

Review

Detecting Wear and Tear in Pedestrian Crossings Using Computer Vision Techniques: Approaches, Challenges, and Opportunities

Gonçalo J. M. Rosa ¹, João M. S. Afonso ¹, Pedro D. Gaspar ^{2,3}, Vasco N. G. J. Soares ^{1,4} and João M. L. P. Caldeira ^{1,4,*}

¹ Polytechnic Institute of Castelo Branco, Av. Pedro Álvares Cabral, n° 12, 6000-084 Castelo Branco, Portugal; goncalo.rosa@ipcbcampus.pt (G.J.M.R.); joao.afonso2@ipcbcampus.pt (J.M.S.A.); vasco.g.soares@ipcb.pt (V.N.G.J.S.)

² Department of Electromechanical Engineering, University of Beira Interior, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal; dinis@ubi.pt

³ Centre for Mechanical and Aerospace Science and Technologies (C-MAST), Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal

⁴ Instituto de Telecomunicações, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal

* Correspondence: jcaldeira@ipcb.pt

Abstract: Pedestrian crossings are an essential part of the urban landscape, providing safe passage for pedestrians to cross busy streets. While some are regulated by timed signals and are marked with signs and lights, others are simply marked on the road and do not have additional infrastructure. Nevertheless, the markings undergo wear and tear due to traffic, weather, and road maintenance activities. If pedestrian crossing markings are excessively worn, drivers may not be able to see them, which creates road safety issues. This paper presents a study of computer vision techniques that can be used to identify and classify pedestrian crossings. It first introduces the related concepts. Then, it surveys related work and categorizes existing solutions, highlighting their key features, strengths, and limitations. The most promising techniques are identified and described: Convolutional Neural Networks, Histogram of Oriented Gradients, Maximally Stable Extremal Regions, Canny Edge, and thresholding methods. Their performance is evaluated and compared on a custom dataset developed for this work. Insights on open issues and research opportunities in the field are also provided. It is shown that managers responsible for road safety, in the context of a smart city, can benefit from computer vision approaches to automate the process of determining the wear and tear of pedestrian crossings.

Keywords: pedestrian crossings; smart cities; computer vision; state-of-the-art; performance evaluation



Citation: Rosa, G.J.M.; Afonso, J.M.S.; Gaspar, P.D.; Soares, V.N.G.J.; Caldeira, J.M.L.P. Detecting Wear and Tear in Pedestrian Crossings Using Computer Vision Techniques: Approaches, Challenges, and Opportunities. *Information* **2024**, *15*, 169. <https://doi.org/10.3390/info15030169>

Academic Editors: Francesco Beritelli and Marco Leo

Received: 23 January 2024

Revised: 5 March 2024

Accepted: 15 March 2024

Published: 20 March 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

It is important to acknowledge that pedestrians are more vulnerable than other users of public roads due to their lack of physical protection. To ensure their safety, it is necessary to create designated areas for them. One such solution is the implementation of pedestrian crossings (i.e., zebra crossings or crosswalks), which are marked points on the road that allow pedestrians to cross the carriageway safely. Pedestrian crossings are typically marked on the pavement with parallel white stripes perpendicular to the road's direction. These stripes are usually around 50 cm wide and can vary in length between 250, 300, and 400 cm, depending on their location. The distance between the stripes is approximately 50 cm [1].

According to the annual report by the Portuguese National Road Safety Authority, Portugal experienced 27,725 road accidents in 2020. It is important to note that 13.3% of these accidents involved victims being hit by a car, and unfortunately, 17.7% of the road accident victims were fatal [2]. In this context, safety measures are necessary to address the alarming statistics. In Portugal, local authorities have implemented measures to enhance visibility in pedestrian crossing areas, such as replacing static vertical signs with dynamic

light-emitting diode (LED) signs or using flashing LEDs near pedestrian crossings. It is worth noting that while these measures are a significant contribution, their effectiveness is most noticeable during the night [3].

Although some pedestrian crossings are regulated by timed signals and are marked with signs and lights, others are only simply marked on the road. Over time, the markings may experience wear and tear due to weather conditions, heavy traffic, and other factors, which can make them difficult to be viewed by drivers. Hence, it may be necessary to consider interventions to address this issue. Therefore, it is important to develop solutions that can effectively identify and classify the wear and tear of pedestrian crossings.

The work presented in this paper represents a first step in a project aimed at developing a system that uses computer vision techniques and solutions to address the challenge of timely detection of wear and tear on pedestrian crossings. Such a system could identify pedestrian crossings that may require repair and notify the responsible entities promptly. This solution could be utilized in local authority vehicles traveling on city roads, as part of smart cities' initiatives that aim to explore new strategies for optimizing available resources towards a more sustainable future [4]. It is of utmost importance that municipal authorities effectively integrate and implement these technologies to ensure efficient urban management and the development of resilient urban communities that can meet the increasing demands of the future.

In Portugal, the responsibility of monitoring and maintaining road markings lies with Infraestruturas de Portugal (IP) and the municipal authorities. However, it is worth noting that technology can now assist these organizations in identifying and diagnosing the state of road markings, which can complement citizen reports. This approach has the potential to improve the efficiency and accuracy of road maintenance. According to information provided by IP [5], enforcement is more prevalent in areas with high traffic volume and commercial activity, among other criteria, which can exclude many urban areas. It is important to foster cooperation between local authorities and IP to address these issues.

The integration of computer vision techniques into the mobile inspection and support units from IP or local authority vehicles can facilitate the collection of information on road conditions, such as the location and the wear and tear of pedestrian crossings. This information may be useful to competent authorities in the maintenance and restoration of pedestrian crossings if deemed necessary. This paper surveys commonly used computer vision techniques that have been used to address these problems and related topics. It discusses their strengths and limitations. Then, it identifies the most promising techniques and evaluates their performance using a new dataset that has been created in the context of this work.

The rest of the paper is organized as follows. Section 2 introduces the computer vision techniques that form the basis of this work. Section 3 presents related work. Section 4 presents a performance evaluation study of the most promising techniques. Section 5 outlines the challenges and opportunities. Finally, Section 6 presents conclusions and identifies potential areas for future research.

2. Computer Vision Techniques

Computer vision is a field of artificial intelligence that focuses on extracting visual data, such as images and videos, which can be used to perform various tasks, including object detection and classification. Cameras are often the primary source of this data, and a significant amount of information is required to achieve accurate recognition. Moreover, algorithms play a crucial role in facilitating autonomous learning of the model without the need for human intervention. This technology has a wide range of applications, including machine learning [6].

This section is divided into three subsections. Firstly, it will address the area of convolutional neural networks (CNN), covering their respective types and architectures. Subsequently, it will discuss image processing techniques, detailing how they work and

demonstrating their use in the context of pedestrian crossings. Finally, a section on classification architectures will be presented.

2.1. Convolutional Neural Network Architectures

In recent years, computer vision has made significant progress, largely due to the development of new deep learning techniques. This field aims to enable computers to recognize complex patterns in images, texts, sounds, and other objects, like the human brain [7]. Simultaneously, there has been an enhancement in the processing capacity of Graphics Processing Units (GPUs), which has enabled researchers to train deeper networks. Regarding the issue of object detection, CNNs have demonstrated their effectiveness and can be classified into two categories: One-Stage Detectors and Two-Stage Detectors.

CNNs are widely recognized for their crucial role in computer vision, as they are optimized to process visual data and recognize complex features. Due to their ability to automatically learn data characteristics, they have become powerful tools for tasks such as object recognition, image segmentation, and visual classification. CNNs have found various applications in fields such as automobile automation, healthcare, and document analysis [8].

CNNs are composed of three types of layers, as illustrated in Figure 1, and can be replicated. Each layer has a crucial role in both image reduction and processing. The first layer, known as the input layer, receives the image to be processed. Its function is to provide image input to the network, and it also has channels for the different red, green, and blue (RGB) colors. However, it is not considered a part of the CNN structure. The convolutional layer is a crucial element of a CNN. It applies the weights of the previous layers to extract image details through a convolutional mathematical operation [9]. The layer uses a matrix, known as a kernel, which contains learned data and where the learning process occurs. The output is then transformed into a feature map and forwarded to the next layer, where new characteristics such as textures and colors are learned. The pooling layer is designed to simplify the subsequent layers, much like reducing the resolution of an image. Its primary objective is to reduce computing time [10]. The process of image classification takes place in the fully connected (FC) layer, which makes use of information from the previous layers and applies either a sigmoid or softmax activation function [11].

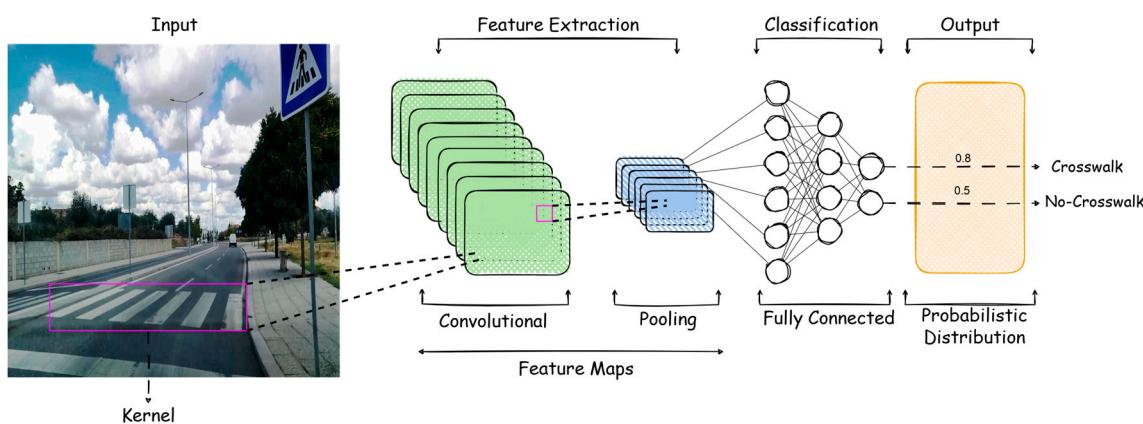


Figure 1. Neural network architecture with convolutional layers.

The following sections will discuss two types of CNNs: One-Stage and Two-Stage. Furthermore, examples of CNNs and their architectures will be provided.

2.1.1. One-Stage Detection

The process of One-Stage Detection models involves the direct processing of the image upon receipt by the network. The network generates anchor boxes, which are areas identified as probable locations of objects to be detected. These anchor boxes are later converted into bounding boxes, representing the areas that the model identifies as

the objects present. While it is true that One-Stage models may offer faster processing times than Two-Stage models, it is also important to note that they tend to have lower accuracy [12]. Therefore, it is crucial to consider both speed and accuracy when selecting a model. In the context of this paper, the models that adopt the One-Stage Detection approach are You Only Look Once (YOLO), Single Shot Detector (SSD), and EfficientDet. Although these use similar processing methods, they have specific variations.

The YOLO model [13], also known as ‘You Only Look Once’, is a highly efficient method for object detection that utilizes a single pass through the neural network. It can identify objects in images and videos with great accuracy thanks to various optimization strategies, such as dividing the image into a grid of cells. This approach allows for precise identification even when objects overlap. Furthermore, during network training, techniques such as L2 regularization [14] can be employed to prevent the weights from assuming excessively high values. This aids in preventing overfitting [14], which will be discussed in later sections, and enhances the model’s generalizability. The detection process of the YOLO model is depicted in Figure 2.

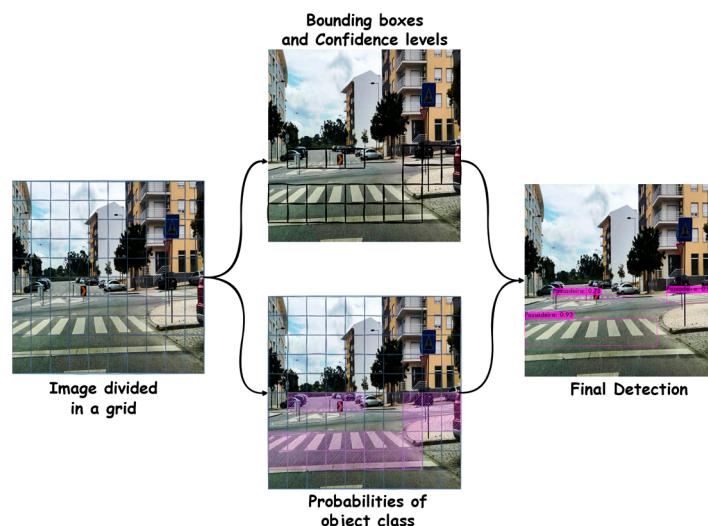


Figure 2. YOLO model detection process.

The Single Shot Detector (SSD) model [15] is based on feed-forward neural networks, where the information follows one direction, unlike other networks where the nodes of the network form cycles. These networks produce a collection of bounding boxes and a score, usually in percentage format, which indicates the probability that the object present in these bounding boxes is the object it was trained to identify. According to [16], this model has better accuracy and inference time than YOLO [13] and Fast R-CNN [16].

The SSD model [15] is characterized by a backbone where a CNN architecture is used to extract important features from the image. This phase is essential for capturing information at different levels of detail, and models such as VGG-16 [17], ResNet [18], or MobileNet [19] are often adopted as backbones, adapted according to the specific needs of precision and computational efficiency. The SSD contains several detection layers operating at different scales. Each layer is responsible for detecting objects of different sizes, applying convolutional filters adapted to each scale. The detection layers produce feature maps that provide information about the presence of objects in different regions and scales of the image. These maps are then used to make predictions about bounding boxes and associated classes. Each point on the feature map makes predictions for different sizes of bounding boxes and classes, allowing the model to deal efficiently with objects of different scales and shapes. The SSD also incorporates anchor boxes to facilitate multi-scale detection. Predictions include information on the location of the bounding boxes and the confidence scores associated with each class. The use of activation functions, such as softmax, is common to obtain normalized probabilities. Finally, non-maximum

suppression is applied to eliminate duplicate detections and select the most relevant bounding boxes corresponding to the detected objects. Figure 3 illustrates the architecture of the SSD model.

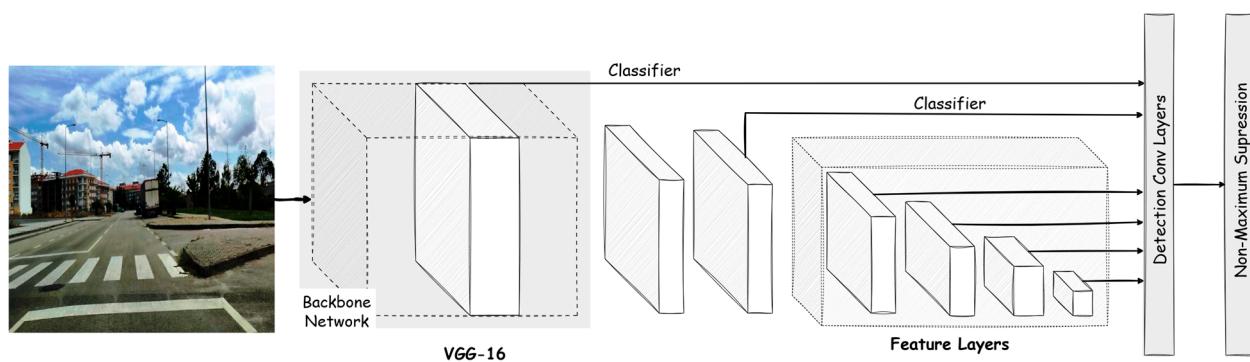


Figure 3. Illustration of the architecture of the SSD model. Adapted from [15].

The EfficientNet model [20] is a one-stage convolutional neural network that utilizes a systematic scaling approach. This approach carefully adjusts the depth, width, and resolution dimensions through a composite coefficient. Unlike conventional methods found in most CNNs, which perform arbitrary scaling of these factors, the EfficientNet approach uniformly modifies the width, depth, and resolution of the network. This is achieved using predetermined coefficients derived from a small grid search in the original model, which is smaller in size. For example, if an increase of 2^N computing resources is needed, the grid's depth can be increased by α^N , its width by β^N , and the image size by γ^N . These coefficients are constant and are determined by a grid search on the original model. EfficientNet uses a compound coefficient to uniformly scale the width, depth, and resolution of the grid. One possible justification for using the compound scaling method could be that if the input image is larger, the network may require more layers to increase the receptive field and more channels to capture finer patterns in the larger image. The EfficientNet-B0 network is built on the inverted bottleneck residual blocks of MobileNetV2, along with compression and excitation blocks. Figure 4 provides a comparison of different types of CNN architectures. The standard network architecture is shown in Figure 4a. Figure 4b–d illustrate conventional scaling methods that increase only one dimension, namely width, depth, or resolution. On the other hand, the EfficientNet network employs a compound scaling method that increases all dimensions, as depicted in Figure 4e.

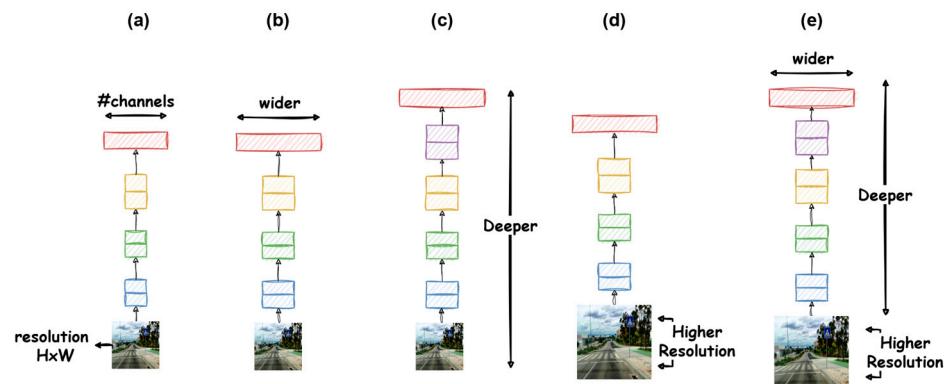


Figure 4. Comparison of CNN architectures. Adapted from [21].

2.1.2. Two-Stage Detection

Models based on the Two-Stage Detection paradigm are highly accurate at detecting and localizing objects. However, it is worth noting that these methods may have a high computational cost, which may make them unsuitable for real-time processing systems [21]. On the other hand, One-Stage Detection methods have lower computational cost and

are often considered real-time detectors. However, it is worth noting that their level of detection accuracy may be comparatively lower [22,23].

The Mask R-CNN model [24] is an example of Two-Stage Detection. It is a development of Faster R-CNN, using the same architecture but with an extra step. Faster R-CNN works in two stages. In the first stage, the network receives the image and identifies proposed regions [25] that may contain the objects to be detected, determined using Region Proposal Networks (RPNs). The proposed regions are sent to the ROIAlign layer, which aligns the features of a region of interest (ROI) with the spatial grid of the output feature map. In the subsequent stage, the bounding boxes are classified and refined to enhance detection accuracy. Additionally, Mask R-CNN introduces a third step known as the ‘Mask Head’. The system aims to perform image segmentation, allowing for the identification of objects that may be partially or completely obscured by other elements or by the proximity of multiple objects of the same class. To achieve this, the Mask Head employs the ROIAlign technique to predict individual binary masks for each object, creating distinct areas for each pixel. For further clarification on the architecture of the Mask R-CNN model, please refer to Figure 5.

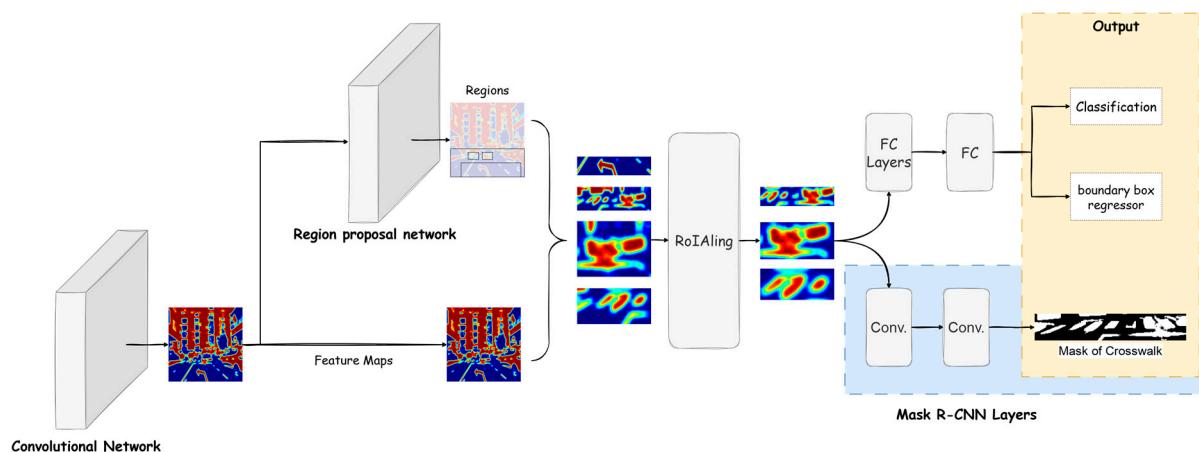


Figure 5. Illustration of the architecture of the Mask R-CNN model. Adapted from [26].

2.2. Image Processing

The Histogram of Oriented Gradients (HOG) [27,28] is a feature descriptor that extracts useful information from images. It allows for the selection of areas of interest and the elimination of unnecessary information, which facilitates the detection of object edges present in these images. This includes Canny Edge and Scale-Invariant Feature Transform, which are discussed in this section. This technique can be used to detect objects in images or process them for later use in CNN models.

The concept behind this technique is to describe the appearance and shape of objects in an image through the distribution of intensity gradients or edge directions. The image is divided into cells, and for each cell, a histogram of gradient directions is generated. The feature descriptor is obtained by concatenating these histograms. To further improve the accuracy of the histograms, it may be beneficial to consider normalizing them based on contrast. This can be achieved by using intensity measurements on larger image blocks, which can help to mitigate the effects of variations in lighting and shadows. This can be accomplished by using intensity measurements on larger image blocks, which can help to mitigate the effects of variations in lighting and shadows.

To apply the HOG, the original image in Figure 6a was first converted to grayscale. Then, a cell containing the edge of a pedestrian crossing was selected to distinguish between the area representing the pedestrian crossing and the area representing the road surface. The directions in which these variations are most marked are shown in Figure 6b, while Figure 6c shows the selected cell. Figure 6d displays several angles where there are

noticeable changes in gradient tone. The histogram presented in Figure 6e demonstrates the various angles at which the most significant changes occur.

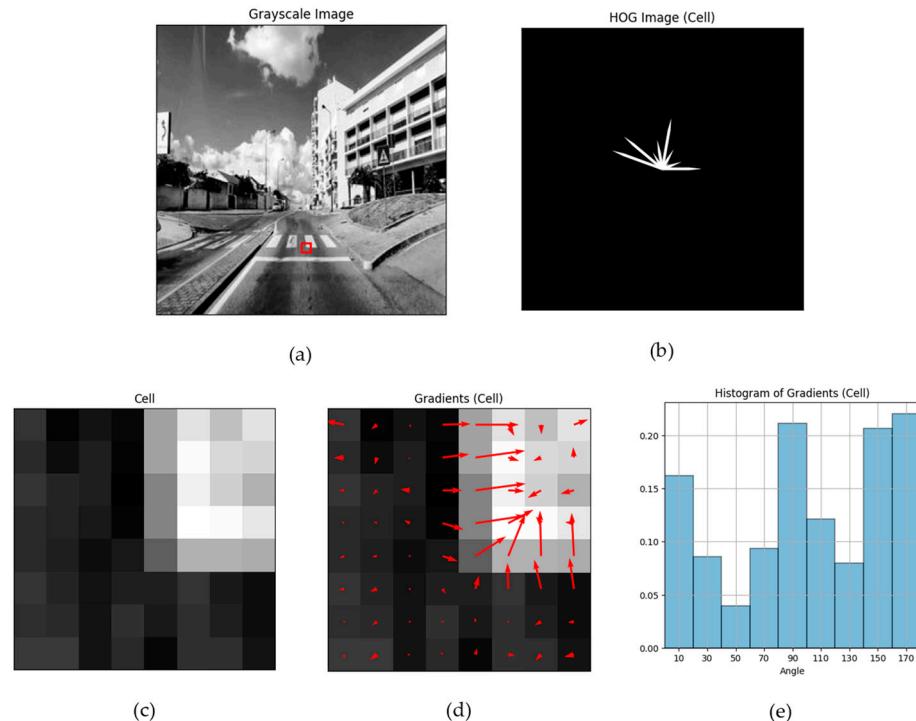


Figure 6. Example of a HOG application: (a) grayscale image; (b–d) selected pixel and gradient tone; (e) histogram graph of gradient tone. Using code available in [29,30].

The study [27] emphasizes the importance of adjusting image dimensions to the 128×64 pixel standard. According to [31], this is a crucial step in achieving better performance by obtaining 8×8 cells. The original images before the HOG application are depicted in Figure 7a,c. Figure 7b,d were evaluated using HOG to assess the state of degradation of pedestrian crossings in the original images. It was observed that Figure 7b exhibits greater wear, while Figure 7d shows good preservation.

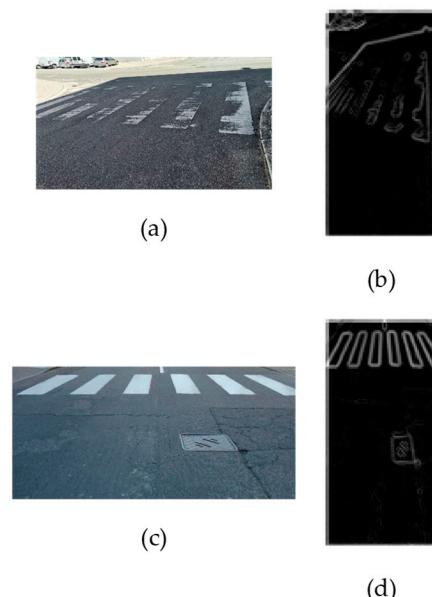


Figure 7. The application of HOG was used to verify the preservation of a pedestrian crossing: (a,c) original images; (b,d) images after HOG.

Maximally Stable Extremal Regions (MSERs) [32] can be used to identify areas in an image with differing properties. To apply this technique, the image must first be converted to grayscale, which allows for the application of various threshold values. This process enables the identification of regions that remain between these thresholds. Pixels with an intensity above the defined threshold will be converted to white, while the rest will be displayed in black. Later, techniques can be applied to find the contours using the defined regions. For instance, the contours can be represented as a list of points using `findContours` [33], which can then be used to extract features. This technique has been demonstrated in the road context, as shown in [34,35]. An example of the application of this technique is provided in Figure 8.

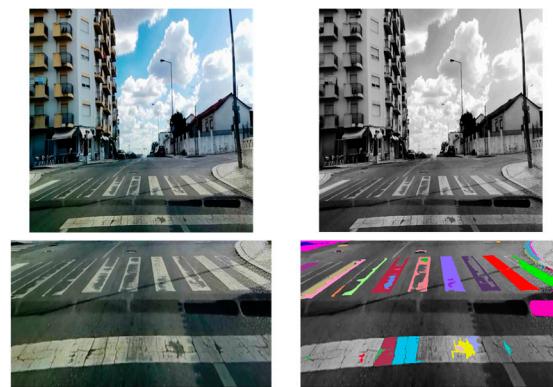


Figure 8. An example showcasing the application of the MSER technique.

Another technique widely used in these contexts is Image Perspective Transformation (IPT). IPT plays a significant role in optimizing CNN models by allowing adjustments to be made to the perspective of images. It is often necessary to emphasize or eliminate less important details and focus on the most relevant areas. Below is a description of some perspective techniques that typically improve object identification in the road context.

The Bird's Eye View technique was developed to provide an aerial view [36], with a specific focus on the road surface. In [37], its effectiveness in detecting road markings while excluding other objects that may appear on the sides of an image was tested. An example of the application of this technique is provided in Figure 9.



Figure 9. Example of the application of the Bird's Eye View technique: (a) is the original image; (b) is the image after application of the technique.

Another example of this technique's application was demonstrated by [38]. Bird's Eye View was used in conjunction with HOG to identify road lanes in the context of self-driving cars.

The Fish Eye View technique can be applied to create a panoramic or spherical image by distorting lines and magnifying the curvature of objects. This technique can be useful for extracting additional information about an object. In this case, it allows for the observation

of the current state of pedestrian crossings. An example of the application of this technique is provided in Figure 10.



Figure 10. Examples of the application of the Fish Eye View technique: (a) the original image; (b) the image after applying the technique.

The techniques mentioned above could potentially be combined with a CNN to improve the detection of objects in images or videos, as noted in [39].

Image Threshold is a technique commonly used for object detection, defect checking, and medical imaging. In this context, it can be used to assess the degradation of pedestrian crossings. The technique involves replacing pixels in an image with either white or black pixels based on their intensity level. The following describes the application of this technique:

- If the intensity of a pixel, represented by I_{ij} , is lower than a fixed threshold value, T, then that pixel will be replaced with the color black.
- Otherwise, it will be replaced with the color white.

Figure 11 presents the outcomes of a combination of techniques, including Threshold, Bird's Eye View, Region of Interest, and Contour Detection, as provided by [33]. It is essential to segment the irrelevant areas and focus on the areas of interest, namely the pavement. To achieve this, methods such as those described above can be applied. Subsequently, the Threshold algorithm is utilized to extract information about the geometry of pedestrian crossings. Figure 11a appears to show some signs of wear on the pedestrian crossing, while Figure 11b seems to show no such signs. Figure 11c,d show the crosswalk stripes in green using the `findContours` function.

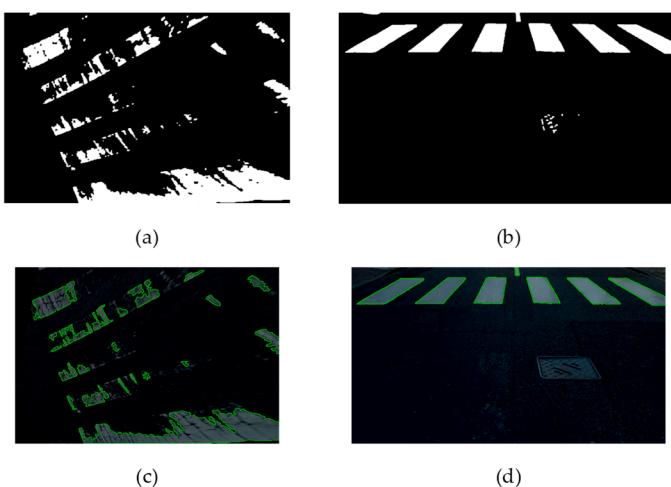


Figure 11. Examples of applying Threshold to images: Binary images are represented in (a,b) where Threshold is applied. Contour Detection is represented in (c,d).

Although the previous results were positive in detecting and classifying the state of pedestrian crossings, it is important to note that certain factors can affect the recognition of the edges of these road markings. These factors include low light conditions, wet surfaces, and shaded areas covering the markings, as can be observed in Figure 12.



Figure 12. Illustration of images featuring less favorable elements.

The Canny Edge Detector [40] is a technique that is widely used for identifying object edges in images. The process involves multiple stages, beginning with the removal of image noise using a Gaussian filter. Next, the image gradient is calculated using operators such as Sobel, Prewitt, or Roberts [41]. One approach to achieve this is by utilizing two threshold values, namely a minimum and a maximum. Any pixels with an intensity greater than the defined maximum will be labeled as strong pixels, while those between the minimum and maximum values will be labeled as weak pixels and subsequently eliminated. It is recommended to consider removing any pixels from the image that may be deemed irrelevant. It is also advisable to evaluate whether the pixels designated as weak are necessary to retain in the final image. Validation is carried out by verifying whether the pixel is connected to a strong pixel. If this is the case, the pixel is retained in the image. This technique is commonly referred to as Edge Tracking by Hysteresis [40]. An illustration of how this technique could be applied is presented in Figure 13.



Figure 13. Example of edge detection using the Canny Edge algorithm.

2.3. Architectures for Classification

Support Vector Machine (SVM) [42] is commonly used for data classification and regression analysis. Its main objective is to identify the hyperplane that best separates data points from different classes. SVM employs two vectors, known as support vectors, which are important for determining the position and orientation of the hyperplane [43]. Its objective is to achieve the largest possible margin, or maximum margin, between data points of different classes, which enables better generalization of unseen data. The SVM algorithm is commonly employed as a classifier based on previously extracted features, often utilizing the HOG and MSER algorithms. Figure 14 illustrates how this algorithm is applied.

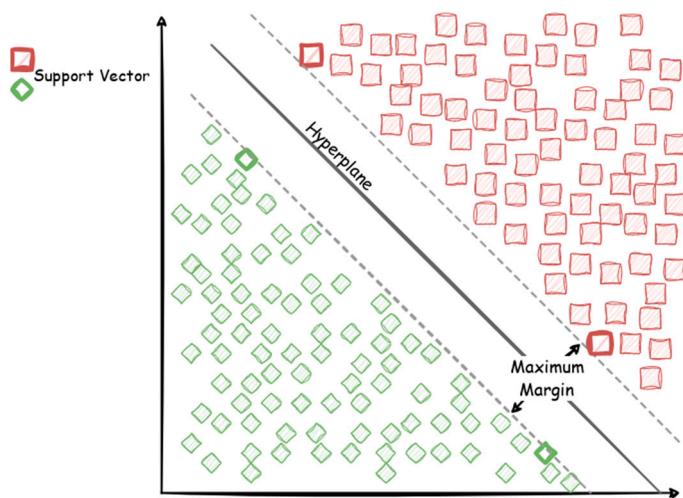


Figure 14. Illustration of the Support Vector Machine concept.

MobileNetV2 [44] is the second iteration of the MobileNet model [19]. Similar to its predecessor, this model performs well on mobile devices due to its architecture, which includes an inverted residual structure with connections between bottleneck layers. The inputs and outputs of these residual blocks are connected via a linear layer, which helps to maintain small feature spaces and minimize the risk of information loss. MobileNetV2 utilizes depthwise separable convolutions [45], which are employed in new CNN architectures. This approach replaces the fully connected layer with a factorized version that splits the standard convolutional layer into two: depthwise convolution and pointwise convolution. By doing so, the standard convolutional layer applies filters and transforms the inputs into a new set of data in two separate steps, resulting in improved efficiency and reduced computational cost. The MobileNet architecture differs from traditional CNN architectures in its use of depthwise convolution, which is separated into two layers: one for filtering and the other for combining inputs. This reduces the model size and computational cost. Additionally, a 3×3 convolutional layer is used, followed by a batch normalizer (BN) and an ReLU. The MobileNet architecture utilizes a separation of the 3×3 convolutional layer into 3×3 depthwise convolution and 1×1 pointwise convolution. MobileNetV2 is frequently employed as a backbone for the SSD model, especially in devices with limited resources, like mobile phones or Internet of Things (IoT) devices. Figure 15 depicts the distinctions between the architectures described.

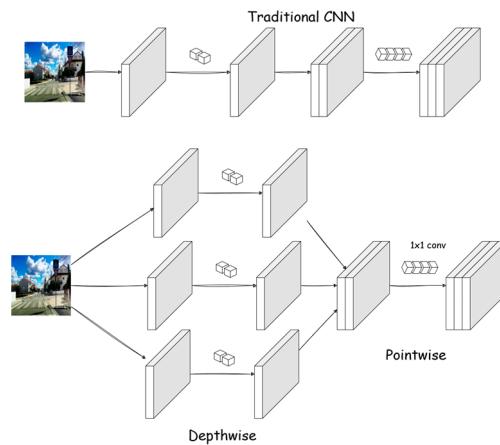


Figure 15. Traditional CNN architecture compared to a MobileNet model architecture.

VGG-16 [17] uses a conventional convolutional approach, which has resulted in a highly resilient architecture. Its remarkable performance in computer vision tasks is due to its 16-layer architecture, which consists of 13 convolutional layers and 3 fully connected

layers. The VGG-16 model is designed to process RGB images of a fixed size, typically 224×224 pixels. Its 13 convolutional layers are organized into 5 blocks, each consisting of 3×3 layers, which are then followed by a max-pooling layer. After the convolutional blocks, there are 3 fully connected (FC) layers. The first two FC layers have 4096 channels, while the last layer has only 1000 channels per class. The architecture uses a softmax layer that predicts the class of the input image [17]. However, it is worth noting that this architecture has some weaknesses, particularly in terms of model size and number of parameters, which result in longer training times [46]. This characteristic may be undesirable for devices with limited resources. Figure 16 illustrates the network's architecture.

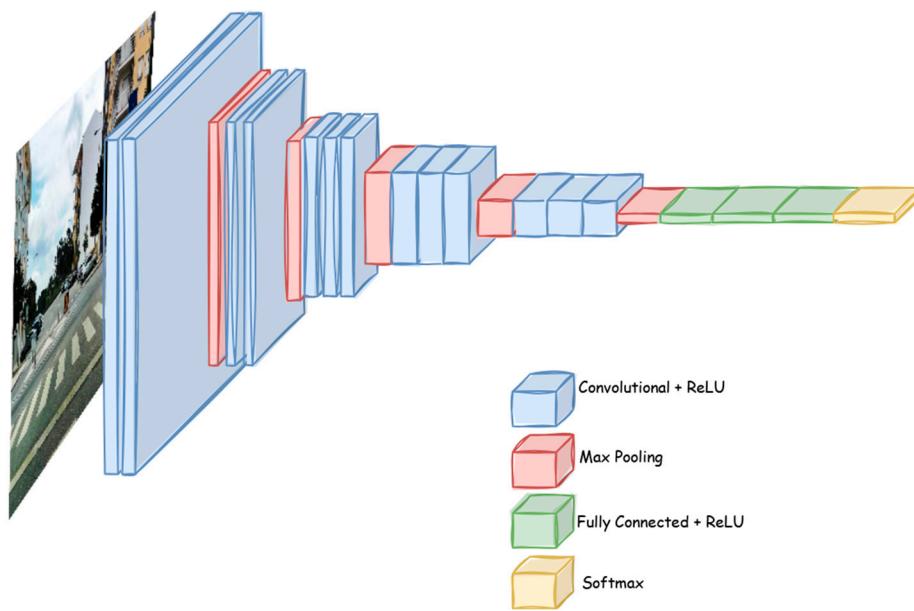


Figure 16. Illustration of the architecture of the VGG-16 model. Adapted from [17].

3. Related Work

In a study conducted in [47], pedestrian crossings were detected in real time using the YOLOv5 model, which was implemented on a Jetson Nano device. The dataset used in the study was created by capturing videos using a camera mounted on a vehicle in various atmospheric conditions. Additionally, to ensure a diverse dataset, the Synthetic Fog augmentation algorithm was employed. The model's robustness was further improved by creating images with a fog scenario. To handle complex environments, such as excessive brightness, fog, and wet ground, a Squeeze-and-Excitation unit [48] was incorporated to improve accuracy. Furthermore, the Negative Sample Training and Region of Interest (ROI) algorithms were implemented to enhance the model's detection speed. Hence, the YOLOv5 model was adapted, leading to an F1 score of 94.83% in complex scenarios and 98% in less complex scenarios.

In [37], a study was carried out to evaluate the effectiveness of detecting road markings using segmentation and object detection algorithms. The researchers created a new dataset due to the inadequacy of existing ones, which lacked visible markings and appropriate resolutions and contained few images in various atmospheric environments. The study assessed the performance of various models, including SSD MobileNet version 1, SSD Inception version 2, and two versions of Mask RCNN, with the Inception-v2 and ResNet 50 architecture being applied. Furthermore, the MobileNet networks underwent optimization using the Image Perspective Transformation (IPT) technique, which transformed the images into a 'bird's-eye view', eliminating unnecessary background elements and allowing the model to focus on the object of interest. Based on the results, it was found that the Mask R-CNN networks outperformed the other models. Both the ResNet 50 and Inception-v2 models have shown impressive performance in different scenarios. The ResNet 50 model

has demonstrated superior performance in scenarios with rain, shadows, and nearby obstacles, while the Inception-v2 model has excelled in nighttime environments.

In [38], it was proposed to use an algorithm based solely on computer vision techniques and data obtained from a camera to detect lanes. The proposed approach suggested using color segmentation, a fundamental technique for identifying objects or specific regions of an image with distinct colors, in a minimalist manner. This method allowed the algorithm to focus solely on predetermined color segments of the image, simplifying processing and analysis. The approach used in this study was found to be effective in identifying information related to carriageway markings. This is because road markings around the world are mostly in shades of yellow or white. To distinguish these markings from the rest of the road surface, the Canny Edge Detector algorithm was applied. This algorithm identifies the pixels of different intensities that make up the distinction between road markings and the road surface. The authors proposed two approaches for identifying regions of a given frame with the highest density of non-zero pixels. It was found that the bird's-eye view technique with sliding window search was less effective in identifying curved markings, while the second approach, histogram analysis, was more robust and less susceptible to adverse conditions.

In [49], an architecture called Vanishing Point Prediction (VPP) was proposed to identify carriageways and other markings, creating an illusion of a three-dimensional surface in a two-dimensional image. The proposed architecture aimed to accomplish four tasks: grid regression, object detection, multi-label classification, and vanishing point prediction. The authors also created a dataset of approximately 20,000 images with 17 classes. The system was developed based on four different scenarios: heavy rain, normal rain, no rain, and nighttime rain. The collected images had a resolution of 1288×728 . The authors suggested using grid-level annotation instead of pixel-level annotation. This involved dividing the image into an 8×8 grid and labeling the grid cell if any pixel from the original annotation is inside it. This approach allowed for the integration of two independent objectives: identifying road markings and carriageway boundaries. In the context of fully convolutional networks, it has been demonstrated that grid-level annotations outperform pixel-level annotations. The authors concluded that this architecture was robust in various weather conditions and operated in real time.

In [50], an approach was presented for the real-time detection of road markings, with emphasis on classifying damaged or undamaged markings, using object detection algorithms in Edge Computing environments. The study began by constructing a dataset, obtained by capturing images from cameras installed in garbage collection trucks. The initial use of the Deep-on-Edge (DOE) algorithm on a Raspberry Pi 3, with the camera pointed at the ground, was found to be unsuitable for the intended application. To overcome this limitation, the YOLO algorithm was proposed, which allowed the camera to be configured on the garbage collection trucks. Furthermore, the VGG-16 architecture was employed to extract the necessary visual characteristics for identifying road markings. The authors have proposed other architectures, such as Mask R-CNN and SegNet. However, it is worth noting that these architectures may have a higher inference time, which could potentially pose challenges in detecting other road markings.

In [51], a model was proposed to detect pedestrian crossings using Mask R-CNN. The instance segmentation architecture employed ResNet-101 as its backbone. The dataset used to train and validate the model was created by collecting images using a dashcam and was divided into two parts: 80% for training and 20% for validation. The model was also tested using 30 images with a resolution of 640×480 . The architecture was configured with a learning rate of 0.001 and a batch size of 100 and trained for 30 epochs. Mask images were used to calculate the reverse loss and optimize the model. The model was trained using an Intel (R) Core (TM) i7-7700k CPU, 2 NVIDIA GeForce 1080 Ti for GPU acceleration, and 32 G of memory with the Matterport framework. The test images yielded an mAP of 97%.

In [52], the authors presented an algorithm that identified damage to road markings. To ensure privacy, the algorithm was implemented using a half-celestial camera positioned

on the underside of a vehicle, specifically a rubbish lorry. The algorithm was based on two methods: Measure Outline Percentage, which created a border around the road mark to indicate how many objects are present inside it, and Measure Detect Percentage, which formed a binary image containing only black and white to allow the shape of the road mark to be discerned. The study involved the collection of 242 images, out of which 164 were found to be damaged while the remaining 78 were undamaged. Two methods were employed to measure the accuracy of the road mark detection. The Measure Outline Percentage method demonstrated an accuracy of 76.6%, whereas the Measure Detect Percentage exhibited 51.8%. It is worth noting that both methods had limitations related to atmospheric conditions, as an appropriate threshold needed to be defined, and complex road markings.

A pavement damage detection model was proposed in [53], based on the fusion of thermal and RGB images. This fusion allowed for the collection of important information from a set of images. For instance, while an RGB image may detect n objects, thermal images can detect the same objects and others not detected in the RGB image, thereby enhancing the model. The dataset was created by using a FLIR ONE camera attached to a mobile phone. The captured images were extracted using the MATLAB library API, resulting in 1500 RGB images and 1500 thermal images. To expand the dataset, the authors applied Gaussian noise to the RGB images and randomly reduced or expanded the color bar by 20% in the thermal images. These procedures resulted in a total of 13,500 final images. The EfficientNet model was trained using a high-performance CPU and GPU. The Stochastic Gradient Descent (SGD) algorithm with a momentum of 0.7 was chosen to update the model's weights. For the learning rate, the authors chose an initial value of 0.01, which was adjusted over time using a multiple learning rate strategy: the initial learning rate was multiplied by 0.6 in each iteration until it reached a maximum of 30,000, after which the learning rate remained constant. The study suggested that combining fused images with RGB images may result in higher accuracy rates, as demonstrated by the achieved 98.34% accuracy. Furthermore, the use of an augmented dataset appears to have improved the stability of the detection model, resulting in an accuracy rate of 98.35%, a recovery rate of 98.34%, and an F1 score of 98.34%.

In [54], a study proposed the use of image processing techniques to identify pedestrian crossings. Two techniques, namely the Canny Edge Detector and Hough Transform, were considered for this purpose. These techniques facilitated the identification of the edges present in the stripes of pedestrian crossings and their respective geometric shapes. To determine whether the identified object was a pedestrian crossing, the authors used the Hough technique to draw parallel lines on the edges of the pedestrian crossings. The results were obtained using the Canny Edge technique. If the total number of parallel lines exceeded a certain constant, the image was considered to contain a pedestrian crossing. Furthermore, the authors utilized ROI and Inverse Perspective Mapping (IPM) to eliminate unwanted objects and enhance the study's outcomes. The techniques used allowed the identification of 96% of pedestrian crossings in 51 test images, all of which were under favorable lighting conditions.

In [55], a method was proposed for detecting and recognizing text on road markings. The method was based on the MSER technique, which was used to identify possible candidate areas where road markings may be present. Additionally, IPM was used to create a top-down view, which facilitated the extraction of text from road markings and provided a better perspective. The authors divided the detection process into two stages. In the first stage, they used HOG and SVM techniques to detect the symbols on the road markings. In the second stage, they used optical character recognition to detect the text on these markings. The authors tested the performance of these techniques on videos and achieved an F1 score of 85% for text and 91% for symbols.

In [56], a study was carried out to detect pedestrian crossings in urban environments, considering both the pedestrian and driver perspectives. The dataset was divided into three sets for training, validation, and testing, with an 80%, 10%, and 10% split, respectively.

Data augmentation was applied to increase the dataset. Two models, YOLOv7 and Faster R-CNN (versions R101-FPN and X101-FPN), were evaluated. The results obtained were similar for both models. The YOLOv7 model achieved an accuracy of 98.6%, whereas the Faster R-CNN model achieved an accuracy of 98.29%.

In [57], the U-Net model was proposed for pedestrian crossing segmentation in images. A dataset of 150 training images and 30 validation images was created. To extract information from the images, ResNet-34 was initially applied to improve the U-Net model. The convolution layer was expanded to increase the receptive field of the feature points, while still preserving the resolution of the feature map. Following this, the abstract features, such as shapes, colors, tones, or textures, were restored to their original size by utilizing the original U-Net subsampling network, in addition to complementary information from the jump link. As a result, the model achieved an accuracy of 94.6%. It has been suggested that the use of ResNet-34 may improve the training of the U-Net model, potentially mitigating degradation over time.

In [58], a method was proposed for detecting pedestrian crossings. The method employed various techniques, including adaptive histogram equalization, Flood Fill operation, and Hough Transform. Subsequently, the SVM classification algorithm was applied. The authors first applied adaptive histogram equalization to enhance the contrast of the pedestrian crossing markings. Then, they converted the resulting image into a binary image with only black and white pixels using the Otsu method. To reduce the number of non-candidate objects in the image, they applied a statistical threshold process, which eliminated objects with an edge size below the defined value. To determine whether an image contains a pedestrian crossing, the authors employed the Flood Fill and Canny Edge methods. To identify the edges of the pedestrian crossing stripes, which would later be used to extract information, they used Uniform Local Binary in conjunction with ROI with a dimension of 128 by 128 pixels. Under varying atmospheric and lighting conditions, this method was shown to achieve high accuracy rates of 97.95% and 98.17%, respectively.

Table 1 summarizes the articles analyzed in the state-of-the-art review.

Table 1. Summary of the studies included in the state-of-the-art review.

References	Year of Publication	Dataset	Purpose of Study	Methodology	Results
[37]	2022	Dataset created by authors	To help make up for the lack of public datasets related to road markings.	SSD MobileNetV1, SSD InceptionV2, Mask-RCNN-InceptionV2, and Mask-RCNN-ResNet50	The ResNet 50 version performed better in scenarios with rain, shadows, and nearby obstacles. The Inception-v2 excelled in nighttime environments.
[38]	2020	N/A	Propose an algorithm for identifying road markings using only histogram analysis and perspective transformation techniques.	Perspective transformation, Threshold.	The results indicate that the curved markings approach outperformed the minimalist approach. Conversely, the second approach demonstrated greater resilience and was less vulnerable to adverse conditions.
[47]	2022	CDNet dataset, created by authors	Detect pedestrian crossings in real time and optimizing algorithms for devices with limited resources.	YOLOv5 with NST, ROI, SVM, and FOG algorithms.	YOLOv5 achieved an F1 score of 94.83% in complex scenarios. In less complex scenarios, 98% was achieved.
[49]	2017	Dataset created by authors	Propose a multi-task network to detect carriageways and road markings in adverse weather conditions.	VPGNet for road mark detection and MSER, FAST, and HOG to extract features and SVM to produce labels.	VPGNet has proved to be robust in different weather conditions, in real time.
[50]	2017	Dataset created by authors	Create a solution to identify road markings using the YOLO and VGG-16 algorithms.	VGG-16 as a way of extracting characteristics and using YOLO to identify brands.	It showed an average precision of 22.4% for pedestrian crossings.

Table 1. Cont.

References	Year of Publication	Dataset	Purpose of Study	Methodology	Results
[51]	2019	Dataset created by authors	Detect pedestrian crossings using Instance Segmentation algorithms.	Mask R-CNN using ResNet-101 for the network backbone.	Mask R-CNN with ResNet-101 as the backbone, recorded an mAP of 97%.
[52]	2015	Dataset created by authors	Use of an image recognition algorithm to identify possible damage to road markings.	Image processing techniques.	Measure Outline Percentage showed a precision of 76.6%, Measure Detect Percentage achieved 51.8%. Using an appropriate threshold value for various atmospheric scenarios is a problem.
[53]	2022	Dataset created by authors	Propose a pavement damage detection model based on the fusion of RGB thermal images, using EfficientNet as a backbone.	EfficientNet B4 and EfficientNet B5	Using image fusion to detect damage to road markings can achieve an accuracy of 98.34%. When using the augmented dataset, the detection model appears to be more stable, achieving 98.35% accuracy, 98.34% recovery, and 98.34% F1 score.
[54]	2021	N/A	Propose an algorithm capable of identifying pedestrian crossings using image processing techniques.	ROI, Hough Transform line, Canny Edge, and Inverse perspective Mapping.	There were 96% of pedestrian crossings identified in 51 test images, all of which were in good lighting conditions.
[55]	2015	N/A	Create a solution to identify road markings and recognize text on them, using image processing techniques.	MSER, SVM, HOG, inverse perspective mapping, and optical character recognition.	An F score of about 85% was obtained for text and 91% for symbols.
[56]	2023	Dataset created by authors	Create a solution to identify pedestrian crossings using the YOLO and Fast R-CNN models.	Faster R-CNN (R101-FPN and X101-FPN) and YOLOv7.	YOLOv7 obtained an accuracy of 98.6%, and Faster R-CNN obtained an accuracy of 98.29%.
[57]	2020	Dataset created by authors	Using an image segmentation model to identify pedestrian crossings.	U-Net together with ResNet-34.	The U-Net model showed an accuracy of 94.6%. The use of ResNet-34 improved the training of the U-Net model, which could prevent the model from degrading over training time.
[58]	2019	N/A	Propose an algorithm capable of identifying pedestrian crossings using image processing techniques.	Hough Transform line, Canny Edge, adaptive histogram equalization, SVM, and Flood Fill.	In different atmospheric conditions, the accuracy was 97.95%, while in different lighting conditions the accuracy was 98.17%.

The work presented in this paper represents a first step in an ongoing effort to develop a prototype that is appropriate for use in vehicles owned by local councils or other entities, which uses computer vision techniques to collect information on road conditions, such as the location and the wear and tear of pedestrian crossings. Therefore, compact, modular, and mobile hardware solutions, such as Raspberry, Jetson, and Google Coral, are the most suitable options for the prototype. To make the best use of the limited computing resources of such devices, the above studies suggest that the most promising models for detecting pedestrian crossings are YOLOv4-tiny [59], SSD-MobileNetV2 [60], and SSD-EfficientDet-D0 [60]. Although the Mask R-CNN model was an option, it was not considered due to its Two-Stage architecture.

4. Performance Evaluation

This section presents a performance comparison of Histogram of Oriented Gradients, Maximally Stable Extremal Regions, Canny Edge, thresholding methods, and Convolutional Neural Networks models. It begins by introducing a new dataset that was developed purposefully to assess their performance for detecting pedestrian crossings. Then, it pro-

ceeds to the performance assessment study. Results and discussion are provided for the conducted experiments.

4.1. Description of the Dataset

The preparation procedure for the dataset to be used in this study began with an examination of the Roboflow platform [61] to identify datasets that could be suitable. However, it was found that these datasets mainly contained images of pedestrian crossings from the perspective of a pedestrian. Since this project aimed to create a device for vehicle integration, the use of such datasets could affect the detection capability. Therefore, a new dataset had to be created.

For this reason, a mobile phone holder was incorporated into the vehicle's windscreen, as shown in Figure 17, to ensure an appropriate image perspective. To ensure safe driving, a decision was made to capture videos instead of taking static photos, at a resolution of 1920×1080 at 30 FPS. The videos were recorded at a resolution of 1920 by 1080 at 30 FPS and were then divided into frames. From these frames, the most relevant ones were selected. The approach was deemed suitable and efficient in capturing images of pedestrian crossings under realistic traffic conditions, meeting the requirements of this work.



Figure 17. Cell phone setup for capturing videos.

Table 2 presents the surface conditions and recording times, which are important factors in building a robust dataset. It is recommended to consider potential obstacles, excessive luminosity, and shadows that may hinder pedestrian crossing detection. Therefore, it is advisable to include a variety of images with different characteristics.

Table 2. Dataset description.

Date of Capture	Surface Conditions	Time of Capture	Number of Images
8 September 2023	Dry	Morning	151
9 September 2023	Wet	Morning	137
12 September 2023	Dry	Afternoon	156
2 October 2023	Dry	End of the day	143
3 October 2023	Dry	Morning	55
			Total: 642

It is worth noting that some of the recordings were made during a time of day when the sun was often lower on the horizon. This decision was made to avoid potential visibility issues and to account for a higher volume of users, which could make it difficult to see pedestrian crossings. Furthermore, this approach helped to diversify the range of images captured.

The city roads and routes, illustrated in Figure 18, used in this study for the dataset construction process are from the city of Castelo Branco, Portugal. It is worth mentioning that the study did not include the areas of the city of Castelo Branco with cobbled

surface, as they were not the focus and did not exhibit significant signs of wear in the pedestrian crossings.

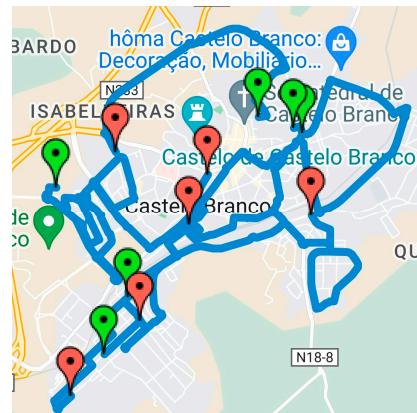


Figure 18. Routes taken to create the dataset in the city of Castelo Branco, Portugal.

To compile the dataset, areas with a high concentration of pedestrian crossings were strategically selected. This approach allowed for the capture of a diverse set of images and information related to pedestrian crossings, contributing to a more comprehensive and representative dataset. The annotation tool available on the Roboflow platform was then used. This task aimed to determine the location and classification of objects. The resulting data were used for model training and performance evaluation. Figure 19 illustrates the process of generating annotations in Roboflow.



Figure 19. Example of creating annotations on the Roboflow platform.

To further improve the dataset, several pre-processing techniques outlined in Section 2 were implemented, in addition to utilizing the data augmentation tools available on the Roboflow platform. The specific techniques employed are detailed in Table 3. The resulting differences between each version are illustrated in Figure 20, and hence the modifications made to the dataset. Figure 20a shows the original image without any changes. Figure 20b shows the changes using Version A of Table 3. Figure 20c shows the Version B of the dataset. The contrast between the road markings is greater than in Version A.

Table 3. Techniques applied to the dataset.

Techniques	Version A	Version B
Resize ¹	616 × 616	608 × 608
Auto-Adjust Contrast ¹	Histogram Equalization	Adaptive Equalization
Flip (Horizontal) ²	Used	Used
Grayscale ²	Applied to 25% of images	Applied to 25% of images
Noise ²	Up to 1% of pixels	Not used
Brightness ³	Not used	Between -10% and +10%
Exposure ³	Not used	Between -10% and +10%

¹ Pre-processing, ² Data augmentation, ³ Bounding box data augmentation.

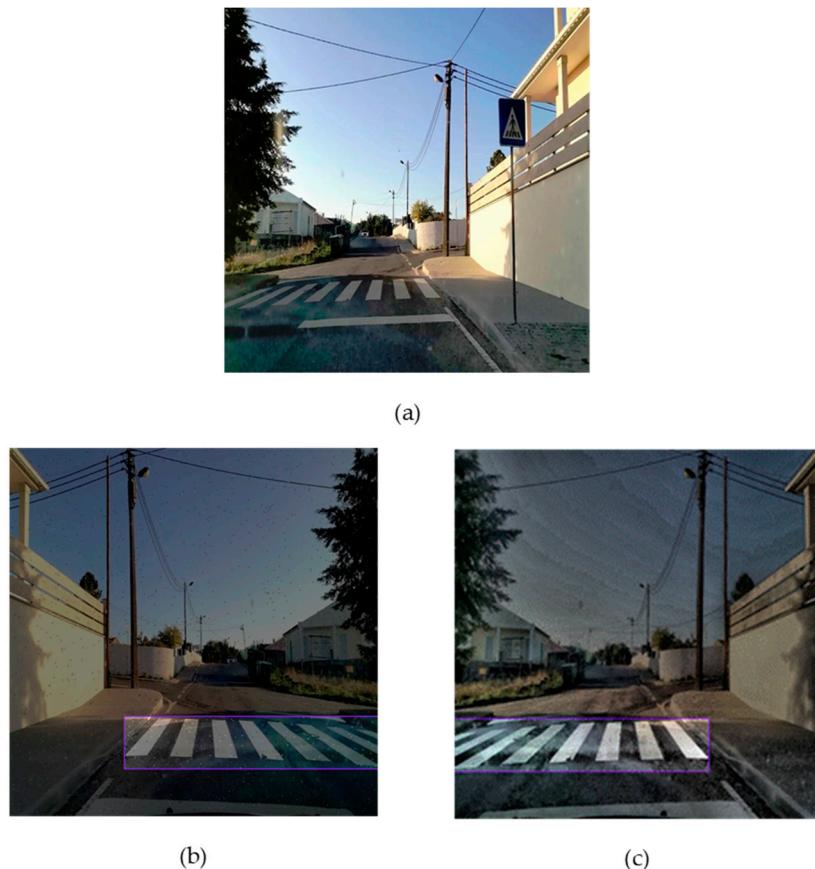


Figure 20. Original image (a) and images (b,c) after the changes have been applied.

The purpose of this updated dataset was to improve the contrast between the road markings and the surrounding environment by pre-processing the images. To achieve this, the Adaptive Equalization technique available on the Roboflow platform was utilized.

Additionally, the dataset was divided into training, validation, and test sets to facilitate model learning and validation. This separation facilitated the analysis of the model's behavior to identify signs of overfitting. Additionally, it enabled the evaluation of the model's capability to detect objects, particularly pedestrian crossings, in images that were not previously seen.

Table 4 presents the distribution of images across the sets. The dataset created in the context of this work is available on the Roboflow platform [62], and it is divided into three partitions: 80% for training, 15% for validation, and the remaining 5% for testing.

Table 4. Number of images in the training, validation, and test sets.

Class	Train	Validation	Test	Total
Passadeira *	1190 ¹	259 ¹	55 ¹	1504 ¹
	1250 ²	279 ²	20 ²	1549 ²

* Crosswalk in English, ¹ Version A, ² Version B.

4.2. Image Processing

This section presents the image processing techniques that can be used to identify pedestrian crossings and determine their wear and tear. Demonstrative examples will also be provided. Figure 21 illustrates one possible approach to these techniques.

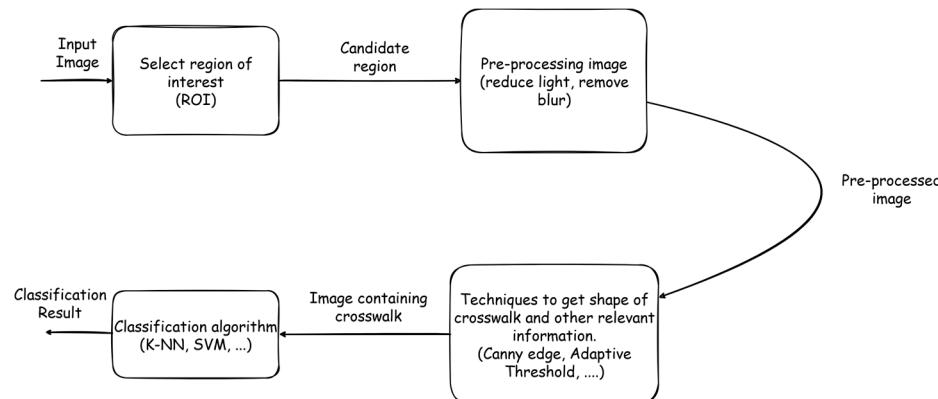


Figure 21. Method for detecting signs of usage on pedestrian crossings.

Figures 22 and 23a show that the HOG technique can be used to obtain the shape of pedestrian crossings and visualize the wear and tear caused by vehicles. This technique provides valuable information as a feature descriptor, which can be utilized by classification algorithms such as k-Nearest Neighbor (K-NN), SVM, or decision trees. As can be seen in Figure 23b, it can be challenging to use this technique in areas with high levels of light.



Figure 22. Example of the application of HOG in the context of pedestrian crossings together with the ROI technique.

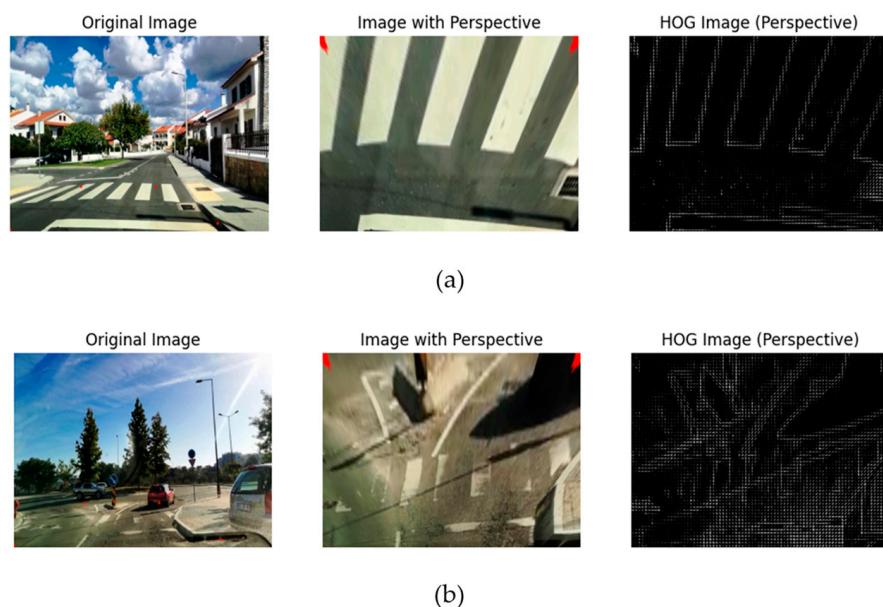


Figure 23. Example of combining HOG, ROI, and perspective techniques in favorable (a) and unfavorable (b) crosswalk conditions.

Figure 24 displays examples of the image processing techniques described in this paper. In Figure 24a(2), the Adaptive Threshold technique was used to identify a pedestrian crossing. The threshold of the THRESH_BINARY type was provided by the OpenCV library [33]. The observed outcome indicated signs of wear on nearly all the stripes of the pedestrian crossing, which were not discernible in the original image Figure 24a(1). To address this issue, a threshold of type THRESH_OTSU was utilized, as shown in Figure 24a(3). This method automatically calculated a threshold value between the two peaks of the histogram, as demonstrated in Figure 25. This approach enabled finding a threshold value that can adjust to varying lighting conditions.

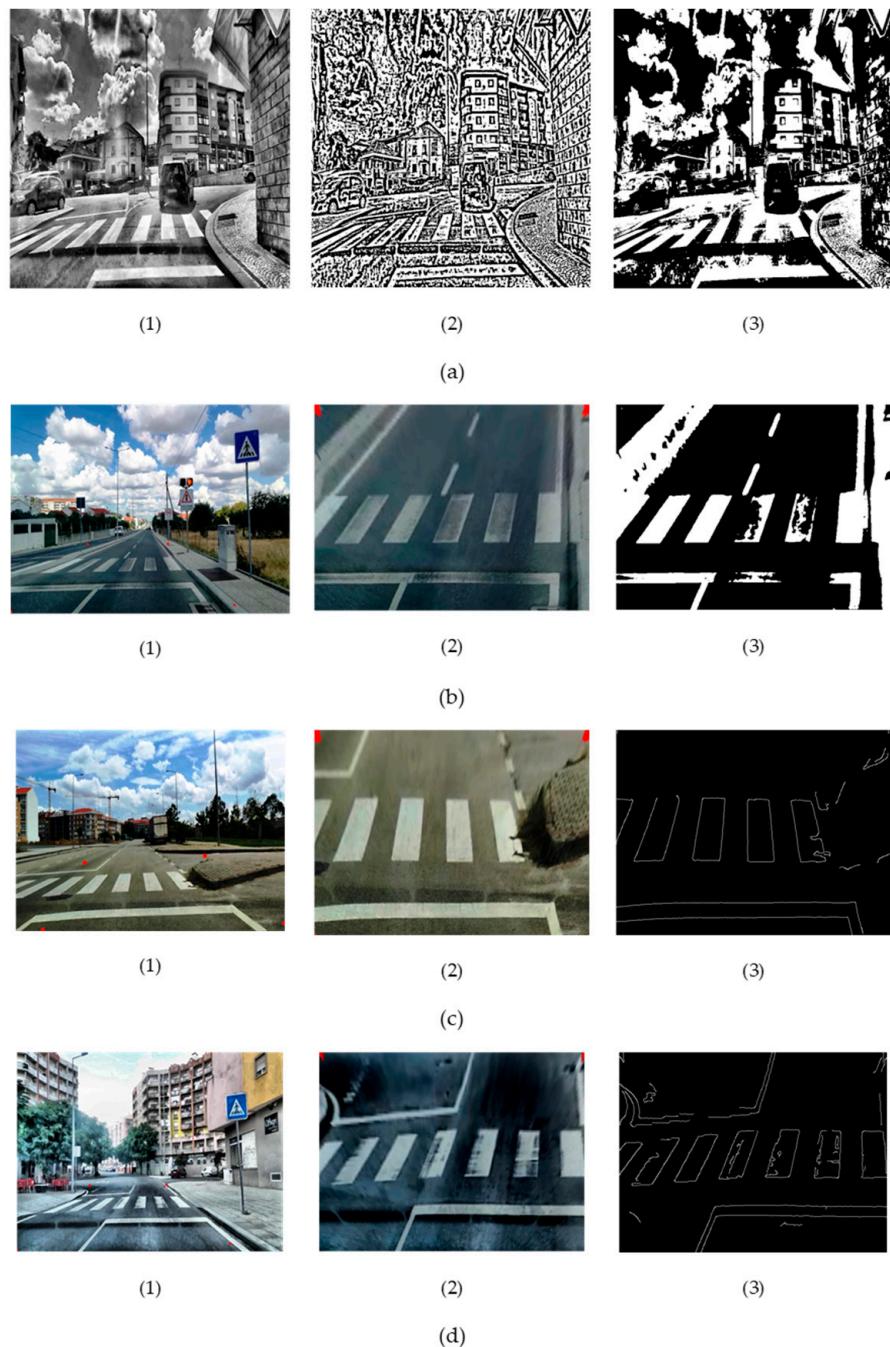


Figure 24. Examples of the application of image processing techniques. Different Threshold methods applied on (a) and (b), (c) and (d) shows the use of Canny Edge algorithm.

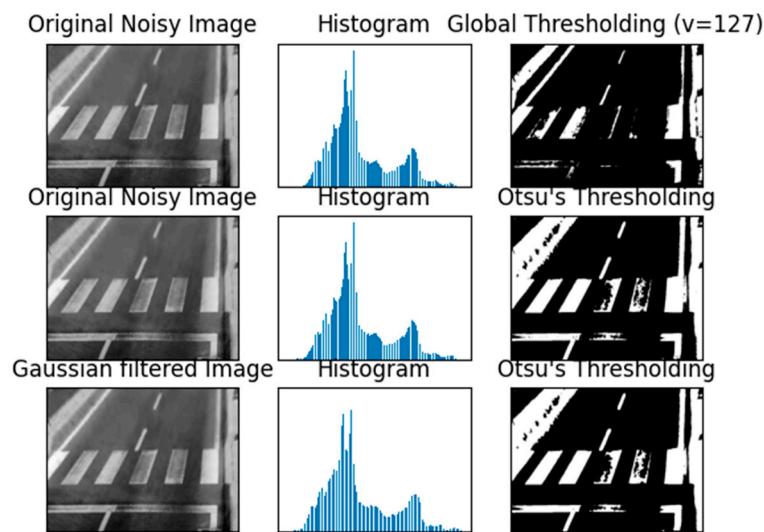


Figure 25. Comparison of a fixed threshold value with the automatically calculated threshold.

It could be argued that the image contains some extraneous information. Therefore, the use of ROI could be considered advantageous as it allows for the selection of the specific area where the object is present. To enhance the identification of wear on pedestrian crossings, it is important to consider a perspective that offers such conditions. Figure 24b(1) shows a pedestrian crossing where ROI, or region of interest, is applied, with the region delimited by red dots. Subsequently, in Figure 24b(2), a perspective is applied based on the selected region to obtain a better perception of wear. By applying the same thresholding concept described above and using a perspective, Figure 24b(3) shows the wear and tear on the pedestrian crossing. Another technique that can be used is Canny Edge Detection, which identifies the edges of objects. In this case, it identified the edges of the pedestrian crossing stripes and the wear and tear within them, as seen in Figure 24c(3),d(3). Where an ROI is initially selected on the original images Figure 24c(1),d(1) to obtain better results, the results are shown in Figure 24c(2),d(2).

During the conducted tests, it was observed that the use of these techniques in environments with abrupt lighting variations may present some challenges, as shown in Figure 26a(1),b(1). Figure 26a,b illustrate that the use of a threshold, even when automatically calculated using the OTSU method, was more effective in low-light or wet surface environments, as demonstrated in Figure 26b(3). On the other hand, Canny Edge presents challenges in both environments, which could be attributed to the use of ROI with perspective, as illustrated in Figure 27a,b.

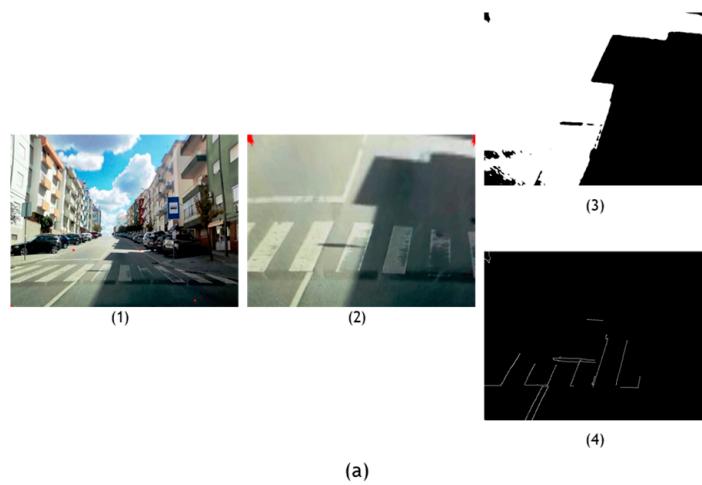


Figure 26. Cont.

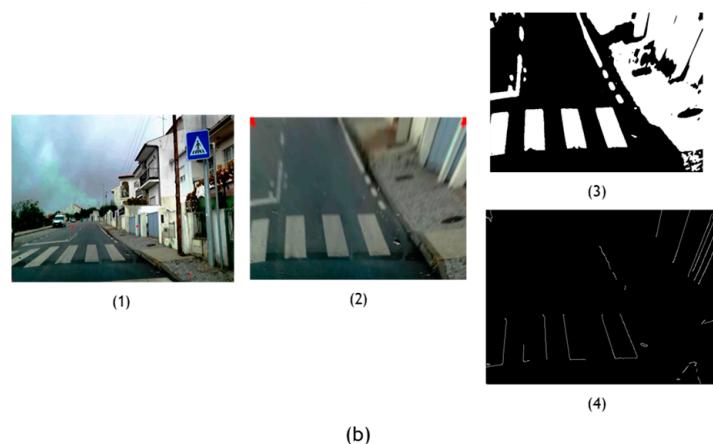


Figure 26. Visualization of Canny Edge and Adaptive Threshold behavior in the presence of different types of lighting, (a(1–4)) shows the behavior of Canny Edge and Adaptive Threshold on shadows; (b(1–4)) illustrates behavior on wet floor conditions.

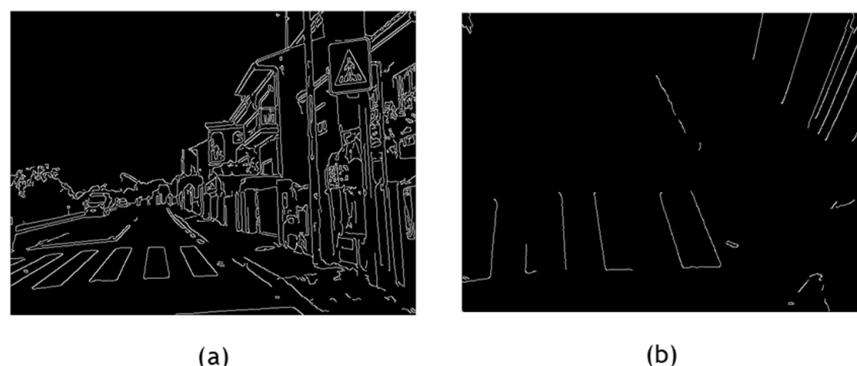


Figure 27. Comparison of Canny Edge without using a perspective and ROI, (a) uses Canny Edge and (b) uses Canny Edge and ROI.

With regards to the MSER technique, a perspective, and a region of interest (ROI) were applied to enhance the results. Figure 28a displays the original image, Figure 28b shows the image after the perspective and ROI selection were applied. Figure 28c,d are similar, except that Figure 28d represents the mask that can be obtained using the MSER technique. It is worth noting that Figure 28d shows that some elements, such as lane separation lines, were obtained unnecessarily. To address these issues and potentially improve the results, it may be helpful to consider filtering elements based on their minimum and maximum area values. This approach would exclude regions that fall outside of these values, as demonstrated in Figure 28d.



Figure 28. Cont.

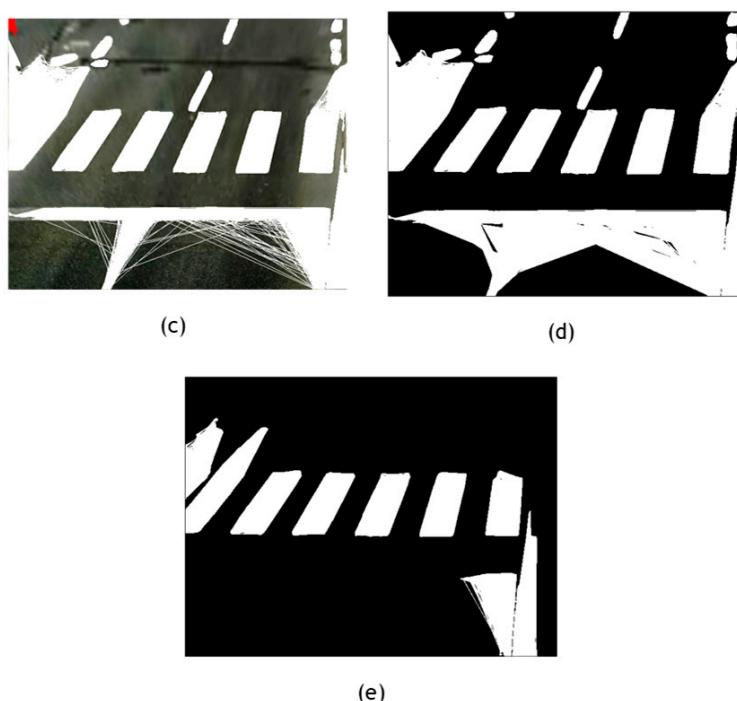


Figure 28. Application of the MSER technique, (a) original image; (b) ROI selected; (c,d) areas detected by MSER technique; (e) fine tuned parameters to reduce irrelevant areas.

In conclusion, it is recommended to apply the presented techniques in a controlled environment, taking care to avoid sudden changes in lighting or wet surfaces. Even if the image is pre-processed to reduce illumination or increase contrast, it is important to consider that each image may exhibit unique characteristics. For instance, it is possible that the effectiveness of these techniques may vary depending on the time of day and whether the image has been pre-processed. This suggests that a universal logic cannot be applied to all variables in each environment, which may result in less favorable outcomes.

4.3. Convolutional Neural Network Models

This subsection presents a performance assessment study of the CNN models YOLOv4-tiny, SSD-MobileNetV2, and SSD-EfficientDet-D0, which were selected in Section 3 as the most promising. Firstly, the benchmark scenario is presented. After that, the performance metrics are described. Then, the results obtained after training the models are presented, along with some tests carried out to improve the identification of pedestrian crossings. Examples of pedestrian crossings detected by these models are presented.

4.3.1. Benchmark Scenario

The YOLOv4-tiny model was trained and tested on the Google Colab platform [63], while the other models were trained on the Kaggle platform [64]. The Kaggle platform was preferred for the other models as they required more time to complete their training, and this platform allowed the process to continue running in the background until it reached a 12 h time limit.

The training on the Google Colab platform was carried out using an Intel(R) Xeon(R) CPU @ 2.00 GHz, 12 GB of RAM, and a Tesla T4 GPU. The Kaggle platform made use of an Intel(R) Xeon(R) CPU @ 2.20 GHz, 32 GB of RAM, and two Tesla T4 GPUs, which increased its computing power.

4.3.2. Performance Metrics

To evaluate the performance of these models in object detection and classification, YOLOv4-tiny was trained for 6000 iterations with a batch size of 32, which is equivalent to

141 epochs, as per Equation (1). The SSD-MobileNet-V2 model underwent 50,000 iterations, resulting in 640 epochs. The SSD-EfficientDet-D0 model underwent 200,000 iterations, resulting in 640 epochs.

$$\text{Epochs} = \frac{\text{number of interactions}}{\frac{\text{number of training images}}{\text{batch}}} \quad (1)$$

The model's performance was assessed using the mean Average Precision (mAP) metric. The Average Precision was calculated using Equation (2), while Precision (P) and Recall (R) were calculated using Equation (3) and Equation (4), respectively. Equation (3) and Equation (4) provide definitions for TP, FP, and FN, representing True Positives, False Positives, and False Negatives, respectively. Additionally, Equation (5) was utilized to calculate mAP.

$$AP = \int_0^1 P_{(R)} dR \quad (2)$$

$$P = \frac{TP}{TP + FP'} \quad (3)$$

$$R = \frac{TP}{TP + FN'} \quad (4)$$

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k \quad (5)$$

Overfitting [65] can be a common occurrence when training CNNs. The model may perform well when trained on specific data but may not perform as well on new and unseen data. This phenomenon can result from models that memorize the training data, including irrelevant noise, but fail to learn the underlying patterns [66]. To address this issue, it is important to find a balance between accurately capturing meaningful patterns and avoiding an overly complex model that overfits the training data [66]. Figure 29 illustrates the concept of overfitting.

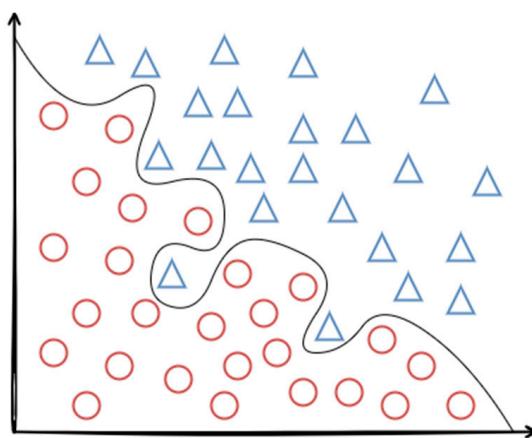


Figure 29. Illustration of the overfitting concept.

One possible approach to mitigate overfitting is to halt the training of the model at a specific point, commonly referred to as early stopping [65]. This entails observing the model's performance on a validation dataset throughout training to determine the point at which it starts to plateau or exhibit signs of deterioration. Once it is determined that the validation dataset is no longer improving or is deteriorating, indicating an increased number of errors, training should be terminated. This prevents the model from memorizing the characteristics of the training dataset. This concept is depicted in Figure 30.

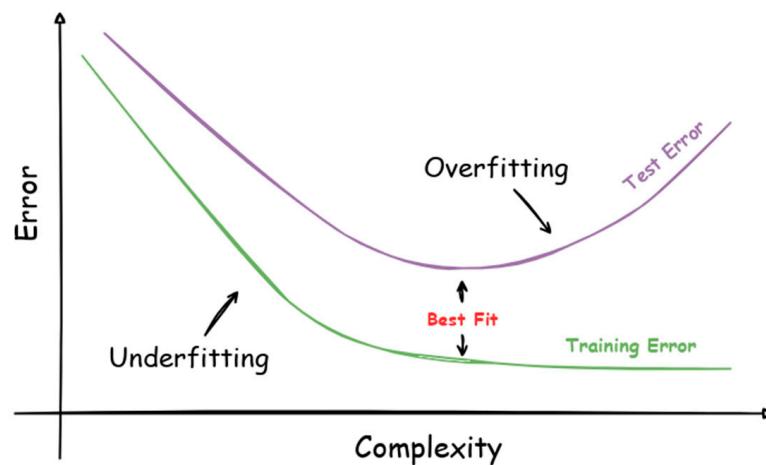


Figure 30. Illustration of the early stopping solution.

4.3.3. Results and Discussion

This section evaluates the performance results achieved by the YOLOv4-tiny [59], SSD-MobileNetV2, and SSD-EfficientDet-D0 models. The pre-trained versions of the SSD-MobileNet-V2 and SSD-EfficientDet-D0 models were obtained from the COCO 2017 dataset [67], which was made available by the Tensorflow 2.0 library's zoo model [68]. Subsequently, these models were trained with the dataset created in this work, using transfer learning [69]. Version B of the dataset was chosen due to the occurrence of false positives (FPs) in some of the trained models. For example, oblique lanes were incorrectly identified as pedestrian crossings due to their visual similarities. An example of an FP is shown in Figure 31.

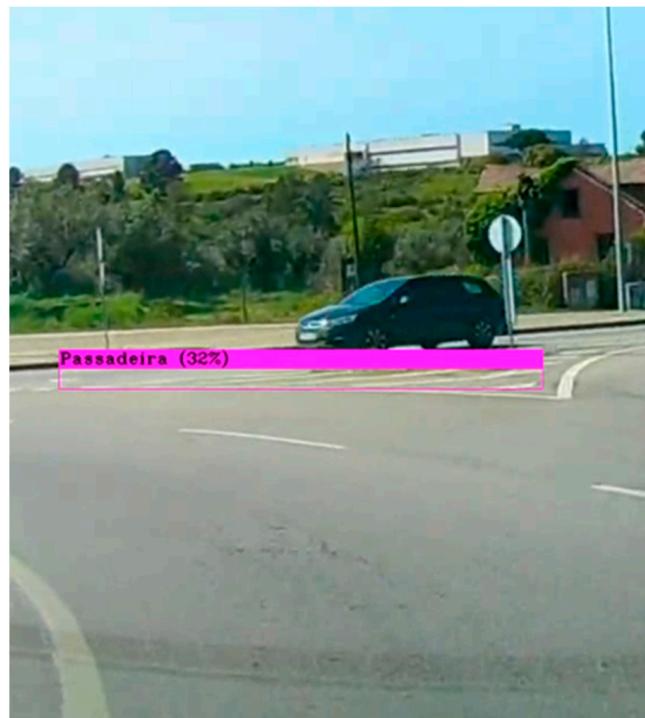


Figure 31. Example of a detected false positive.

Table 5 shows the preliminary results obtained after training the models with their predefined configuration.

Table 5. A comparison of models that were trained using predefined epochs.

Model	Input Size	Batch_Size	mAP (%)	Epochs
YOLOv4-tiny	608 × 608	32	77.3	51
SSD-MobileNet-V2	320 × 320	16	31.24	640
SSD-EfficientDet-D0	512 × 512	4	26.36	640

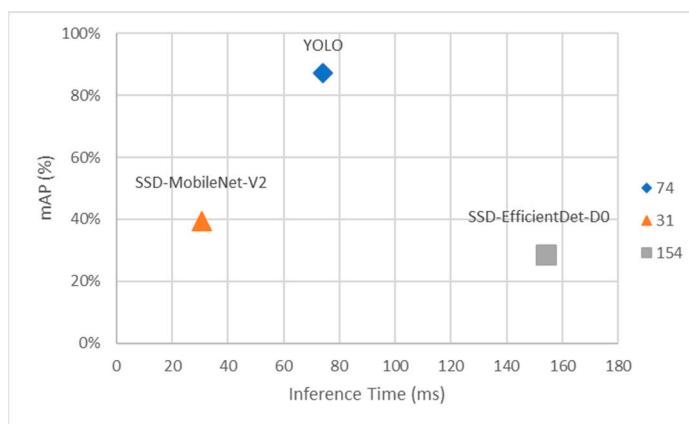
The experimental results showed that the YOLOv4-tiny model obtained a higher mAP than the other models, suggesting better performance in the pedestrian crossing detection task. It is worth noting, however, that the number of epochs used during training can have an impact on the results. The SSD-MobileNet-V2 and SSD-EfficientDet-D0 models may benefit from longer training sessions to improve their performance.

To explore the impact of additional epochs on the mAP value, further training sessions were conducted. The new results are outlined in Table 6.

Table 6. Comparison of models trained with a higher number of epochs.

Model	Input Size	Batch_Size	mAP (%)	Epochs
YOLOv4-tiny	608 × 608	32	87	153
SSD-MobileNet-V2	320 × 320	16	39.29	1024
SSD-EfficientDet-D0	512 × 512	4	28.35	736

Additionally, the inference time was calculated to assess the duration required for the model to detect pedestrian crossings in new images. The inference times of the trained models are compared in Figure 32. It is worth noting that the SSD-EfficientDet-D0 model achieved an inference time of 154 ms, which may be considered high and could potentially affect the detection of pedestrian crossings on devices with limited resources. On the other hand, the SSD-MobileNet-V2 model had a faster inference time than the SSD-EfficientDet-D0 model by 41 ms. The backbone architecture of the model was lightweight, which resulted in less processing time. However, it is worth noting that the mAP value was lower compared to other models.

**Figure 32.** Comparison of inference time.

Among the models tested, the YOLOv4-tiny model achieved the highest mAP. Considering its inference time and its ability to detect pedestrian crossings in video format, it can be concluded that this model was the most suitable for the context of this work.

Figure 33 displays examples of pedestrian crossing detection using the previously discussed models. Figure 33a displays the results obtained by the SSD-MobileNet-V2 model. Figure 33b shows examples of the SSD-Efficient-D0 model, and Figure 33c displays the use of the YOLOv4-tiny model. These models could detect multiple pedestrian crossings per image, as well as identify obstacles that may partially obstruct them.

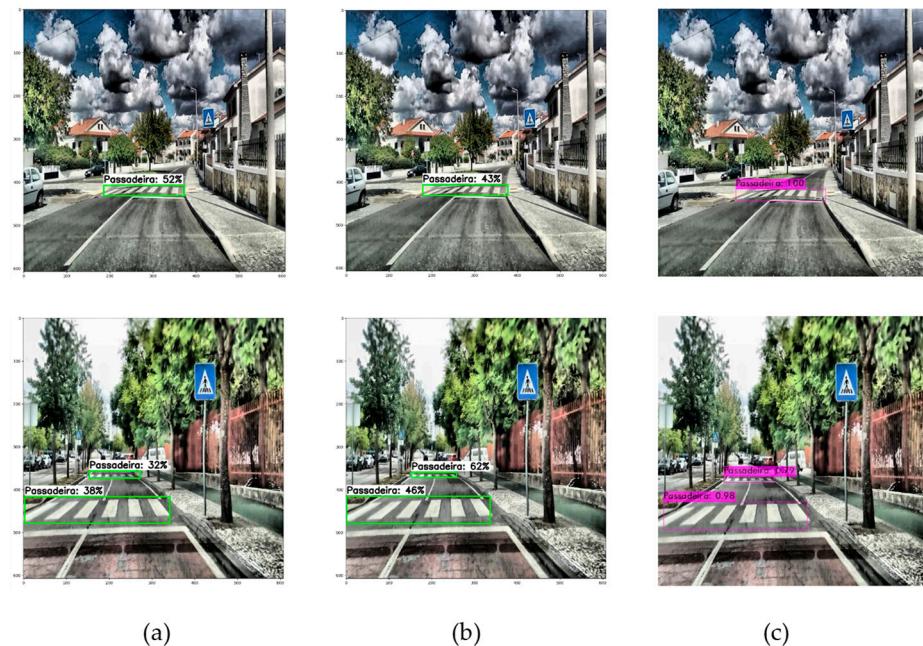


Figure 33. Pedestrian crossing detection examples, (a) SSD-MobileNet-V2; (b) SSD-Efficient-D0; (c) YOLOv4-tiny.

5. Discussion

The following section discusses the limitations that persist despite the various proposed solutions. As demonstrated in the literature, real-time detection remains a challenge due to obstacles related to processing time and/or computational cost, which often renders them incompatible with the intended solutions.

Pedestrian crossing detection in urban scenarios presents several challenges. These include occlusions caused by vehicles, pedestrians, or other objects partially or completely obstructing the crossing. Additionally, scale and rotation variations result from the movement of these objects along the camera's perspective. Illumination changes caused by environmental factors such as shadows or reflections are also a challenge. Finally, there are variations in the types of markings and colors used to delimit these crossing zones, which differ between countries.

The limitations of these systems can sometimes be attributed to the computational capacity of detection devices. It is important to ensure that sufficient processing resources are available to guarantee their proper functioning. Furthermore, the quality of the image used to detect defects in road surfaces can pose a significant challenge, as well as the capture of images without the consent of other road users. In this context, the use of high-definition cameras positioned externally on vehicles or in the vehicle cabin (as shown in Figure 17) could potentially contribute to the effectiveness of the proposed solution by capturing only road surfaces.

This paper presents the initial phase of an ongoing research project. The project aims to develop an autonomous system based on computer vision to detect, classify, and georeference defects in road surfaces, particularly the wear and tear on pedestrian crossings, in real time. The solution also includes notifying the responsible entities of any necessary maintenance. The research conclusions of this paper are of great significance as they have identified the computer vision methodologies with the highest potential for the proposed solution [38,47,49,54,56,58]. These methodologies will be subject to testing and performance evaluations in the next project development phase.

6. Conclusions

Since pedestrians are the most vulnerable road users, safe pedestrian crossings are essential. However, over time they are subjected to wear and tear due to weather conditions,

heavy traffic, and other factors, which may affect their visibility and thus compromise the pedestrian's safety. Therefore, it is important to explore technological solutions that can help the entities responsible for their conservation and maintenance.

This paper demonstrated that computer vision approaches can play a major role in improving road safety and urban infrastructure maintenance. Managers responsible for road safety, in the context of a smart city, can use them to automate the process of determining the wear and tear of pedestrian crossings. In summary, the main contributions resulting from this article are: (1) a survey on various computer vision approaches for detecting pedestrian crossings; (2) the creation and sharing of a new dataset with pedestrian crossings; and (3) a performance assessment study of various image processing techniques and CNN models for detecting pedestrian crossings.

The conducted performance evaluation concluded that the YOLOv4-tiny CNN model has good potential. This model demonstrated an mAP of 87% and the capability to detect data in video format. Even though the other evaluated models registered lower mAP results, it may be premature to dismiss them at this stage as they still hold potential. It is worth noting that these results were obtained using the Google Colab and Kaggle platforms, which have limitations when it comes to training the models.

It should be noted that the work presented in this paper is the first step of an ongoing research project that aims to develop a computer vision system approach to detect and classify the wear and tear of pedestrian crossings. The objective is to develop a prototype to be installed in vehicles, and to test and demonstrate it in real-world scenarios. Therefore, as for future work, it is fundamental to carry out tests to assess which of these techniques can be used to classify and categorize wear and tear. Furthermore, it may be necessary to test other computer vision techniques.

Author Contributions: Conceptualization, G.J.M.R. and J.M.S.A.; methodology, G.J.M.R. and J.M.S.A.; validation, P.D.G., J.M.L.P.C. and V.N.G.J.S.; formal analysis, P.D.G., J.M.L.P.C. and V.N.G.J.S.; investigation, G.J.M.R. and J.M.S.A.; writing—original draft preparation, G.J.M.R. and J.M.S.A.; writing—review and editing, P.D.G., J.M.L.P.C. and V.N.G.J.S.; supervision, J.M.L.P.C. and V.N.G.J.S.; funding acquisition, J.M.L.P.C. and V.N.G.J.S. All authors have read and agreed to the published version of the manuscript.

Funding: J.M.L.P.C. and V.N.G.J.S. acknowledge that this work is funded by FCT/MCTES through national funds and, when applicable, co-funded EU funds under the project UIDB/50008/2020. P.D.G. appreciates the support provided by the Center for Mechanical and Aero-space Science and Technologies (C-MAST) under the project UIDB/00151/2020.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Seco, Á.; Macedo, J.; Costa, A. *Manual de Planeamento das Acessibilidades e da Gestão Viária*; CCDRN: Coimbra, Portugal, 2008.
2. Segurança Rodoviária, A.N. Relatório Anual 2020; ARSPE: Barcarena, Brazil, 2020.
3. Patella, S.M.; Sportiello, S.; Carrese, S.; Bella, F.; Asdrubali, F. The Effect of a LED Lighting Crosswalk on Pedestrian Safety: Some Experimental Results. *Safety* **2020**, *6*, 20. [[CrossRef](#)]
4. Yin, C.; Xiong, Z.; Chen, H.; Wang, J.; Cooper, D.; David, B. A Literature Survey on Smart Cities. *Sci. China Inf. Sci.* **2015**, *58*, 100102. [[CrossRef](#)]
5. Conservação | Infraestruturas de Portugal. Available online: <https://www.infraestruturasdeportugal.pt/pt-pt/conservacao> (accessed on 4 September 2023).
6. What Is Computer Vision? IBM. Available online: <https://www.ibm.com/topics/computer-vision> (accessed on 8 December 2023).
7. What Is Deep Learning?-Deep Learning Explained-AWS. Available online: <https://aws.amazon.com/what-is/deep-learning/> (accessed on 12 January 2024).
8. Top 5 Applications of Convolution Neural Network. Available online: <https://indiaai.gov.in/article/top-5-applications-of-convolution-neural-network> (accessed on 21 December 2023).

9. Mayank Mishra Convolutional Neural Networks, Explained. Available online: <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939> (accessed on 20 December 2023).
10. MK Gurucharan Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network. Available online: <https://www.upgrad.com/blog/basic-cnn-architecture/> (accessed on 20 December 2023).
11. Wang, J.; Turko, R.; Shaikh, O.; Park, H.; Das, N.; Hohman, F.; Kahng, M.; Chau, P. CNN Explainer. Available online: <https://poloclub.github.io/cnn-explainer/#article-relu> (accessed on 21 December 2023).
12. Lohia, A.; Dhananjay Kadam, K.; Raghvendra Joshi, R.; Bongale, A.M.; Dhananjay, K.; Raghvendra, R. Bibliometric Analysis of One-Stage and Two-Stage Object Detection. *Libr. Philos. Pract.* **2021**, *49*, 10, 34.
13. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
14. Cortes, C.; Research, G.; York, N.; Mohri, M.; Rostamizadeh, A. L2 Regularization for Learning Kernels. *arXiv* **2009**, arXiv:1205.2653.
15. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9905, pp. 21–37.
16. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
17. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
18. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
19. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
20. Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 10691–10700.
21. Carranza-García, M.; Torres-Mateo, J.; Lara-Benítez, P.; García-Gutiérrez, J. On the Performance of One-Stage and Two-Stage Object Detectors in Autonomous Vehicles Using Camera Data. *Remote Sens.* **2020**, *13*, 89. [CrossRef]
22. Zhao, Z.-Q.; Zheng, P.; Xu, S.-T.; Wu, X. Object Detection with Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
23. Carrio, A.; Sampedro, C.; Rodriguez-Ramos, A.; Campoy, P. A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles. *J. Sens.* **2017**, *2017*, 3296874. [CrossRef]
24. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 386–397. [CrossRef]
25. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
26. Computer Vision: Instance Segmentation with Mask R-CNN | by Renu Khandelwal | Towards Data Science. Available online: <https://towardsdatascience.com/computer-vision-instance-segmentation-with-mask-r-cnn-7983502fcad1> (accessed on 3 January 2024).
27. HOG (Histogram of Oriented Gradients): An Overview | by Mrinal Tyagi | Towards Data Science. Available online: <https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f> (accessed on 5 September 2023).
28. Alhindi, T.J.; Kalra, S.; Ng, K.H.; Afrin, A.; Tizhoosh, H.R. Comparing LBP, HOG and Deep Features for Classification of Histopathology Images. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–7.
29. Nemutlu, D. HOG Feature Descriptor with Python and OpenCV. Available online: <https://github.com/dahinemutlu/hog-feature-descriptor> (accessed on 6 September 2023).
30. Nemutlu, D. HOG Feature Descriptor. Available online: <https://medium.com/@dnemutlu/hog-feature-descriptor-263313c3b40d> (accessed on 6 September 2023).
31. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
32. Kaspers, A. *Blob Detection*; Image Sciences Institute: Utrecht, The Netherlands, 2011.
33. OpenCV-Open Computer Vision Library. Available online: <https://opencv.org/> (accessed on 7 November 2023).
34. Mammeri, A.; Boukerche, A.; Tang, Z. A Real-Time Lane Marking Localization, Tracking and Communication System. *Comput. Commun.* **2016**, *73*, 132–143. [CrossRef]
35. Jia, W.; Zhang, H.; He, X. Region-Based License Plate Detection. *J. Netw. Comput. Appl.* **2007**, *30*, 1324–1333. [CrossRef]
36. Venkatesh, M.; Vijayakumar, P. A Simple Bird's Eye View Transformation Technique. *Int. J. Sci. Eng. Res.* **2012**, *3*, 5.
37. Jayasinghe, O.; Hemachandra, S.; Anhettigama, D.; Kariyawasam, S.; Rodrigo, R.; Jayasekara, P. CeyMo: See More on Roads—A Novel Benchmark Dataset for Road Marking Detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2022.
38. Muthalagu, R.; Bolimera, A.; Kalaichelvi, V. Lane Detection Technique Based on Perspective Transformation and Histogram Analysis for Self-Driving Cars. *Comput. Electr. Eng.* **2020**, *85*, 106653. [CrossRef]
39. Toth, T.; Bauer, D.; Sukosd, F.; Horvath, P. Fisheye Transformation Enhances Deep-Learning-Based Single-Cell Phenotyping by Including Cellular Microenvironment. *Cell Rep. Methods* **2022**, *2*, 100339. [CrossRef] [PubMed]
40. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *6*, 679–698. [CrossRef]

41. Jose, A.; Deepa Merlin Dixon, K.; Joseph, N.; George, E.S.; Anjitha, V. Performance Study of Edge Detection Operators. In Proceedings of the 2014 International Conference on Embedded Systems (ICES), Coimbatore, India, 11 July 2014; pp. 7–11.
42. Cortes, C.; Vapnik, V. Support-Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
43. Support Vector Machine—Introduction to Machine Learning Algorithms | by Rohith Gandhi | Towards Data Science. Available online: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47> (accessed on 10 November 2023).
44. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 1 June 2018; pp. 4510–4520.
45. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 1 July 2017; pp. 1800–1807.
46. Zebin, T.; Scully, P.J.; Peek, N.; Casson, A.J.; Ozanyan, K.B. Design and Implementation of a Convolutional Neural Network on an Edge Computing Smartphone for Human Activity Recognition. *IEEE Access* **2019**, *7*, 133509–133520. [CrossRef]
47. Zhang, Z.-D.; Tan, M.-L.; Lan, Z.-C.; Liu, H.-C.; Pei, L.; Yu, W.-X. CDNet: A Real-Time and Robust Crosswalk Detection Network on Jetson Nano Based on YOLOv5. *Neural. Comput. Appl.* **2022**, *34*, 10719–10730. [CrossRef]
48. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef] [PubMed]
49. Lee, S.; Kim, J.; Yoon, J.S.; Shin, S.; Bailo, O.; Kim, N.; Lee, T.-H.; Hong, H.S.; Han, S.-H.; Kweon, I.S. VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), 22 October 2017; pp. 1965–1973.
50. Kawano, M.; Mikami, K.; Yokoyama, S.; Yonezawa, T.; Nakazawa, J. Road Marking Blur Detection with Drive Recorder. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; pp. 4092–4097.
51. Malbog, M.A. MASK R-CNN for Pedestrian Crosswalk Detection and Instance Segmentation. In Proceedings of the 2019 IEEE 6th International Conference on Engineering Technologies and Applied Sciences (ICETAS), Kuala Lumpur, Malaysia, 20–21 December 2019; pp. 1–5.
52. Kawasaki, T.; Iwamoto, T.; Matsumoto, M.; Yonezawa, T.; Nakazawa, J.; Takashio, K.; Tokuda, H. A Method for Detecting Damage of Traffic Marks by Half Celestial Camera Attached to Cars. In Proceedings of the 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, Coimbra, Portugal, 22–24 July 2015.
53. Chen, C.; Chandra, S.; Han, Y.; Seo, H. Deep Learning-Based Thermal Image Analysis for Pavement Defect Detection and Classification Considering Complex Pavement Conditions. *Remote Sens.* **2021**, *14*, 106. [CrossRef]
54. Fang, N.; Zhang, Z.; Xia, B.; Yao, Z. Polite Zebra Crossing Driver Reminding System Design. In Proceedings of the 2021 International Conference on Bioinformatics and Intelligent Computing, Harbin, China, 22–24 January 2021; ACM: New York, NY, USA; pp. 390–394.
55. Greenhalgh, J.; Mirmehdi, M. Detection and Recognition of Painted Road Surface Markings. In Proceedings of the International Conference on Pattern Recognition Applications and Methods, Lisbon, Portugal, 10–12 January; 2015; Volume 1, pp. 130–138.
56. Kaya, Ö.; Çodur, M.Y.; Mustafaraj, E. Automatic Detection of Pedestrian Crosswalk with Faster R-CNN and YOLOv7. *Buildings* **2023**, *13*, 1070. [CrossRef]
57. Zhong, J.; Feng, W.; Lei, Q.; Le, S.; Wei, X.; Wang, Y.; Wang, W. Improved U-Net for Zebra-Crossing Image Segmentation. In Proceedings of the 2020 IEEE 6th International Conference on Computer and Communications (ICCC), Chengdu, China, 11–14 December 2020; pp. 388–393.
58. Meem, M.I.; Dhar, P.K.; Khaliluzzaman, M.; Shimamura, T. Zebra-Crossing Detection and Recognition Based on Flood Fill Operation and Uniform Local Binary Pattern. In Proceedings of the 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox'sBazar, Bangladesh, 7–9 February 2019; pp. 1–6.
59. GitHub-AlexeyAB/Darknet: YOLOv4/Scaled-YOLOv4/YOLO-Neural Networks for Object Detection (Windows and Linux Version of Darknet). Available online: <https://github.com/AlexeyAB/darknet> (accessed on 23 October 2023).
60. Models/Research/Object_detection/G3doc/Tf2_detection_zoo.Md at Master Tensorflow/Models GitHub. Available online: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md (accessed on 29 December 2023).
61. Roboflow: Give Your Software the Power to See Objects in Images and Video. Available online: <https://roboflow.com/> (accessed on 5 October 2023).
62. Crosswalks Dataset > Overview. Available online: <https://universe.roboflow.com/projeto-5fy5m/crosswalks-zbjgg> (accessed on 5 December 2023).
63. Google Colab. Available online: <https://colab.google/> (accessed on 6 November 2023).
64. Kaggle: Your Machine Learning and Data Science Community. Available online: <https://www.kaggle.com/> (accessed on 6 November 2023).
65. Ying, X. An Overview of Overfitting and Its Solutions. *J. Phys. Conf. Ser.* **2019**, *1168*, 022022. [CrossRef]
66. Pothuganti, S. Review on Over-Fitting and under-Fitting Problems in Machine Learning and Solutions International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering Review on over-Fitting and under-Fitting Problems in Machine Learning and Solutions. *Artic. Int. J. Adv. Res. Electr. Electron. Instrum. Eng.* **2018**, *7*, 3692–3695.
67. COCO-Common Objects in Context. Available online: <https://cocodataset.org/#home> (accessed on 12 January 2024).

68. TensorFlow. Available online: <https://www.tensorflow.org/> (accessed on 12 January 2024).
69. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **2019**, *109*, 43–76. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.