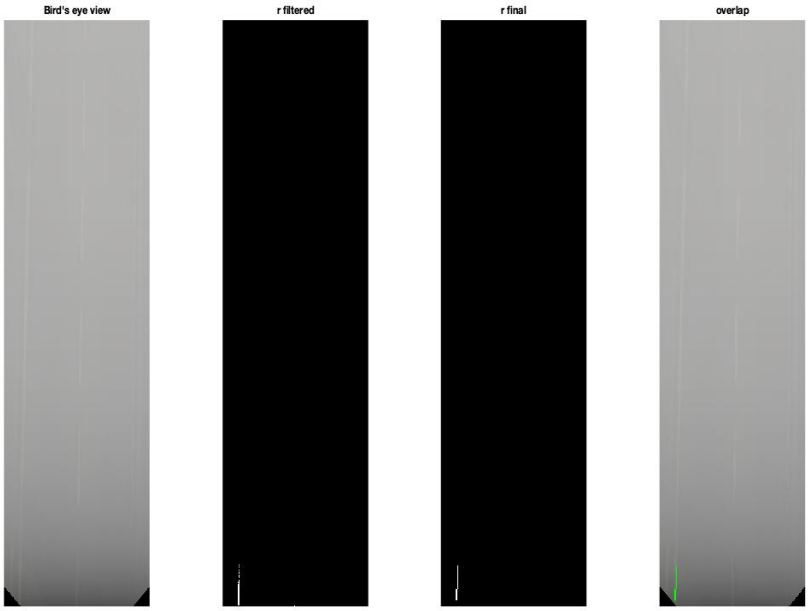

Figure 4.27. Output images obtained using as input the image reported in Figure 4.26

4.4.2 Heavy Rain

As anticipated during the introduction to this section, besides the fog, also the heavy rain will be considered. This particular weather condition can dramatically affect the visibility because of the water drops, which can reduce the field of view, and of the water on the camera glass, which can introduce an additional distortion to the image. In order to reproduce the rain a custom filter has been developed, then, to reproduce the effect generated by the water on the camera glass, the image has been slightly blurred. As done with the fog, also in this case both the daytime (Figure 4.29, 4.30) and the nighttime (Figure 4.31, 4.32) have been considered.

The results of the tests shown that in rainy conditions there is a significant degradation of the performances. Looking at figures 4.29 and 4.31 it is possible to see that, as happened with the thick fog, after a certain point the system is not able to detect the lane. Moreover, the tests shown that, especially during the nighttime, passing from the original image to the bird's eye view, the rain produces some vertical lines that can generate spurious detections. In order to reduce these spurious detections and improve the overall performances, the sensitivity has been reduced using a *threshold* value equal to 8, allowing the system to reject a bit more the lines generated by the rain. The results obtained with a higher sensitivity (*threshold* = 4) can be seen in Figure 4.28, which shows the overlap of the detected lines (highlighted in green) and the bird's eye view. It is immediate to see that there are much more spurious detections and that the performances using a lower sensitivity are considerably better. During all these tests the input images have been produced introducing just a soft blur trying to reproduce the effect of the water on the camera glass. It is clear that, for a vehicle that travels during a violent rainstorm, the water that covers the camera glass can produce a significantly stronger distortion on the input image with respect to one that has been considered during the tests. However, since the results obtained until now shown that the system does not provide sufficient performances to work in rainy conditions, were not considered further worse cases.

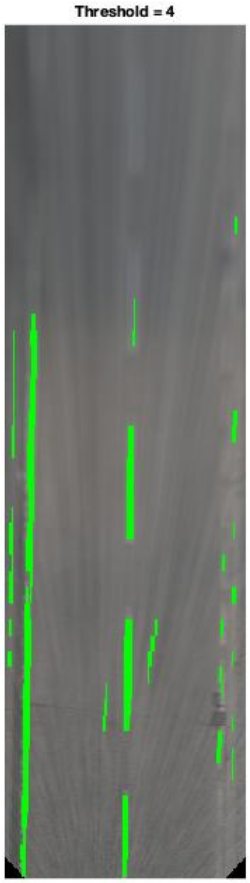

Figure 4.28. Final result in heavy rain conditions during daytime with threshold = 4, (higher sensitivity)

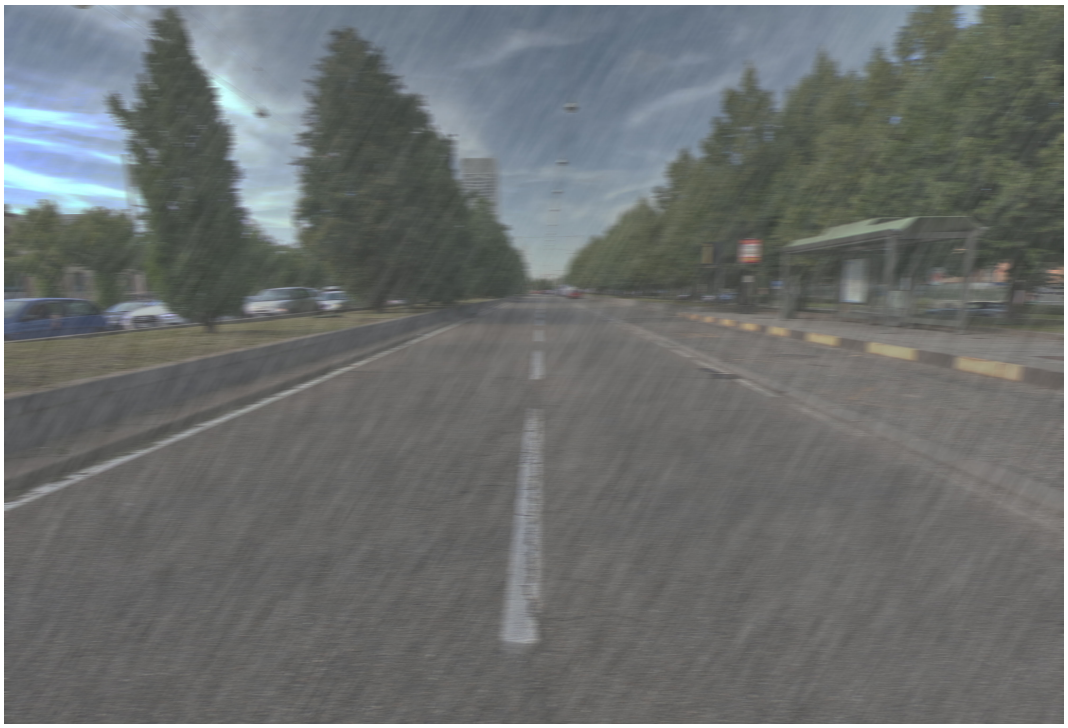

Figure 4.29. Input picture of Corso Castelfidardo during the daytime with heavy rain

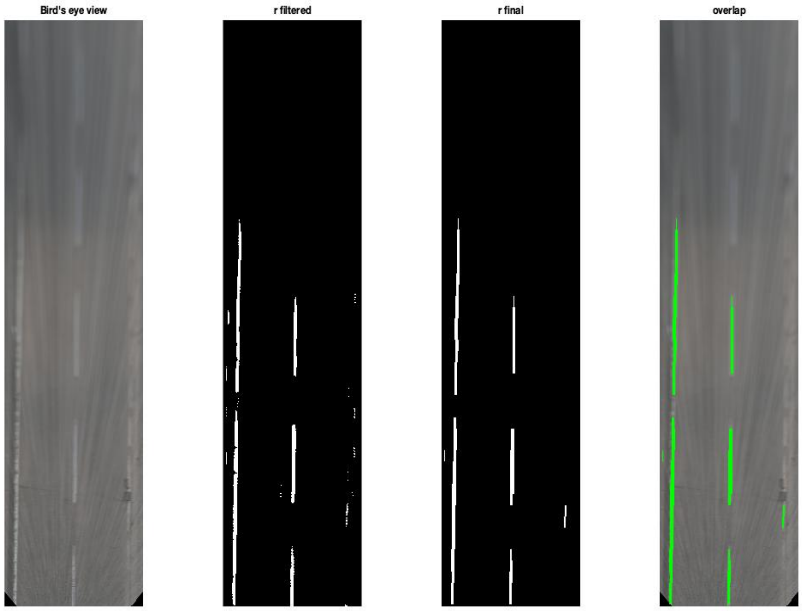

Figure 4.30. Output images obtained using as input the image reported in Figure 4.28

Figure 4.31. Input picture of Corso Castelfidardo during the nighttime with heavy rain

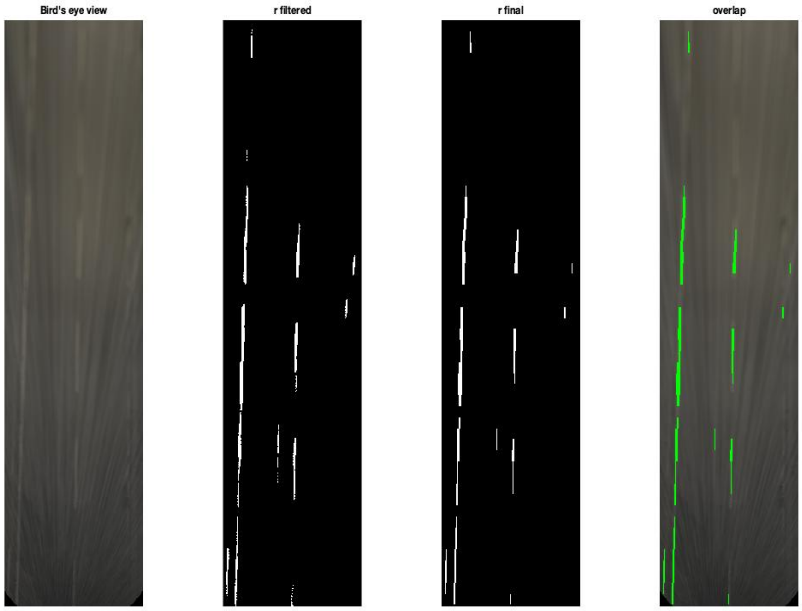

Figure 4.32. Output images obtained using as input the image reported in Figure 4.30

4.5 Bad asphalt conditions and colored road markings

During the previous phases of this validation procedure, several atmospheric conditions have been taken into account considering always a street in good conditions with clear road markings. In this last part instead, the goal is to evaluate the effects of a ruined asphalt with colored road markings in optimal weather conditions. In order to do that, since the characteristic of the road are almost perfect for the purposes explained previously, a picture of Via Sant'Antonio da Padova (Figure 4.33) has been taken during the daytime. The results obtained using this picture as input are reported in Figure 4.34, the first thing that can be noticed is the fact that only the yellow lines have been properly detected while the blue ones were not individuated by the system. Another thing that it is worth noting is that, since they are parked very close to the lane, the vehicles along the sides of the road produced some dark shapes on the bird's eye view, leading to spurious detections that could not be removed even reducing the sensitivity of the system. In general, the performances of the algorithm in this kind of scenario cannot be absolutely considered satisfactory, the only positive thing that can be noticed from this test is that the bad asphalt conditions did not lead to any spurious detection.

4.6 Conclusion

Even if many other possible scenarios with many other variables can be considered, this validation procedure allowed to evaluate the robustness of this lane detection system and to highlight its limits. In general the quality of the results was good, with an acceptable level of efficiency for most of the "standard" conditions. Nevertheless, all the limits of an ADAS based exclusively on vision came out, with a degradation of the performances every time that there was a significant reduction of the visibility (e.g. macro defects, fog or heavy rain). Moreover, the performances in streets with bad asphalt conditions and unclear road markings was very bad, with the system that has not detected most of the lines. Another aspect that was highlighted by the tests is the fact that, in order to obtain the best possible performances, there is the need of a sensors system which allows to set the level of sensitivity depending on the different scenarios.

At the end of this validation procedure it is possible to conclude that, considering the level of performance obtained in all the different conditions, the system can be used only in streets with very good asphalt conditions and only in non-critical situations, where the visibility is just slightly reduced.

Figure 4.33. Input picture of Via Sant'Antonio da Padova

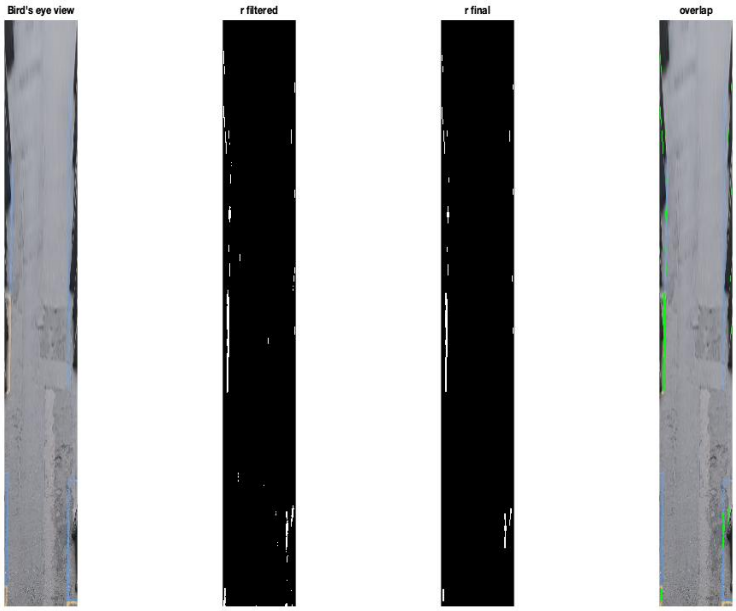

Figure 4.34. Output images obtained using as input the image reported in Figure 4.33

Part IV

Validation of the YOLOv3 object detection system

Chapter 5

Introduction

One of the main goals of this work is to analyze vision processes used to achieve a certain level of autonomous driving in order to evaluate their robustness. While previously the focus has been set on the lane detection process, in this second part object detection will be considered. Since it allows to detect and recognize different classes of objects inside an input image, the capability of perform this process represents a crucial feature for what concern autonomous driving. During the bibliographic analysis several object detection methods have been considered, explaining their working principles and reporting their performances in terms of speed and precision. Between all the methods seen during this analysis the one with the best overall performances is YOLOv3, which is the latest version of the YOLO object detection system. One of the features that makes YOLOv3 particularly interesting is its speed, in fact it is one of the few object detection systems able to work in real time with a decent level of mAP. The ability of working in real time is a must when one works inside the automotive field, especially for safety applications, where there are stringent constraints for what concerns timing.

Usually, the approach adopted to validate systems that use neural networks is based on the analysis of huge datasets. Considering object detection, after a certain period of training, the system is tested using a series of input images, which allow to estimate the mean average precision (mAP) for each considered object class. Both the processes of training and validation require a lot of time, even when are used machines with a very high computational power. Moreover, since the input images used for training cannot be used again for the validation process, in order to avoid overfitting, the creation of a training dataset can be very long. Overfitting is a phenomenon where the system shows better performances than the actual ones. This is mainly due to the fact that, when one uses images belonging to the training dataset, the system already "knows" these images and can individuate all the objects easily. The results of many validation processes performed in this "traditional" way can be found in the bibliography, for example Figure 5.1

[9] shows the scores obtained by many different object detection systems with the Pascal VOC 2012 challenge. Since the main goal of this work is to evaluate how

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [27]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [28]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [32]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

Figure 5.1. Pascal VOC 2012 Leaderboard taken from [9]

the system under test reacts in terms of performances to the variation of different parameters, the validation process will be performed with an approach similar to the one used previously for the lane detection system. In particular, instead of considering thousands of different input images, a small but much more specific dataset has been produced. In order to do that in a safe way, a generic automotive scenario has been recreated inside a parking area, then a picture has been taken for each different moment of the day, allowing to consider all the possible lighting conditions. The final dataset has been finally obtained modifying these images to take into account also the effects of micro and macro defects, different resolutions and different weather conditions.

Figure 5.2 shows the automotive scenario used for the tests, it contains: three different kind of cars (i.e. a van, a station wagon and a small utilitarian car), two different road signs and two pedestrians. All these elements have been chosen to recreate a realistic and heterogeneous environment, trying to consider as many objects as possible. To evaluate the system range, all the different kind of objects have been arranged at different distances: the two pedestrians are placed at about 10 and 30 meters respectively, the stop sign are placed at 10 meters while the other road sign is 30 meters away, the yellow car is placed at 6 meters with respect to the camera while the van and the station wagon are approximatively 10 and 17

meters away respectively. The camera used during the tests is a Casio® Exilim with a 12.1 Mega pixels sensor. This choice is due to the fact that both the cost and the quality of the sensor inside this device are suitable for automotive purposes.

Each section of the following chapter will treat the effects of a different "variable" (i.e. lighting, weather conditions, micro and macro defects) processing the images contained into the dataset with a pre-trained version of YOLOv3. Every variable will be analyzed considering two different resolutions, full size (4000×3000) and VGA (640×480), in order to understand if a higher resolution leads to better results. During this validation procedure all the critical issues will be highlighted, trying, if possible, to speculate about how they can be solved or at least mitigated.

Figure 5.2. Automotive scenario recreated for the tests

Chapter 6

Dependencies on image quality and corner cases

6.1 Lighting

Like for every system based on a vision process, lighting is crucial for object detection. As explained during the bibliographic analysis, object recognition is often performed through edge detection, which is a process where the direction and the intensity of the light can affect heavily the final result. So the first parameter that is going to be evaluated is the lighting, in order to do that four different moments of the day characterized by different light conditions will be considered (i.e. morning, afternoon, sunset and night). For the tests performed in the nighttime, car lights have been considered both on and off to evaluate how they affect the performances.

6.1.1 Morning

Let's start in a chronological order with the first part of the day: the morning. During the morning there are the best lighting conditions, in fact the light is strong and uniformly distributed, an ideal condition for any vision process. Figure 6.1 shows the results of the tests performed during the morning, on the left column there are the input images while on the right one the respective output images can be seen. The difference between the two input images is the pedestrians orientation, which has been changed to evaluate if the system somehow obtains better results in one case rather than the other. As can be seen from these images, the quality of the results is very good, in fact all the objects have been properly recognized with the only exception of the white van, which is partially hidden by the yellow car in the foreground, and the triangular road sign, which is evidently too far to be recognized. The only mistake made by the system is a spurious detection of a person inside the yellow car, which was strangely detected only in the image with the pedestrians in profile at full resolution.

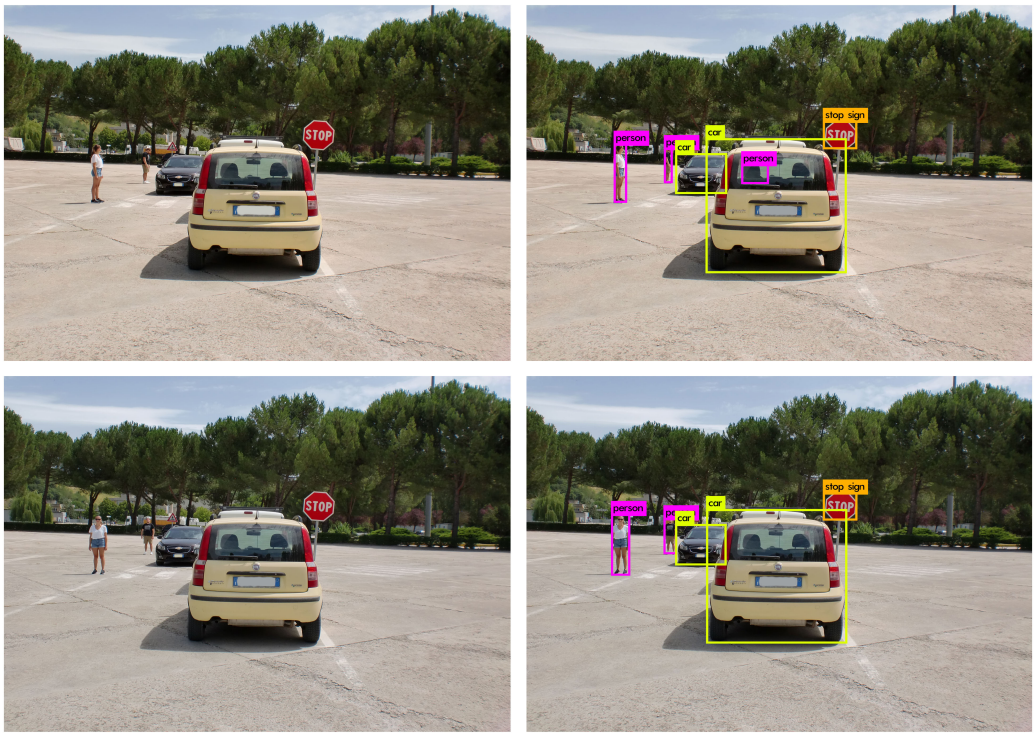

Figure 6.1. Results of the tests performed during the morning

As mentioned previously, the tests have been performed with two different resolutions (full size and VGA), in both cases the system worked approximatively in the same way providing satisfactory results. Surprisingly the test that produced the worst results in terms of confidence score was the one at full resolution with the pedestrians in profile. The following table reports all the confidence scores obtained during the tests ("profile" and "front" are referred to the orientation of the pedestrians):

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	100%	100%	100%
person (boy)	99%	99%	99%	98%
yellow car	87%	92%	90%	94%
black car	100%	100%	100%	98%
person (error)	64%	//	//	//

6.1.2 Afternoon

The second part of the day that has been considered is the afternoon, in this case the amount of light is slightly smaller with respect to the morning but, because of the sun position, there are more shadows and in general a worst distribution of the light. Figure 6.2 shows the results obtained during this part of the day, as expected, the system detected all the object that have been detected previously during the morning. The unexpected fact, instead, is that in this case the confidence score for every resolution and orientation of the pedestrians is higher with respect to the morning. Probably the reason of this improvement of the results is that the sun position made the asphalt a bit darker, increasing the contrast between the asphalt and the contour of the objects. As done for the morning, all the results are reported in the table below:

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	100%	100%	100%
person (boy)	100%	100%	100%	100%
yellow car	97%	98%	97%	98%
black car	100%	100%	100%	99%

6.1.3 Sunset

Proceeding in chronological order, the analysis continues with the sunset. In this case the lighting conditions are very close to those that one has during the afternoon but a bit more emphasized (i.e. darker images and more shadows). As expected, the quality of the results is comparable to the one obtained during the afternoon, with just a little decrease of the confidence score, especially for the yellow car. The results are reported in Figure 6.3 and reassumed in the following table:

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	99%	99%	100%	100%
person (girl)	100%	100%	100%	100%
person (boy)	100%	99%	99%	99%
yellow car	94%	91%	94%	92%
black car	99%	99%	99%	99%

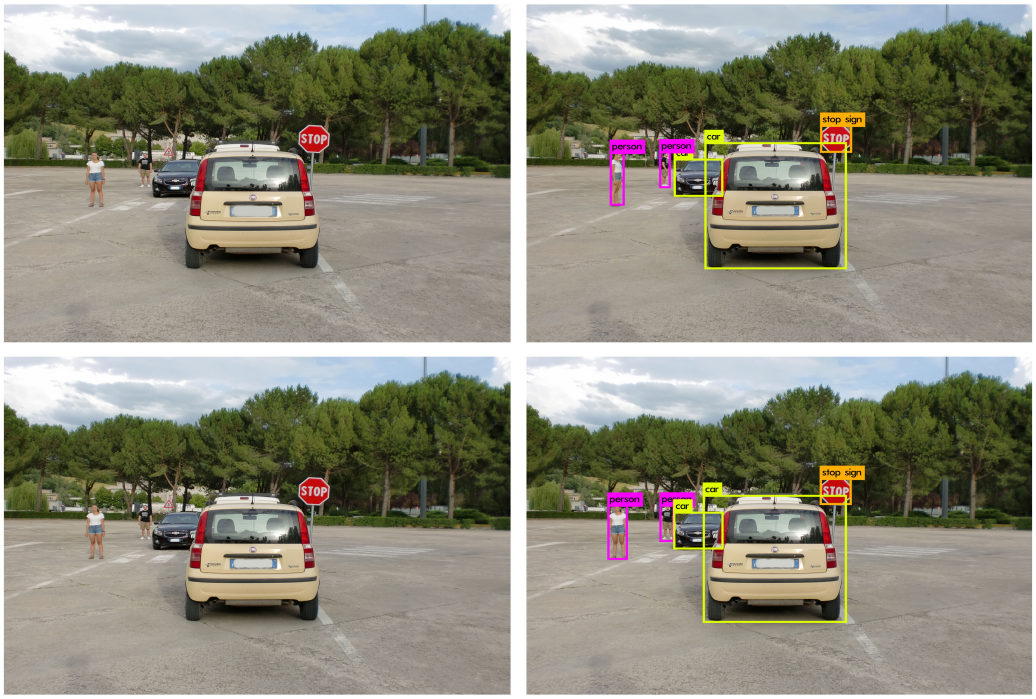

Figure 6.2. Results of the tests performed during the afternoon

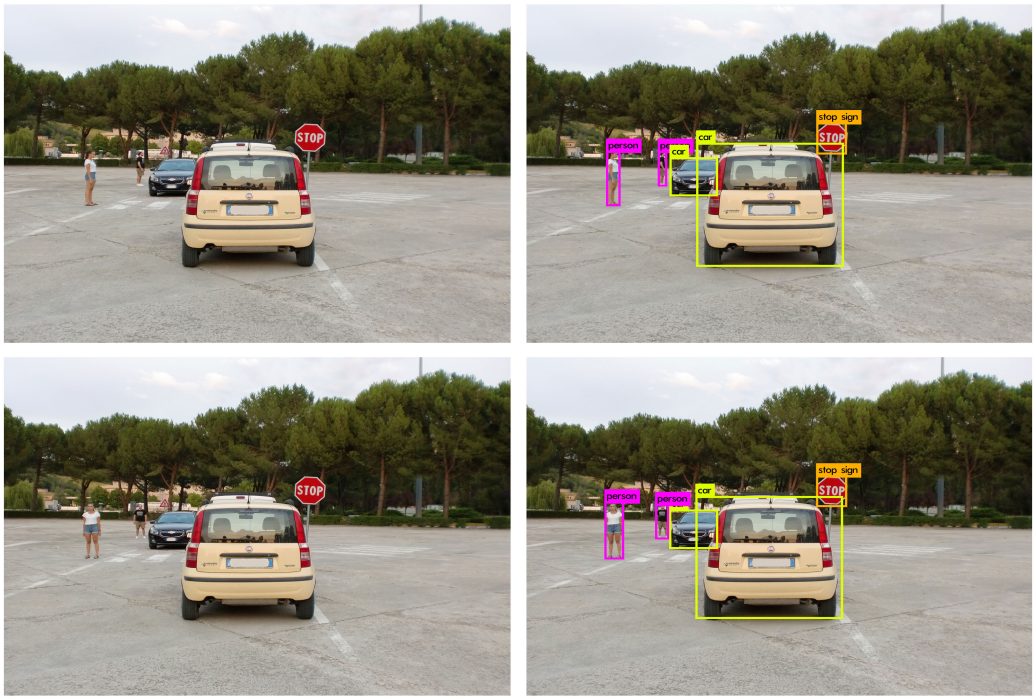

Figure 6.3. Results of the tests performed during the sunset

6.1.4 Night

Finally, it is time to analyze the most critical condition for what concerns the lighting condition, i.e. the night. In the nighttime the only sources of light are the artificial road lights and the car lights (and sometimes the moon). As anticipated previously the tests have been performed considering both the cases with the car lights turned on and off, trying to evaluate how and if they can improve the performances. Surprisingly the quality of the results remained mostly unchanged in both cases, the only object that shown a reduction of the confidence score was the yellow car, which is in shadow because of the location of the artificial lights. Anyway the use of car lights improved the overall performances, allowing to increase the confidence score in most of the cases. A curious thing is that, at full resolution and with car lights turned on, the system supposed also that the yellow car could be a truck with the 58% of probability. This is probably due to the "boxy" shape of the car and to the low light, anyway, it is not a big issue but just an imprecision.

Figures 6.4 and 6.5 reports the results obtained with car lights turned on and off with full resolution, figures 6.6 and 6.7, instead, reports the same results but using images with VGA resolution. The following tables summarize the results obtained in both the conditions:

Car lights OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	99%	99%	99%	99%
person (girl)	100%	100%	100%	100%
person (boy)	99%	99%	100%	100%
yellow car	66%	85%	58%	79%
black car	99%	99%	99%	96%

Car lights ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	99%	99%	100%	100%
person (boy)	98%	98%	98%	99%
yellow car	87%	89%	86%	84%
black car	99%	98%	99%	98%
truck	58%	//	//	//

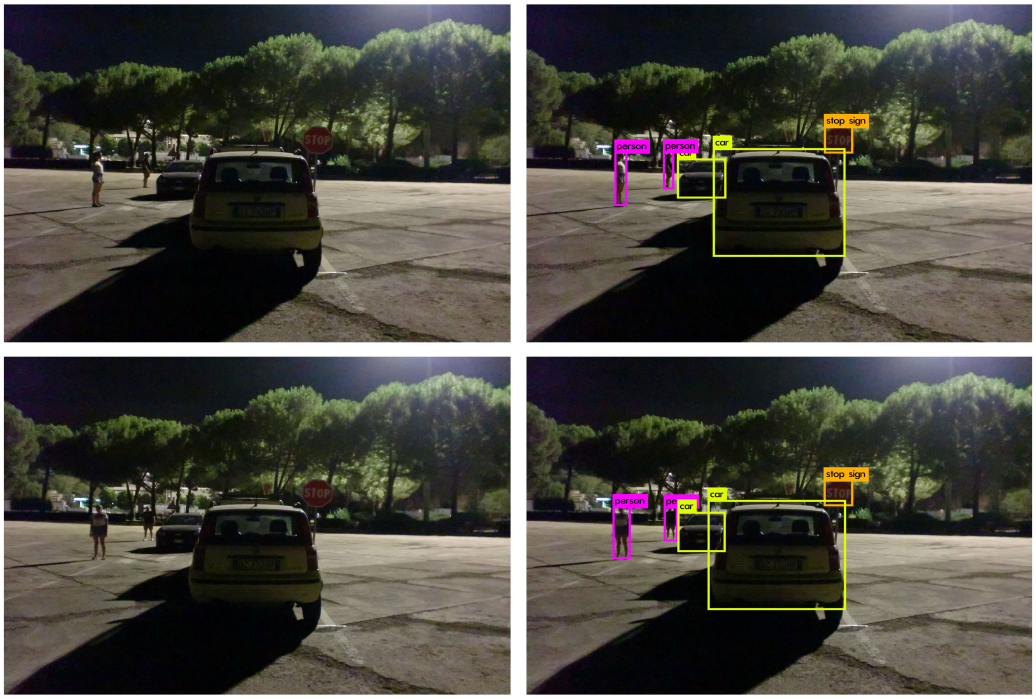

Figure 6.4. Results of the tests performed during the night with full resolution (car lights OFF)

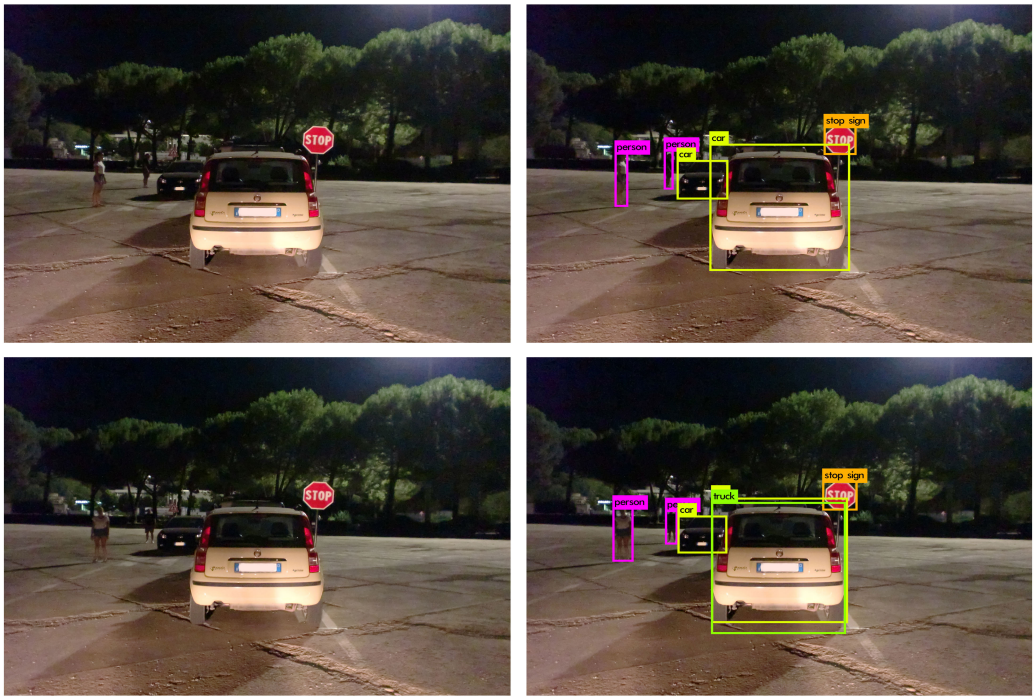

Figure 6.5. Results of the tests performed during the night with full resolution (car lights ON)

Figure 6.6. Results of the tests performed during the night with VGA resolution (car lights OFF)

Figure 6.7. Results of the tests performed during the night with VGA resolution (car lights ON)

6.2 Defects

Now that the impact on the performances of different lighting conditions have been evaluated, the validation process goes on with the analysis of another important aspect which can affect considerably the performances, that is the presence of defects. When one deals with an outdoor environment, there is the possibility of having external agents that can make the lens of the camera dirty (e.g. mud, dust and other particles, etc...), generating some macro defect on the input images like spots or marks. Moreover one can have also micro defects, due, for example, to digital noise or burned pixels. During the following analysis, all these defects will be evaluated to understand how and if they can affect the performances.

6.2.1 Micro defects

The first kind of defects that will be considered are micro defects. Even if the dimension of these defects is very small, when one deals with a system that analyzes small groups of pixels, they can lead to a significant performance degradation. In order to add micro defects to the original images the *imnoise()* Matlab® function has been used, specifying a "salt & pepper" noise.

The following figures (from 6.8 to 6.15) show the results obtained with this kind of defects considering different lighting conditions (morning, afternoon and night). While during the previous analysis, where has been considered the effect of the illumination, the results were mostly coherent with the expectations, in this case a lot of curious things came out from the test results, especially concerning the tests performed during the nighttime. In fact, if the results with a high level of illumination (morning and afternoon) were almost identical to the ones obtained without any defects, the tests with low lights highlighted a substantial loss of performance.

As can be seen from figures 6.12 and 6.13, the system did not detect the yellow car at every resolution, while at VGA resolution the system did not detect even the stop sign. The black car instead were not detected only at VGA resolution with the pedestrians oriented in front of the camera. This result is very significant if we think that a typical micro defect is the digital noise due to low light conditions. So, during the design procedure, a camera with a high noise rejection will be advisable for autonomous driving purposes. Anyway, since the law imposes to have car lights turned on during night driving, discarding the possibility of a failure, the tests performed in this condition should be more realistic. Luckily, as it is possible to see in figures 6.14 and 6.15, the results obtained in this case show that the better illumination allows to improve significantly the performances. Indeed the system detected all the objects correctly, with the only exception of the image at VGA resolution with the pedestrians in profile, where the female pedestrian (the closer

one) was not found by the system. Another thing that can be noticed is the fact that, during the tests performed in the afternoon at VGA resolution, the system assumed also the hypothesis that the yellow car could be a truck. The same thing happened also without micro defects but only during the nighttime at full resolution and with car lights turned on.

The following tables reports all the results in terms of confidence score:

Morning)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	99%	100%	100%
person (boy)	100%	97%	98%	93%
yellow car	89%	87%	83%	89%
black car	100%	100%	100%	100%

Afternoon)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	99%	100%	100%
person (boy)	100%	99%	100%	99%
yellow car	77%	84%	92%	66%
black car	99%	100%	100%	99%
truck	//	59%	//	66%

Night with car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	97%	//	96%	//
person (girl)	99%	97%	100%	99%
person (boy)	99%	97%	99%	99%
yellow car	//	//	//	//
black car	97%	65%	94%	//

Night with car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	99%	100%	100%
person (girl)	98%	//	100%	94%
person (boy)	98%	79%	91%	93%
yellow car	75%	64%	89%	89%
black car	96%	84%	98%	91%

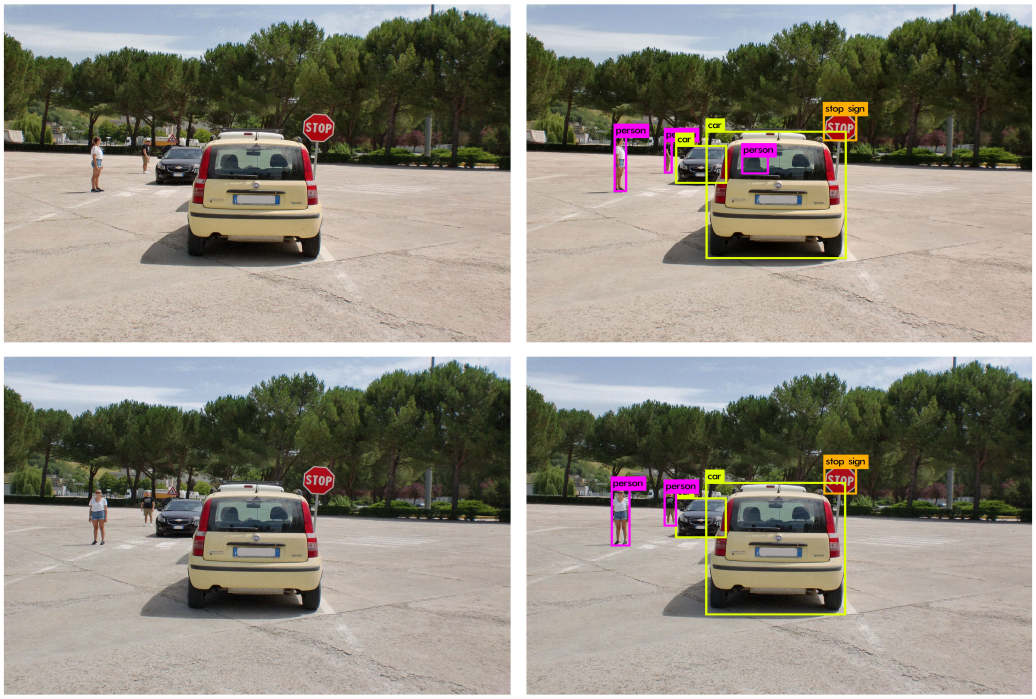

Figure 6.8. Results of the tests performed during the morning with micro defects at full resolution

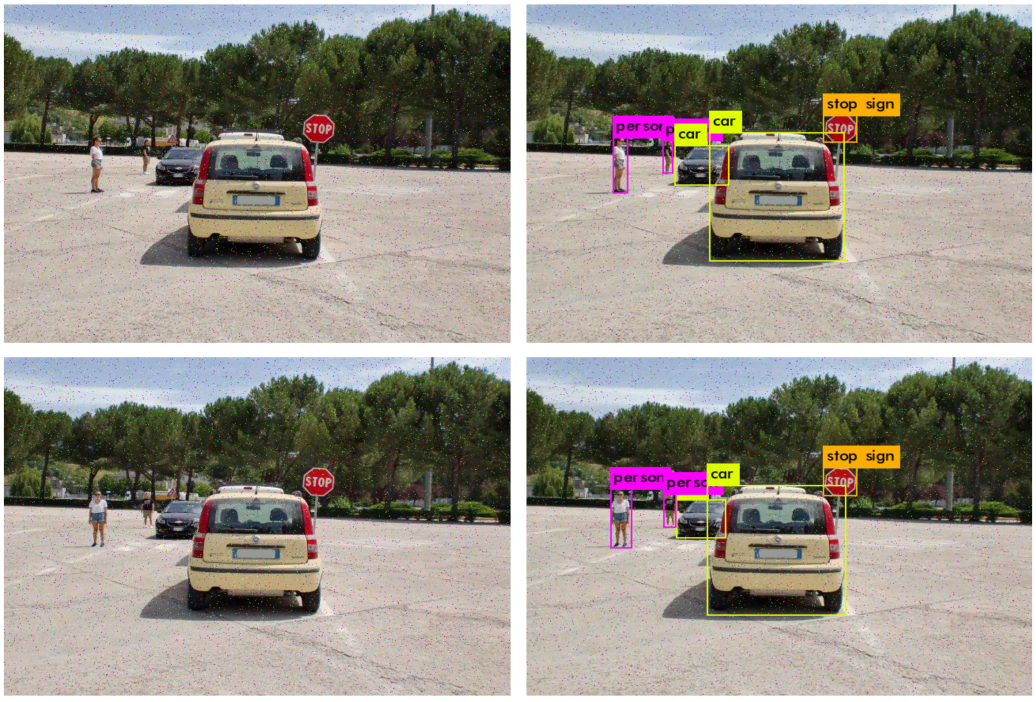

Figure 6.9. Results of the tests performed during the morning with micro defects at RGB resolution

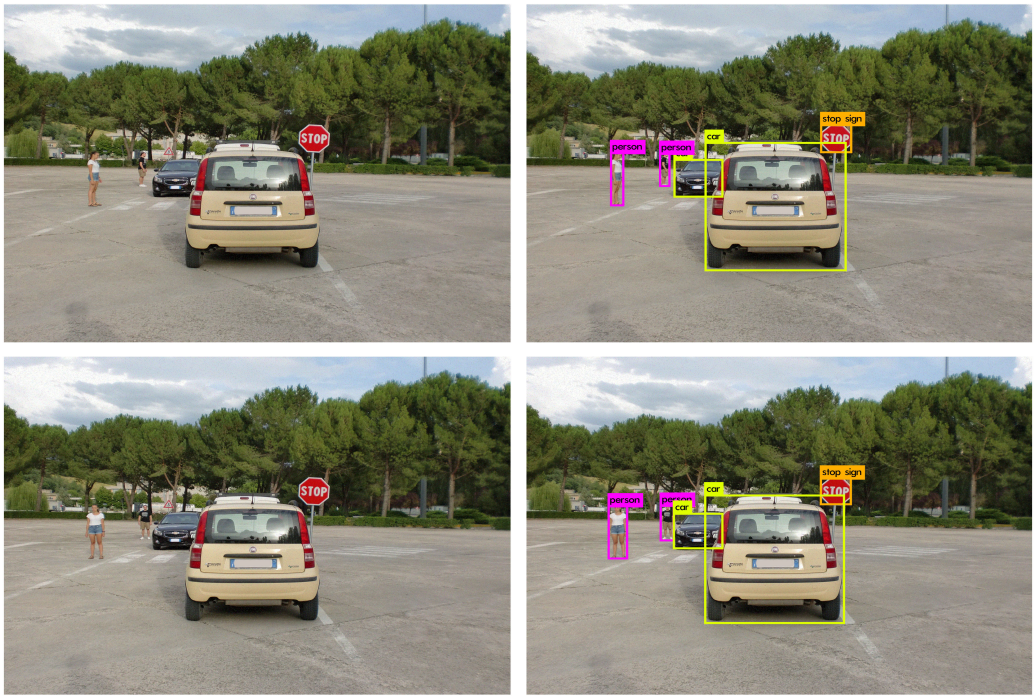

Figure 6.10. Results of the tests performed during the afternoon with micro defects at full resolution

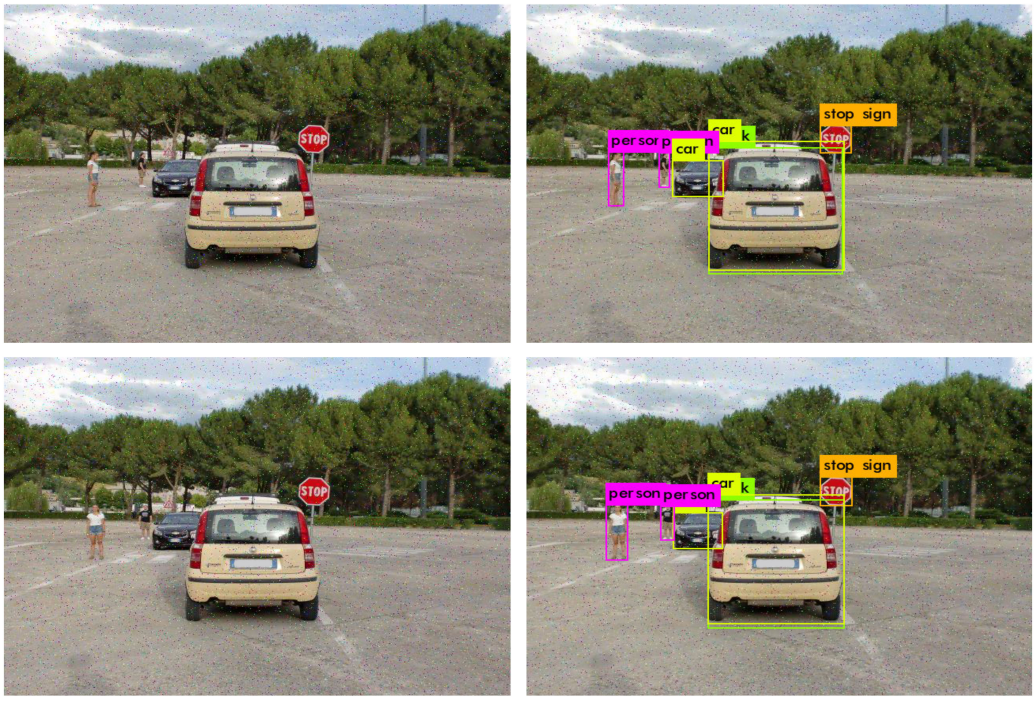

Figure 6.11. Results of the tests performed during the afternoon with micro defects at VGA resolution

Figure 6.12. Results of the tests performed during the night with micro defects at full resolution (car lights OFF)

Figure 6.13. Results of the tests performed during the night with micro defects at VGA resolution (car lights OFF)

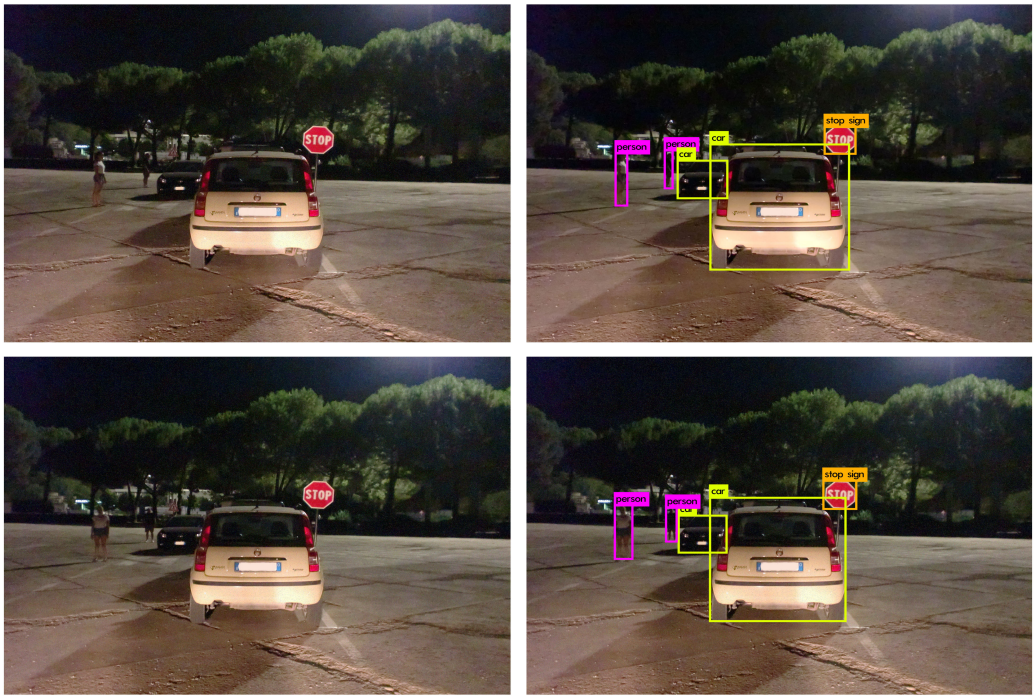

Figure 6.14. Results of the tests performed during the night with micro defects at full resolution (car lights ON)

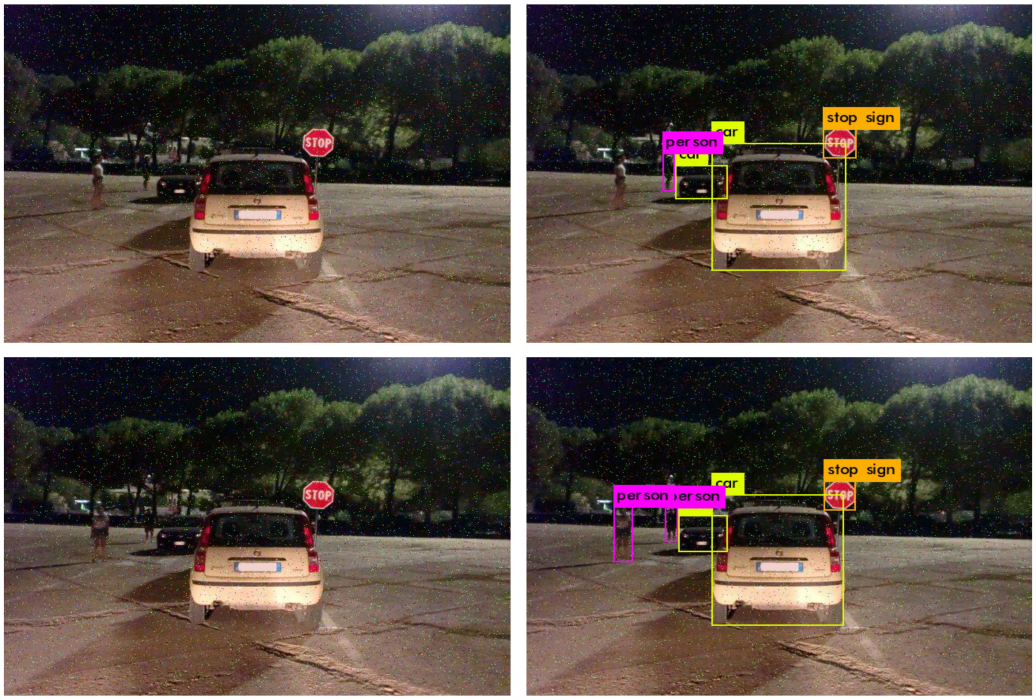

Figure 6.15. Results of the tests performed during the night with micro defects at VGA resolution (car lights ON)

6.2.2 Macro defects

The second and last kind of defects that will be treated are macro defects. To reproduce macro defects like spots and marks, the original images have been modified using Photoshop®, in particular some brown spots have been added to partially cover some of the objects inside the input images. The main goal of this test is to evaluate the ability of the system to recognize objects that are not completely visible considering different light conditions.

Figures from 6.16 to 6.23 show the results obtained, the tests have been performed considering the same conditions evaluated during the analysis of micro defects. Since the macro defects have been added using always the same "mask", the position of the spots is the same in every image. The position of the objects, instead, can be slightly different from an image to another because of the small movements of the camera easel. So, depending on the different moment of the day, the spots will cover the objects in a slightly different way. This thing is not necessarily bad, in fact it allows to evaluate more levels of overlap between the objects and the spots. In general the system produced decent results considering the entity of the defects, however there are a lot of errors and imprecisions that have been noticed during the tests and that it is worth to report.

Going in chronological order, during the morning (figures 6.16 and 6.17) the system produced two overlapped bounding boxes for the yellow car: one around the visible part of the car and another one including both the car and the spot close to it. The curious thing is that the biggest bounding box (the one including both the car and the spot) achieved a higher confidence score with respect to the smaller one including only the car (which is the right bounding box). As regards the afternoon (figures 6.18 and 6.19), the original image is such that the spots on the mask cover in a more considerable way the farthest pedestrian and the stop sign. The result is that, at every resolution, the stop sign were not detected while the pedestrian were detected only when in profile. Concluding with the night (figures 6.20 and 6.21), the tests shown that in low light conditions the performance degradation is much higher, moreover the car lights did not help to improve the results. The most relevant errors are probably that the farther pedestrian were never detected and that the closest pedestrian was not detected when in profile at VGA resolution. Moreover, at full resolution with the car lights turned off and with the pedestrian in profile, the system confused the biggest spot with a person. Another mistake made by the system is that the stop sign, which also in this case is more covered by the spot, was never detected (but it was predictable because it happened also for the afternoon).

A recap of all the confidence scores is reported in the following tables:

Morning)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	99%	100%	98%
person (girl)	96%	98%	99%	94%
person (boy)	96%	97%	98%	94%
yellow car	//	80%	72%	87%
black car	99%	99%	98%	99%
person (error)	58%	//	//	//
car (error)	74%	70%	70%	60%

Afternoon)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	//	//	//	//
person (girl)	93%	95%	99%	96%
person (boy)	79%	77%	//	//
yellow car	96%	97%	97%	98%
black car	99%	98%	98%	98%

Night with car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	//	//	//	//
person (girl)	87%	55%	68%	78%
person (boy)	//	//	//	//
yellow car	70%	74%	69%	58%
black car	90%	97%	92%	73%
person (error)	//	//	51%	//

Night with car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	//	//	//	//
person (girl)	//	//	74%	79%
person (boy)	//	//	//	//

yellow car	93%	92%	94%	94%
black car	80%	50%	76%	79%

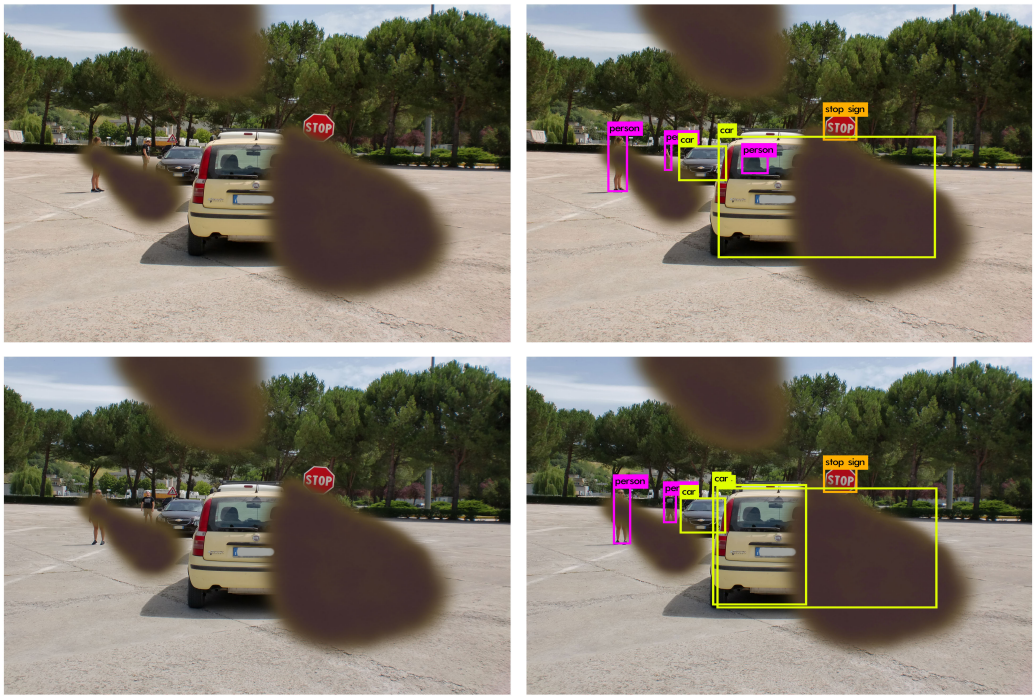

Figure 6.16. Results of the tests performed during the morning with macro defects at full resolution

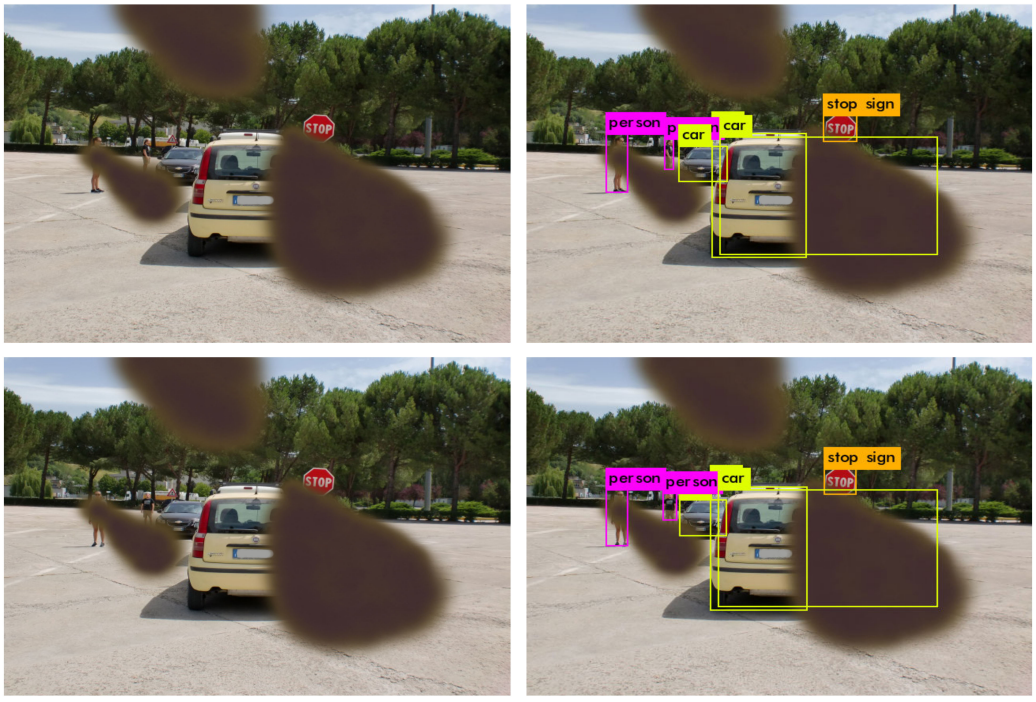

Figure 6.17. Results of the tests performed during the morning with macro defects at VGA resolution

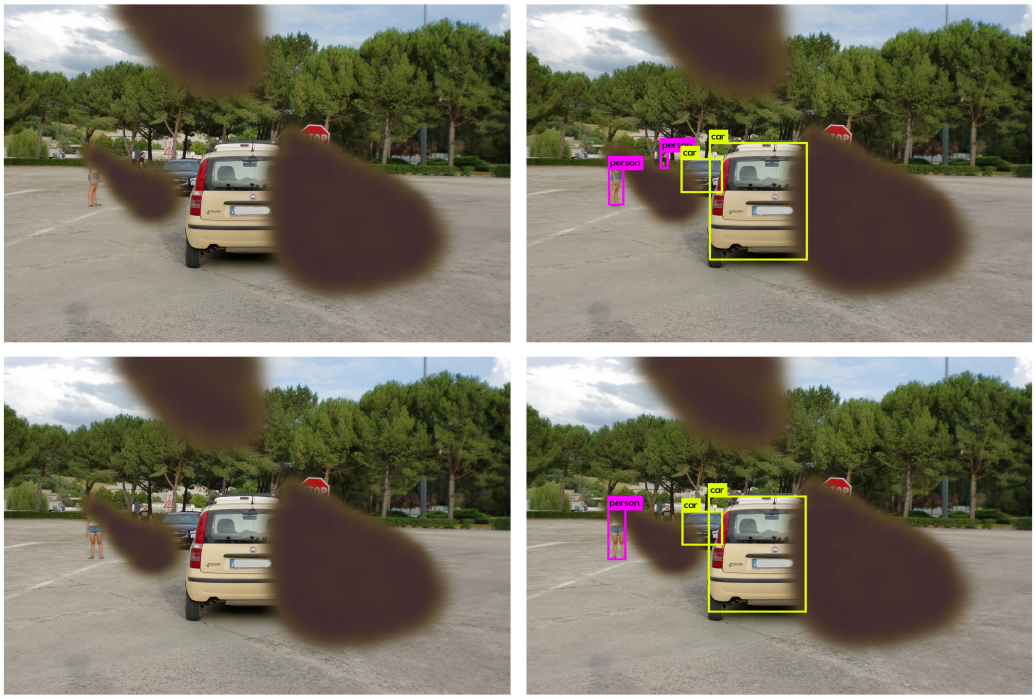

Figure 6.18. Results of the tests performed during the afternoon with macro defects at full resolution

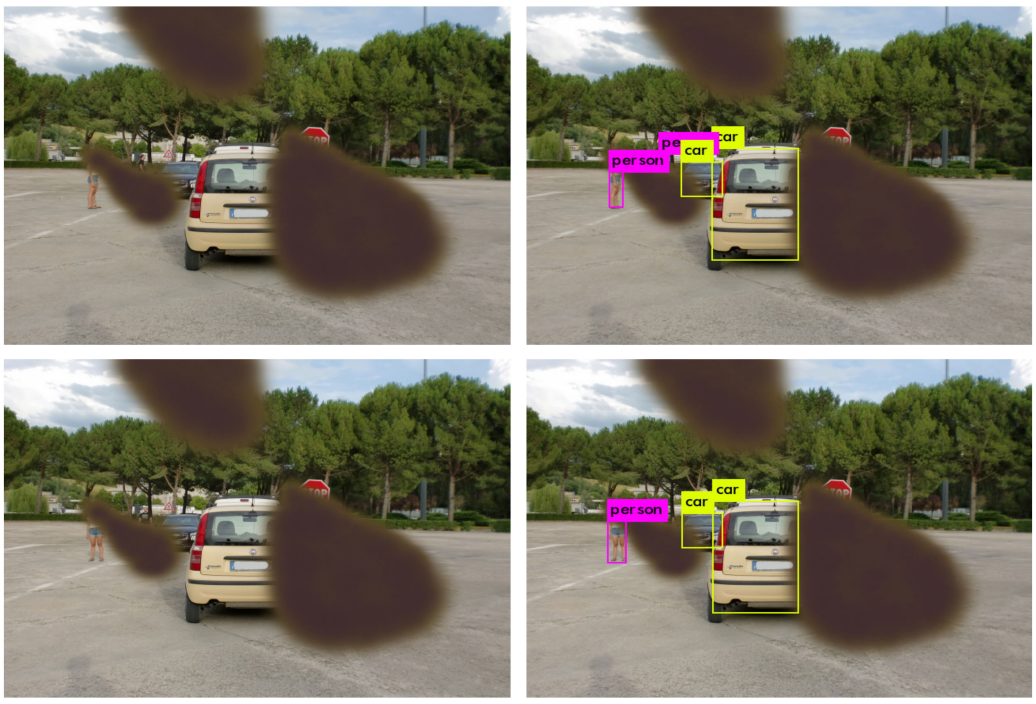

Figure 6.19. Results of the tests performed during the afternoon with macro defects at VGA resolution

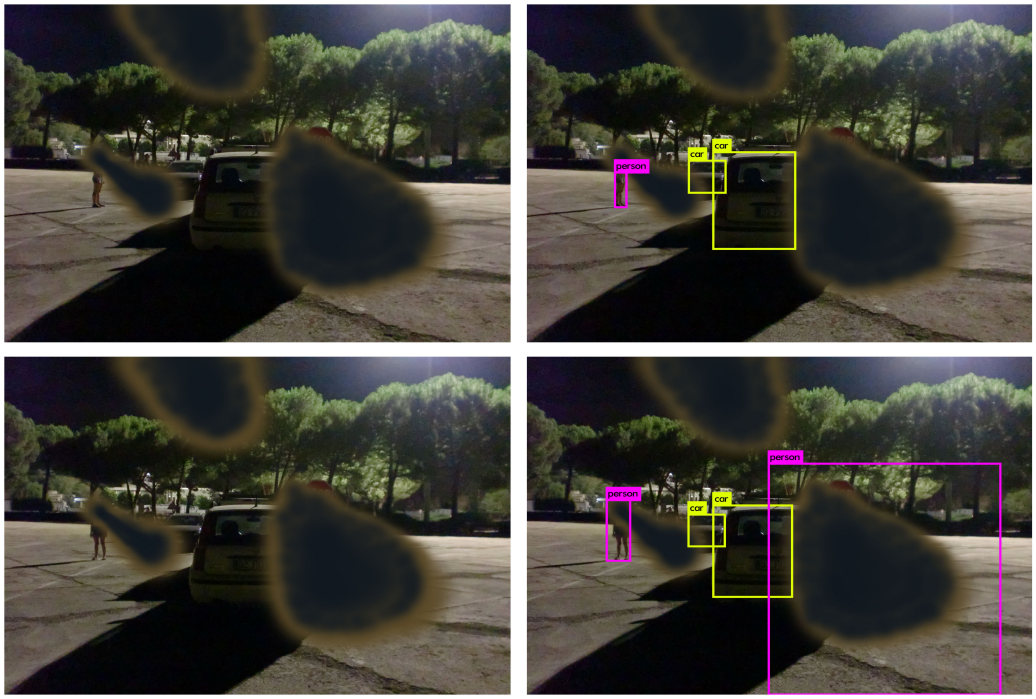

Figure 6.20. Results of the tests performed during the night with macro defects at full resolution (car lights OFF)

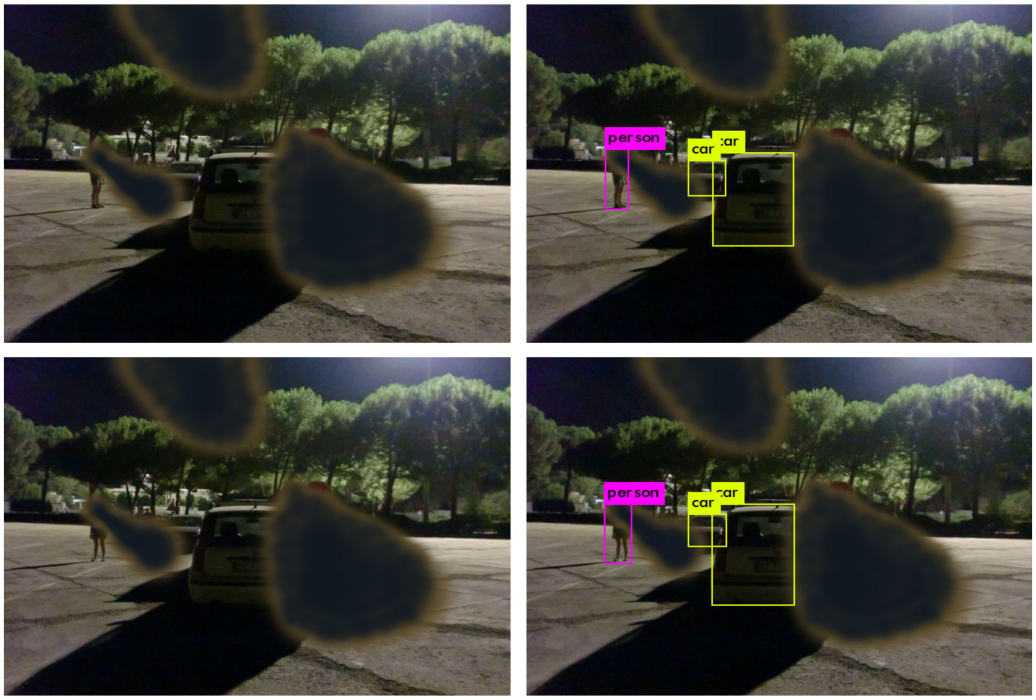

Figure 6.21. Results of the tests performed during the night with macro defects at VGA resolution (car light OFF)

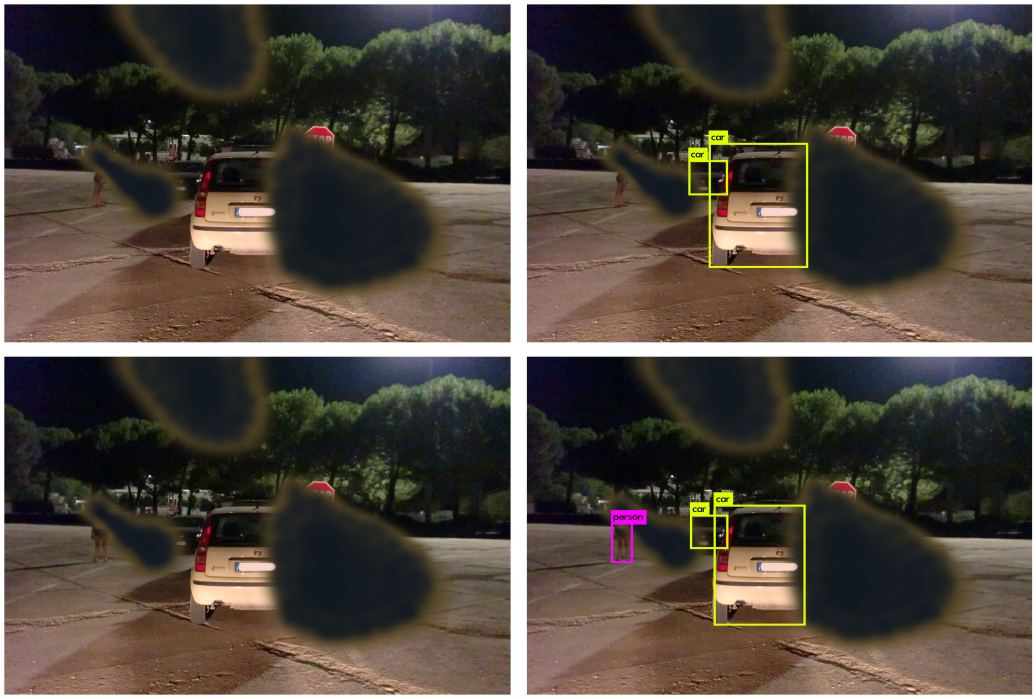

Figure 6.22. Results of the tests performed during the night with macro defects at full resolution (car lights ON)

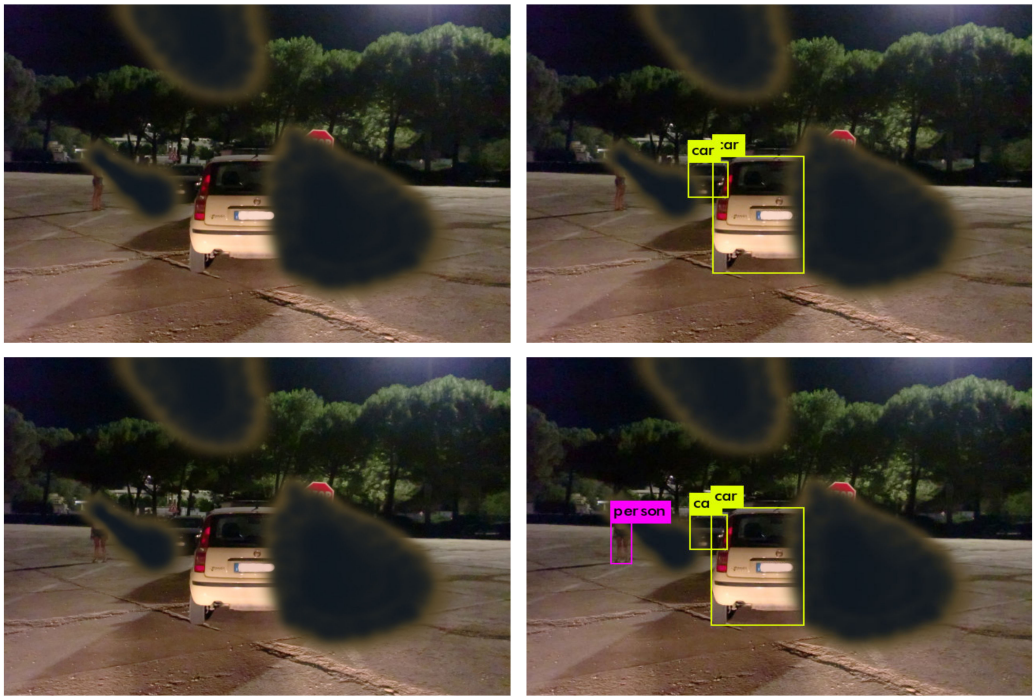

Figure 6.23. Results of the tests performed during the night with macro defects at VGA resolution (car light ON)

6.3 Weather conditions

Continuing with the validation procedure, the next step is the evaluation of different weather conditions. As done for the validation of the lane detection system, two weather conditions have been considered: heavy rain and fog. These two particular conditions are the most critical in terms of visibility and, in the worst cases, they can strongly reduce the visibility. The following analysis will evaluate how the system reacts in these cases considering each weather condition in different moments of the day (daytime and nighttime). Also in this case, for technical and safety reasons the dataset used for the tests has been obtained modifying with Photoshop® the pictures used for the lighting condition analysis.

6.3.1 Fog

The first weather condition that is going to be analyzed is the fog. For each lighting condition, two different levels of fog have been considered: light fog and thick fog. The results obtained in both cases are reported in the figures from 6.24 to 6.30, each of them shows the outcomes obtained at full resolution with the only exception of Figure 6.30, which is the only case where the results produced at full resolution (Figure 6.27) were different with respect to those at VGA resolution. From these images it is immediate to see that with light fog (figures 6.24, 6.26 and 6.28), regardless of the amount of light, the level of performance is almost unchanged. The only mistake registered in this condition is that the stop sign in Figure 6.26 (the one inside the image with the pedestrians in profile) was not detected. Regarding the thick fog, during the daytime the performances are comparable with the ones obtained in light fog conditions. Unfortunately, in low light conditions, the system is not efficient in the same way. In fact, during the nighttime with car lights turned off (Figure 6.27) the system did not detect the stop sign and the black car in any of the considered cases, the yellow car, instead, has been detected only at full resolution with the pedestrians in profile. The situation has improved with car lights turned on, in this case system did not show any particular issue. Once again, the tests shown that the car lights have a beneficial effect on the performances, allowing to achieve good results also in critical situations.

For what concerns the confidence scores, as can be seen in the tables below, the performance degradation due to the fog is very small:

Daytime with light fog)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	100%	100%	100%

person (boy)	99%	99%	98%	98%
yellow car	74%	88%	92%	93%
black car	99%	100%	99%	99%

Daytime with thick fog)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	99%	99%	100%	99%
person (girl)	99%	99%	100%	100%
person (boy)	94%	92%	91%	93%
yellow car	85%	88%	93%	91%
black car	99%	99%	98%	98%

Nighttime with light fog and car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	//	//	63%	61%
person (girl)	98%	98%	99%	99%
person (boy)	97%	97%	99%	99%
yellow car	91%	90%	89%	88%
black car	92%	92%	91%	93%

Nighttime with thick fog and car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	//	//	//	//
person (girl)	95%	95%	97%	97%
person (boy)	92%	94%	96%	96%
yellow car	61%	//	//	//
black car	//	//	//	//

Nighttime with light fog and car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	99%	98%	99%	99%
person (boy)	92%	94%	92%	95%

yellow car	93%	93%	89%	93%
black car	93%	92%	95%	95%

Nighttime with thick fog and car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	99%	99%	99%	98%
person (girl)	93%	91%	96%	95%
person (boy)	71%	77%	62%	80%
yellow car	81%	84%	76%	81%
black car	88%	86%	93%	92%

Figure 6.24. Results of the tests performed during the morning with light fog at full resolution

Figure 6.25. Results of the tests performed during the morning with thick fog at full resolution

Figure 6.26. Results of the tests performed during the night with light fog at full resolution car lights OFF)

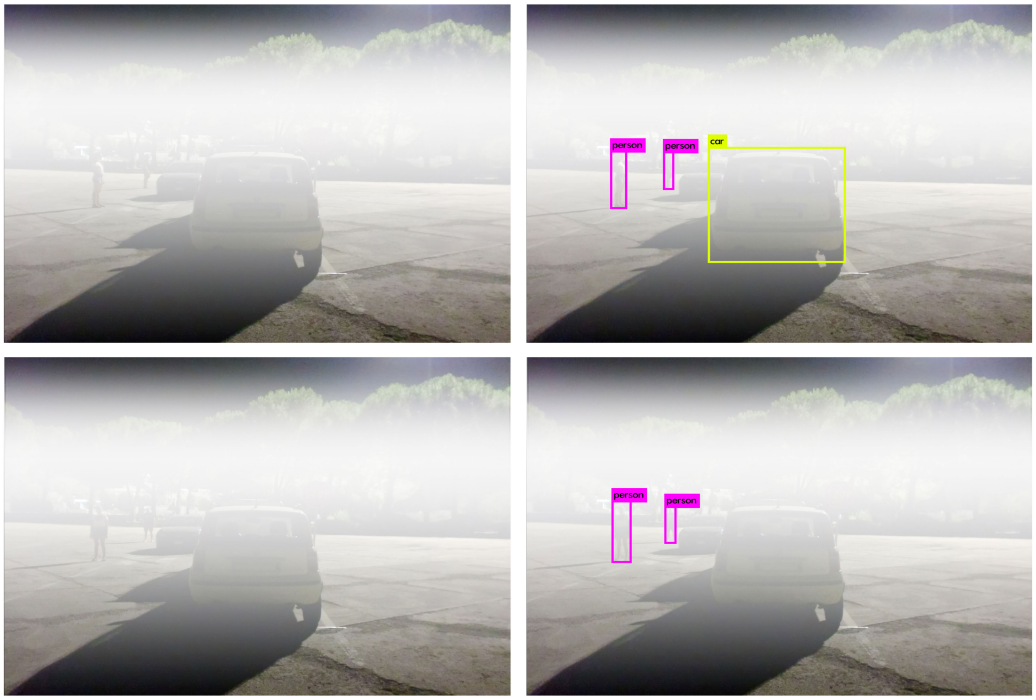

Figure 6.27. Results of the tests performed during the night with thick fog at full resolution car lights OFF)

Figure 6.28. Results of the tests performed during the night with light fog at full resolution (car lights ON)

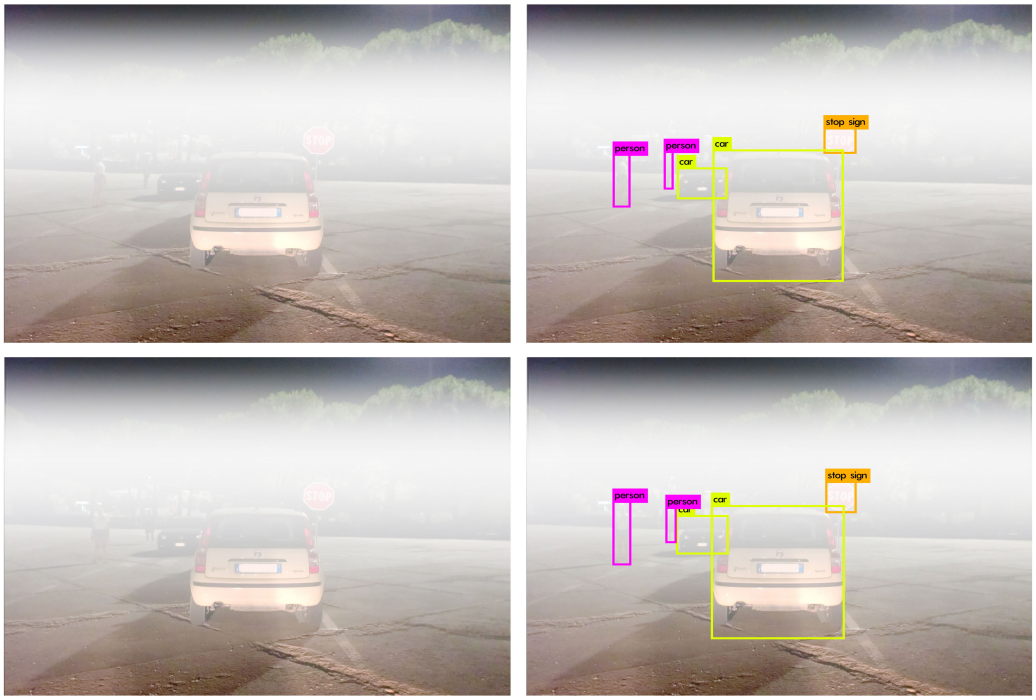

Figure 6.29. Results of the tests performed during the night with thick fog at full resolution (car light ON)

Figure 6.30. Results of the tests performed during the night with macro defects at VGA resolution (car lights OFF)

6.3.2 Heavy rain

Continuing the analysis of critical weather conditions, as anticipated, after the fog the second and last weather condition that must be evaluated is the heavy rain. As done for the fog, both the daytime and the nighttime have been considered in order to evaluate the performances with different light conditions. Moreover, to reproduce the effect of eventual water drops on the camera glass, the images previously modified to add the rain have been blurred using a specific filter, allowing to test the performances also with an extremely distorted input image.

All the results obtained during these tests are reported in the figures from 6.31 to 6.37. In particular figures from 6.31 to 6.33 show the results obtained introducing only the water drops and reducing a little bit the luminosity, while the others reports the results obtained considering also the effect of the water on the camera glass. In the first case, with the only exception of the nighttime with car lights turned off (Figure 6.32), the quality of the results is very good, with the system that detected the same objects detected in the absence of rain. In fact the rain acts as a sort of micro defect, which does not affect excessively the correct functioning of the system when there is enough light. Instead, as happened with micro defects, without a proper illumination there is a significant performance degradation, with the system that did not detect both the black car and the yellow car. Passing to the second case, where the input images are heavily distorted, the tests shown a bit more significative performance degradation also in good lighting conditions. However, considering the extent of the disturbance, the system worked surprisingly well despite the adverse conditions. For what concerns the effects of the resolution, the gap between the full resolution and the VGA resolution is absolutely negligible. The only difference that is worth reporting is the fact that, during the daytime, considering the effect of water on the camera glass, the furthest pedestrian were not detected when in profile at full resolution (Figure 6.34).

All the confidence scores obtained during the tests are summarized in the tables below:

Heavy rain during the daytime)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	100%	100%	100%	100%
person (boy)	100%	100%	100%	98%
yellow car	99%	99%	99%	99%
black car	100%	100%	100%	100%

Heavy rain during the nighttime with car lights OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	96%	94%	93%	96%
person (girl)	99%	99%	100%	100%
person (boy)	99%	99%	99%	99%
yellow car	//	//	//	//
black car	//	//	//	//

Heavy rain during the nighttime with car lights ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	99%
person (girl)	99%	99%	99%	99%
person (boy)	98%	97%	95%	93%
yellow car	80%	74%	94%	82%
black car	89%	86%	95%	93%

Heavy rain during the daytime + water distortion)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person (girl)	94%	96%	98%	99%
person (boy)	//	55%	94%	97%
yellow car	98%	99%	99%	99%
black car	99%	99%	99%	99%

Heavy rain during the nighttime with car lights OFF + water distortion)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	61%	69%	//	//
person (girl)	96%	96%	97%	99%
person (boy)	93%	95%	97%	98%
yellow car	64%	56%	//	//
black car	//	//	//	//

Heavy rain during the nighttime with car lights ON + water distortion)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	99%
person (girl)	68%	66%	60%	68%
person (boy)	//	//	60%	57%
yellow car	93%	91%	85%	82%
black car	98%	97%	98%	97%

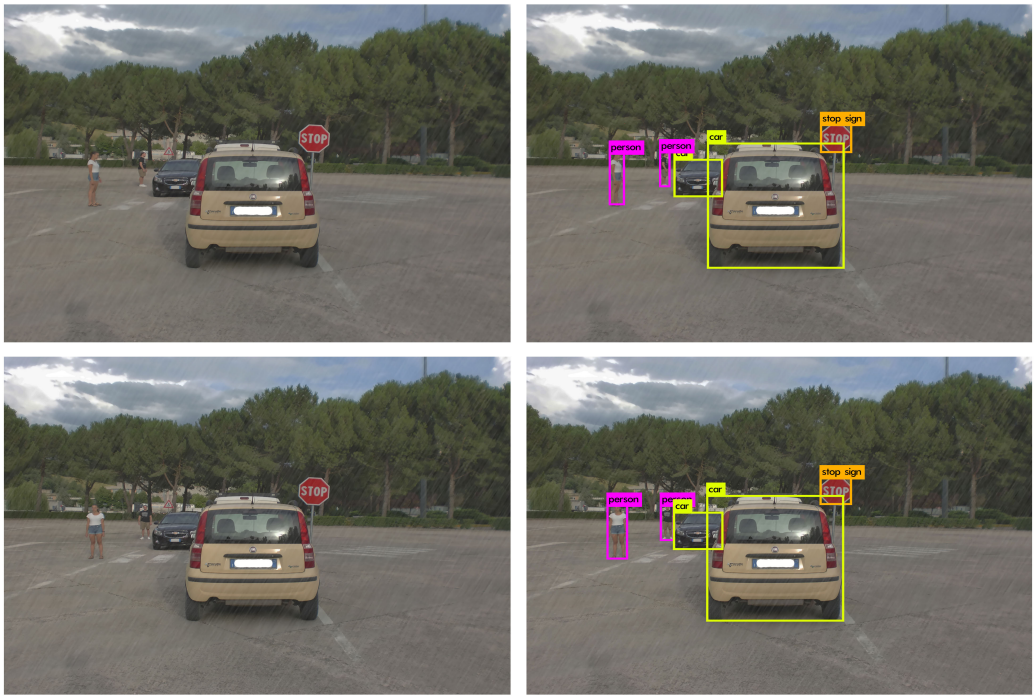

Figure 6.31. Results of the tests performed during the morning with heavy rain at full resolution

Figure 6.32. Results of the tests performed during the night with heavy rain at full resolution (car lights OFF)

Figure 6.33. Results of the tests performed during the night with heavy rain at full resolution (car lights ON)

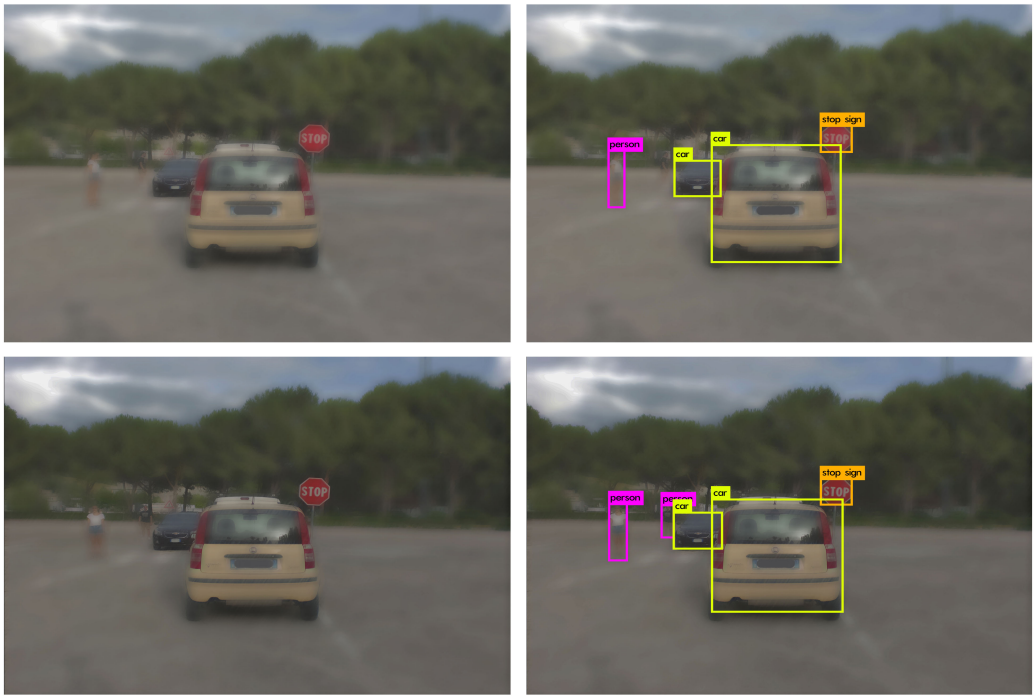

Figure 6.34. Results of the tests performed during the morning with heavy rain simulating the presence of water on the camera glass at full resolution

Figure 6.35. Results of the tests performed during the morning with heavy rain simulating the presence of water on the camera glass at VGA resolution

Figure 6.36. Results of the tests performed during the night with heavy rain simulating the presence of water on the camera glass at full resolution (car lights OFF)

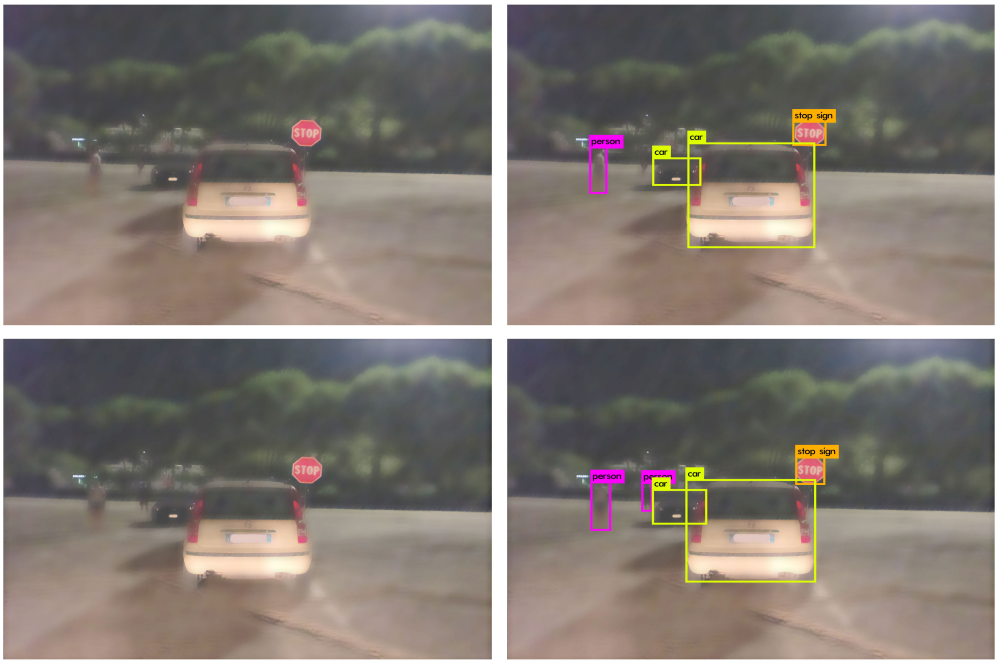

Figure 6.37. Results of the tests performed during the night with heavy rain simulating the presence of water on the camera glass at full resolution (car lights ON)

6.4 Corner cases

The last part of this validation procedure will be focused on some corner cases, which means that some critical scenarios will be evaluated in order to see how the system acts in the worst possible conditions. In fact, when one deals with safety, every possibility, even if it is very infrequent, must be considered, because no errors are admitted. During this analysis the following cases will be evaluated:

- Very low light conditions;
- Color overlapping on a white street;
- Low resolutions.

6.4.1 Very low light conditions

In the night tests previously performed, the presence of artificial lights produced a minimal illumination that helped the system to improve the quality of the results. However, especially during the extra-urban driving, often one drives on unlit roads, so, with the aim of evaluating also these conditions, some tests have been performed on a country road without artificial lights. Two scenarios were considered: crossroad with a pedestrian dressed with dark clothes and a car in front of the driven vehicle, crossroad without any other vehicle considering only the pedestrian. Also in this case the car lights have been considered both turned on and off.

The results of these tests are shown in the figures from 6.38 to 6.45, where it is possible to see that, even if the lighting was very poor, YOLO provided decent results detecting most of the objects in all the different conditions. The most relevant mistake is that in some cases the pedestrian has not been detected when in profile (figures 6.39, 6.42 and 6.43). In general, the tests shown that the system achieved lower confidence scores for the pedestrian every time that it was in profile. This is not a good thing if we think that, usually, the pedestrians are in profile with respect to the vehicle when they cross the road, so, an eventual mistake of this kind can lead to serious accidents. Other mistakes have been registered in the case without the second car, where YOLO produced some spurious detections, confusing the road signs with other objects (i.e. the sports ball in Figure 6.42 and the person in Figure 6.44).

All the confidence scores registered during the tests are reported in the following tables (also in this case the terms "profile" and "front" are referred to the orientation of the pedestrian):

Low light with another car in front of the vehicle and car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	99%	96%	99%
person	83%	//	97%	96%
car	100%	100%	100%	100%

Low light with another car in front of the vehicle and car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	95%	100%	99%
person	51%	60%	99%	96%
car	99%	99%	100%	100%

Low light with car lights turned OFF)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person	//	//	94%	99%
sports ball	//	//	58%	//

Low light with car lights turned ON)

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
stop sign	100%	100%	100%	100%
person	91%	97%	99%	100%
person (error)	57%	//	//	//

Figure 6.38. Results of the tests performed during the night with a car in front of the vehicle in low light conditions at full resolution (car lights OFF)

Figure 6.39. Results of the tests performed during the night with a car in front of the vehicle in low light conditions at VGA resolution (car lights OFF)

Figure 6.40. Results of the tests performed during the night with a car in front of the vehicle in low light conditions at full resolution (car lights ON)

Figure 6.41. Results of the tests performed during the night with a car in front of the vehicle in low light conditions at VGA resolution (car lights ON)

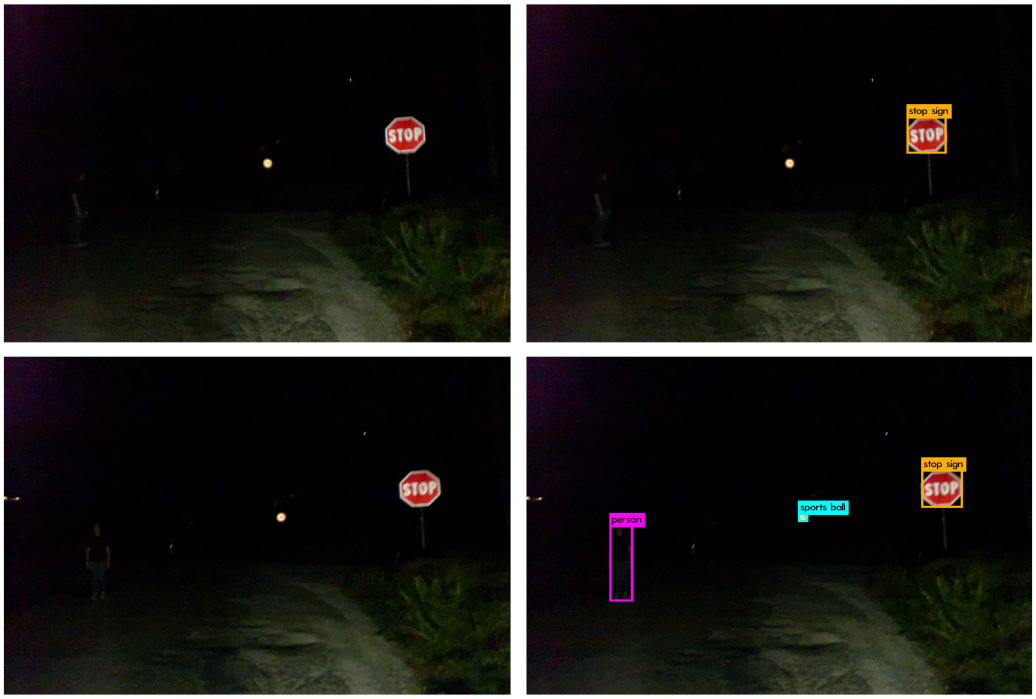

Figure 6.42. Results of the tests performed during the night in low light conditions at full resolution (car lights OFF)

Figure 6.43. Results of the tests performed during the night in low light conditions at VGA resolution (car lights OFF)

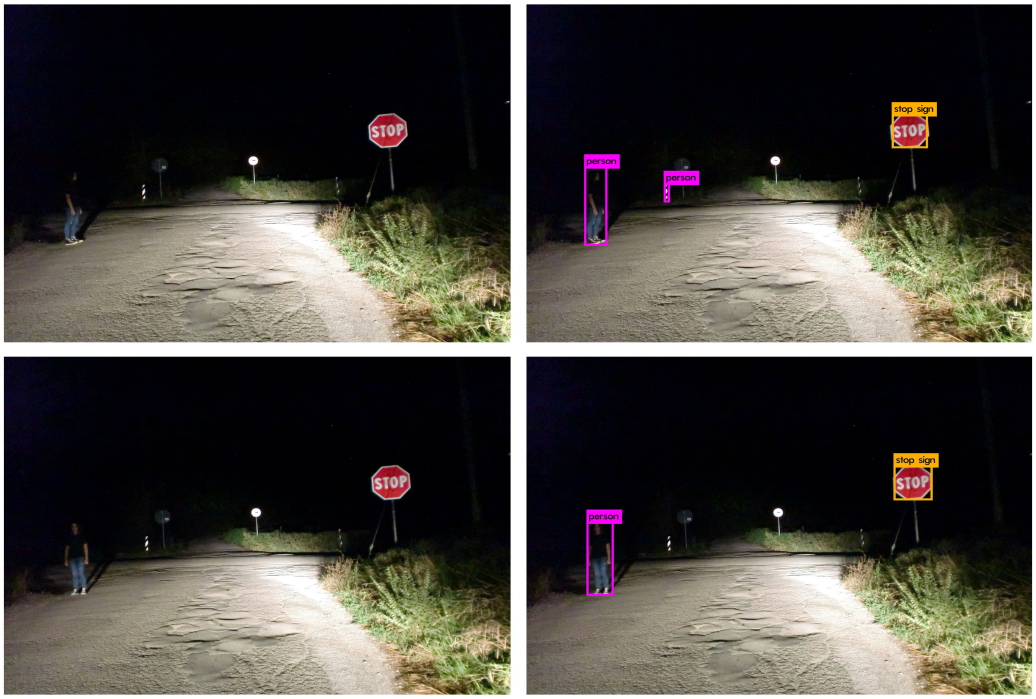

Figure 6.44. Results of the tests performed during the night in low light conditions at full resolution (car lights ON)

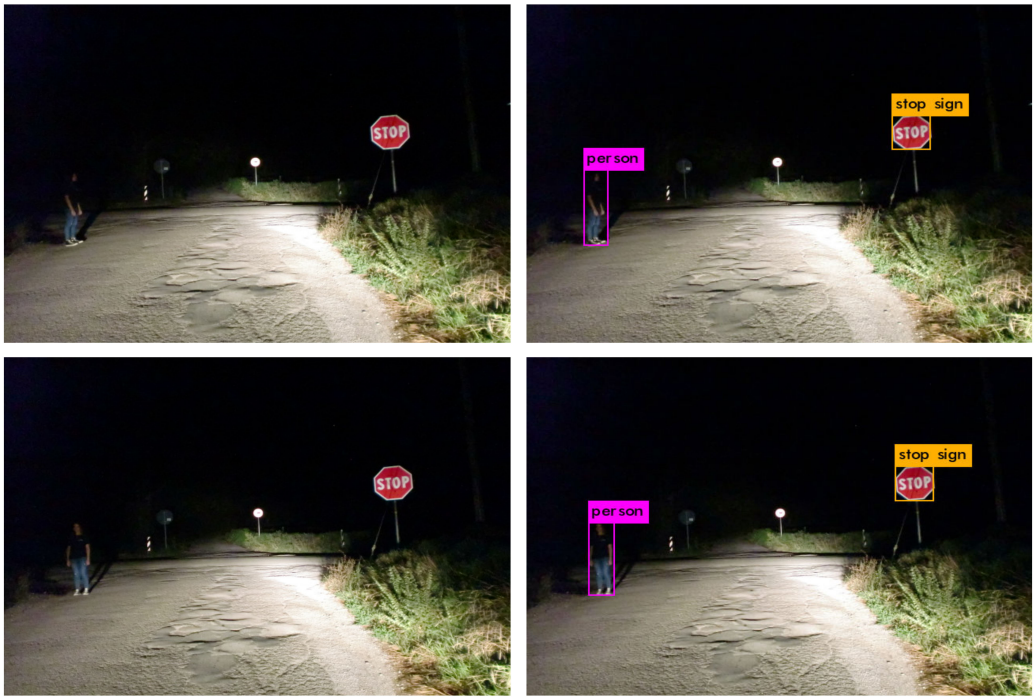

Figure 6.45. Results of the tests performed during the night in low light conditions at VGA resolution (car lights ON)

6.4.2 Color overlapping on a white street

Sometimes the objects inside a picture can have a colour similar to the background or can be aligned with something that matches its colors, making it difficult to be found. Even if this case has been partially covered in the previous subsection, with the pedestrian dressed with dark clothes in low light conditions, it can be interesting to perform some tests to see how YOLO reacts in this situation. For this reason a similar test has been repeated during the daytime, with a pedestrian dressed in white on a white street. The pedestrian has been placed at a distance of about 30 meters, which is a reasonable braking distance, also in this case the tests have been performed with two levels of resolution, i.e. full resolution and VGA.

The results achieved are shown by the figures 6.46 (full resolution) and 6.47 (VGA resolution). As it is possible to see, the system did not detect the pedestrian only when in profile at VGA resolution. This can be considered in any case a success considering the distance of the pedestrian and the fact that, in these conditions, it is difficult to individuate it also for a human eye. The confidence scores registered during the tests are reported below:

Object	Profile (full res)	Profile (VGA)	Front (full res)	Front (VGA)
person	98%	//	99%	81%

Figure 6.46. Results of the tests performed during the daytime on a white street with a pedestrian dressed in white at full resolution

Figure 6.47. Results of the tests performed during the daytime on a white street with a pedestrian dressed in white at VGA resolution

6.4.3 Low resolutions

During all the tests performed previously, two levels of resolution have been considered (i.e. 4000×3000 and 640×480) to understand if a very high resolution produced significantly better results with respect to a lower one. The tests performed until now shown that the gap between the two resolutions is negligible in most of the cases, with the lower resolution that sometimes provided even better results that the higher one. So, in order to understand the limits of this object detection system, some other tests will be performed, decreasing the resolution until a tangible performance degradation is not registered.

Three resolutions have been considered: 480×360, 320×240 and 144×108. In order to evaluate only the effects of the resolution, the best possible condition in terms of lighting has been chosen, i.e. the morning. The results are shown by the figures 6.48 (with the pedestrians in profile) and 6.49 (with the pedestrians in front), the images inside each figure are ordered with a descending resolution from the left to the right, each input figure has below it the corresponding output. Surprisingly, even at the lowest possible resolution (144×108), which is so low that YOLO does not have enough pixels to place the entire bounding boxes but only the labels, the tests provided positive results. Nevertheless, even if the results are good, at the lowest resolution the system did not detect the farther pedestrian when in front and confused the yellow car with a truck. This means that, if in the best possible conditions the system did not work as expected, this level of resolution is not suitable for a real application. Anyway, in the light of the results obtained during the previous tests, the 640×480 VGA resolution can be considered a good tradeoff between computational speed and quality of the detections. At full resolution instead, even if sometimes the system achieved better results, the time required to process a single frame is too high to use YOLO in real time, even if it runs on a GPU. Probably a good level of performance can be obtained also using a 480×360 resolution, which allows to decrease the weight of the single frames and so to increase the overall speed of the system, without sacrificing too much the level of detail and, hopefully, the overall performance. In any case, this is just a supposition that must be confirmed through further studies.

All the confidence scores registered for each resolution are reported in the tables below:

Pedestrians in profile)

Object	480x360	320x240	144x108
stop sign	100%	100%	99%
person (girl)	100%	100%	99%

person (boy)	100%	99%	75%
yellow car	88%	93%	53%
black car	100%	99%	97%
truck	//	//	91%

Pedestrians in front)

Object	480x360	320x240	144x108
stop sign	100%	100%	99%
person (girl)	100%	100%	100%
person (boy)	98%	98%	//
yellow car	91%	95%	//
black car	100%	100%	99%
truck	//	//	92%

Figure 6.48. Results of the tests performed with different resolutions (pedestrians in profile)

Figure 6.49. Results of the tests performed with different resolutions (pedestrians in front)

6.5 Conclusion

When one deals with object detection, the number of variables is too high to evaluate accurately every single possibility. During this validation procedure, all the tests have been performed trying to recreate some particularly significative scenarios in the most realistic manner as possible, with the main goal of evaluating the effects of the environmental conditions on the overall performances. The results reported during the tests shown that YOLO is a very robust object detection system, capable of providing surprisingly good results even in the most adverse conditions. However, the system shown its limits and some issues came out, like, for example, the performance degradation in case of very low light conditions and in case of macro defects.

In order to summarize what emerged from the tests, some graphs have been reported in figure 6.50. All of them are referred to the results obtained using the VGA resolution, this choice is mainly due to the fact that it is the most suitable resolution for real time uses, and so, for an actual application. These graphs reports the confidence scores achieved for each object class in all the different conditions evaluated during the validation procedure, with the only exception of the macro defects, which gave unrelated results in the different moments of the day because of the little shifts of the spots and that, for this reason, must be evaluated looking directly to the input and output images. All the reported scores have been obtained computing the average between the values obtained using the images with the pedestrians in profile and the ones obtained using the images with the pedestrians in front.

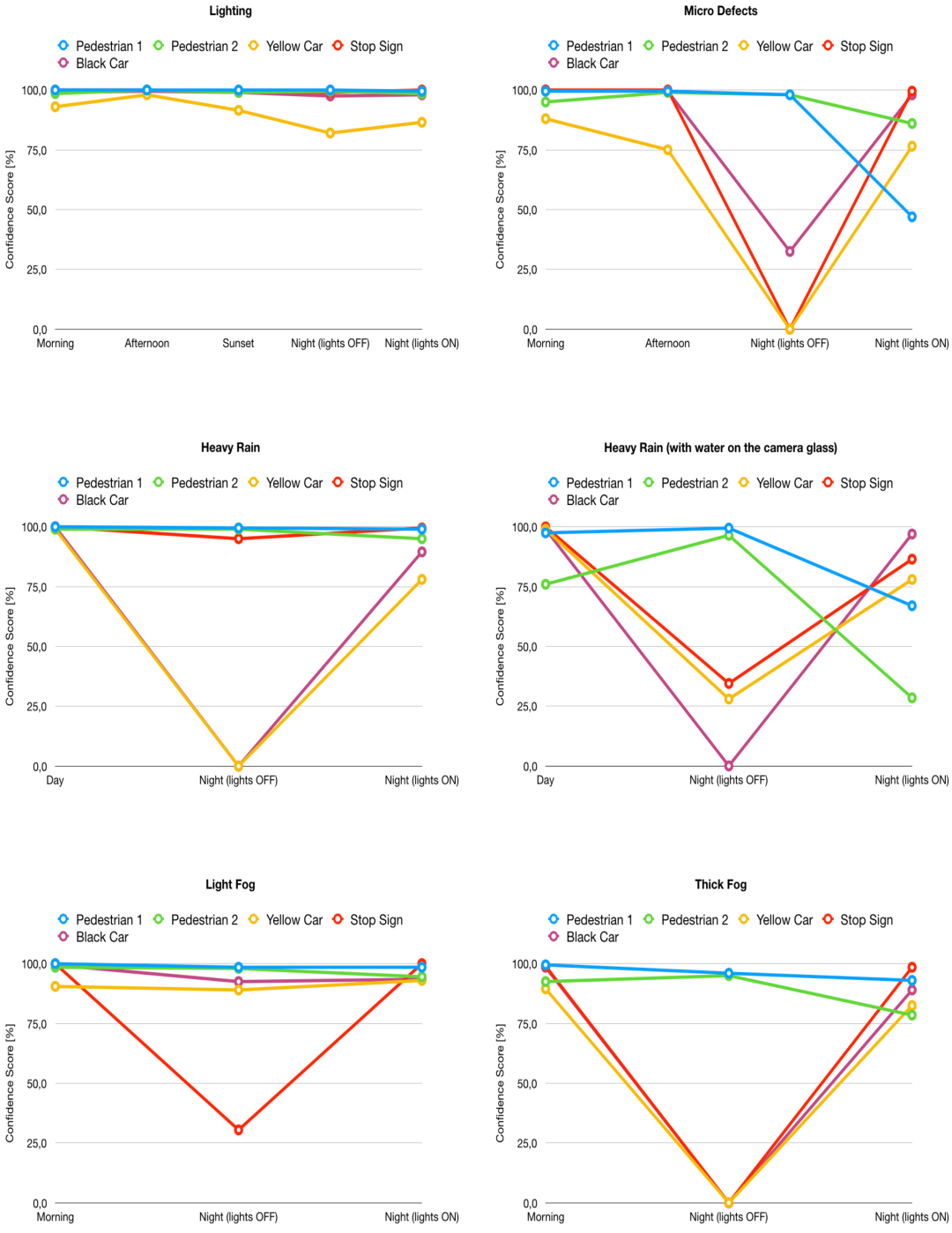

Figure 6.50. Graphs that summarize some of the results obtained during the tests

Part V

Conclusion

In the light of the results obtained during this work, there are many conclusions that can be made. The first one is that both the analyzed systems cannot accomplish by themselves to the functions needed to implement the related ADAS. In fact, every time that something disturbed the input provided by the camera, e.g. defects or critical weather conditions, the loss of performance has been too high to comply the standards needed for safety purposes. So, during the design phase, there will be the need to find something that compensates the shortages of all the systems based only on vision, like, for example, an additional system which uses radars and other alternative sensors. Considering the single systems, instead, for what concerns the lane detection one, it shown all its weaknesses when dealing with bad asphalt conditions and unclear road markings, while achieved a reasonable level of performance in non-critical conditions. In any case, the one analyzed in this work, is a very basic implementation of a system based on a simple working principle, so it is clear that, using something more sophisticated, better results could be achieved. Nevertheless it is important to underline that the main goal of this thesis is to evaluate the impact of the environmental conditions on vision systems for autonomous driving, regardless of their complexity. On the other hand YOLO shown to be a much more robust and mature technology, especially if one thinks about the fact that the tests have been performed with a pre-trained network, which means that no specific training was performed. So it is reasonable to think that, with a targeted training, a significative improvement in the results could be achieved. Naturally, even if the overall performances can be considered satisfactory, also YOLO had some issues in critical conditions, especially with low light, where the system did some mistakes like spurious or missing detections.

All the problems and the malfunctions that came out from this analysis represent a starting point to understand where there is the need to invest in terms of research in order to reach a higher level of efficiency. By carrying on this process of constant improvement, one day, will be possible to reach the safety standards needed for autonomous driving. The hope is that the conclusions and the results obtained thanks to this work will represent a help inside this process of development for all the coming researchers.

Bibliography

[1] Massimo Bertozzi, Alberto Broggi. *GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection*. IEEE, 1998.

[2] Chris Kreucher, Sridhar Lakshmanan. *LANA: A lane extraction algorithm that uses frequency domain features*. IEEE, 1999.

[3] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester and Deva Ramanan. *Object Detection with Discriminatively Trained Part-Based Models*. TPAMI, 2010.

[4] J. Uijlings, K. van de Sande, T. Gevers, A. Smeulders. *Selective Search for Object Recognition*. IJCV, 2013.

[5] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik. *Rich feature hierarchies for accurate object detection and semantic segmentation*. CVPR, 2014.

[6] Ross Girshick, Microsoft Research. *Fast R-CNN*. ICCV, 2015.

[7] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. arXiv preprint arXiv:1506.01497, 2015.

[8] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun. *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks*. ICLR, 2014.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. *You Only Look Once: Unified, Real-Time Object Detection*. CVPR, 2016.

[10] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. *SSD: Single Shot MultiBox Detector*. ECCV, 2016.

[11] Joseph Redmon, Ali Farhadi. *YOLOv3: An Incremental Improvement*. arXiv preprint arXiv:1804.02767, 2018.

[12] Wikipedia: Hough transform.
`"https://en.wikipedia.org/wiki/Hough_transform"`

[13] Matlab: Bird's eye view.
`https://it.mathworks.com/help/driving/ref/birdseyeview.html`

[14] Matlab: Camera parameters.
`https://it.mathworks.com/help/vision/ref/cameraparameters.html`

[15] Matlab: Camera calibration.

https://it.mathworks.com/help/vision/ug/single-camera-calibrator-app.html

[16] Wikipedia: Support Vector Machine.
"https://en.wikipedia.org/wiki/Support-vector_machine"