Received: April 14, 2023. Revised: May 23, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Parking Space Availability Detections from Two Overlapping Cameras Using YOLOv5 and Image Stitching Methods

Misbachul Falach Asy'ari[1] **Chastine Fatichah[1]*** **Nanik Suciati[1]**

[1]*Department of Informatics, Institut Teknologi Sepuluh Nopember, Indonesia*
* Corresponding author's Email: chastine@if.its.ac.id

Abstract: Capturing a large parking lot's entire area requires the installation of more than one camera. In a parking availability detection system, it is crucial to identify the overlapping area recorded by two cameras. This study aims to identify the overlapping area on two cameras using the image stitching method. The you only look once version 5 (YOLOv5) method is then used to determine whether parking spaces on the stitched image are empty or occupied by cars. The experiments using six YOLOv5 configurations and three different image stitching methods showed that the system could detect the availability of parking slots with the best mean average precision (mAP) score of 0.953. The total number of parking spaces before and after stitching is also compared in this study to demonstrate the accuracy of the number of overlapping parking slots compared to the actual number.

Keywords: Parking space detection, Overlapping camera, Image stitching, YOLO, Smart city.

1. Introduction

Nowadays, there are many car parks that are inadequate to accommodate the capacity of parked cars. There are still many cars that will park, having difficulty getting an empty parking space. This problem is usually encountered during rush hours such as morning and evening in several malls and tourist spots in several big cities [1]. The problem with the parking lot can cause many vehicles to queue and the drivers need to go around looking for an empty parking slot. This will cause quite long queues and can cause loss of productive time. One way to overcome this issue is to apply technology that can detect the availability of parking lots (to know whether the slot is empty or occupied by a vehicle).

Technology could be used to solve problems that exist in everyday life, such as the problem of the availability of parking slots. In the case of car park availability problems, there are several techniques that have been implemented before, such as applying ground-mounted sensors to the parking slots. These sensors include ultrasonic sensors, geomagnetic sensors, and infrared sensors. The ultrasonic sensor is projected on the grid map to measure the observation target space to determine whether it is feasible to park the vehicle or not. However, a parking system that uses ultrasonic sensors requires a lot of money due to the complexity of the installation of the sensors [2]. On the other hand, a previous study shows that integrating related technologies such as ZigBee, geomagnetic sensors, and recurrent neural network (RNNs) could address parking space availability. However, the limitation of this research is that it required the use of many geomagnetic sensors to get good results, so it required additional costs and many components [3]. In another word, the technique of detecting the availability of parking slots using ground-mounted sensors requires expensive installation and maintenance costs [4]. Other techniques, such as employing closed-circuit television (CCTV), could be used to determine whether parking spaces are available without using sensors.

Most parking lots have CCTV as a surveillance function (such as for detecting crime). Besides that, CCTV can also be used as a technology that can solve the problem of parking lot availability. It could detect the availability of parking slots [5]. With various techniques available in the field of computer vision, such as the object detection method, we could recognize the location of the car parking slot and could also identify whether the parking slot is empty or occupied by cars. There have been a number of prior research on the availability detection for vehicle parking lots, including using the image segmentation-based method like mask regional convolutional neural network (Mask RCNN) [6] and bounding box-based method like you only look once (YOLO) method [7].

Several object detection techniques use deep learning methods such as YOLO or Mask RCNN to detect object locations in a frame/image and classify them. Deep learning methods are proven better than traditional methods (motion-based recognition and machine learning) and are making tremendous progress in the development of car parking lot availability detection [6]. Compared to Mask RCNN (instance segmentation-based), YOLO (bounding box-based) is more suitable in the case of parking space availability detection because we don't need to segment the car/parking slot and we only need the location of the parking slot by simply using a bounding box, so it will reduce computational costs. A study using the YOLO version 3 (YOLOv3) method and the Lite AlexNet classifier has performed automatic parking space availability detection by utilizing the YOLOv3 bounding box [7]. On the other hand, the YOLO algorithm has been improved with the YOLO version 5 (YOLOv5). YOLOv5 has been proven capable of producing more object detection system performance compared to its previous versions (YOLOv3 and YOLOv4) [8]. The YOLOv5 algorithm has also been shown to have better performance compared to object detection methods such as faster region-based convolutional neural networks (Faster RCNN) [9]. By that, YOLOv5 could also be used for the problem of the parking lot availability detection to achieve the best result.

Another challenge that is often encountered is that many parking lots usually install more than one CCTV camera to get a wider perspective, such as PKLot dataset [10], IP Camera dataset [11], PLds dataset [12], and CNRPark dataset [13]. This serves as better surveillance due to the wider field of view coverage. Some of the cameras installed are adjusted to the conditions in that place and sometimes it is possible for these cameras to capture overlapping areas, such as CNRPark. In order to be used as a system to determine the occupancy of parking spaces, a method is required to cope with these overlapping areas. The aim is to allow the system to detect overlapping parking slots on two cameras that are actually the same parking slot.

Image stitching is a method that can combine two images (or more) that have overlapping areas. This method detects pixels with the same intensity value in both images and uses those pixels as key points to join the overlapping regions. A previous study used an image stitching method to combine multiple CCTV cameras within a mine with overlapping camera angles to provide seamless visualization results [14]. The visualization results of this method can be affected by feature detectors. This feature detector is a process of image stitching used to detect key points. Experiments using three feature detectors (Accelerated KAZE (AKAZE), binary robust invariant scalable keypoints (BRISK), and oriented FAST and rotated BRIEF (ORB)) on multiple image datasets prove to be more efficient compared to other feature detectors. [15]. These studies led the authors to use an image stitching method with trials of the three feature detectors to overcome the case of parking slot availability detection with overlapping camera regions.

Our proposed task is to train the system to identify the availability parking spaces that are either empty or occupied using the YOLOv5 algorithm. Meanwhile, an image stitching method with each feature detector (ORB, BRISK, AKAZE) is applied to combine two images of a pair of cameras with overlapping areas. Image stitching results are tested using YOLOv5 training weights. Thus, the system could recognize the same parking slot in overlapping regions. We evaluate the performance of the system by comparing the sum of all the parking lots in the two images before and after the image stitching process. The system was also tested using the mean average precision (mAP) score to see how well the system detected parking slot availability.

Our main contribution is to detect the overlapping area in two images from two different cameras using the image stitching method. We also combine it with the YOLOv5 method to identify whether the car parking lot is free or filled with the car. This study applies a newer version of YOLO than previous research in the case of detecting the occupancy of parking lots. We also compared the performance of several feature descriptors in the image stitching process.

This essay's remaining sections are organized as follows. Section 2 reviews the literature on previous studies regarding parking space availability detection. Section 3 is a methodology that explains the dataset used, data preprocessing, overlapping detection using the image stitching method, the training process using the YOLOv5 method, and the testing process. Section 4 discusses the experimental results and their analysis regarding the results of stitched images and the results of parking space availability detection from two overlapping cameras. Finally, the conclusion section wraps up this work's primary findings and provides a brief overview of future work.

2. Related works

2.1 Parking space availability detections

There are already some work studies regarding the detection of parking space availability, especially using deep learning methods. Research by [11] uses variations of the LeNet, AlexNet, mLeNet, and mAlexNet methods to detect parking spots using the CNRPark dataset and private IP. Data collection was carried out from morning to evening. The best accuracy was obtained at 0.9315 on the mAlexNet method. Another study by [16] used a deep extreme learning machine (DELM) algorithm to detect roadside parking spots. The accuracy obtained is 0.9125. However, the two studies only used one camera angle.

On the other hand, several studies have used different camera angles to get a wider field of view. Research by [17] uses Faster RCNN to detect vacant parking slots. The data used PKLot dataset with 3 different camera angles. With a reasonably high epoch, training accuracy can increase to 0.9 at the 20,000th epoch. There is a dataset with 9 camera angles published by [13]. The authors use the convolutional neural network (CNN) method with the mLeNet and mAlexNet architectures. The PKLot dataset (training data) was tested using the PKLot dataset to get an accuracy of 0.989 on mAlexNet. This dataset can be explored further because it uses 9 different camera angles.

Bounding box-based object detection methods, such as YOLO, can be used in the case of detecting parking space vacancies. Another study by [7] divides into two stages in the process of identifying the parking lot occupancy, which are the marking stage using YOLOv3 and the classification stage using several methods such as Mini AlexNet, Lite AlexNet, AlexNet, and VGG16. The dataset uses CNRPark in 3 different weather conditions: sunny, rainy, and overcast. An average accuracy of 0.9233 was obtained at the classification stage using Alexnet. However, the YOLO method is a one-stage detector that, apart from being able to pinpoint the location of the object, can also classify the object, (in this case, a parking lot that is empty or filled with cars). In addition, there are studies that use YOLOv3 modifications by adding residual blocks for further feature extraction. The data used the PASCAL VOC, COCO, and PKLot datasets [4].

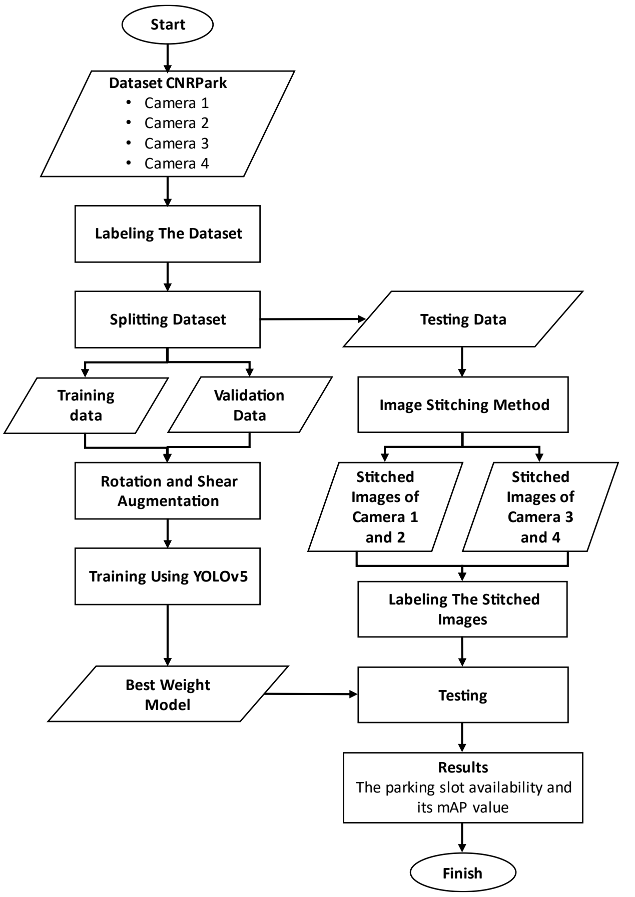


Figure. 1 Methodology flowchart

Table 1. The number of datasets

Camera	Total Images	Training Data	Validation Data	Testing Data
1	456	329	82	45
2	456	335	76	45
3	460	323	88	49
4	460	321	90	49
Total	**1832**	**1308**	**336**	**188**

Those studies concentrate on using a single camera to identify the availability of parking spaces. As mentioned in the contribution section of this paper, this study detects the availability of parking space on two overlapping cameras. Therefore, this paper needs to discuss the image stitching method in previous research which can be used to overcome this problem.

2.2 Image stitching applications

There are numerous applications for the image stitching technique. Scale invariance is handled by research by [18] using a modified ORB feature detector with Gaussian pyramid. An endoscopic image with an overlapping area was used as the source of the data. This technique thereby outperforms the scale invariant feature transform (SIFT) feature descriptor in terms of registration accuracy and stitching process speed, outperforming SIFT by a factor of roughly ten. As a result of this study, ORB can be employed as a feature detector that performs better than SIFT.

Mineralogical analysis is a further use of the image stitching method [19]. This research takes overlapping images of the rocks with the camera. By using the speeded-up robust features (SURF) feature detector, a collection of overlapping images could be stitched together to make it easier to analyze the mineral content in the rock. The results of this study were compared with the other three methods, both visually and in computational time. This study not only stitched two images into one, but several images (mosaic).

The authors combined the image stitching technique with YOLOv5 based on previous studies. Image stitching could be applied in the case of parking lots that have overlapping cameras. Meanwhile, to detect parking slots that are empty or occupied by cars, the YOLOv5 object detection method could be applied as the proposed method.

3. Methodology

Several methods were used in this study. First, we selected the dataset as needed. Furthermore, a pre-processing stage is required before the training process. On the other hand, the image stitching method is applied to testing data to get stitched images. These images are used to test the parking space availability detection system. All the steps are shown in Fig. 1.

3.1 CNRPark datasets

The datasets originated from a public dataset called CNRPark [13]. The CNRPark dataset consists of 9 cameras with viewing angles. Of the nine cameras, there are two sets of cameras with overlapping areas. Those are Camera 1 with Camera 2, and Camera 3 with Camera 4. Therefore, we are only using images from those four cameras with a total data of 1820 images. Each camera was recorded in three different weather conditions, which are overcast, rainy, and sunny. The data were collected on different days between November 2015 to February 2016. An example of data is shown in Fig. 2.

3.2 Data preprocessing

After preparing the datasets, the next step is the preprocessing stage. This starts with creating the ground truth. Ground truth is required to train the machine to recognize objects in empty or occupied parking spaces and their locations in the image. This process is carried out by using ‘app.roboflow.com’ to create a bounding box/annotation/labeling for each parking slot in each image. Each image in the dataset is labeled based on the parking space availability, whether empty or occupied by a car. From the total data, there are around 3000 empty classes and 3500 occupied classes. For the record, we only label existing parking slots based on the parking space box. This is because some cars do not park correctly, for example on Camera 1 and Camera 2.

After labeling all the images, the next process is to split the datasets into training data, validation data, and testing data. This process also uses the help of Roboflow. We only select a few images from each camera with overlapping regions to use as test data. Therefore, only about 10% of the total datasets are testing data. The rests are training and validation data with a ratio of 80:20. See Table 1 for details.

The dataset image size is 1000×750 pixels, so image resizing is required. We resized the image to 640×640 pixels according to the YOLOv5 input image. A geometry augmentation process was then performed on the training and validation data. This augmentation is necessary because the process of rotation and shear occurs when two images from two different cameras are combined in the image stitching process. For rotation, the matrix transformation is used to obtain new coordinates after rotating against the center of the image by θ degrees. The equation is:

$$R(\theta) = \begin{bmatrix} cos\,\theta & sin\,\theta & -cos\,\theta \cdot center\ x - sin\,\theta \cdot center\ y \\ -\,sin\,\theta & cos\,\theta & sin\,\theta \cdot center\ x - cos\,\theta \cdot center\ y \end{bmatrix} \qquad (1)$$

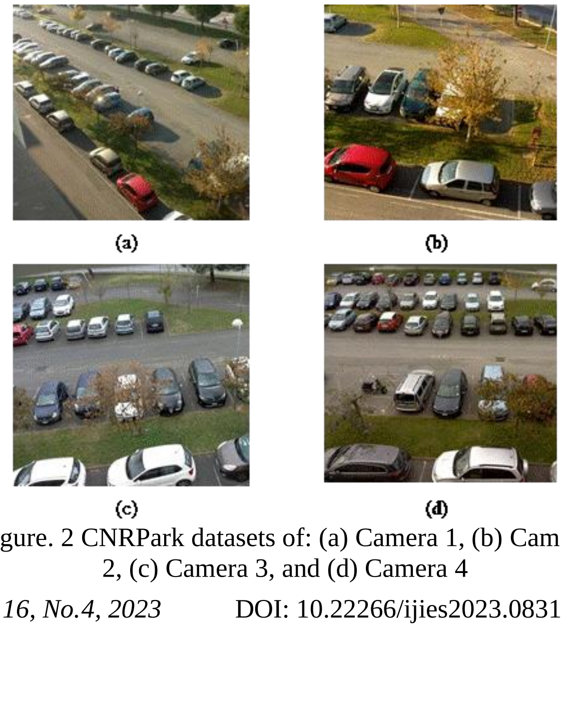


Figure. 2 CNRPark datasets of: (a) Camera 1, (b) Camera 2, (c) Camera 3, and (d) Camera 4

For Cameras 1 and 2, $\theta = \pm 40°$ and $\theta = \pm 20°$ for Cameras 3 and 4. Shearing turns the rectangular image into parallelogram image. For horizontal shearing (x-shearing), the transformation matrix is:

$$S(\theta) = \begin{bmatrix} 1 + cot\,\alpha \cdot cot\,\beta & cot\,\alpha \\ cot\,\beta & 1 \end{bmatrix} \quad (2)$$

Where α is the angle of shear parallel to the x-axis and β is the angle of shear parallel to the y-axis. We used $\alpha, \beta = \pm 45°$. As a result, the number of images from each camera has tripled compared to the original.

3.3 Image stitching methods

Image stitching is a technique that is useful for combining two or more images that have overlapping areas to produce a wider field of view (FOV). This technique is divided into two categories, which are pixel-based methods and feature-based methods. This feature-based method is widely used now because this method is faster than the pixel-based method. The advantages of this method are obtained by selectively extracting sparse feature points from all pixels in overlapping regions, becoming more robust by building different feature descriptors effectively, and calculating the closeness relationship between input images automatically [20]. In general, the feature-based method is divided into 3 stages: feature description, feature matching, and finding homography matrix.

Feature points are generally selected from any part of the overlapping area. Perhaps there are different overlapping areas from each other in the image due to camera shake or photographer movement (noise). Therefore, the selected feature point must be robust for these differences and prove that the feature point is the same area. The distinctive patterns exhibited by the discovered features' surrounding pixels are then used to explain them. Because each feature is described by being given a unique identity that facilitates efficient matching, this procedure is known as feature description. The feature description algorithm already includes a number of feature descriptors, such as oriented FAST and rotated BRIEF (ORB), binary robust invariant scalable keypoints (BRISK), and accelerated AKAZE (AKAZE).

The ORB feature description combines the FAST (features from accelerated segment test) keypoint detector and the BRIEF (binary robust independent elementary features) feature description. As a feature detector, ORB uses the center of intensity to calculate the image patch orientation. The degree of the vector of the centroid can be utilized to relate orientations because it is assumed that the angular intensity is displaced from its center [21]. All the keypoints discovered by FAST are combined by BRIEF into a binary feature vector, allowing them to collectively represent an object [22]. We use several parameters in computing, namely $nfeatures$ = 10000, $fastThreshold$ = 20, and $patchSize$ = 31. The $nfeatures$ shows the maximum number of features obtained. The $fastThreshold$ means the intensity threshold in the FAST algorithm and the $patchSize$ is the size of the image patch used by the oriented BRIEF descriptor.

Another feature detector is BRISK. The BRISK algorithm detects and identifies the characteristic direction of each feature to obtain scale invariants and rotation invariants. This feature detector works in three ways: keypoint detection, keypoint description, and descriptor matching [23]. The keypoint detection process is like AGAST (adaptive and generic accelerated segment test). Keypoints are described by building a space-scale pyramid to obtain a FAST score that matches the keypoint criteria. Using the grayscale connections of random pixel pairings throughout the image, the keypoint descriptor constructs the feature descriptor of the identified keypoint. For the last stage, descriptor matching is obtained by comparing the similarity between two feature point descriptors using the hamming distance [24]. We use AGAST detection threshold score equal to 30 and octave (space-scale pyramid) equal to 3.

Accelerated KAZE (AKAZE) is a feature descriptor algorithm that is a variation of the accelerated KAZE algorithm. AKAZE consists of 3 parts: calculating the contrast factor, constructing a nonlinear scale-space, and feature detection [25]. The contrast factor could be calculated by calculating the absolute gradient value of the gradient for each pixel, then multiplying it by the gradient histogram. Next, the nonlinear scale-space is created by solving the partial differential equation using the fast explicit diffusion scheme (FED). The contrast factor is used in the conductivity function in the partial differential equation. This study used the number of octaves equal to 4. The AKAZE feature detector uses the determinant of Hessian (DoH) blob detector by comparing it with neighboring windows measuring 3×3. If the pixel is larger than its eight neighbors, that pixel becomes the keypoint. AKAZE generates a feature descriptor for each of these keypoints [26].

After getting the feature description for each image, next is the feature matching procedure. Every point in one image is compared to every point in another, and matched points are identified. Feature matching requires a matching object. A commonly

Received: April 14, 2023. Revised: May 23, 2023.

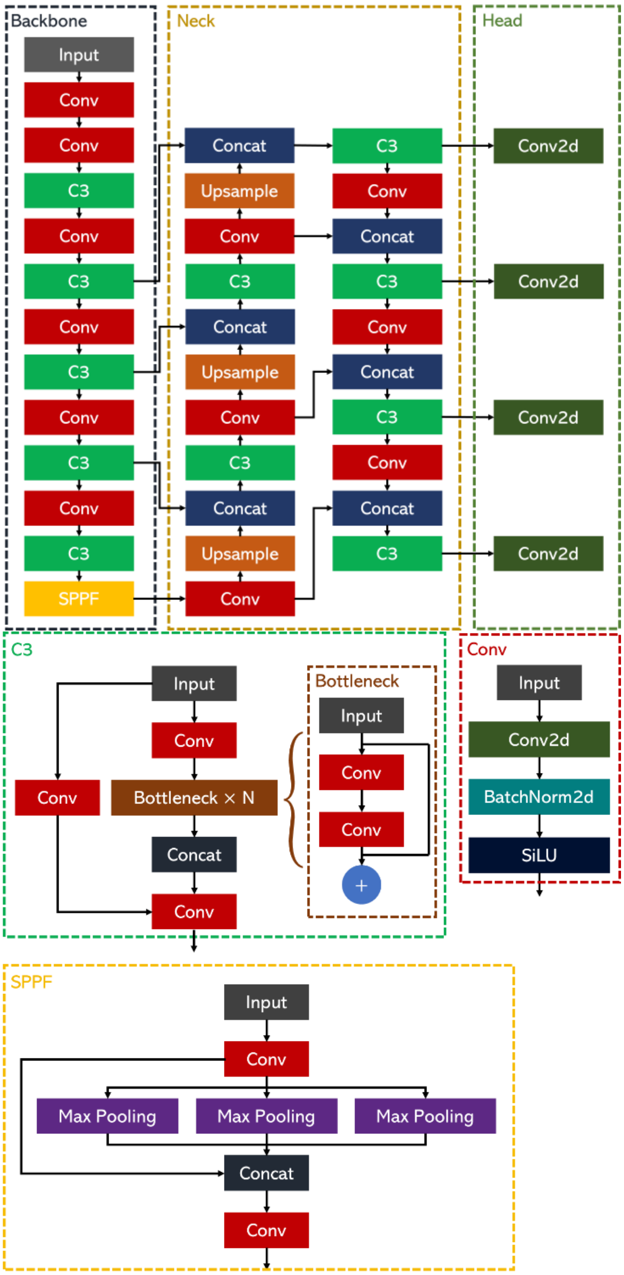


Figure. 3 YOLOv5 architecture

used feature matching algorithm is the brute force matcher (BF Matcher) [27]. The BF matcher investigates all possibilities and selects the best match. It compares the feature descriptor in one image to all the other features in the second image using the Hamming distance. Keypoints that match between images are used to represent the closest points [28].

The final step in the image stitching method is to find the homography matrix. This matrix is a reversible transformation from the real projection plane to the projection plane that maps straight lines to straight lines. One algorithm to compute the homography value is using the random sample consensus (RANSAC). RANSAC is a method of classifying each correspondence as an inlier or an outlier. The homography matrix could be calculated from all correspondences that are considered inliers [29]. Once the homography matrix is obtained correctly, each image is associated with it. Then use the homography matrix to get the stitched image (image stitching).

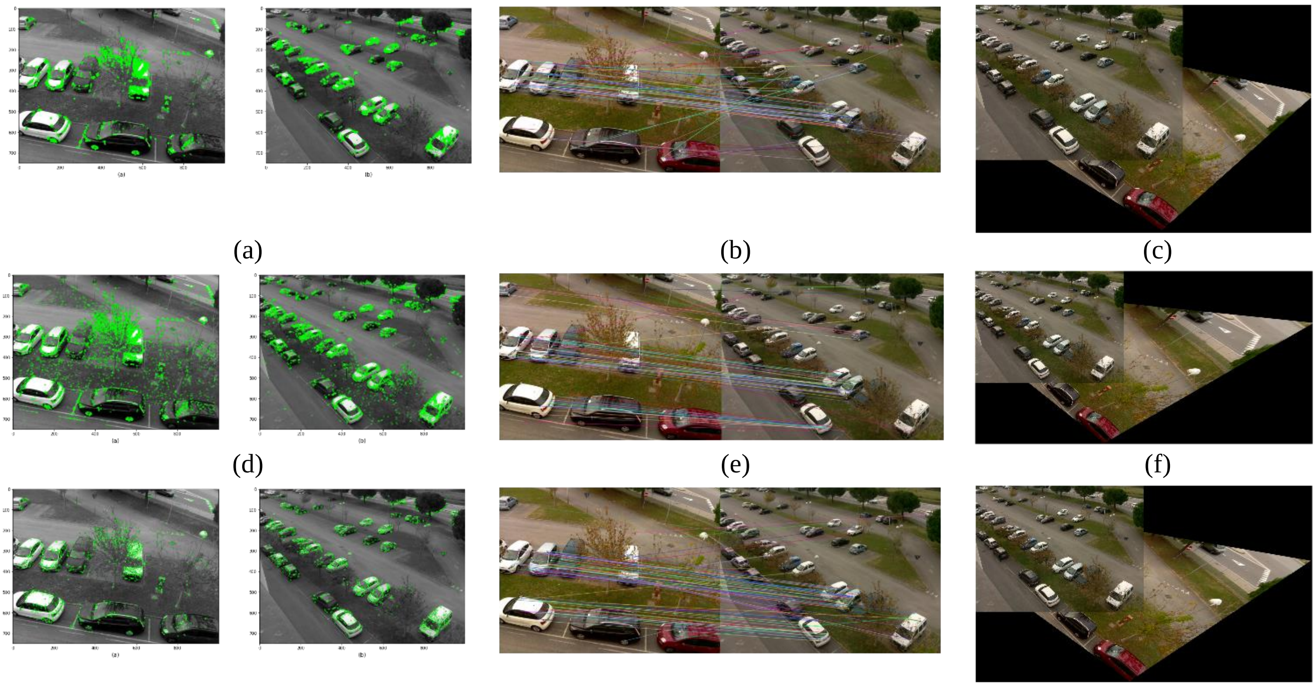
3.4 Training process using YOLOv5

The you only look once (YOLO) method is a method used in object detection problems. The YOLO algorithm was first introduced by [30]. In 2020, [31] released YOLOv5 which is a two-stage detector algorithm consisting of 3 main parts, which are backbone, neck, and head as shown in Fig. 3. The backbone is used to extract important features from a given input image. YOLOv5 implements cross stage partial (CSP) networks on the backbone. The CSP architecture (C3 layer) is used to overcome the gradient vanishing problem and reduce the number of training parameters. In this backbone, there is also spatial pyramid pooling (SPP) which is a pooling layer to remove fixed network size limitations. YOLOv5 implements path aggregation network (PANet) as neck to improve information flow. PANet employs a new feature pyramid network (FPN) with multiple bottom-up and top-down layers. The head section in YOLOv5 is the same as the previous version of YOLO which produces three different outputs for the detection process (predicting class and bounding box) [32–34].

In YOLOv5 version 6.0 there are several improvements, including (1) replacing the Focus layer with a Conv layer (kernel=6, stride=2, padding=2); (2) replacing SPPF with SPP layer to reduce FLOPS and increase speed; (3) reduction in Conv backbone layer; and (4) reorder places SPPF at the end of the backbone [31]. In addition, the number of layers and neuron parameters depends on the configuration/architecture. In this study, the authors used 6 different architectures as testing parameters, which are YOLOv5n6, YOLOv5s6, YOLOv5m6, YOLOv5l6, YOLOv5x6, and YOLOv5p7. A letter after YOLOv5 means a different type of configuration and the number '6' after it indicates an improved version of YOLOv5 (which is the 6.0 version), except p7. It represents a feature level with $1/2^i$ resolution of the input image as in the EfficientDet architecture [35]. The letter 'n' through 'x' represents nano, small, medium, large, and extra-large respectively. As the name suggests, YOLOv5n6 has fewer neurons and layers than YOLOv5s6, and so on. The difference between those five configurations with YOLOv5p7 is that YOLOv5p7 is free anchor-

Table 2. The results of the image stitching process in numbers

Feature Descriptor	Feature Detected		Feature Detecting Time (s)		Feature Matched	Feature Matching Time (s)	Outliers Rejected	Outlier Rejection and Homography Calculation Time (s)	Total Image Stitching Time (s)
	1st Image	2nd Image	1st Image	2nd Image					
Camera 1 and 2									
ORB	4990	4978	0.055	0.045	1170	0.301	1094	0.279	0.636
BRISK	4948	4489	0.459	0.319	1017	0.363	956	0.183	1.006
AKAZE	1553	1423	0.216	0.149	438	0.033	397	0.064	0.313
Camera 3 and 4									
ORB	9574	9268	0.081	0.064	2224	0.719	2146	0.324	1.125
BRISK	4738	2473	0.476	0.466	738	0.186	719	0.162	0.825
AKAZE	1006	861	0.205	0.187	280	0.057	267	0.137	0.399



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure. 4 The result of the image stitching process on Camera 1 and 2: (a), (d), (g) are features detected of ORB, BRISK, and AKAZE respectively, (b), (e), (h) are feature matched of ORB, BRISK, and AKAZE respectively, and (c), (f), (i) are homography calculations and stitched image results of ORB, BRISK, and AKAZE respectively

based, while the others require an anchor as a parameter.

3.5 Testing process

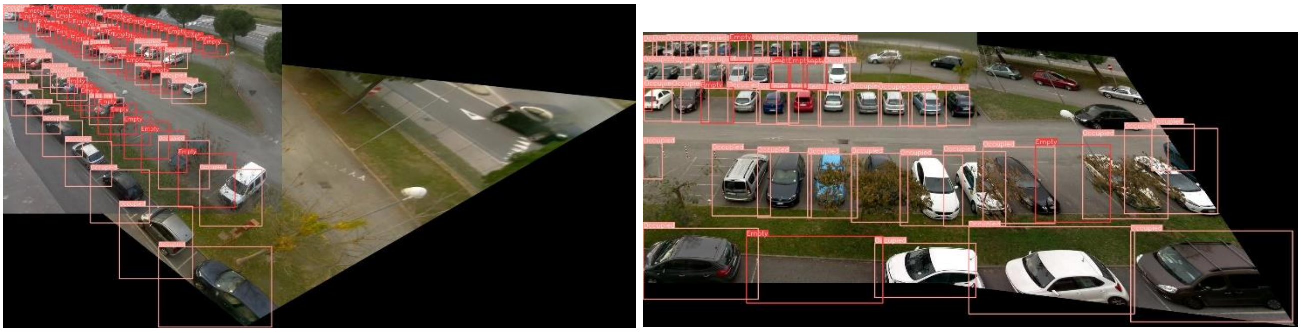
The testing process is performed after training the data using several YOLOv5 configurations and obtaining stitched images (test data). The augmented and non-augmented training and validation data were compared. Both are also compared with each model of the YOLOv5 configuration. The mean average precision (mAP) value is required to know which model is the best. How mAP works is that it computes the average precision (AP) for each label/object class (N) by first computing precision and recall. AP could be calculated using the equation:

$$AP = \sum_{i=0}^{i=n-1}(Recall(i) - Recall(i+1)) \cdot Precision(i) \quad (3)$$

where n is the threshold value, $Precision(n) = 1$, and $Recall(n) = 0$. The mAP score could be found by the following formula:

$$mAP = \frac{1}{N}\sum_{j=1}^{N} AP_j \quad (4)$$

N is the number of object classes and AP_j is the



(a) (b)

Figure. 5 The result of parking slot occupancy detection from two overlapping cameras on: (a) stitched image of Camera 1 and 2 and (b) stitched image of Camera 3 and 4

AP of class j. For object detection, the amount of intersection over union (IoU) is used to determine the threshold. IoU is calculated by dividing the intersection of the predicted and ground truth bounding boxes by their union. The IoU threshold greatly affects the calculation of the mAP value. This study used 0.5 as an IoU threshold for calculating the mAP, so the metric is called mAP0.5.

4. Results and discussions

4.1 Stitched image

The image stitching method is used to detect overlapping areas on a pair of Camera 1 and Camera 2 as well as Camera 3 and Camera 4 on the test data. To get good results from stitched images, we tested three feature detector algorithms in our experiments. The next stage of this method is feature matching and finding homography. We use the same technique for both processes, so this study only compares the performance of the three feature detectors. The results of stitched images can be compared quantitatively and qualitatively. Table 2 compares the performance of the three feature detector algorithms in numbers. Whereas Fig. 4 compares them visually.

In Table 2, this study compares two images that have overlapping areas in them. The first image is an image from Camera 1 (or Camera 4 in other pairs) which is the key image or query image. While the second image is Camera 2 (or Camera 3 in the other pair) which is a train image, the image that will be stitched to the query image. As a result, ORB could detect the most features compared to BRISK and AKAZE, but the time required is the fastest. If we look at Fig. 4, the features detected by ORB looked almost as numerous as BRISK, but they are more densely located. As the corner detections, all three feature detectors could detect corners of objects such as cars and trees. But BRISK also detects grass as a feature. Fig. 4 (b), (e), and (h) show the results of the BF Matcher algorithm in feature matching. The majority of the lines that connect from one feature to another one in both images could connect the corresponding features. Although there are several wrong connections between one feature and another due to differences in color grading. AKAZE's computation time is relatively the fastest compared to the others because it has the fewest matched features. Due to the large number of features detected in ORB and BRISK, there are also many features with outliers compared to AKAZE. Computational time is also directly proportional to the three feature detectors. Total image stitching time is the total average time required to stitch one pair of images which is the sum of the three processes of the image stitching method. The results could be seen in Fig. 4 (c), (f), and (i). Visually, AKAZE looks the most seamless in the views compared to the results from the other two feature detectors. But overall, the three feature detectors are able to create good stitched images on two camera pairs.

4.2 Parking space availability detection

An example of the results of detecting the availability of parking slots from two overlapping cameras could be seen in Fig. 5. The red bounding box indicates an empty parking slot, while the pink bounding box represents a parking slot occupied by a car. Each camera's training and validation data are used in the training process. There are two types of data: augmented (aug) data, which is synthetic data added by performing rotation and shear augmentations; and original (ori) data, which is original data without augmentation.

In this study's experiments, the results of testing (stitched images) on data with and without augmentation were compared. Model weights with different YOLOv5 configurations were also used in the testing procedure, as indicated in Table 3. With the exception of YOLOv5n6, the augmented dataset outperformed the original dataset in terms of precision, recall, and mAP0.5 when compared

Table 3. Comparing the result of parking lot availability detection from the stitched image of Camera 1 and Camera 2

YOLO Config	Feature Detector	Data	Precision	Recall	mAP0.5
YOLOv5n6	ORB	Ori	0.758	0.885	0.894
		Aug	0.712	0.874	0.832
	BRISK	Ori	0.748	0.877	0.888
		Aug	0.696	0.890	0.828
	AKAZE	Ori	0.774	0.891	0.884
		Aug	0.721	0.882	0.835
YOLOv5s6	ORB	Ori	0.830	0.842	0.918
		Aug	0.817	0.901	0.930
	BRISK	Ori	0.790	0.902	0.926
		Aug	0.823	0.917	0.934
	AKAZE	Ori	0.820	0.852	0.910
		Aug	0.828	0.921	0.932
YOLOv5m6	ORB	Ori	0.838	0.834	0.890
		Aug	0.823	0.916	0.943
	BRISK	Ori	0.829	0.793	0.911
		Aug	0.825	0.917	0.939
	AKAZE	Ori	0.842	0.850	0.918
		Aug	0.828	0.923	0.938
YOLOv5l6	ORB	Ori	0.840	0.866	0.916
		Aug	0.821	0.928	0.942
	BRISK	Ori	0.811	0.800	0.908
		Aug	0.825	0.920	0.944
	AKAZE	Ori	0.829	0.841	0.912
		Aug	0.833	**0.933**	0.941
YOLOv5x6	ORB	Ori	0.845	0.890	0.919
		Aug	0.879	0.909	0.945
	BRISK	Ori	0.815	0.800	0.910
		Aug	0.862	0.904	0.946
	AKAZE	Ori	0.844	0.817	0.916
		Aug	**0.890**	0.902	0.947
YOLOv5p7	ORB	Ori	0.847	0.873	0.926
		Aug	0.849	0.914	0.949
	BRISK	Ori	0.837	0.834	0.923
		Aug	0.864	0.912	**0.953**
	AKAZE	Ori	0.851	0.847	0.914
		Aug	0.856	0.922	0.949

Table 4. Comparing the result of parking lot availability detection from the stitched image of Camera 3 and Camera 4

YOLO Config	Feature Detector	Data	Precision	Recall	mAP0.5
YOLOv5n6	ORB	Ori	0.325	0.180	0.193
		Aug	0.511	0.616	0.519
	BRISK	Ori	0.350	0.218	0.207
		Aug	0.471	0.680	0.478
	AKAZE	Ori	0.273	0.138	0.142
		Aug	0.360	0.456	0.303
YOLOv5s6	ORB	Ori	0.358	0.161	0.169
		Aug	0.532	0.599	0.555
	BRISK	Ori	0.349	0.170	0.168
		Aug	0.520	0.611	0.537
	AKAZE	Ori	0.326	0.137	0.140
		Aug	0.409	0.451	0.357
YOLOv5m6	ORB	Ori	0.596	0.334	0.447
		Aug	0.881	0.850	0.906
	BRISK	Ori	0.614	0.370	0.468
		Aug	0.873	0.866	0.902
	AKAZE	Ori	0.537	0.243	0.324
		Aug	0.784	0.746	0.792
YOLOv5l6	ORB	Ori	0.880	0.703	0.781
		Aug	**0.884**	0.845	0.908
	BRISK	Ori	0.862	0.774	0.828
		Aug	0.881	0.870	0.926
	AKAZE	Ori	0.799	0.615	0.685
		Aug	0.873	0.806	0.877
YOLOv5x6	ORB	Ori	0.854	0.685	0.749
		Aug	0.800	0.817	0.883
	BRISK	Ori	0.866	0.701	0.774
		Aug	0.858	0.828	0.895
	AKAZE	Ori	0.722	0.555	0.619
		Aug	0.796	0.731	0.826
YOLOv5p7	ORB	Ori	0.848	0.780	0.792
		Aug	0.874	0.914	**0.942**
	BRISK	Ori	0.840	0.765	0.804
		Aug	0.862	**0.916**	0.934
	AKAZE	Ori	0.806	0.693	0.741
		Aug	0.872	0.863	0.907

without the augmentation data. Based on evaluation metric scores on the original data, this model detected more accurately in YOLOv5n6, although the value is still very low when compared to other YOLO configurations. The more characteristics used while comparing different YOLO configurations, the better the outcomes. At BRISK using augmented data, YOLOv5p7 had the best performance with a mAP0.5 of 0.953. Without the need for prior knowledge of the typical sizes and shapes of objects in the data set, YOLOv5p7's lack of anchor boxes enables the model to better adapt to a wide range of object sizes and aspect ratios. As a result, the architecture is more adaptable and effective because the model does not have to be trained on how to change the anchor box. The three feature detectors produced evaluation metrics for the Camera 1 and Camera 2 pair that were nearly the same in each YOLO configuration.

Table 5. Counting overlapping parking slot

Camera	Total Slots	Overlapping Parking Slots	Ground Truth
Cam 1	108	9	9
Cam 2	11		
Stitched of Cam 1 and 2	110		
Cam 3	26	16	16
Cam 4	43		
Stitched of Cam 3 and 4	53		

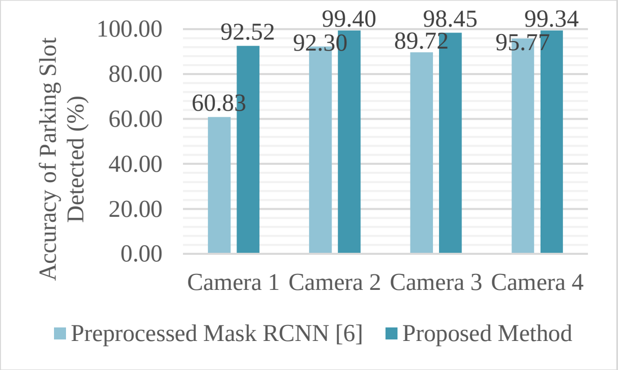


Figure. 6 Accuracy of parking slot comparison on four cameras

Nevertheless, BRISK generated the best mAP0.5. This is due to the fact that, when compared to the other two feature detectors, BRISK created stitched images with the highest width-to-height ratio. The model might recognize long and narrow objects like cars and trees more precisely if the width-to-height ratio were bigger [36].

This experiment also analyzes the results of images that were stitched on Cameras 3 and 4, as shown in Table 4. In all YOLO setups and the three feature detectors, it was obvious that augmented data provides better mAP0.5 than unaugmented data. This is due to the angle area that overlaps on Camera 3 (train image) being sufficiently sheared, indicating the requirement for shear augmentation to improve the model's ability to learn. Except for YOLOv5x, where it turns out that the mAP0.5 value is less than YOLOv5l6, the more YOLO parameters, the better the mAP0.5 value, yet YOLOv5p7 yields the highest score. This demonstrates that the YOLOv5p7 model, which doesn't have anchor boxes, worked the best in this situation. While comparing the performance of the feature detectors, ORB outperformed the other two feature detectors in terms of the mAP0.5 score. This is due to the fact that ORB could detect the most features in overlapping areas, making the output more seamless. On the other side, BRISK's recall is the best, meaning that BRISK has a low false negative rate and it means that BRISK is good at recognizing objects in parking spaces that are either empty or occupied by cars.

4.3 Overlapping parking slot counting

This study also tested each individual camera as a comparison with previous studies. From each of the four cameras used as a dataset, we look for the accuracy of the detected parking slots and compare them with the preprocessed mask RCNN method [6]. The results are shown in Fig. 6. Overall, the parking slot accuracy detected using YOLOv5 (proposed method) outperforms the Preprocessed Mask RCNN method. The parking slot accuracy detected on Camera 1 is slightly lower compared to other cameras because there are the greatest number of parking slots. In addition, the RCNN Mask preprocessing method is used to overcome several cars that are not detected when covered with trees. On the other hand, the YOLOv5 method can also detect empty or filled parking slots behind trees.

Each camera tested was also used to measure the number of parking slots in overlapping areas compared to the actual number. In this study, the total number of slots in both cameras before and after stitching is compared to the average count of total slot. The best mAP0.5 score from each camera pair was used for the comparison. Parking spaces that were detected as overlapping areas are called overlapping slots and the actual number of parking slots are called ground truth. For instance, we may be determined how many overlapping parking slots in the two cameras by calculating the average total slots on Camera 1, Camera 2, and the stitched images of both. Overlapping slots (x) can be calculated by the formula:

$$x = Total_{cam1} + Total_{cam2} - Total_{stitched} \quad (5)$$

So that the outcomes match those in Table 5. The pairs of Cameras 3 and 4 are likewise affected by this. The difference between the number of overlapping parking slots and the ground truth is zero for both pairs of stitched cameras. This experiment showed that the system could recognize the overlapping parking slot area precisely.

5. Conclusion

This study has succeeded in detecting overlapping areas on two overlapping cameras using the image stitching method. The result of this process is the stitched images from the pairs of two cameras. The three feature detectors can detect areas of overlap and stitch image pairs with seamless stitched image results. The next objective is to detect empty or occupied parking slots in the stitched image using the YOLOv5 method with various configurations. The configurations could affect the value of mAP0.5. Due to the results of the stitched image depending on the angle of the camera placement, rotation and shear augmentation are required on the training and validation data. As a result, the augmented data obtained a higher mAP0.5 score than the unaugmented data. In addition, the YOLOv5p7 configuration which did not require an anchor got the best mAP0.5 score on both camera pairs. However, the performance of the feature detector to obtain the highest mAP0.5 value is different. In the stitched image of Camera 1 and Camera 2, BRISK is better than ORB and AKAZE. Meanwhile, for Camera 3 and Camera 4 pair, the best mAP0.5 value was obtained by ORB. This experiment obtained higher accuracies of parking slots detected than the previous research on four tested cameras. Each of the four tested cameras is used to calculate the total overlapping parking slots on the two camera pairs. As a result, this study precisely calculates the number of overlapping parking slots compared to the actual number.

This study stitched two images, thus in future studies the number of images that can be stitched could be more than two cameras simultaneously. Moreover, this study used one parking space location, therefore it should add more datasets from different parking space locations in future works.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

As the first author, Misbachul Falach Asy'ari contributed to the preparation of the manuscript, the formulation of the methodology, the application of the method, and the implementation of the experiment. The second author, Chastine Fatichah, contributed to supervising the application of the proposed method and the preparation of the articles. The third author, Nanik Suciati, contributed to supervising the implementation of the experiment and drafting the manuscript.

Acknowledgments

The authors would like to express our sincere gratitude to all those who have contributed to the completion of this research project, especially the Institut Teknologi Sepuluh Nopember which sponsored the first author's Fresh Graduate scholarship and the Department of Informatics, Institut Teknologi Sepuluh Nopember which has funded this research.

References

[1] R. Yusnita, F. Norbaya, and N. Basharuddin, “Intelligent Parking Space Detection System Based on Image Processing”, *International Journal of Innovation, Management and Technology*, Vol. 3, No. 3, 2012.

[2] Y. Shao, P. Chen, and T. Cao, “A Grid Projection Method Based on Ultrasonic Sensor for Parking Space Detection”, In: *Proc. of IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3378–3381, 2018.

[3] C. Ren, S. Lee, D. Jeong, H. Chen, and Y. Xiao, “Parking Guidance System Based on Geomagnetic Sensors and Recurrent Neural Networks”, *J. Sens.*, Vol. 2022, p. 7481064, 2022, doi: 10.1155/2022/7481064.

[4] X. Ding and R. Yang, “Vehicle and Parking Space Detection Based on Improved YOLO Network Model”, *Journal of Physics: Conference Series*, Institute of Physics Publishing, Nov. 2019. doi: 10.1088/1742-6596/1325/1/012084.

[5] J. Cho, J. Park, U. Baek, D. Hwang, S. Choi, S. Kim, and K. Kim, “Automatic parking system using background subtraction with CCTV environment international conference on control, automation and systems (ICCAS 2016)”, In: *Proc. of International Conference on Control, Automation and Systems*, IEEE Computer Society, Jan. 2016, pp. 1649–1652. doi: 10.1109/ICCAS.2016.7832520.

[6] A. A. Naufal, C. Fatichah, and N. Suciati, “Preprocessed Mask RCNN for Parking Space Detection in Smart Parking Systems”, *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 6, pp. 255–265, 2020, doi: 10.22266/ijies2020.1231.23.

[7] E. Tanuwijaya and C. Fatichah, “Modification of AlexNet Architecture for Detection of Car Parking Availability in Video CCTV”, *Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)*, Vol. 13, No. 2, pp. 47–55, 2020.

[8] U. Nepal and H. Eslamiat, “Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs”, *Sensors*, Vol. 22, No. 2, Jan. 2022, doi: 10.3390/s22020464.

[9] J. A. Kim, J. Y. Sung, and S. H. Park, “Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition”, In: *Proc. of 2020 IEEE International Conference on Consumer Electronics - Asia, ICCE-Asia 2020*, Institute of Electrical and Electronics Engineers Inc., 2020. doi: 10.1109/ICCE-Asia49877.2020.9277040.

[10] P. Almeida, L. S. Oliveira, A. S. Britto, E. J. Silva, and A. Koerich, “PKLot-A Robust Dataset for Parking Lot Classication”, *Expert Systems with Applications*, Vol. 42, No. 11, pp. 4937-4949, 2015, doi: 10.1016/j.eswa.2015.02.009.

[11] A. Farley, H. Ham, and Hendra, “Real Time IP Camera Parking Occupancy Detection using Deep Learning”, *Procedia Computer Science*, Elsevier B.V., pp. 606–614, 2021, doi: 10.1016/j.procs.2021.01.046.

[12] R. M. Nieto, Á. G. Martín, A. G. Hauptmann, and J. M. Martínez, “Automatic Vacant Parking Places Management System Using Multicamera Vehicle Detection”, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 20, No. 3, pp. 1069–1080, 2019, doi: 10.1109/TITS.2018.2838128.

[13] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and C. Vairo, “Car Parking Occupancy Detection Using Smart Camera Networks and Deep Learning”, In: *Proc. of 2016 IEEE Symposium on Computers and Communication (ISCC)*, IEEE, 2016.

[14] Z. Bai, Y. Li, X. Chen, T. Yi, W. Wei, M. Wozniak, and R. Damasevicius, “Real-time video stitching for mine surveillance using a hybrid image registration method”, *Electronics (Switzerland)*, Vol. 9, No. 9, pp. 1–18, Sep. 2020, doi: 10.3390/electronics9091336.

[15] S. A. K. Tareen and Z. Saleem, “A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK”, In: *Proc. of 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pp. 1–10, 2018, doi: 10.1109/ICOMET.2018.8346440.

[16] S. Y. Siddiqui, M. A. Khan, S. Abbas, and F. Khan, “Smart occupancy detection for road traffic parking using deep extreme learning machine”, *Journal of King Saud University - Computer and Information Sciences*, Vol. 34, No. 3, pp. 727-733, 2020, doi: 10.1016/j.jksuci.2020.01.016.

[17] G. Khan, M. A. Farooq, Z. Tariq, and M. U. G. Khan, “Deep-Learning Based Vehicle Count and FreeParking Slot Detection System”, In: *Proc. of 22nd International Multitopic Conference (INMIC)*, 2019.

[18] Z. Zhang, L. Wang, W. Zheng, L. Yin, R. Hu, and B. Yang, “Endoscope image mosaic based on pyramid ORB”, *Biomed Signal Process Control*, Vol. 71, p. 103261, 2022, doi: https://doi.org/10.1016/j.bspc.2021.103261.

[19] S. H. Ro and S. H. Kim, “An image stitching algorithm for the mineralogical analysis”, *Miner Eng*, Vol. 169, p. 106968, 2021, doi: https://doi.org/10.1016/j.mineng.2021.106968.

[20] W. Lyu, Z. Zhou, L. Chen, and Y. Zhou, “A survey on image and video stitching”, *Virtual Reality & Intelligent Hardware*, Vol. 1, No. 1, pp. 55–83, 2019, doi: https://doi.org/10.3724/SP.J.2096-5796.2018.0008.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: an efficient alternative to SIFT or SURF”, In: *Proc of the IEEE International Conference on Computer Vision*, May 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.

[22] T. Rao and T. Ikenaga, “Quadrant segmentation and ring-like searching based FPGA implementation of ORB matching system for Full-HD video”, In: *Proc. of 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, 2017, pp. 89–92. doi: 10.23919/MVA.2017.7986797.

[23] Y. Liu, H. Zhang, H. Guo, and N. N. Xiong, “A FAST-BRISK feature detector with depth information”, *Sensors (Switzerland)*, Vol. 18, No. 11, Nov. 2018, doi: 10.3390/s18113908.

[24] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary Robust invariant scalable keypoints”, In: *Proc. of 2011 International Conference on Computer Vision*, 2011, pp. 2548–2555. doi: 10.1109/ICCV.2011.6126542.

[25] S. K. Sharma and K. Jain, “Image Stitching using AKAZE Features”, *Journal of the Indian Society of Remote Sensing*, Vol. 48, No. 10, pp. 1389–1401, 2020, doi: 10.1007/s12524-020-01163-y.

[26] L. Kalms, K. Mohamed, and D. Göhringer, “Accelerated Embedded AKAZE Feature Detection Algorithm on FPGA”, In: *Proc. of the 8th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies*, in HEART2017. New York, NY, USA: Association for Computing Machinery, 2017. doi: 10.1145/3120895.3120898.

[27] S. A. Bakar, X. Jiang, X. Gui, G. Li, and Z. Li, “Image Stitching for Chest Digital Radiography Using the SIFT and SURF Feature Extraction by RANSAC Algorithm”, *Journal of Physics: Conference Series*, IOP Publishing Ltd, 2020, doi: 10.1088/1742-6596/1624/4/042023.

[28] M. D. Lakshmi, P. Mirunalini, R. Priyadharsini, and T. T. Mirnalinee, “Review of feature extraction and matching methods for drone image stitching”, *Lecture Notes in Computational Vision and Biomechanics*, Springer Netherlands, 2019, pp. 595–602. doi: 10.1007/978-3-030-00665-5_59.

[29] Y. Wu, X. Su, and X. Hu, “Image Stitching Based on ORB Feature and RANSAC”, *ICIC Express Letters Part B: Applications ICIC International c*, Vol. 7, No. 7, pp. 1397–1403, 2016.

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection”, In: *Proc of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* IEEE Computer Society, Dec. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[31] G. Jocher, “ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements”, *Zenodo*, 2020, doi: 10.5281/ZENODO.4154370.

[32]V. M. Krešňáková, A. Kundrát, Š. Mackovjak, P. Butka, S. Jaščur, I. Kolmašová, and O. Santolík, “Automatic Detection of Atmospherics and Tweek Atmospherics in Radio Spectrograms Based on a Deep Learning Approach”, *Earth and Space Science*, Vol. 8, No. 11, p. e2021EA002007, 2021, doi: https://doi.org/10.1029/2021EA002007.

[33] M. Antonakakis, A. Tzavaras, K. Tsakos, E. G. Spanakis, V. Sakkalis, M. Zervakis, and E. G. M. Petrakis, “Real-Time Object Detection using an Ultra-High-Resolution Camera on Embedded Systems”, In: *Proc. of 2022 IEEE International Conference on Imaging Systems and Techniques (IST)*, pp. 1–6, 2022, doi: 10.1109/IST55454.2022.9827742.

[34] L. Zhu, X. Geng, Z. Li, and C. Liu, “Improving YOLOv5 with Attention Mechanism for Detecting Boulders from Planetary Images”, *Remote Sens (Basel)*, Vol. 13, No. 18, 2021, doi: 10.3390/rs13183776.

[35] M. Tan, R. Pang, and Q. V. Le, “EfficientDet: Scalable and Efficient Object Detection”, *CoRR*, Vol. abs/1911.09070, 2019, [Online]. Available: http://arxiv.org/abs/1911.09070

[36] P. Y. Simard, D. Steinkraus, and J. C. Platt, “Best practices for convolutional neural networks applied to visual document analysis”, In: *Proc. of Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pp. 958–963, 2003, doi: 10.1109/ICDAR.2003.1227801.