TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 18, No. 4, August 2020, pp. 1942~1953
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i4.15009

Object detection for KRSBI robot soccer using PeleeNet on omnidirectional camera

Winarno[1], Ali Suryaperdana Agoes[2], Eva Inaiyah Agustin[3], Deny Arifianto[4]
[1]Faculty of Science and Technology, Universitas Airlangga, Indonesia
[2]Medical Robotic Laboratory, Universitas Airlangga, Indonesia
[3,4]Faculty of Vocational, Universitas Airlangga, Indonesia

Article Info

Article history:

Received Dec 18, 2019
Revised Mar 25, 2020
Accepted Apr 7, 2020

Keywords:

Deep learning
Object detection
Robot soccer

ABSTRACT

Kontes Robot Sepak Bola Indonesia (KRSBI) is an annual event for contestants to compete their design and robot engineering in the field of robot soccer. Each contestant tries to win the match by scoring a goal toward the opponent's goal. In order to score a goal, the robot needs to find the ball, locate the goal, then kick the ball toward goal. We employed an omnidirectional vision camera as a visual sensor for a robot to perceive the object's information. We calibrated streaming images from the camera to remove the mirror distortion. Furthermore, we deployed PeleeNet as our deep learning model for object detection. We fine-tuned PeleeNet on our dataset generated from our image collection. Our experiment result showed PeleeNet had the potential for deep learning mobile platform in KRSBI as the object detection architecture. It had a perfect combination of memory efficiency, speed and accuracy.

Corresponding Author:

Deny Arifianto,
Department of Engineering, Faculty of Vocational,
Universitas Airlangga,
Dharmawangsa Dalam Street 28-30 (Kampus B), Surabaya 60286, Indonesia.
+6231-5033869
Email: deny-a@vokasi.unair.ac.id

1. INTRODUCTION

Kontes Robot Sepak Bola Indonesia (KRSBI) is held annually by The Ministry of Research, Technology, and Higher Education of the Republic of Indonesia (KEMENRISTEKDIKTI) starting from 2017. KRSBI is part of the Kontes Robot Indonesia (KRI) for the enthusiastic to compete their design and engineering in the field of robot soccer. KRSBI participants are institutions or teams from state and private Universities under KEMENRISTEKDIKTI, which sends four Diploma and/or Bachelor and/or Postgraduate students. Each team prepares three autonomous wheeled robots consisting of one goalkeeper and two striker robots. Although KRSBI's has smaller field size to accommodate the Indonesian participant's circumstances, the robot size is in accordance with the middle size league (MSL) rule [1]. The robots that have been built will later compete against team opponent without any human control interference. The winner of soccer matches is determined by the number of balls that have been successfully scored into the opponent's goal. The robot task to find and locate the ball will gradually increase, as the current rule is using orange or yellow colored ball. Eventually, the color cues will be removed with the use of real soccer match ball [2].

KRSBI robot uses an omnidirectional vision camera or in short omnidirectional camera as part of its sensor. The omnidirectional camera is a camera that utilizes a hyperbolic mirror as a light collector from

the robot surrounding environment. The advantage of omnidirectional camera is its ability to imaging 360° per frame [3]. Thus, robot able to quickly detect ball position required for robot movement planning. The 360° imaging without the need for circular robot's or camera's movement, this is beneficial in the terms of information processing and in terms of power consumption. The omnidirectional camera offers a richer perception for robots to render objects in order to decide its movement during the soccer match. However, these advantages accompanied by a large amount distortion caused by the parabolic mirror gather 360° surrounding into a rectangular sensor. The distorted projection resulting the spatial dimension no longer similar compared with the normal lens image [4].

We divide ball detection hardware approaches in recent years into the use of a normal lens-based camera and omnidirectional based camera. First, we convey the approaches using a normal lens camera. Ball detection carried out by [2] performed in the absence of color features. They utilize a classifier that trained with 4 square features in the input image. Training process conducted on images that are converted into grayscale images. Another method proposed by [5], they performed ball detection using the curvature (arc) feature extracted from grayscale images. The extracted features were obtained through the difference pixel intensity. From here, the acquired edges will be strengthened, so only the curve edge processed. In [6], uses the Aibo camera which provides YUV images. They use edge detection algorithms that take into account the contrast patterns and color classes around. While [7] use a blob detection method based on the nearest neighbor component to detect the ball.

Secondly, we convey the deployment of an omnidirectional camera for ball detection. In recent years omnidirectional camera utilization is increasing in the robot researcher community. In [8], utilized omnidirectional camera in obtaining environmental information. In their report, they mention the use two-color variations to obtain that likely resemble contour points of a ball. The method they did was comparing two outcomes contour from rotary and radial scan. Another example is in [9], they used an omnidirectional camera to further processed based on color features to distinguish the ball from other objects. All the approaches stated before as in normal and omnidirectional camera used the handcrafted features. The use of handcrafted feature is often followed by the risk of ambient lighting fluctuation, the object is partially covered by shadows, and object's appearance is blocked by another object. These hindrances are not desirable during the robot soccer match.

The recent success of Alexnet [10] spurs the convolutional neural network (CNN) utilization for image classification, object detection and semantic segmentation in intelligent transportation system, robot vision and medical application. The robustness of deep learning feature extraction method that overcomes most problems occur by handcrafted feature attract many robot vision researchers. Ball detection using CNN has done by several groups, such as in [11]. They use CNN to detect ball from images captured by the camera. Their CNN model consists of four convolutional layers and followed by two fully connected layers. The output from CNN is a plot that can predict where the ball is. This ball's position obtained from a mapping possibility graph of a pixel where the ball located. This method resembles image segmentation in the fashion of pixel prediction. However, it gives a low-level accuracy. Another work as in [12], they present a dataset to train CNN. They reported the result in a simple CNN network to detect balls. However, the dataset is designated in the use for bipedal robot soccer, these data contain patches with ball and patches without ball. The information extracted from the dataset will not provide other than the ball visual feature representation.

In this paper, we present a method for overcoming the shortcomings of omnidirectional camera application and KRSBI's object detection by handcrafted feature. The limited computation resource on robot soccer will confine the deep learning model choice. Especially model deployed for any real-time vision application. Therefore, we describe our approach to tackle the object detection task by a deep learning method using fine-tuned PeleeNet [13]. PeleeNet is a deep learning architecture designed specifically for mobile devices that have a balance for its performance and its computational cost. Together with a carefully designed dataset that reflect the object's visual feature needed for the robot vision in a robot soccer match. We train the network using images that we collect separately. We show that the utilization 360° perception camera combined with CNN's trained on the normal images is feasible to be implemented in the robotic vision for KRSBI's object detection.

2. PROPOSED METHOD

The KRSBI's object detection problem is correlated closely with the development of computer vision branch named object detection. In the early development of object detection, researchers tended to treat it as a repetitive task of object classification, by imposing sliding windows and performing object classification with the neural networks in the window's region. The development of CNN based object detection triggers an approach to cut the computational cost of sliding windows. Instead of spending too

much resource for calculating the classification in every sliding window, people tried to find a better candidate location of sliding windows. As a result, region proposal along with CNN based object detection introduced in [14]. Later on [15], the CNN flow had been reworked, the 2000 times feedforward for each image in the previous approach, is no longer used. Rather, the image passes through the model one time and generate multiple region of interest (ROI). While on the other spectrum of deep learning-based object detection. A single shot detector proposed by [16], they employed a single architecture model for training and inference. An end to end training is possible to be done because each of features maps consists of class and location confident level. Recently, PeleeNet [13] had been proposed, it has a compact memory size and outperformed the previous approach. We employ the PeleeNet as our CNN object detection with the transfer learning method to detect ball, goal, center circle, robot cyan, and robot magenta. These five classes are our main interest in robot navigation during the match, the dataset will be described in subsection 2.2.

2.1. PeleeNet architecture

An amount of efficient architectures has been proposed in recent years, for example, SSD, MobileSSD, and PeleeNet. PeleeNet is a variant of DenseNet. It follows the connectivity pattern and some of key design principles of DenseNet. It was designed to meet strict constraints on memory and computational budget of mobile devices. PeleeNet has some key features such as: two-way dense layer, stem block, dynamic number of channels in bottleneck layer, transition layer without compression, and composite function. The first two approaches in the PeleeNet features were composed to increase the richness of visual patterns recognized by the model. The two-way dense layer emits two channels feature detectors with double 3x3 kernel, preceded by 1x1 filter. The key contribution of PeleeNet's accuracy, speed and model size is achieved by:

- Feature map selection: Dissimilar with SSD, their object detection network does not contain 38x38 filter. PeleeNet avoid this, in order to reduce computational cost.
- Residual prediction block: The residual prediction block (ResBlock) that consist of 2 stream convolutional filter. The first stream is comprised of 1x1x128, 3x3x128, and 1x1x256 convolutional filter. The second stream is 1x1x256 convolutional filter. The usage of ResBlock right after each 5 scale feature maps is to pass features into prediction layer.
- Small convolutional kernel for prediction: Further PeleeNet parameter number reduction is achieved by deployment of 1x1 convolutional kernel to predict category scores and box deployment instead of 3x3 kernel.

The PeleeNet architecture consists of four stages of feature extractor, instead of using three stage model. They avoid one stage parameter reduction during the feature extraction step, to keep the object representation. The PeleeNet model description is shown as follows in Table 1.

Table 1. Overview of PeleeNet architecture

Stage	Block	Layer	Output Shape
	Input		224x224x3
Stage 0	Stem block		56x56x32
Stage 1	Dense block	Dense layer x3	
	Transition layer	1x1 Conv, stride 1	
		2x2 Average pool, stride 2	28x28x128
Stage 2	Dense block	Dense layer x4	
	Transition layer	1x1 Conv, stride 1	
		2x2 Average pool, stride 2	14x14x256
Stage 3	Dense block	Dense layer x8	
	Transition layer	1x1 Conv, stride 1	
		2x2 Average pool, stride 2	7x7x512
Stage 4	Dense block	Dense layer x6	
	Transition layer	1x1 Conv, stride 1	
		2x2 Average pool, stride 2	7x7x704
	Classification layer	7x7 Global average pool	
		1000D Fully-con, softmax	1x1x704

2.2. Dataset description

We present our dataset in the direction of deep learning model development. It comprises of the component involved in the robot soccer match in particular the KRSBI. Therefore, the data is suitable for CNN based computer vision algorithms for object detection intended to be deployed in the robot. We then assume the future trained weight deployment is not limited to the omnidirectional camera. Therefore, our

data consists of normal images acquired with a mobile phone's camera during the KRSBI's regional preliminary round and national final round. The images are recorded from the side and the back of the field, illuminated by robot soccer field lighting. Consequently, any stage lighting nearby is counted in. We argue this condition is the best possible way to mimic the robot soccer match lighting environment condition. However, the competition rule for stage lighting is subject to change year after year.

We ensure the data set is suitable for training, validation and testing the architecture while maximizing possibility for future development. Our dataset consists of five objects and one background. The class's labels are "Ball", "Goal_post", "Center_circle", "Robot_cyan", and "Robot_magenta". We mark the object inside each image using LabelImg [17]. Human input is needed to operate LabelImg to generate the corresponding object's bounding boxes. During the ground truth generation, each image is displayed individually. Human expert then manually segments each class that occurred in the image within the bounding box coincide with the outermost limit of the object. Each image information is saved as .xml format, we follow the annotation rule by the PASCAL VOC dataset convention. The annotation file contains image's resolution information, image's name, image's path, and objects inside the image with the corresponding x_{min} y_{min} x_{max} y_{max}. Figure 1 shows annotated images taken from our dataset, the square markings are the ground truth bounding box consistent with the object's perimeter.

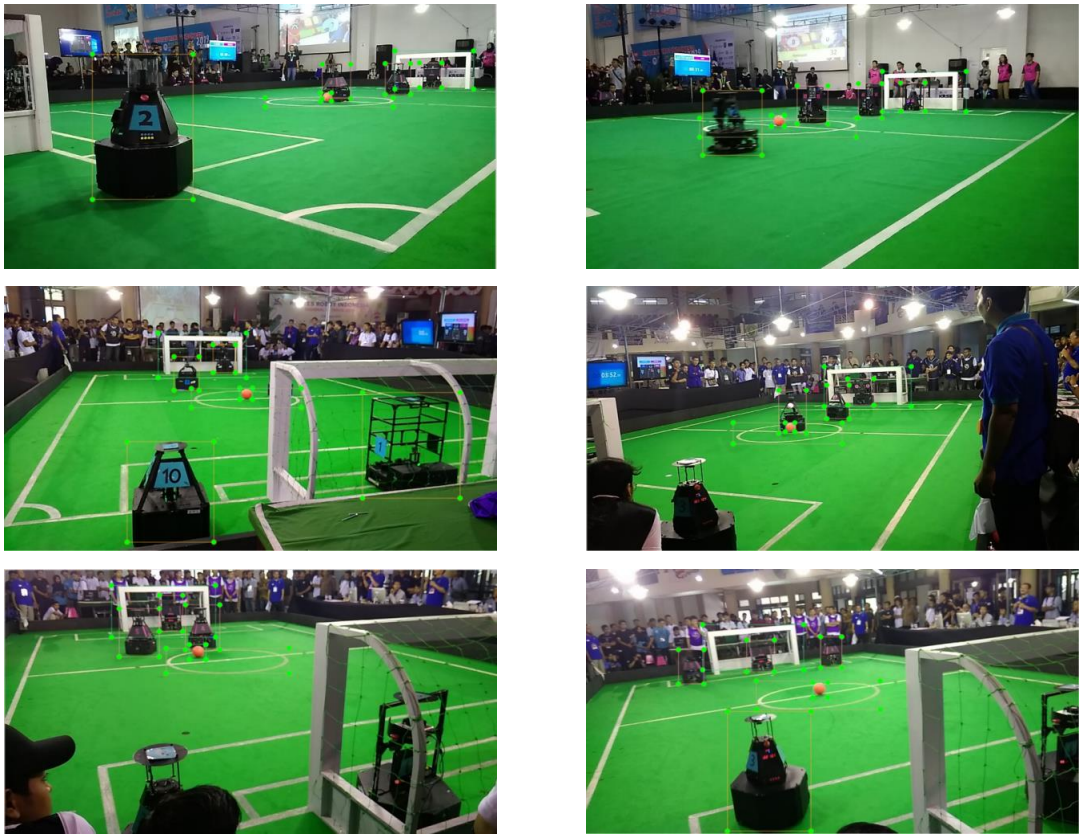

Figure 1. The annotated images show boundary boxes correspond with each class's label. Samples show images taken from a different perspective

Our data set comprises of 1280x720 500 RGB .jpg formatted images. Whereas "Ball" object is found on 387 images, "Goal_post" is located on 409 images, "Center_circle" on 457 images, "Robot_cyan" on 411 images, and "Robot_magenta" is found on 452 images. We then split the data in such that 70% is used for training, 15% is used for validation, and 15% for testing. Therefore, a total of 425 images are used for training and validation, and 75 images are intended for testing purposes.

2.3. Omnidirectional vision camera calibration

A 360° view of the robot's environment surrounding is provided in a single image by omnidirectional camera. Figure 2 (a) shows the omnidirectional camera, it usually combines a convex mirror, focusing lens, aperture, and a light sensor aligned on an axis. The convex mirror deployed in omnidirectional

camera ranging from conic, spherical, parabolic, or hyperbolic [18–20]. Placement of the light sensor is pointing to mirror at the end of omnidirectional camera system. Figure 2 (b) shows the omnidirectional camera is mounted on a robot soccer. The omnidirectional image is usually formed during the omnidirectional camera deployment. It produces images with a circular projection of robot's nearby environments centered on the robot. This characteristic makes the robot's visual main interest easier to track, since they remain longer in the field of view compared to the normal lens camera image.

(a) (b)

Figure 2. (a) Omnidirectional camera, (b) Camera mounted on the robot soccer

Like we mentioned before, omnidirectional camera offers benefits that followed by a distortion. The heavy distortion cause object's geometrical information not as the same as the normal lens camera projection. We use the OpenCV module to calculate the camera intrinsic and extrinsic parameter, then unwrap the image into a cylindrical perspective. The module base on the MATLAB toolbox done by [21]. However, we use the checkerboard pattern instead of feature descriptor-based calibration patterns. The checkerboard pattern utilization is inline with the unified projection model [22] assumption. Where the catadioptric coefficient $\xi = 1$ for initialization. The checkerboard pattern provides a known projection of a straight line.

We start our omnidirectional camera calibration with the process of taking several pictures around the robot. The camera is fixed on top of a camera tripod with height adjusted to be the same as robot soccer final dimension, as in Figure 2. We use a 10x7 checkerboard printed on an A4 plain paper. We attach the checkerboard to a 3 mm mica board. Afterward, we placed the board in eight different directions. In each direction, we set upper and lower board position. Then we move the board at each position on three different surfaces facing toward the omnidirectional camera. Some images taken during omnidirectional camera calibration preparation are shown in Figure 3. A total of 48 images is taken during the process.

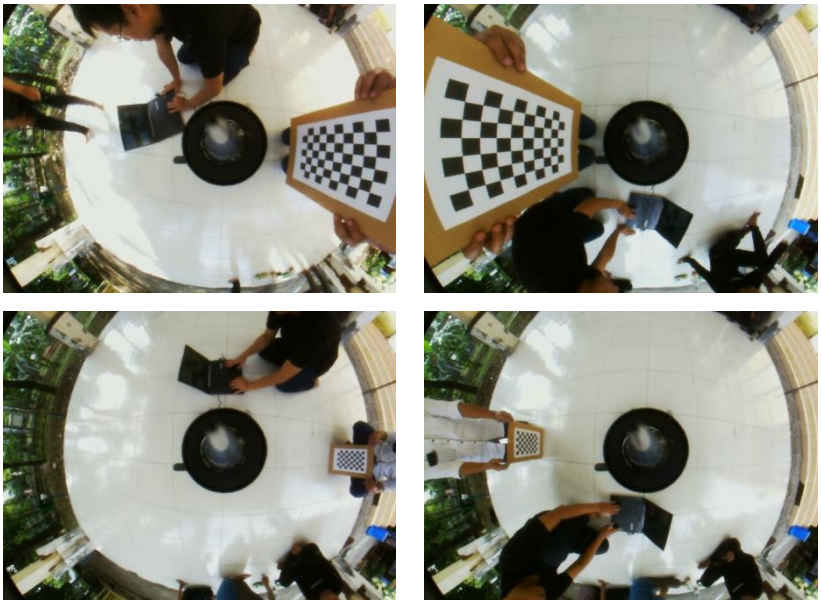

Figure 3. Examples of checkerboard picture taking during the omnidirectional camera calibration preparation

3. RESEARCH METHOD

In this section, we present our method to verify the usability of PeleeNet to detect objects. We initialized the CNN architecture with the generic object dataset PASCAL VOC training. Then we proceed to fine-tune the network for the soccer match object detection task. Fine-tuning a CNN model is a procedure based on the concept of transfer learning [23]. This practice is believed could extract a meaningful representation of the intended object in a compact manner dataset, as in [24]. Although our task is significantly different from the originally trained model, and there is a possibility of the lack of training data. A CNN model's features are first initialized with the broader spectrum of object. Then we replace the spatial and confidence layers of the model which will be trained next. This process is expected to minimize errors in more specific task domain. Under this experimental setup, we investigate and report our object detection task experiment for the KRSBI soccer match robot.

In order to fine-tune a model, we started with removing a "mbox_loss" layer and replace it with a new layer. Figure 4 illustrates transfer learning method deployed in this paper. We used the commonly available trained model weight for Caffe [25] platform. We choose the models trained in PASCAL VOC 2007-2012. The previously trained model is designed to detect 21 objects as in the PASCAL VOC Challenge. The number of classes is reflected in the loss layer. We replace this layer with five classes and one background of total of six numbers of classes. The "mbox_loss" layer is a layer that couple "mbox_loc", "mbox_conf" and "mbox-priorbox". As these layers are functioned as object proposal contained within a boundary box, valued with the confidence level. It needs to learn its weight from a randomly generated number. However, the features and parameters before the loss layer don't need to be trained from scratch. The weight gets a slight adjustment during the training with our data and complete the refinement after the training is finished. During the training session, we set the "mbox_los" parameter and rename it as a new layer. Some of parameter setting examples deployed in the training are confidence loss type: Softmax; Overlap threshold: 0.5; Mining type: Max negative. Next, we report our experiment hardware and library setup. We employ the GTX 1080 Ti as our Deep Learning platform. The GTX 1080 Ti RAM allocation is 11 GB with 3584 cores. We set the batch size 32, with the max iteration set to 6000. Our learning rate is initialized on 0.005 with the weight decay 0.0005. We use a high momentum of 0.9 and gamma 0.1.

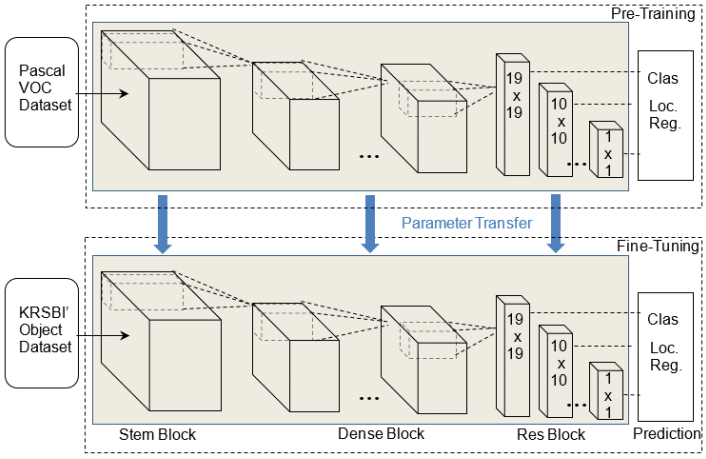

Figure 4. In this simplified display, the whole parameter before the prediction layer is fine-tuned on KRSBI's object dataset with parameter transferred from model pre-trained on the PASCAL VOC dataset

4. RESULTS AND ANALYSIS

In this section, we introduce our result and analysis for PeleeNet's object detection. The main purpose of PeleeNet utilization is the deployment in robot soccer. Therefore, a balance of accuracy, speed and size is a must. Firstly, we analyze the performance of PeleeNet to detect objects in the comparison with two other deep learning models for object detection, I.e SSD and MobileNetSSD. Secondly, we test the inference speed for each model to discover the best timing across all three.

4.1. Quantitative results

We report our detection accuracy based on the metric used in [26] with precision x recall curve on every point interpolation. The precision value is the model capability to recover only the relevant object

complete with its location across all detection. The formula is given in (1). Where true positive (TP) is an object guessed correctly with the intersection over union (IOU) value above some threshold. False positive (FP) is an object guessed wrongly with the IOU value below some threshold. While the recall value is the model capability to recover all the relevant objects. It is given by the (2). Where true positive (TP) is an object guessed correctly with the IOU value above some threshold. False negative (FN) is not detected ground truth.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths} \tag{2}$$

Before precision and recall were calculated, IOU needs to be determined beforehand. IOU is the Jaccard Index value based on the two bounding boxes that were overlapped. Both of the bounding box of object's ground truth and the bounding box of detection are needed. The area under of each then reveals the validation of detection. IOU is given in (3).

$$IOU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{3}$$

Where B_p is a prediction bounding box and B_{gt} is a ground truth bounding box. Subsequently, precision x recall curve is composed of each predicted class. Followed by the all point interpolation in (4), we get the Average Precision (AP) as the approximation of the area under the curve. P_{interp} value calculated with (5). Where $p(\tilde{r})$ is the measured precision at recall $\tilde{r}$.

$$\sum_{r=0}^{1} (r_{n+1} - r_n) p_{interp}(r_{n+1}) \tag{4}$$

$$p_{interp}(r_{n+1}) = p(\tilde{r}) \tag{5}$$

We run a forward pass toward PeleeNet, MobileNetSSD, and SSD. All 75 of the test images are passed through the model once. The confidence level and location then recorded. We report the Mean Average Precision (mAP) followed by the AP for each model on each class in the Table 2. Best overall performance is given by the SSD, with the mAP 95.59%, followed by PeleeNet at 94.13% and the last one is MobileNetSSD at 92.68%. In these comparisons all of three models performed comparable.

Table 2. KRSBI's object test detection result. All of the value shown below are in (%)

Model	mAP	Ball	Center_circle	Goal_post	Robot_cyan	Robot_magenta
PeleeNet	94.13	85.96	97.19	**98.22**	95.32	93.95
MobileNetSSD	92.68	83.86	**97.83**	94.03	93.63	94.08
SSD	**95.59**	**90.56**	97.64	96.00	**95.70**	**98.06**

Next, we further examine each class AP. Each detected object has a different amount in image resolution, the ball has a relatively small area compared to the other class. We first take the ball's AP as the performance measurement point. The PeleeNet's ball AP is 4.6% behind SSD, and 2% better toward MobileNetSSD. This is because of SSD architecture has a bigger scale prediction features map. It consists of 38x38, 19x19, 10x10, 5x5, 3x3, and 1x1. While the PeleeNet lack of the biggest feature maps. The 38x38 feature map helps to give more object's representation in particular a small object like the ball. Secondly, we observe the Goal_post class, this label has the average largest appearance among others. PeleeNet is 2.22% better against SSD and 4.19% more precise against MobileNetSSD. This trend is slightly different for less big object. This shows that five prediction feature maps are not significantly contributed to object detection AP. While keeping only the maximum 19x19 feature maps, it gives PeleeNet a balance between speed and accuracy needed for KRSBI's robot deployment.

4.2. Qualitative results

In this subsection, we deliver our experimental results qualitatively. We show our results on two different test images. The first test images are images that contained in our dataset. These data are taken using a normal lens camera. Then, the second test images are images that recorded employing an omnidirectional camera. In Figure 5 we present our inference result on the dataset's test image. Here we show that PeleeNet performs remarkably well, almost all of the desired KRSBI'e object are detected. Except for the robot cyan in front of the goal post is detected as robot magenta. However, PeleeNet success to detect ball in the center of the field.

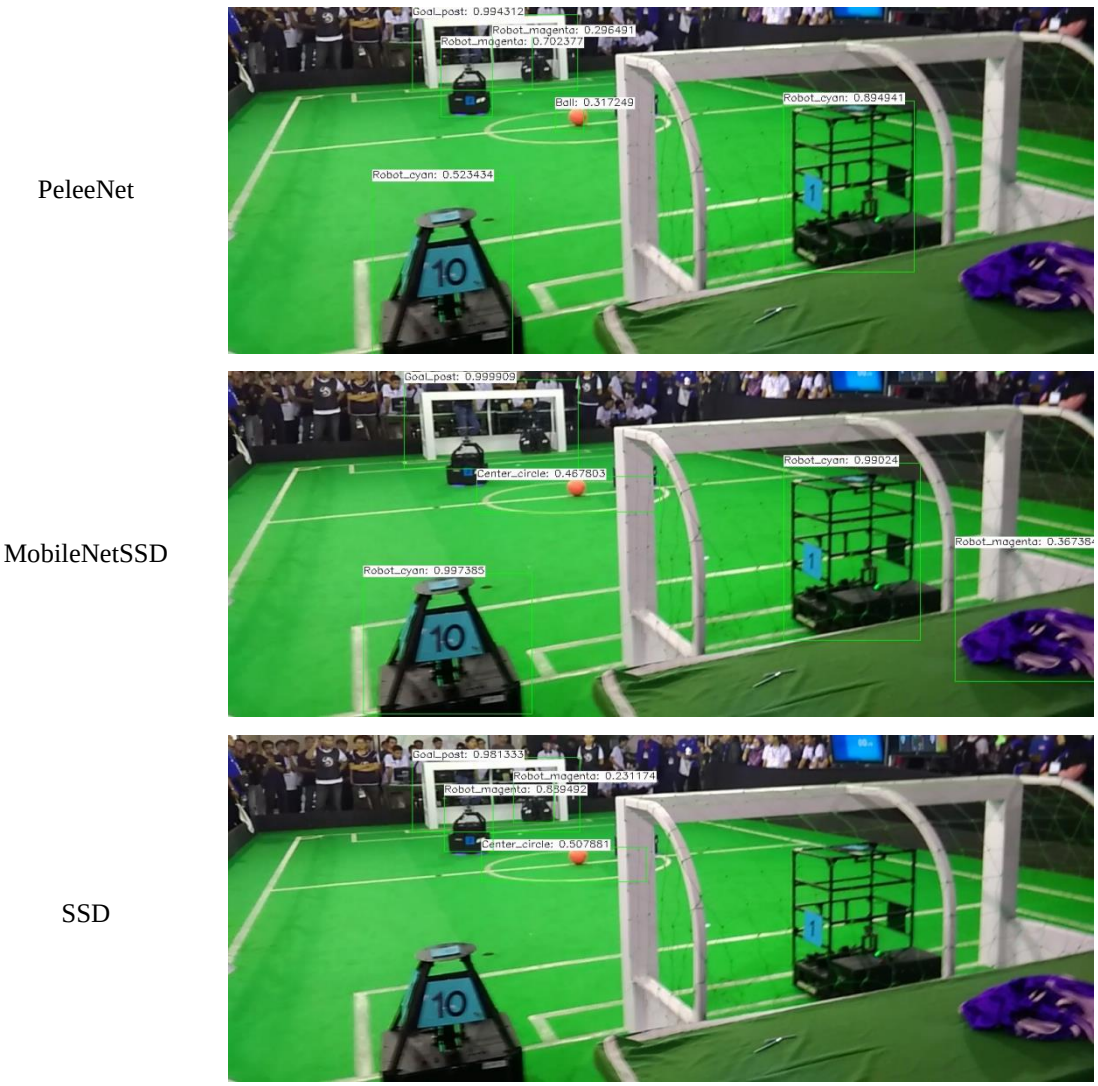

Figure 5. Examples of PeleeNet, MobileNetSSD, and SSD detection result

Next, we convey our qualitative detection result on omnidirectional images. We employ omnidirectional camera as our robot visual perception. We calibrate image taken by a sensor, in order to remove visual defective of extreme mirror curvature incorporated by an omnidirectional camera. The camera calibration procedure was done in an off-line manner. This is rather a risky protocol. Because the movement of aperture ring and lens focuser during the robot movement will alter the focus point and brightness of collimating lens. Consequently, the extrinsic and intrinsic parameters changed. Because it was obtained by not accommodating the displacement of ring focuser and aperture lock. With this problem in the mind, we continue our camera calibration and experiment by keeping the aperture ring and lens focuser fix. Streaming images from camera are calibrated and cropped with a resolution of 300x300. Each region of interest (ROI) contains ball. We accommodate different ball-layout inside the frame.

We take photos of ball in the indoor soccer field. We use the recommended field size for the KRSBI. We put the striker robot in the center field, then the ball is placed 1 m from ahead robot. We vary the ball position by 0.5 m from the previous point. We repeatedly move the ball until it reaches the opponent's goal line. Subsequently, the robot is step backed 1 m from its initial position, and the same procedure for ball placing is repeated. We lay the robot and ball until it covers the whole soccer field. We use two color variations of soccer balls to enrich our data. The first ball is orange, where its widely use for the robot soccer match. The second ball is yellow, we adapted this ball in order to tackle the possibility of ball lighting variation during the match. We also perform the data collection in our lab, we use the white floor as the base.

Subsequently, we begin with the omnidirectional camera calibration. We present our calibrated image collection taken at the soccer field during the data acquisition (see Figure 6). In Figure 6 we show that after the calibration, images from the omnidirectional camera no longer show reflection from the spherical mirror. Rather, it rectifies image with a slight distortion around camera's sensor. This kind of aberration has not affected the testing session. Later in this subsection, we will show our model's inference result for ball positioned near the camera and in the distance. Furthermore, after the images are calibrated. We conducted object detection tests using PeleeNet, MobileNetSSD, and SSD models. The result is shown in Figure 7 are in line with the detection result of the normal images. KRSBI's object detection using the PeleeNet on the calibrated omnidirectional images meet the hindrance toward a small object.

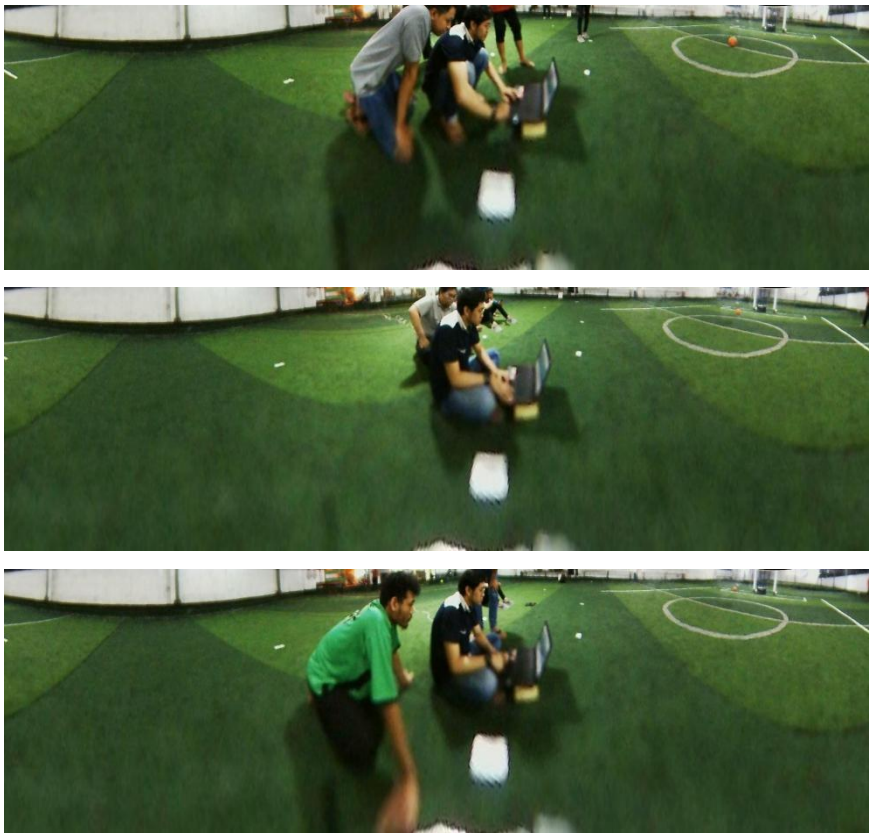

Figure 6. Calibrated images are taken from an omnidirectional camera. Top and bottom image shows different ball position, starting from in the center field to near the goalkeeper area

4.3. Computational speed

The training session for each model consumes time differently. As the model's architecture suggests, SSD which based on the VGG needs 7 days on the GTX 1060 platform. MobileNetSSD which employs MobileNet as their base net requires 2 days to finish training session. While PeleeNet takes also 2 days to accomplish. We then examine the actual inference speed of PeleeNet compared to the other model. The inference time is calculated by average time of 200 images processed by the models. The inference time is taken only during the test images past the network once. It doesn't include the image preprocessing.

We report the inference speed on Table 3. PeleeNet passes the test images within the network requires 0.1 s. This result is 0.3 s better toward the SSD, and 0.033 slower compared to the MobileNetSSD.

Table 3. Average inference time

Model	Input Dimension	Speed (s)
PeleeNet	304x304	0.1
MobileNetSSD	300x300	0.067
SSD	300x300	0.43

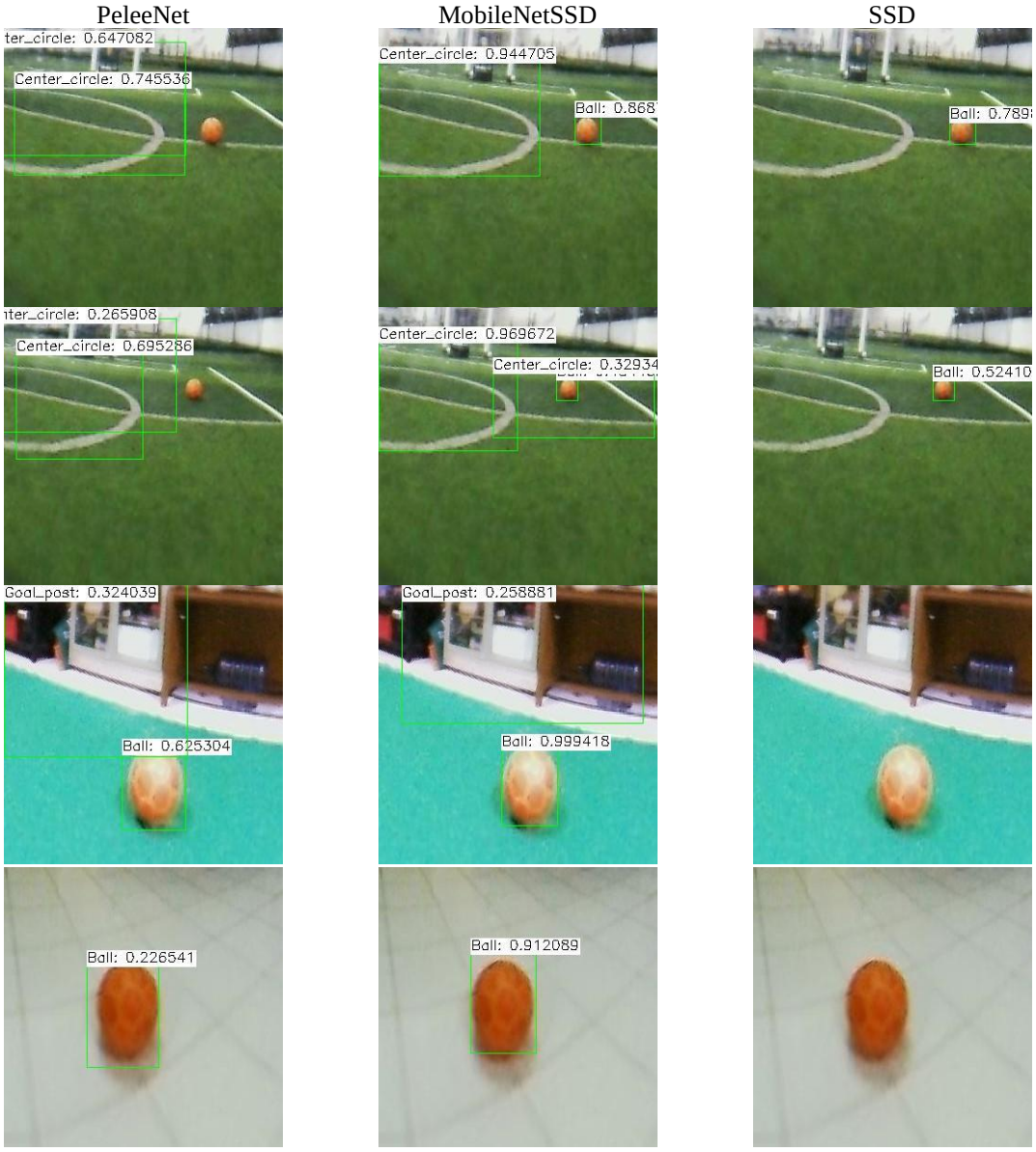

Figure 7. Ball detection result of various ball field's positions of the omnidirectional images

5. CONCLUSION

We deliver our experiment result to detect KRSBI's object using PeleeNet. We compare PeleeNet with SSD and MobileNetSSD to set perspective of PeleeNet performance. PeleeNet has the best balance of its speed, memory and accuracy. Although it has not the best overall performance to predict object. PeleeNet has the potential to be used for mobile deployment in robot soccer, particularly for KRSBI.

REFERENCES

[1] Asada M, Balch T, Bonarini A, Bredenfeld A, Gutmann S, Kraetzschmar G, et al. "Middle Size Robot League Rules and Regulations for 2012," 2011. [Online]. Available: https://msl.robocup.org/wp-content/uploads/2018/08/msl-rules-2011-12-29.pdf.

[2] Treptow A, Zell A., "Real-time object tracking for soccer-robots without color information," *Robotics and Autonomous Systems*, vol. 48, no. 1, pp. 41–48, 2004.

[3] Gaspar J., Winters N., Santos-Victor J., "Vision-based navigation and environmental representations with an omnidirectional camera," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 890–898, 2000.

[4] Nayar S. K., "Catadioptric omnidirectional camera," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 482–488, 1997.

[5] Coath G., Musumeci P., "Adaptive arc fitting for ball detection in robocup," *Proceedings of APRS Workshop on Digital Image Analysing*, Brisbane, Australia, pp. 63–68, 2003.

[6] Matthias Jüngel, Jan Hoffmann, Martin Lötzsch, "A Real-Time Auto-Adjusting Vision System for Robotic Soccer," *RoboCup 2003: Robot Soccer World Cup VII*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 214–225, 2004.

[7] Rahman A., Widodo N. S., "Colored ball position tracking method for goalkeeper Humanoid Robot Soccer," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 11, no. 1, pp. 11–16, 2013.

[8] Huimin Lu, Hui Zhang, Junhao Xiao, Fei Liu, Zhiqiang Zheng, "Arbitrary Ball Recognition Based on Omni-Directional Vision for Soccer Robots," *RoboCup 2008: Robot Soccer World Cup XII*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 133–144, 2009.

[9] Neves A. J. R., Pinho A. J., Martins D. A., Cunha B., "An efficient omnidirectional vision system for soccer robots: From calibration to object detection," *Mechatronics*, vol. 21, no. 2, pp. 399–410, 2011.

[10] Krizhevsky A., Sutskever I., Hinton G. E., "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[11] Speck Daniel, Barros P., et al., "Ball Localization for Robocup Soccer Using Convolutional Neural Networks". *RoboCup 2016: Robot World Cup XX*. Cham: Springer International Publishing, pp. 19–30, 2017.

[12] O'Keeffe Simon, Villing R., "A Benchmark Data Set and Evaluation of Deep Learning Architectures for Ball Detection in the RoboCup SPL,". *RoboCup 2017: Robot World Cup XXI*. Cham: Springer International Publishing, pp. 398–409, 2018.

[13] Wang R. J., Li X., Ling C. X., "Pelee: A real-time object detection system on mobile devices," *Advances in Neural Information Processing Systems*, pp. 1963–1972, 2018.

[14] Girshick R., Donahue J., Darrell T., Malik J., "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation,". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[15] Girshick R., "Fast R-CNN," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, 2015.

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, et al., "SSD: Single Shot MultiBox Detector". *European Conference on Computer Vision – ECCV 2016*, pp. 21–37, 2016.

[17] Tzutalin, LabelImg, GitHub, 2015. [Online]. Available: https://github.com/tzutalin/labelImg.

[18] Baker S., Nayar S. K., "Theory of catadioptric image formation," *Sixth International Conference on Computer Vision*, pp. 35–42, 1998.

[19] Tomáš Svoboda, Tomáš Pajdla, Václav Hlaváč, "Epipolar geometry for panoramic cameras". *European Conference on Computer Vision — ECCV'98*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 218–231, 1998.

[20] Wei S. C., Yagi Y., Yachida M., "Building local floor map by use of ultrasonic and omni-directional vision sensor," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2548–2553, 1998.

[21] Li B., Heng L., Koser K., Pollefeys M., "A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1301–1307, 2013.

[22] Mei C., Rives P., "Single view point omnidirectional camera calibration from planar grids," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3945–3950, 2007.

[23] Donahue J., Jia Y., Vinyals O., Hoffman J., Zhang N., Tzeng E., et al. "DeCAF: A deep convolutional activation feature for generic visual recognition," *31st International Conference on Machine Learning - ICML 2014*, pp. 988–996, 2014.

[24] Agoes A. S., Hu Z., Matsunaga N., "Fine tuning based squeezenet for vehicle classification," *ICAIP 2017: Proceedings of the International Conference on Advances in Image Processing*, pp. 14–18, 2017.

[25] Jia Y., Shelhamer E., Donahue J., Karayev S., Long J., Girshick R., et al., "Caffe: Convolutional architecture for fast feature embedding," *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 675–678, 2014.

[26] Everingham M., Van Gool L., Williams C. K. I., Winn J., Zisserman A., "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

BIOGRAPHIES OF AUTHORS

Winarno is a lecturer in the Physics department, Faculty of Science and Technology, Universitas Airlangga. He has graduated in Physics from Universitas Airlangga and received his Master of Engineering from Institut Teknologi Sepuluh Nopember Surabaya. Winarno's research interests are computer vision, machine learning, electronics and interfacing.

Ali Suryaperdana Agoes received his M.T. degree from Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia in 2014. He gets his Dr. Eng degree from Kumamoto University, Japan in 2018. Currently, he joins as the researcher staff in the Medical Robotics Laboratory, Universitas Airlangga. He is also R & D staff of HMS-Global, japan. His research interest includes deep learning model optimization, 3D surface reconstruction and image semantic segmentation.

Eva Inaiyah Agustin received her Bachelor of Applied Science in Electronics from Politeknik Elektronika Negeri Surabaya and received her Master of Engineering in Electronics from Institut Teknologi Sepuluh Nopember Surabaya. Currently, she is a lecturer in Department of Engineering, Faculty of Vocational, Universitas Airlangga. Her research interest is electronics, sensors and signals, analog circuits and electronics, and soft computing.

Deny Arifianto received his Bachelor of Science in Physics at Universitas Airlangga in 2008 and finishing his Master of Engineering in Biomedical Engineering at Universitas Airlangga in 2018. He works as a lecturer at Department of Engineering, Faculty of Vocational, Universitas Airlangga. His research interests are mechanical and hardware design, electronics, embedded systems and soft computing.