

Article

Not peer-reviewed version

---

# Leveraging Perspective Transformation For Enhanced Pothole Detection in Autonomous Vehicles

---

[Abdalmalek Abu-raddaha](#)\*, [Zaid El-Shair](#), [Samir Rawashdeh](#)

Posted Date: 7 August 2024

doi: 10.20944/preprints202408.0525.v1

Keywords: Autonomous Vehicles; Perspective Transformation; Deeplearning; Pothole Detection; Computer Vision; Mobile Robotics; Object Detection





Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

## Article

# Leveraging Perspective Transformation for Enhanced Pothole Detection in Autonomous Vehicles

Abdalmalek Abu-raddaha \*, Zaid A. El-Shair  and Samir Rawashdeh 

Department of Electrical and Computer Engineering, University of Michigan-Dearborn, Dearborn, MI 48128, USA; abdmalek@umich.edu (A.A.); zelshair@umich.edu (Z.E.); srawa@umich.edu (S.R.)

\* Correspondence: abdmalek@umich.edu

**Abstract:** Road conditions, often degraded by insufficient maintenance or adverse weather, significantly contribute to accidents, exacerbated by the limited human reaction time to sudden hazards like potholes. Early detection of distant potholes is crucial for timely corrective actions, such as reducing speed or avoiding obstacles, to mitigate vehicle damage and accidents. This paper introduces a novel approach that utilizes perspective transformation to enhance pothole detection at different distances, focusing particularly on distant potholes. Perspective transformation improves the visibility and clarity of potholes by virtually bringing them closer and enlarging their features, which is particularly beneficial given the fixed-size input requirement of object detection networks, typically smaller than the raw image resolutions captured by cameras. Our method automatically identifies the region of interest (ROI)—the road area—and calculates the corner points to generate a perspective transformation matrix. This matrix is applied to all images and corresponding bounding box labels, enhancing the representation of potholes in the dataset. This approach significantly boosts detection performance when used with YOLOv5-small, achieving a 45.7% improvement in average precision (AP) at IoU thresholds of 0.5 to 0.95 for a single class, and notable improvements of 30.3%, 78.6%, and 278% for near, medium, and far pothole classes, respectively, after categorizing them based on their distance. This work is the first to employ perspective transformation specifically for enhancing the detection of distant potholes.

**Keywords:** autonomous vehicles; perspective transformation; deeplearning; pothole detection; computer vision; mobile robotics

## 1. Introduction

### 1.1. Pothole Detection Importance for Autonomous Vehicles

Potholes, commonly found in asphalt pavements, are caused by water weakening the underlying soil and repeated traffic wear, leading to depressions or holes in the road surface [1]. These can vary in severity and pose significant hazards, such as suspension damage, tire punctures, and even accidents, by causing loss of control or immobilization of vehicles. The dangers of potholes extend to both vehicles and pedestrians, highlighting the critical need for efficient detection systems.

In 2011, poor road conditions caused around 2,200 deaths in India, while in the U.S., one-third of the 38,824 traffic deaths in 2020 were linked to substandard roads. Michigan, with some of the worst potholes, spent millions annually on repairs, highlighting the widespread impact of this issue. Effective pothole detection is crucial, particularly for autonomous vehicles, which rely on accurate hazard detection to ensure safe operation. This need is underscored by the potential damage and safety risks associated with potholes, emphasizing the importance of accurate and timely detection systems to mitigate these dangers.

### 1.2. Human Response to Potholes

When a driver encounters a pothole, the human reaction time to apply the brakes can vary. For simple tasks, the average human reaction time is often quoted as 0.2 seconds [2]. However, for more complex tasks, such as emergency braking when a pothole is detected, the reaction time is typically longer. These times can be greatly affected by the driver's alertness and the expectation of the need to

brake [3]. Hence, detecting potholes from a greater distance is crucial, as it provides the driver more time to react and navigate safely around the hazard. In addition, far pothole detection allows drivers to prepare and adjust their driving accordingly, reducing the likelihood of sudden maneuvers that can lead to accidents.

In contrast to human drivers, an AV equipped with pothole detection systems can identify and respond to potholes in real time. These systems can modulate vehicle speed and position upon detecting a potential pothole and sometimes ensure the vehicle remains within its lane while minimizing impact. This rapid and precise response can significantly improve road safety by allowing the vehicle to take immediate action to avoid the pothole or minimize impact, thus reducing potential damage and improving overall vehicle control.

### *1.3. Pothole Detection Methods*

The implementation of pothole detection systems in AV offers several advantages. Not only does it improve road safety by reducing the risk of accidents caused by potholes, but it can also contribute to more efficient road maintenance by providing accurate and timely data on the locations and severity of potholes. Moreover, detecting and avoiding potholes can reduce the fuel consumption, wear and tear, and maintenance costs of a vehicle [4]. In addition, it can indirectly decrease the total travel time in some cases [4].

Potholes are detected and observed in different ways, including manual human detection, vibration-based detection, sensor-based detection, and vision-based detection. Human observation is the traditional method for detecting potholes. [5]. Drivers must look out for potholes while driving and react quickly to avoid them, which can lead to dangerous situations. This approach is inconsistent due to human error and is inefficient for detecting multiple objects. In addition, vibration-based detection methods use accelerometers to detect potholes based on the vibration information of the acceleration sensors attached to the vehicle [6]. This method is cost-effective and suitable for real-time processing. However, it has limitations in providing the exact shape of potholes and could provide incorrect results, as road joints can be misidentified as potholes [7]. Moreover, this method is not suitable for detecting potholes in order to avoid or act towards reducing their effect on the vehicle. Sensor-based detection methods, such as light detection and ranging (LiDAR) and radio detection and ranging (RADAR), use electromagnetic waves to detect potholes. LiDAR uses light waves [8], providing high-resolution data and precision [9]. However, it can be expensive, making it less feasible for widespread use in AVs. RADAR, on the other hand, uses radio waves and is superior in terms of cost and ability to monitor large areas [9]. Nevertheless, its lower resolution compared to LiDAR makes it difficult to track and distinguish objects in crowded environments, a common scenario for AVs [10]. Besides, computer vision techniques for pothole detection have gained popularity due to the accessibility and feasibility of cameras, especially for AVs. These techniques use images or videos as input data and apply deep learning and image processing techniques to detect potholes. Although each method has its strengths and weaknesses, computer vision techniques offer significant advantages in terms of cost effectiveness, precision, and the ability to integrate with other data sources for pothole detection. Hence, our work is purely focused on vision-based detection techniques.

### *1.4. Challenges in Potholes Detection*

Deep learning and vision-based approaches for pothole detection face significant challenges, particularly related to the handling of object detection tasks involving small and distant objects. A critical limitation stems from the requirement to resize images to a fixed, smaller size to ensure reasonable processing times. This resizing is necessary because models trained on high-resolution images demand substantial computational resources, leading to impracticalities in real-time applications. As a result, small objects, such as potholes, especially those far away, often become indistinguishable when images are downscaled. The reduction in size leads to a loss of crucial details, making it challenging for the model to accurately detect and classify these objects. This issue is exacerbated by the fact that more

complex models, while potentially offering increased accuracy, do not necessarily resolve the problem of lost detail due to image downscaling.

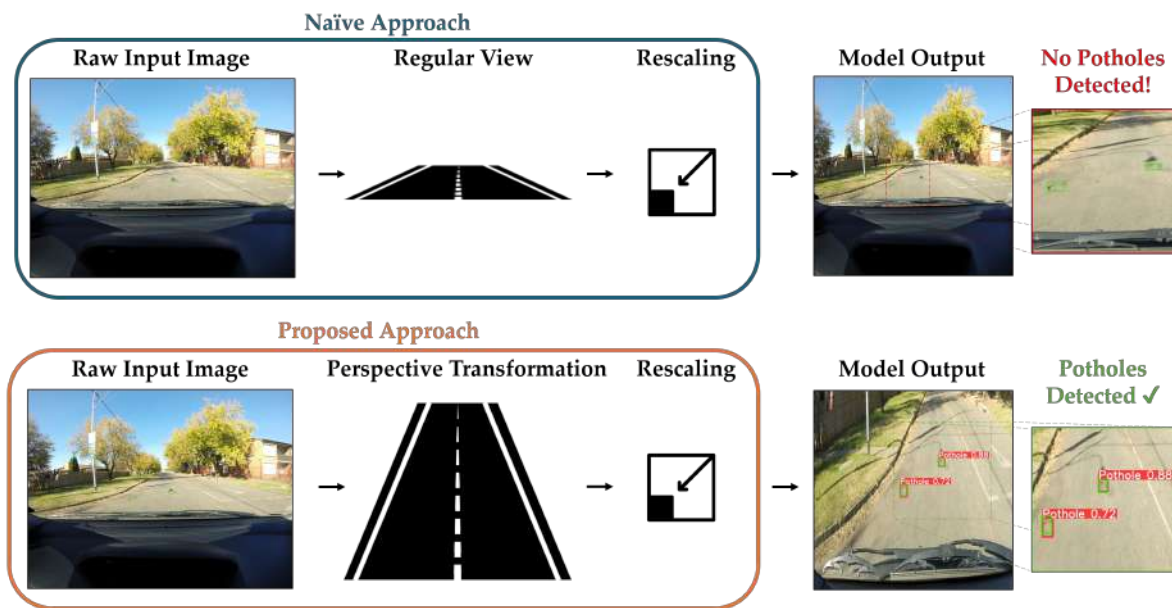
Moreover, the utilization of high-resolution images in training object detection models is hindered by the immense complexity of the search space. High-dimensional data requires more extensive computational resources and can significantly slow down the training and inference processes. This trade-off between image resolution and processing efficiency is particularly problematic in real-time applications such as autonomous driving, where rapid detection and response are crucial for safety. The need to rescale high-definition images to lower resolutions introduces a bottleneck in object detection systems. The act of rescaling can lead to a substantial loss of fine-grained features that are essential for accurately identifying potholes, thereby compromising the model's performance. One proposed solution to mitigate these challenges is the use of perspective transformation. Unlike conventional resizing, perspective transformation selectively focuses on a region of interest (ROI) within an image, such as the area containing a pothole. This approach preserves critical features by altering the viewing angle, effectively enlarging the ROI and reducing the prominence of irrelevant areas. While this method does not introduce new features, it helps retain more of the significant details associated with the potholes, thereby improving detection accuracy. Although theoretically, training a model on full-resolution images would be ideal, it is practically unfeasible due to computational constraints. Hence, perspective transformation offers a practical compromise, allowing the retention of essential features while maintaining manageable processing times, thus enhancing the robustness and effectiveness of pothole detection in autonomous vehicle systems.

#### *1.5. Proposed Method: Distant Pothole Detection with Vision and Perspective Transformation*

To address these challenges, we present a novel approach that leverages perspective transformation to enhance pothole features. Perspective transformation is an image processing technique that alters the viewing angle of potholes from the perspective of the vehicle or driver to a bird's-eye view [11]. This transformation is based on four source points extracted from the values of the bounding box and the location of the pothole pixels. These points represent the ROI to be transformed. Our method proposes an automated selection of these four points by considering all bounding box values in the dataset, thereby reducing human intervention and creating an ROI that is optimal for all images and their bounding boxes.

The perspective transformation modifies the image, enlarging the ROI and minimizing the scale of irrelevant portions. This technique effectively simulates bringing the potholes closer, thereby improving feature extraction. By applying perspective transformation, we generate a magnified and standardized view of the pothole region within the image. This enhanced view enables the computer vision algorithm to more effectively extract crucial features such as cracks, edges, and depth variations, significantly improving the accuracy of pothole detection, particularly for those located at a distance. Figure 1 shows the framework overview of our approach compared to the naive approach. To the best of our knowledge, no previous study has explored the use of perspective transformation in this manner for such an application.





**Figure 1.** Comparison between the naive approach and our proposed approach. The naive approach involves loading the raw input image, and then simply downscaling it to the required input resolution for the object detection network losing significant image features and resulting in undetected potholes. Meanwhile, our approach demonstrates successful and robust pothole detection by transforming the input image to primarily retain the region of interest and minimize irrelevant segments of the image. Ground truth pothole labels and predicted potholes are represented by green and red bounding boxes, respectively. Street and rescaling icons created by Trevor Dsouza and Doodle Icons via TheNounProject.com.

## 2. Related Work

Numerous studies have investigated various computer vision and image processing techniques for automated pothole detection using visual road imagery. Additionally, other research has highlighted the significance of utilizing perspective transformation in object detection applications. The following sections review some of the aforementioned approaches.

### 2.1. Vision Approaches

Basic vision approaches in pothole detection have leveraged simple image-processing techniques to identify road defects. For example, Nienaber et al. [12] utilized road color modeling combined with edge detection, while Pereira et al. [13] employed a basic 4-layer CNN. These methods provide cost-effective solutions without the need for complex sensors or deep learning architectures. However, they suffer from limitations in generalizability across diverse road conditions. Furthermore, the precision needed for real-time application is lacking, and the absence of comprehensive visual analyses and separate test sets undermines the reliability of their results, which might be affected by the lack of a pothole dataset from a vehicle perspective.

Deep learning approaches have shown potential for more robust pothole detection. Studies such as Chen et al. [14], Kumar [15], and Dhiman et al. [16] have employed various deep neural networks, including location-aware CNNs, Faster R-CNN, Mask R-CNN, and YOLOv2. Although these models demonstrate high classification accuracy, they are computationally expensive and struggle with precise localization. Additionally, the two-stage nature of some pipelines, like those in Chen et al., can introduce further computational overhead. The performance of these models is also heavily dependent on high-quality input data, and challenges in real-time application due to processing latency remain a significant drawback.

Stereo vision approaches, explored by Dhiman et al. [16], [17], utilize depth information to identify road defects by analyzing road elevation and depth variations. Although these methods

theoretically provide detailed spatial information, they are highly dependent on the quality of stereo images and the precise camera calibration. Issues such as noise, distortion, and the need for well-aligned image pairs can significantly affect the accuracy of depth estimation and pothole detection. The computational intensity further hinders the practical deployment of these techniques in real-time applications, particularly in the context of autonomous vehicles.

Data augmentation and enhancement techniques have also been explored to improve pothole detection performance. Maeda et al. [18] employed Generative Adversarial Networks (GANs) to synthesize additional training data, combined with a Single Shot multibox Detection (SSD) model. While synthetic data contributed to modest improvements in detection performance, the approach faces challenges such as increased computational complexity, instability in training GANs, and concerns about the generalizability of the models to real-world scenarios. Similarly, Salaudeen et al. [19] used enhanced super-resolution GANs to enhance image quality, feeding the enhanced images into YOLOv5 and EfficientDet [20] models. Despite improved detection metrics, this approach introduces significant computational overhead and risks of overfitting, particularly since the results depend mainly on the quality of the data.

Specific YOLO variants have also been investigated. Al-Shaghouri et al. [21] explored real-time pothole detection using YOLOv3 and YOLOv4, achieving promising precision, but facing limitations related to low IoU thresholds and performance variability across different distances. Buko et al. [22] examined the effectiveness of YOLOv3 and Sparse R-CNN under various challenging conditions, revealing a substantial performance degradation under low light and adverse weather conditions, indicating limited applicability in various real-world scenarios. Nevertheless, this project used the same dataset for training and testing, which affects the generalizability of this approach. Rastogi et al. [23] modified YOLOv2 to address issues such as vanishing gradients and irrelevant feature learning. However, the reliance on close-range smartphone images limits the model's applicability to broader contexts, such as autonomous vehicles where variable distances and angles are encountered.

While previous research in pothole detection has largely concentrated on improving algorithmic architectures or combining multiple techniques to enhance detection accuracy, our contribution addresses a fundamental gap by focusing on the quality and effectiveness of the input dataset. By leveraging perspective transformation, our approach optimizes the dataset, maintains the desired objects' features and enhances them. This method effectively tackles the common issue of limited data without incurring additional computational costs during training. In fact, it often reduces training time compared to using regular or cropped images, making it a practical and efficient solution. Unlike other approaches that are computationally intensive and dependent on high-quality data, our contribution ensures better utilization of the current dataset, enhancing the robustness and real-world applicability of pothole detection systems without the trade-offs associated with complex algorithmic fusions.

## 2.2. Perspective Transformation Technique

In this section, we provide a brief overview of how perspective transformation, a widely used image processing tool, has been used in object detection for various purposes, including enhancing detection accuracy, data augmentation, and others.

In [24], the authors present a method to construct perspective transformations for detecting 3D bounding boxes of vehicles in traffic surveillance, enhancing the accuracy of object detection by extending traditional 2D detectors. Lee et al. [25] introduce a multi-view approach that leverages perspective transformation for pedestrian detection, projecting features onto a ground plane to improve localization accuracy. Additionally, Wang et al. [26] utilize perspective transformation in data augmentation, enhancing object detection by simulating variations in object size and viewpoint. In another study, Hou et al. [27] propose a feature transformation method for multiview aggregation in 3D object detection, focusing on head-foot pair detection. These studies highlight the significance of perspective transformations in addressing challenges related to varying viewpoints in object detection. However, our work goes a step further by directly enhancing object detection quality through an

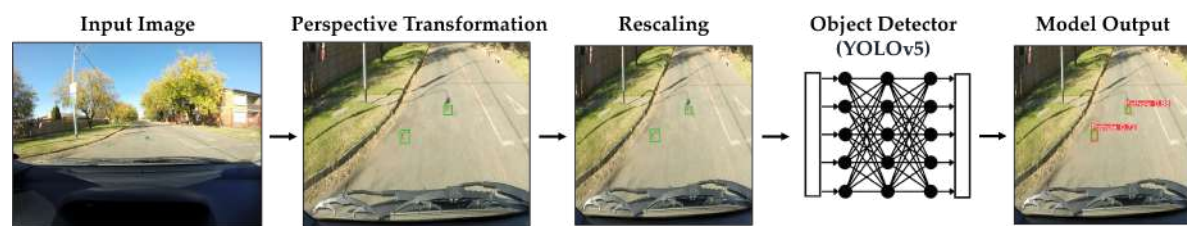
automated application of perspective transformation, improving object feature representation for superior model training and performance.

### 3. Methodology

Potholes significantly impact vehicles and road users, increasing the likelihood of hazardous situations. Therefore, implementing an effective early detection system for potholes is crucial to mitigate potential risks and prevent undesirable or harmful incidents. In this section, we present a detailed breakdown of our proposed approach detailing the techniques employed in this work.

#### 3.1. Framework Overview

When utilizing object detection models, the typical approach is to feed the captured raw images directly once they are rescaled to the desired size. This usually causes a significant loss of image features that are critical for enabling robust pothole detection, especially when using high-resolution input images. Using this naive approach, detecting far potholes, for instance, would be a very challenging task. Instead, our approach, demonstrated in Figure 2, enhances pothole detection from a vehicle's perspective by making potholes appear closer, larger, and with amplified features. Initially, we apply a perspective transformation technique to convert input images from the vehicle's view to a view closer to a bird's-eye view, making them virtually closer and larger. Then, these transformed images are then fed to YOLOv5 [28], a well-known object detector, which uses those images to train and improve the model's ability to detect potholes, especially those at a distance, compared to using regular images.

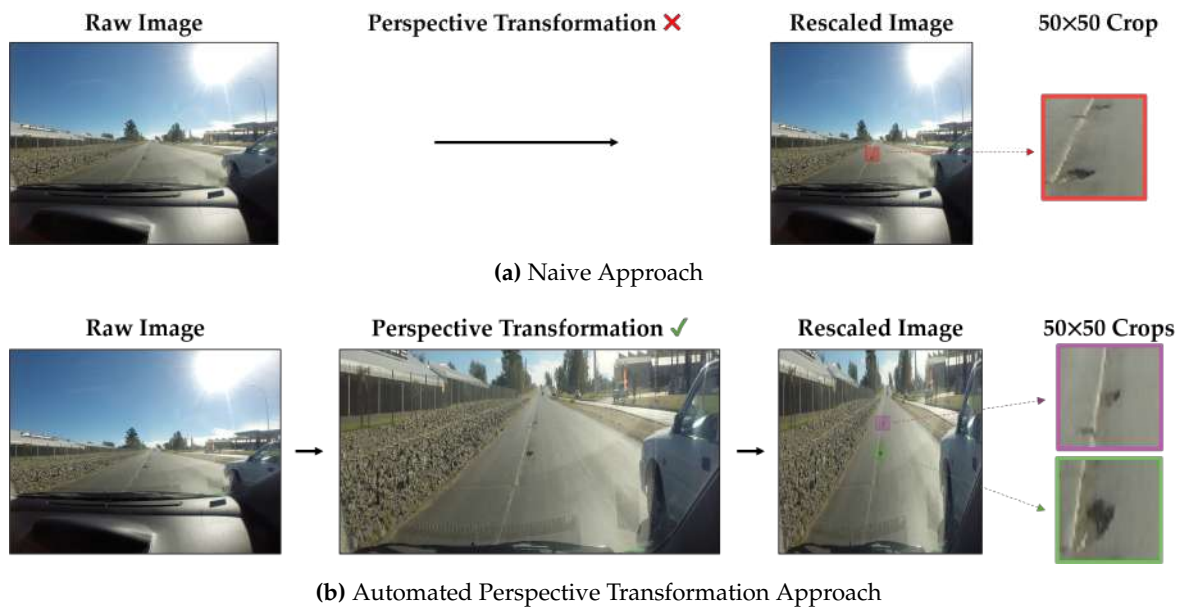


**Figure 2.** Overview of the proposed framework. Raw input images are initially transformed using the transformation matrix generated by our proposed automated algorithm. Then, the resulting images are rescaled to the required input resolution and fed to the object detection network (e.g., YOLOv5). Ground truth pothole labels and predicted potholes are represented by the green and red bounding boxes, respectively. Neural network icon by Lucas Rathgeb via TheNounProject.com.

##### 3.1.1. Perspective Transformation Motivation

Raw images captured by cameras are typically of a high resolution, providing a great level of detail. However, due to design limitations and the need for real-time processing, most object detection networks are trained on fixed, low-resolution images [28–30]. For example, optimized real-time variants of SSD [29] and YOLOv3 [30] are designed for  $300 \times 300$  and  $320 \times 320$  resolutions, respectively. Meanwhile, the more recent YOLOv5 [28] has been tailored for an input spatial resolution of  $640 \times 640$ . Although the trend demonstrates an increase in the models' input resolutions, they are still quite small relative to raw camera output resolutions. Further, training these models on larger input resolution negatively impacts inference times [28,29], which is not ideal for real-time applications. This necessitates resizing the images to a predetermined, smaller size for effective training and generalization. However, when these images are resized, the features of small objects, such as potholes, can become significantly less discernible. Potholes may appear very small relative to the overall image size, resulting in insufficient features for the model to detect and differentiate them effectively, as shown in Figure 3. Furthermore, portions of the image that do not contain regions of interest, such as sidewalks or the sky, are often retained, limiting the focus on the relevant areas necessary for pothole detection. To improve detection accuracy, it is beneficial to adjust the image perspective to emphasize the road—the primary ROI—while minimizing the inclusion of non-essential parts of the image. This

approach ensures that more useful features are preserved after rescaling, enhancing the model's ability to detect potholes without significantly increasing computational overhead [31,32].



**Figure 3.** Comparison of the resulting preprocessed input images between (a) the naive approach and (b) the automated perspective transformation approach. The naive approach involves reading the image as-is and then downscaling to a fixed input resolution ( $800 \times 800$  in this example). A  $50 \times 50$  image crop demonstrates very low resolution for the potholes in the scene. Instead, our proposed approach transforms the image to mainly focus on the ROI (i.e., the street) where, after rescaling to the same input resolution, the resulting spatial resolutions of the potholes are much larger with clearer image features as depicted by the  $50 \times 50$  image crops.

### 3.1.2. YOLOv5: Key Features and Functionality

In the field of computer vision, deep learning has become the preferred approach for object detection, particularly for complex and variable tasks like pothole detection. Traditional handcrafted image processing methods often struggle with the variability in pothole shapes and sizes, and the diverse lighting conditions found in real-world environments. Deep learning-based detectors, such as the YOLO (You Only Look Once) series, provide a more robust and dynamic solution that can generalize well across different scenarios. Among these, YOLOv5 stands out for its balance of speed and accuracy, making it ideal for applications requiring real-time performance. This study employs YOLOv5, implemented in PyTorch [33], for its efficiency, speed, and relatively low computational demands, with versions available in n, s, m, l, and x configurations [28].

The architecture of YOLOv5 follows a similar structure to previous YOLO versions, with a backbone network and detection heads. The backbone, based on the efficient CSPDarknet53 architecture, extracts features from the input image using 53 convolutional layers. The detection heads then predict bounding boxes and class probabilities. YOLOv5 employs a modified YOLO head, consisting of convolutional layers that vary depending on the model configuration (e.g., YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x). Unlike earlier methods, YOLOv5 performs object detection in a single forward pass, ensuring high speed and efficiency that are crucial for real-time applications like surveillance and autonomous driving. With different variants representing varying model scales and complexities, YOLOv5 offers options like YOLOv5s, which has 7.2M parameters and 16.5 GFLOPs, and YOLOv5l, with 46.5M parameters and 109.1 GFLOPs. Notably, YOLOv5s performs better when applied to our approach compared to larger models with a naive approach, highlighting that a more complicated model is not always necessary for good results. We focus on using YOLOv5s for its balanced real-time



performance and accuracy, while also evaluating other YOLOv5 variants to assess the impact of our methods on their performance

YOLOv5 offers several key advantages, including enhanced speed and accuracy due to its lightweight architecture and efficient training through focus modules and data augmentation. Its flexibility and usability are boosted by its user-friendly and customizable PyTorch framework, along with a modular design that allows easy customization for various hardware platforms. Additional benefits include improved anchor box prediction for more accurate bounding box localization and strong community support for collaboration, quick bug fixes, and feature development. Although newer versions such as YOLOv7 [34] and YOLOv8 [35] offer advancements, the speed, efficiency, and strong community support of YOLOv5 make it the preferred choice for this study. While this work is applicable to other detection networks, we chose YOLOv5 for these reasons.

### 3.2. Automated Perspective Transformation Algorithm

The perspective transformation is a fundamental tool in computer vision that allows for the alteration of an image's viewpoint, simulating a change in the observer's position. This is done using a 3x3 transformation matrix that maps points from the original image plane to a new plane. The matrix is calculated by identifying four points in the original image and their corresponding locations in the transformed image. By solving a set of linear equations, the matrix adjusts the coordinates of these points, allowing the image to appear as though it is viewed from a different angle. This method preserves the straightness of lines and their intersections, making it useful for applications like image rectification [36], object tracking [37], and traffic surveillance [38].

For pothole detection, this technique concentrates on the region of interest (ROI), specifically the street, while reducing the prominence of irrelevant areas such as the roadside and sky. By employing perspective transformation, objects that are distant are virtually brought closer, thereby making potholes appear significantly larger in the processed image. This enhancement is accomplished by detecting and matching features, estimating homography, and warping the image using a  $3 \times 3$  homography matrix  $M$ . This process emphasizes relevant features, thereby enhancing detection accuracy, as follows:

$$t' = M \cdot t, \quad (1)$$

where  $t'$  and  $t$  are the coordinates of the ROI in the transformed and source images, respectively. The homography matrix  $M$  is calculated using the corresponding points as explained earlier, aiding in feature extraction and pothole bounding-box regression by mimicking a bird's eye view for more robust pothole detection in AVs.

Applying perspective transformation to a set of images requires manually selecting the boundary points of the ROI. In our application, this ROI would primarily be the road as viewed from the perspective of the vehicle itself. However, the boundaries of the road change significantly from one scene to another depending on many factors, such as the type of road, curvature, number of lanes, etc. Therefore, an ideal application of this technique would be to select the corners of the road for each given image. However, this approach is not feasible for object detection applications due to its time-consuming and error-prone nature, especially when dealing with different scenes that include not only straight streets but street curvatures, u-turns, etc. Manually specifying four points for each image based on the shape of the road is labor-intensive and introduces significant variability and inaccuracies, making it unsuitable for large-scale datasets. Furthermore, due to its unfeasibility, a new transformation matrix is required to be generated for each image, which would add run-time overhead. Therefore, automating this process is essential to ensure a usable workflow and to generalize the technique to most, if not all, datasets with similar structures. To address this challenge, we propose an algorithm that automatically finds a set of ROI corner points that enables the generation of the perspective transformation matrix  $M$ . Using this matrix, a set of images from the same source can be

similarly transformed to better represent the ROI. Consequently, this approach ensures consistency and precise ROI selection in any given dataset of the same source.

To achieve an optimal transformation, it is crucial to accurately identify the best ROI that encompasses all bounding boxes of potholes within an image, ensuring that no potholes are excluded. The automatic transformation process involves determining the ROI by calculating the coordinates of all bounding boxes to establish four defining points. Specifically, the boundary coordinates of the ROI are determined as follows: the top-left point is defined by the minimum  $x$  and  $y$  coordinates among all bounding boxes, the top-right point by the maximum  $x$  and minimum  $y$  coordinates, the bottom-left point by the minimum  $x$  and maximum  $y$  coordinates, and the bottom-right point by the maximum  $x$  and  $y$  coordinates. An offset value ( $\alpha$ ) is added to each of these points to ensure that the ROI extends slightly beyond the boundaries of the bounding boxes. This offset allows for full coverage of the image or ROI boundaries, depending on how much extension the user desires. This approach guarantees that the ROI fully covers the outermost boundaries of the bounding boxes, thereby ensuring comprehensive inclusion of all potholes.

To implement automatic perspective transformation, it is expected that we have a labeled object detection dataset. The algorithm could then be applied to the selected images and their corresponding labels, following the subsequent steps outlined in the algorithm structure below Algorithm 1. The transformed images and label files were then saved to their designated output directories.

---

#### Algorithm 1 Automatic Perspective Transformation for Images and Bounding Boxes

---

**Input:** Images, Ground Truth Labels, ROI Offset  $\alpha$

**Output:** Transformed images and bounding boxes, Perspective transformation matrix  $M$

---

##### Initialize lists

- 1: Initialize lists:  $all\_x\_min, all\_y\_min, all\_x\_max, all\_y\_max, all\_w, all\_h$ 
  - ▷  $x\_min, y\_min$ : top-left corner coordinates
  - ▷  $w, h$ : width and height of bounding boxes
  - ▷  $x\_max, y\_max$ : bottom-right corner coordinates

##### Read bounding boxes

- 2: **for** each *image* and *labels* **do**
- 3:   Read labeled bounding boxes ( $x\_min, y\_min, w, h$ )
- 4:    $(x\_max, y\_max) \leftarrow (x\_min + w, y\_min + h)$
- 5:   Append to respective lists
- 6: **end for**

##### Calculate ROI offsets

- 7:  $(x\_offset, y\_offset) \leftarrow (\alpha \times \max(all\_w), \alpha \times \max(all\_h))$

##### Determine ROI corners

- 8:  $top\_left \leftarrow (\min(all\_x\_min) - x\_offset, \min(all\_y\_min) - y\_offset)$
- 9:  $top\_right \leftarrow (\max(all\_x\_max) + x\_offset, \min(all\_y\_min) - y\_offset)$
- 10:  $bottom\_left \leftarrow (\min(all\_x\_min) - x\_offset, \max(all\_y\_max) + y\_offset)$
- 11:  $bottom\_right \leftarrow (\max(all\_x\_max) + x\_offset, \max(all\_y\_max) + y\_offset)$
- 12: Source points  $src\_pts \leftarrow [top\_left, top\_right, bottom\_left, bottom\_right]$

##### Clip ROI corners to be within image boundaries

- 13: **for** each *point* in  $src\_pts$  **do**
- 14:   **if**  $point_x < 0$  **then**
- 15:      $point_x \leftarrow 0$
- 16:   **else if**  $point_x > image_w$  **then**
- 17:      $point_x \leftarrow image_w$
- 18:   **end if**
- 19:   **if**  $point_y < 0$  **then**
- 20:      $point_y \leftarrow 0$
- 21:   **else if**  $point_y > image_h$  **then**
- 22:      $point_y \leftarrow image_h$
- 23:   **end if**
- 24: **end for**

##### Define target points based on image dimensions

- 25: Target points  $trg\_pts \leftarrow [(0,0), (image_w,0), (0,image_h), (image_w,image_h)]$

##### Calculate perspective transformation matrix

- 26:  $M = \text{getPerspectiveTransform}(src\_pts, trg\_pts)$

##### Transform images and bounding boxes

- 27: **for** each *image* and *labels* **do**
- 28:   Transform *image* using  $M$
- 29:   Transform *labels'* bounding box coordinates using  $M$
- 30:   Save the transformed images and bounding boxes
- 31: **end for**

- 32: **return**  $M$
-

Algorithm 1 automates the process of determining the ROI and generating the perspective transformation matrix  $M$ . The main steps of the algorithm are as follows:

1. **Initialize lists:** Store coordinates and dimensions of bounding boxes for all images, including minimum and maximum  $x$  and  $y$  coordinates, width, and height for each bounding box.
2. **Read bounding boxes:** Extract bounding box data from each image's corresponding label file, calculate all ROI boundary points, and update the respective lists.
3. **Calculate offsets:** Determine the ROI offsets using a specific  $\alpha$  value and the max width and height of all bounding boxes to define a slightly larger ROI.
4. **Determine ROI corners:** Use the minimum and maximum coordinates from the lists, along with the calculated offsets, to determine the corners of the ROI. These corners are the source points ( $src\_pts$ ) for the perspective transformation.
5. **Clip ROI corners:** Ensure ROI corners stay within image boundaries.
6. **Define target points:** Set target points ( $trg\_pts$ ) based on the image dimensions, representing the transformed image corners.
7. **Calculate transformation matrix:** Compute the perspective transformation matrix  $M$  using the source and target points. This matrix is used to transform the coordinates of the ROI to the new perspective.
8. **Transform images and bounding boxes:** Apply the transformation matrix  $M$  to each image and its bounding boxes. This involves transforming the image and adjusting the bounding box coordinates accordingly. The transformed images and bounding boxes are then saved.

The algorithm detailed in Algorithm 1 was initially applied to the training dataset to compute the perspective transformation matrix  $M$ . Subsequently, this matrix was utilized to transform the images and bounding boxes in the testing dataset. The proposed algorithm automates the perspective transformation process, ensuring consistent and precise selection of the region of interest (ROI) across extensive datasets. This automation minimizes manual intervention and the associated variability in selecting ROI corners for each image, making it an effective and scalable solution for applications like pothole detection in autonomous vehicles.

## 4. Experiment Design

### 4.1. Evaluation Dataset

To evaluate our proposed method, we utilized the dataset introduced by Nienaber et al. in [12,39]. This dataset, one of the few publicly available labeled pothole datasets, comprises 4405 images extracted from video footage captured with a GoPro camera mounted on a vehicle's windshield. Unlike most of the other pothole datasets collected using mobile phones or drones, this dataset provides a realistic representation of South African road conditions from a driver's perspective, making it particularly relevant for applications involving AVs and ground mobile robots. The dataset is split into two positive and negative directories. Positive samples are samples that include at least one instance of potholes and include a total of 1119 images, while negative samples are samples without any potholes and include a total of 2658 images. Each image is provided with a label file with a bounding box's format (class label, bounding box coordinates, width, and height). Moreover, the dataset is divided into training and testing subsets, with 628 images designated for testing. All images are provided in JPEG format with a resolution of  $3680 \times 2760$  pixels. Figure 4 showcases six representative samples from the dataset, illustrating the challenges posed by varying illumination levels and pothole appearances, which are critical for developing robust and accurate pothole detection systems.

The dataset presents several significant challenges, particularly given the nature of the objects it aims to detect—potholes. Potholes vary widely in size, shape, and appearance, making them inherently difficult to detect. These variations are further exacerbated by the fact that potholes at greater distances appear smaller, complicating the task of accurately identifying them. Moreover, the color of potholes can differ depending on the surrounding environment, such as sandy areas, pavement, or other types

of ground surfaces. This variation in appearance makes it challenging for a model to generalize across different scenarios, as the model must learn to recognize potholes in various contexts and lighting conditions. The difficulty of this task is amplified by the relatively small size of the dataset. Detecting small objects like potholes typically requires a large dataset to effectively learn the complex features necessary for accurate detection. To overcome this challenge, we use various augmentation techniques to enlarge the size of the training samples, and enhance the robustness and accuracy of the pothole detection model. Furthermore, we used several augmentation techniques including affine scaling, rotation, and shearing to adjust the image size, orientation, and viewpoint to help the model recognize potholes of different sizes and angles. Whereas, horizontal flipping provides different perspectives, while Gaussian blur mimics motion blur to handle imperfect image captures. Adjustments to gamma contrast, brightness, and contrast normalization manage varying lighting conditions, ensuring that the model performs well under different environments. Additionally, additive Gaussian noise is added to make the model resilient to grainy images, and crop and pad transformations simulate occlusions and varying distances from the camera. As a result, the number of training images increased from 1119 images to 2658 images. These augmentations simulate real-world conditions, helping the model generalize better and improve pothole detection performance under diverse scenarios encountered by autonomous vehicles. By creating a diverse and representative training dataset, the model becomes more robust and capable of accurately detecting potholes in various challenging conditions.



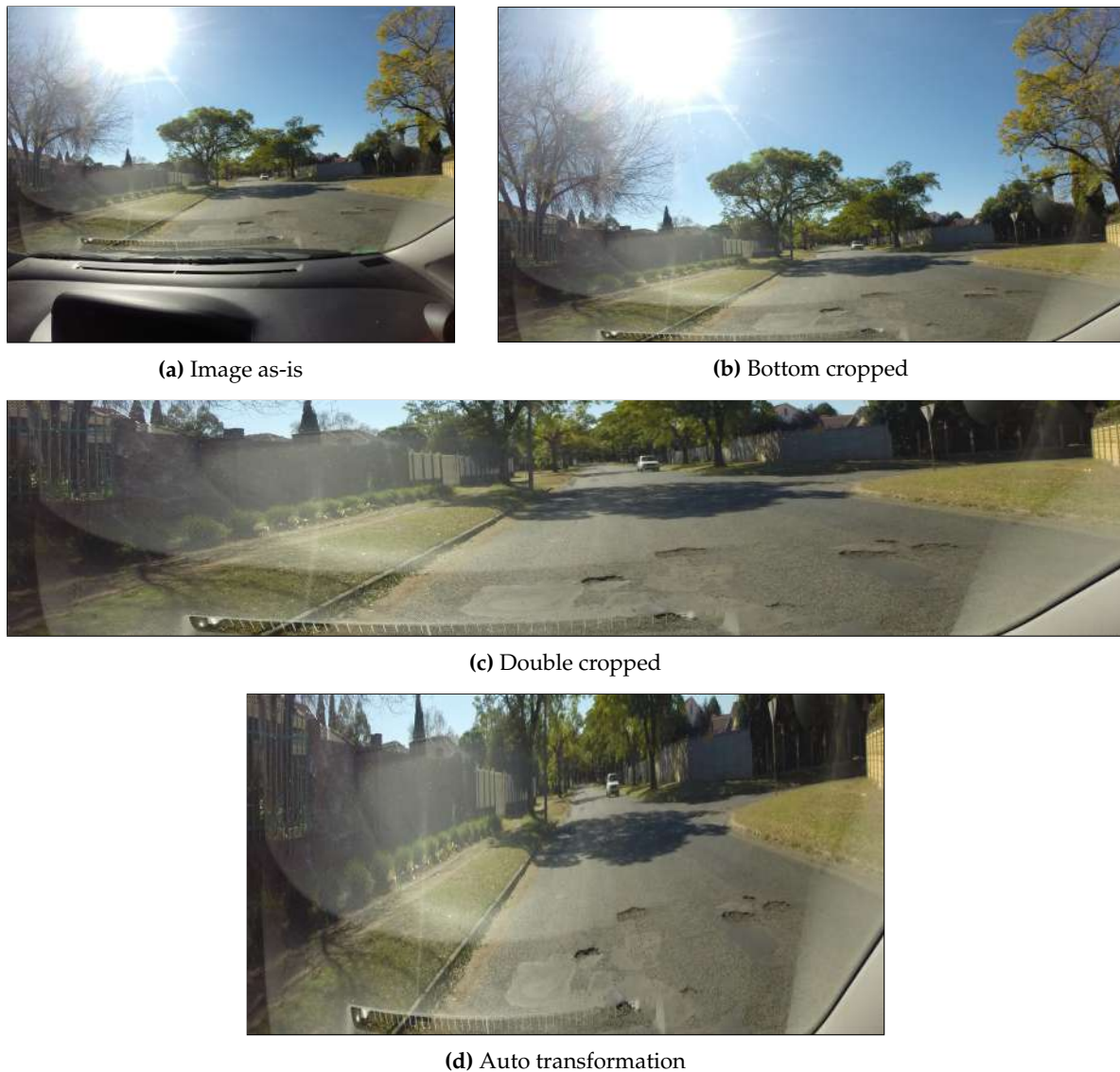
**Figure 4.** Demonstration of different samples of the dataset used in this work. These samples are some examples of the variance in lighting intensity and road conditions observed in this dataset.

Given the critical role of data quality and quantity in model performance, we explored several preprocessing methods to maximize the utility of the dataset in comparison to our proposed approach, Automatic Transformation. The methods evaluated include "Image As Is," "Dashboard Cropping," and "Dashboard and Sky Cropping." The "Image As Is" method involves using the images without any alterations, while "Dashboard Cropping" entails cropping the bottom portion of the images to exclude the dashboard, and "Dashboard and Sky Cropping" involves cropping both the top and bottom parts of the images to remove the sky and dashboard. These preprocessing techniques were employed to eliminate extraneous elements, such as the dashboard and sky, which can lead to misclassifications and increased computational load. This step was crucial in optimizing the model's efficiency and ensuring the dataset provided the best possible training conditions for pothole detection.



To determine optimal cropping locations and minimize the loss of bounding boxes, we analyzed their distribution and found that 99% fell within the range of 1200–1800 pixels of their y-coordinates. Values above this range typically corresponded to the dashboard, while values below included sky regions.

Figure 5 visually compares the original image with two different cropping cases, in addition to the proposed method, demonstrating the impact of image composition on computational efficiency and model performance. This highlights the importance of preprocessing techniques in optimizing the detection pipeline. In Image As is Figure 5a, the dashboard and sky occupy significant portions, introducing irrelevant information and increasing computational load. This results in unnecessary overhead and prolonged training times, negatively affecting model performance. To address this, image Figure 5b representing the Dashboard Cropping method, was cropped from the bottom to remove the dashboard, reducing false detections. However, this still left a substantial portion of the sky, contributing minimal information. Consequently, the Dashboard and Sky Cropping in Figure 5c was cropped to focus solely on the road surface, eliminating both the sky and the dashboard. Finally, Figure 5d shows our approach focusing only on the street where potholes are present.



**Figure 5.** Visualization of the different comparison methods employed in our experiments: (a) Naive approach (image as-is); (b) Fixed Cropping – Dashboard (bottom cropped); (c) Fixed Cropping – Dashboard and Sky Cropping (double-cropped); (d) Automated perspective transformation approach. The presented methods are demonstrated using the same input image.

#### 4.2. Evaluation Metrics

In object detection, key evaluation metrics include Intersection over Union (IoU), Precision, Recall, Average Precision (AP), and Average Recall (AR). These metrics are crucial for assessing the performance of detection models.

**Intersection over Union (IoU)** is a fundamental metric that measures the overlap between the predicted bounding box and the ground truth bounding box. It is calculated as the ratio of the area of intersection to the area of union of the two boxes, as shown in Equation 2. IoU is a threshold-based measure, typically used to determine whether a detection is considered a true positive or a false positive.

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} \quad (2)$$

**Precision** is the ratio of True Positives (TP) to the sum of True Positives and False Positives (FP), indicating the accuracy of the positive predictions made by the model. It is defined as shown in Equation 3.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3)$$

**Recall** measures the proportion of actual positives correctly identified by the model, calculated as the ratio of True Positives to the sum of True Positives and False Negatives (FN). This is expressed in Equation 4.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4)$$

**Average Precision (AP)**, derived from the Precision-Recall (P-R) curve, is calculated by integrating the area under this curve. AP at a specific IoU threshold (e.g.,  $AP_{50}$  for  $\text{IoU} \geq 0.5$ ) represents the precision averaged across different recall levels at that threshold.  $AP_{50:95}$  refers to the average precision computed at multiple IoU thresholds ranging from 0.5 to 0.95 with a step size of 0.05. This metric provides a comprehensive evaluation of model performance across various IoU thresholds.  $AP_{50}$  and  $AP_{75}$  specifically denote AP at IoU thresholds of 0.5 and 0.75, respectively, offering insights into model precision at different levels of overlap criteria.

**Average Recall (AR)** reflects the average recall over different numbers of detections per image, providing an aggregate measure of the model's ability to identify relevant instances among all actual positives.  $AR_{\max=1}$  and  $AR_{\max=10}$  denote the average recall when considering a maximum of one detection per image and ten detections per image, respectively, across IoU thresholds of 0.5 to 0.95. These metrics help to evaluate the model's recall capability, considering different levels of detection strictness.

These metrics collectively offer a detailed assessment of the detection model's performance, highlighting its strengths and weaknesses across various detection thresholds and conditions.

#### 4.3. Evaluation Strategy

To assess the effectiveness of our proposed method for detecting potholes at varying distances, we employed two evaluation strategies. Initially, we evaluated the performance using a single class (pothole). Subsequently, we expanded the analysis to include three classes (near, medium, and far) by categorizing the bounding boxes based on the  $y$ -coordinates of their top-left corners, with each region representing a different class. This classification aimed to measure the effectiveness of our approach in enhancing the detection of potholes at different distances. We conducted a comparative analysis against other dataset processing techniques, applying these evaluation strategies to each dataset using predefined thresholds as follows:

For the Image As Is and Dashboard Cropping:

- **Far:**  $1196 < y < 1350$

- **Medium:**  $1351 < y < 1500$
- **Near:**  $1501 < y < 1963$

For the Dashboard and Sky Cropping:

- **Far:**  $4 < y < 235$
- **Medium:**  $236 < y < 370$
- **Near:**  $371 < y < 771$

For the Automatic Transformation:

- **Far:**  $0 < y < 285$
- **Medium:**  $286 < y < 430$
- **Near:**  $431 < y < 800$

#### 4.4. Implementation Settings

Throughout all of our experiments, we trained the object detection models using the following hyperparameters: 100 epochs, a stochastic gradient descent (SGD) optimizer, a batch size of 16, and a learning rate of 0.01. The learning rate determines the step size, which is the amount the model's parameters are adjusted with respect to the gradient during optimization. This rate was chosen to balance the speed of convergence with stability, ensuring the step size is neither too large, causing overshooting; nor too small, leading to slow convergence. The SGD optimizer was selected based on its superior performance compared to the ADAM optimizer in our tests. Additionally, after data augmentation, the dataset, consisting of 2658 images, was divided into 80% for training and 20% for validation. For the experiments, we utilized the YOLOv5 small, medium, and large variants to assess the model's performance across different scales and complexities. The best-performing model on the validation set, determined based on the results from each epoch, was selected as the final model.

The hardware setup for our experiments consisted of a GTX 1080 GPU with 11GB of memory and 64GB of RAM. All experiments utilized images that were downscaled to a resolution of  $800 \times 800$  pixels from the original  $3680 \times 2760$  resolution captured by a high-resolution camera. We chose an input resolution of  $800 \times 800$ , which is marginally larger than the default YOLOv5 base resolution of  $640 \times 640$ . This choice was justified based on our empirical findings, where the larger input size allowed for better feature representation, especially for detecting smaller and more distant potholes. The increased resolution facilitated the model's ability to capture finer details, thus enhancing detection accuracy without significantly compromising computational efficiency or overwhelming the available hardware resources.

To start the training process, we fine-tuned the YOLOv5 model on our dataset, leveraging the pre-trained weights and further training them specifically on our data. The default training augmentations provided by the YOLOv5 framework were employed throughout the experiments, alongside the default hyperparameters, which we kept unchanged for consistency and standardization purposes except the ones we mentioned earlier. The training process exclusively utilized positive images, applying the preprocessing augmentations as detailed in 4.1. This methodological choice, including the exclusion of negative images, is further substantiated by an ablation study presented in the following section, which validates the effectiveness of these decisions in optimizing the model's performance. Additionally, based on our experiments and achieving optimal transformation outcomes, we selected an  $\alpha$  value of 0.2 for the automatic transformation algorithm presented in Algorithm 1.

## 5. Results and Discussion

### 5.1. Experiment 1: Naive vs. Fixed Cropping vs. Automated Transformation Approach

In this experiment, we systematically trained one variant of YOLOv5 (small) on all possible dataset configurations. These configurations were evaluated for both single-class (pothole) and multi-class

(near, medium, and far distance) detection tasks. As detailed in Table 1. Each method was assessed for overall pothole detection as one class, and for each distance-based class separately.

**Table 1.** Experiment 1 results. This experiment compares the different approaches presented in this work by fine-tuning YOLOv5-small under each configuration and then evaluating their performance on the test set using various object detection metrics. Our proposed approach demonstrates superior performance across all metrics and pothole distance categories.

Approach	Pothole Distance	Metric (%)				
		AP <sub>50:95</sub>	AP <sub>50</sub>	AP <sub>75</sub>	AR <sub>max=1</sub>	AR <sub>max=10</sub>
Image as Is	All	17.3	41.8	10.5	15.4	22.6
	Near	20.8	49.4	14.0	18.9	27.0
	Medium	16.8	42.4	9.3	18.8	23.5
	Far	4.5	11.9	1.5	5.8	6.6
Bottom Cropped	All	17.9	42.5	11.4	15.9	23.3
	Near	21.4	49.0	14.1	19.4	27.3
	Medium	18.7	46.2	11.5	20.6	25.9
	Far	3.9	12.4	1.5	5.5	6.6
Double Cropped	All	16.7	43.7	9.0	15.0	22.0
	Near	21.3	53.5	11.6	19.3	27.0
	Medium	17.6	46.9	10.2	20.9	23.9
	Far	8.8	25.4	4.3	10.4	13.4
Auto Transformation	All	<b>25.2</b>	<b>54.2</b>	<b>19.3</b>	<b>20.0</b>	<b>32.0</b>
	Near	<b>27.1</b>	<b>57.1</b>	<b>22.6</b>	<b>22.8</b>	<b>33.6</b>
	Medium	<b>30.0</b>	<b>64.2</b>	<b>23.8</b>	<b>31.6</b>	<b>37.2</b>
	Far	<b>17.0</b>	<b>39.2</b>	<b>10.5</b>	<b>19.0</b>	<b>25.0</b>

The best result per metric and pothole distance is highlighted in **bold**.

As illustrated in Table 1, our novel approach demonstrates superior performance across all metrics, surpassing all other methods. Notably, we observed a substantial increase in AP<sub>50:95</sub>, with a 45.7% improvement in the single class using our proposed approach compared to the Image As Is method. Furthermore, there were increases of 30%, 78.5%, and 278% for the near, medium, and far classes, respectively, at the same IoU threshold. Besides, our approach resulted in an improvement in AP<sub>50</sub> of 29.6%, and in AP<sub>75</sub> of 83.8%. Improvements were also observed in AR, with AR<sub>max=1</sub> increasing by 30% and AR<sub>max=10</sub> by 41.5% at the same (50:95%) IoU threshold. These results underscore the effectiveness of our method in enhancing pothole detection accuracy compared to traditional approaches.

The significant improvements observed in our results are due to the effectiveness of the automatic perspective transformation approach, which virtually brings potholes closer to the vehicle, amplifying their features and making them more discernible to the detection model, as Figure 3 shows. This perspective adjustment enhances the model's ability to learn and recognize pothole patterns, resulting in more accurate and reliable detections. The amplification of pothole features simplifies the learning process for the YOLOv5 model, leading to significant improvements in average precision across various classes and IoU thresholds.

The proposed approach not only improved the overall pothole detection performance, but also excelled in detecting the more challenging cases, particularly medium- and far-distance potholes, which are the most critical for safety. Our method significantly improved the detection accuracy for far potholes, an area where other methods have notably underperformed. The consistent performance gains across different IoU thresholds validate the robustness of our approach. Traditional detection methods struggle with varying perspectives and angles, while our method standardizes these perspectives, offering a more uniform dataset for the model to train on, which is crucial for real-world applications. The success of our approach in enhancing pothole detection accuracy has broader implications for other object detection tasks, potentially leading to advancements in multiple areas of computer vision.



### 5.2. Experiment 2: Effects of Network Complexity/Scale on Performance

As demonstrated in Table 2, we performed an extensive evaluation of our proposed approach compared to the baseline method, where the image remains unchanged. The evaluation was performed using three variants of YOLOv5 (small, medium, and large). Furthermore, we tested these models on three distinct classes (near, medium, and far), as well as on the entire pothole dataset treated as a single class (pothole).

**Table 2.** Experiment 2 results. This experiment compares the naive (i.e., Image As-Is) approach with our proposed approach on three YOLOv5 variants. In each configuration, a YOLOv5 variant is fine-tuned on the corresponding approach's training set and then evaluated on the test set using various object detection metrics. Results show that our proposed approach always surpasses the performance of the naive approach regardless of the utilized variant. Additionally, combining YOLOv5-Small with our proposed approach significantly outperforms the naive approach even when compared to the YOLOv5-Large configuration, for which the model is over six times larger in terms of the number of parameters, across all metrics and pothole distance categories.

Approach	Object Detection Model	Parameters (M)	FLOPs (G)	Pothole Distance	Metric (%)				
					AP <sub>50:95</sub>	AP <sub>50</sub>	AP <sub>75</sub>	AR <sub>max=1</sub>	AR <sub>max=10</sub>
Image As Is	YOLOv5-Small	7.2	16.5	All	17.3	41.8	10.5	15.4	22.6
				Near	20.8	49.4	14.0	18.9	27.0
				Medium	16.8	42.4	9.3	18.8	23.5
				Far	4.5	11.9	1.5	5.8	6.6
	YOLOv5-Medium	21.2	49.0	All	18.3	43.4	12.6	16.2	23.6
				Near	21.5	49.0	16.1	19.3	27.0
				Medium	18.9	45.6	12.0	20.6	26.1
				Far	5.1	16.2	1.8	7.6	8.5
	YOLOv5-Large	46.5	109.1	All	18.4	43.3	12.6	15.7	23.8
				Near	22.4	50.3	16.5	19.5	28.3
				Medium	17.7	45.2	10.6	20.1	25.0
				Far	4.6	13.6	1.5	6.4	7.6
Auto Transformation	YOLOv5-Small	7.2	16.5	All	<b>25.2</b>	54.2	<b>19.3</b>	<b>20.0</b>	<b>32.0</b>
				Near	<b>27.1</b>	<b>57.1</b>	<b>22.6</b>	22.8	<b>33.6</b>
				Medium	<b>30.0</b>	<b>64.2</b>	<b>23.8</b>	<b>31.6</b>	<b>37.2</b>
				Far	17.0	39.2	10.5	<b>19.0</b>	25.0
	YOLOv5-Medium	21.2	49.0	All	23.2	51.2	17.2	19.2	29.4
				Near	25.8	55.0	20.1	22.5	31.7
				Medium	27.4	57.0	21.5	28.8	33.6
				Far	15.9	39.8	10.7	16.7	22.4
	YOLOv5-Large	46.5	109.1	All	24.8	<b>54.6</b>	18.1	<b>20.0</b>	31.4
				Near	26.9	55.9	22.0	<b>22.9</b>	33.5
				Medium	27.9	60.8	19.3	31.0	34.9
				Far	<b>18.8</b>	<b>45.2</b>	<b>11.9</b>	18.7	<b>25.2</b>

The best result per metric and pothole distance is highlighted in **bold**.

The results demonstrate a significant improvement in detection accuracy when comparing the two approaches across all YOLOv5 variants for all classes. Notably, the YOLOv5 small variant, when applied to our approach, outperformed the YOLOv5 large variant applied to the baseline approach, knowing that YOLOv5 large has almost 6x the number of parameters as in YOLOv5 small as explained in subsection 3.1.2. This highlights the effectiveness of our method in detecting potholes across various distances while requiring lower computational resources compared to the traditional approach.

The results in Table 2 reveal that our proposed approach significantly enhances detection accuracy across all YOLOv5 variants for every class compared to the baseline method. The superior performance of the YOLOv5 small variant is particularly noteworthy, which outperformed both the medium and large variants when using our method. We hypothesize that this counterintuitive result stems from the larger and medium YOLOv5 models being more susceptible to the poor quality of some labels, potentially learning and incorporating these inaccuracies into their detection processes more than the smaller variant. Consequently, the small version's relatively simpler architecture may have enabled it to generalize better and avoid overfitting to the noisy data, resulting in enhanced detection accuracy [40]. This finding underscores the effectiveness of our automatic perspective transformation approach

and suggests that smaller, less complex models can be more robust in scenarios where data quality is variable, offering valuable insights for similar projects in object detection.

5.3. Experiment 3: Ablation Study

Given that our dataset contains negative images and the augmentation capabilities inherent in the YOLOv5 framework, we conducted a comprehensive series of experiments to identify the optimal configuration for training our model. The goal was to quantitatively validate the chosen configuration throughout our experiments. We integrated the negative images with the positive images and explored various augmentation strategies, testing the effectiveness of relying exclusively on the YOLOv5 framework’s default augmentations versus supplementing them with additional manual augmentations that were introduced in section 4.1. This methodological investigation aimed to rigorously assess the impact of these different approaches on the model’s performance.

To identify the optimal configuration, we created and evaluated four distinct setups using our automatic perspective transformation approach. The results of these experiments are detailed in Table 3. The chosen configuration was then utilized for subsequent experiments presented in Table 1 and Table 2.

The four configurations tested were as follows:

- 1. **YOLOv5’s Augmentations Only, No Negative Images:** This setup utilized only YOLOv5’s augmentation step without negative images, as illustrated in the first row of Table 3.
- 2. **YOLOv5’s Augmentations with Negative Images:** This setup included negative images alongside YOLOv5’s augmentation step, shown in the second row of the table.
- 3. **Manual Preprocessing and YOLOv5’s Augmentation, Positive Images Only:** This configuration combined manual preprocessing augmentations with YOLOv5’s augmentations, using only positive images. It achieved the best results among all setups.
- 4. **Manual Preprocessing and YOLOv5’s Augmentation with Negative Images:** This setup used both manual and YOLOv5’s augmentations, incorporating negative images into the positive dataset. It resulted in the lowest performance metrics.

**Table 3.** Ablation study results comparing the different preprocessing configurations. These results are based on the automated transformation approach on all test set potholes using YOLOV5 Small. Including the preprocessing augmentations while excluding the negative samples (training images without potholes) produced the best performance across all metrics.

Configuration		Metric (%)				
Preproc. Augs.	Neg. Images	AP <sub>50:95</sub>	AP <sub>50</sub>	AP <sub>75</sub>	AR <sub>max=1</sub>	AR <sub>max=10</sub>
		24.5	53.6	17.9	18.9	31.2
	✓	23.7	52.3	17.1	19.1	30.3
✓		<b>25.2</b>	<b>54.2</b>	<b>19.3</b>	<b>20.0</b>	<b>32.0</b>
✓	✓	23.4	51.5	17.3	19.0	29.7

The best result per metric is highlighted in **bold**.

As illustrated in Table 3, the configuration that utilizes manual pre-processing with only the positive data set during training consistently achieved the best results in all metrics. This approach was subsequently applied to all experiments conducted in this study, confirming its superiority as the optimal method for improving the accuracy of pothole detection.

Moreover, the results show that our approach that combined manual preprocessing with only the positive dataset consistently outperformed all other configurations to improve the accuracy of the pothole detection. This setup, which excluded negative images and relied on extensive augmentations, proved superior across all metrics. We hypothesize that the inclusion of negative images introduced noise into the training process, as these images lack bounding boxes or pothole features, which are

critical for improving the model's robustness and pattern recognition capabilities. Additionally, the extensive use of manual preprocessing augmentations exposed the model to a wider variety of pothole shapes, colors, and orientations, enhancing its ability to generalize across different scenarios. In contrast, relying solely on YOLOv5's framework augmentation step limited the model's exposure to diverse cases, thereby restricting its generalization potential.

By meticulously analyzing these configurations, we established a robust methodology to leverage augmentation techniques and dataset composition to maximize the effectiveness of our automatic perspective transformation approach in pothole detection. This comprehensive evaluation underscores the importance of carefully selecting training configurations to achieve superior performance in computer vision tasks.

## 6. Conclusion

In this paper, we introduced a novel method for improving pothole detection by leveraging perspective transformation to automatically extract ROI from images and their corresponding labels. The transformed dataset was then fed into the YOLOv5 small object detection model. Our approach resulted in a notable improvement in detection accuracy, achieving a 45.7% increase in average precision (AP) for a single class at IoU thresholds of 0.5 to 0.95 ( $AP_{50:95}$ ), compared to the naive use of changed images. In addition, the method significantly improved the detection of potholes at various distances, near, medium, and far, addressing a crucial aspect of road safety. The findings underscore the critical role of preprocessing techniques, such as perspective transformation, in enhancing the performance of object detection tasks.

For future work, we propose developing a deep learning model capable of dynamically regressing the four corner points of the street in each image to generate a perspective transformation matrix. This approach would necessitate labeled data, potentially obtainable from semantic segmentation datasets, to further automate and refine the preprocessing pipeline.

**Author Contributions:** Conceptualization, A.A., Z.E. and S.A.; data curation, A.A.; methodology, A.A., Z.E. and S.A.; software, A.A.; validation, Z.E.; formal analysis, A.A. and Z.E.; investigation, A.A.; resources, S.R.; data curation, A.A.; writing—original draft preparation, A.A. and Z.E.; writing—review and editing, Z.E. and S.R.; visualization, A.A. and Z.E.; supervision, Z.E. and S.R.; project administration, S.R. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The original data presented in the study are openly available in [Kaggle] at [<https://www.kaggle.com/datasets/sovitath/road-pothole-images-for-pothole-detection/data>].

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

## References

1. Potholes could be 'self-repairing' in the next 30 years, say experts — independent.co.uk. Available online: <https://www.independent.co.uk/travel/news-and-advice/potholes-fix-repair-self-repairing-roads-transport-infrastructure-maintenance-a8719716.html> (accessed on 01 08 2024).
2. Visual Expert. Reaction Time. Available online: <https://www.visualexpert.com/Resources/reactiontime.html> (accessed on 01 08 2024), 2024.
3. Remodel or Move. Should You Hit the Brakes When Going Over a Pothole? Available online: <https://www.remodelormove.com/should-you-hit-the-brakes-when-going-over-a-pothole/> (accessed on 01 08 2024), 2023.
4. Balakuntala, S.; Venkatesh, S. An intelligent system to detect, avoid and maintain potholes: A graph theoretic approach. *arXiv preprint arXiv:1305.5522* **2013**.
5. University of Minnesota Twin Cities. Talking Potholes. Available online: <https://twin-cities.umn.edu/news-events/talking-potholes-u-m> (accessed on 01 08 2024), 2023.
6. Kim, Y.M.; Kim, Y.G.; Son, S.Y.; Lim, S.Y.; Choi, B.Y.; Choi, D.H. Review of recent automated pothole-detection methods. *Applied Sciences* **2022**, *12*, 5320.

7. Eduzaurus. Pothole Detection Methods. Available online: <https://eduzaurus.com/free-essay-samples/pothole-detection-methods/> (accessed on 01 08 2024), 2023.
8. Geoawesomeness. Application of Mobile LiDAR on Pothole Detection. Available online: <https://geoawesomeness.com/eo-hub/application-of-mobile-lidar-on-pothole-detection/> (accessed on 01 08 2024), 2013.
9. Samczynski, P.; Giusti, E. *Recent Advancements in Radar Imaging and Sensing Technology*; MDPI, 2021.
10. Outsight. How Does LiDAR Compare to Cameras and Radars? Available online: <https://www.outsight.ai/insights/how-does-lidar-compares-to-cameras-and-radars> (accessed on 01 08 2024), 2023.
11. Zhang, J.; Zhang, J.; Chen, B.; Gao, J.; Ji, S.; Zhang, X.; Wang, Z. A perspective transformation method based on computer vision. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 2020, pp. 765–768. <https://doi.org/10.1109/ICAICA50127.2020.9182641>.
12. Nienaber, S.; Booysen, M.J.; Kroon, R. Detecting potholes using simple image processing techniques and real-world footage. In Proceedings of the 34th South Africa Transport Conference (SATC) Pretoria, South Africa, 6–9 July 2015, 2015.
13. Pereira, V.; Tamura, S.; Hayamizu, S.; Fukai, H. A deep learning-based approach for road pothole detection in timor leste. In Proceedings of the 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI). IEEE, 2018, pp. 279–284.
14. Chen, H.; Yao, M.; Gu, Q. Pothole detection using location-aware convolutional neural networks. *International Journal of Machine Learning and Cybernetics* **2020**, *11*, 899–911.
15. Kumar, A.; Kalita, D.J.; Singh, V.P.; et al. A modern pothole detection technique using deep learning. In Proceedings of the 2nd International Conference on Data, Engineering and Applications (IDEA). IEEE, 2020, pp. 1–5.
16. Dhiman, A.; Klette, R. Pothole detection using computer vision and learning. *IEEE Transactions on Intelligent Transportation Systems* **2019**, *21*, 3536–3550.
17. Dhiman, A.; Chien, H.J.; Klette, R. Road surface distress detection in disparity space. In Proceedings of the 2017 International Conference on Image and Vision Computing New Zealand (IVCNZ). IEEE, 2017, pp. 1–6.
18. Maeda, H.; Kashiyama, T.; Sekimoto, Y.; Seto, T.; Omata, H. Generative adversarial network for road damage detection. *Computer-Aided Civil and Infrastructure Engineering* **2021**, *36*, 47–60.
19. Salaudeen, H.; Çelebi, E. Pothole Detection Using Image Enhancement GAN and Object Detection Network. *Electronics* **2022**, *11*, 1882.
20. Tan, M.; Pang, R.; Le, Q.V. Efficientdet: Scalable and efficient object detection. In Proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10781–10790.
21. Shaghouri, A.A.; Alkhatib, R.; Berjaoui, S. Real-time pothole detection using deep learning. *arXiv preprint arXiv:2107.06356* **2021**.
22. Bučko, B.; Lieskovská, E.; Záborská, K.; Záborský, M. Computer vision based pothole detection under challenging conditions. *Sensors* **2022**, *22*, 8878.
23. Rastogi, R.; Kumar, U.; Kashyap, A.; Jindal, S.; Pahwa, S. A comparative evaluation of the deep learning algorithms for pothole detection. In Proceedings of the 2020 IEEE 17th India Council International Conference (INDICON). IEEE, 2020, pp. 1–6.
24. Kocur, V. Perspective transformation for accurate detection of 3d bounding boxes of vehicles in traffic surveillance. In Proceedings of the Proceedings of the 24th Computer Vision Winter Workshop, 2019, Vol. 2, pp. 33–41.
25. Lee, W.Y.; Jovanov, L.; Philips, W. Multi-View Target Transformation for Pedestrian Detection. In Proceedings of the Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, January 2023, pp. 90–99.
26. Wang, K.; Fang, B.; Qian, J.; Yang, S.; Zhou, X.; Zhou, J. Perspective Transformation Data Augmentation for Object Detection. *IEEE Access* **2020**, *8*, 4935–4943. <https://doi.org/10.1109/ACCESS.2019.2962572>.
27. Hou, Y.; Zheng, L.; Gould, S. Multiview Detection with Feature Perspective Transformation. In Proceedings of the Computer Vision – ECCV 2020; Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J.M., Eds., Cham, 2020; pp. 1–18.
28. Jocher, G. YOLOv5 by Ultralytics. Available online: <https://github.com/ultralytics/yolov5> (accessed on 4 8 2024), 2020. <https://doi.org/10.5281/zenodo.3908559>.



29. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. Springer, 2016, pp. 21–37.
30. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* **2018**.
31. Liu, K.; Fu, Z.; Jin, S.; Chen, Z.; Zhou, F.; Jiang, R.; Chen, Y.; Ye, J. ESOD: Efficient Small Object Detection on High-Resolution Images. *arXiv preprint arXiv:2407.16424* **2024**.
32. Saponara, S.; Elhanashi, A. Impact of image resizing on deep learning detectors for training time and model performance. In Proceedings of the International Conference on Applications in Electronics Pervading Industry, Environment and Society. Springer, 2021, pp. 10–17.
33. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* **2019**, 32.
34. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 7464–7475.
35. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLO. Available online: <https://github.com/ultralytics/ultralytics> (accessed on 4 8 2024), 2023.
36. Nozick, V. Multiple view image rectification. In Proceedings of the 2011 1st International Symposium on Access Spaces (ISAS). IEEE, 2011, pp. 277–282.
37. El Shair, Z.; Rawashdeh, S. High-temporal-resolution event-based vehicle detection and tracking. *Optical Engineering* **2023**, 62, 031209–031209.
38. Kocur, V.; Ftáčnik, M. Detection of 3D bounding boxes of vehicles using perspective transformation for accurate speed measurement. *Machine Vision and Applications* **2020**, 31, 62.
39. Nienaber, S.; Kroon, R.; Booysen, M.J. A comparison of low-cost monocular vision techniques for pothole distance estimation. In Proceedings of the 2015 IEEE symposium series on computational Intelligence. IEEE, 2015, pp. 419–426.
40. Quach, L.D.; Quoc, K.N.; Quynh, A.N.; Ngoc, H.T. Evaluating the effectiveness of YOLO models in different sized object detection and feature-based classification of small objects. *Journal of Advances in Information Technology* **2023**, 14, 907–917.

**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.