

# Pick-and-place application using a dual arm collaborative robot and an RGB-D camera with YOLOv5

Dumrongsak Kijdech, Supachai Vongbunyong

Institute of Field Robotics, King Mongkut's University of Technology Thonburi, Bangkok, Thailand

## Article Info

### Article history:

Received Oct 11, 2022

Revised Oct 30, 2022

Accepted Mar 15, 2023

### Keywords:

Artificial intelligence

Convolution neural network

RGB-D

You Only Look Once

YuMi

## ABSTRACT

Nowadays, many industries use robots and cameras in tandem to detect specific objects and perform specific tasks. However, misdetection can occur due to inconsistencies in lighting, background, and environment. In order to address the aforementioned issues, this study proposes using a dual arm six-degree-of-freedom (6-DoF) collaborative robot, ABB YuMi, and red, green, blue-depth (RGB-D) camera with YOLOv5 in a pick-and-place application. In order to prepare the dataset, the images are collected and labeled. The dataset has been trained with the YOLOv5 machine learning algorithm. It has taken on the role of weight for real-time detection. When RGB images from a camera are sent to YOLOv5, data pertaining to the bottle's position x-y and color are extracted from the depth and color images. The position of the robot is used to control its movement. There are three parts to the experiment. To begin, YOLOv5 is tested with and without trained images. Second, YOLOv5 is tested with real-time camera images. Finally, we assume that YOLOv5 has perfect detection and grasping ability. The results were 95, 90, and 90%.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



## Corresponding Author:

Supachai Vongbunyong

Innovation and Advanced Manufacturing Research Group, Institute of Field Robotics, King Mongkut's

University of Technology Thonburi

Bangkok, Thailand

Email: supachai.von@kmutt.ac.th

## 1. INTRODUCTION

These days, the demand for robots and automation has increased dramatically in many industries due to labor shortages and high wage rates. Robots and automation are able to resolve these issues. However, the general image processing package with industrial robots has limitations when dealing with complex lighting and background. As a result, errors will prone while those conditions occur. Currently, vision system with artificial intelligence (AI) has gained more interest due to the improvement of the performance of computation speed of the graphic card on the computer. Previously, image processing is one of the most popular techniques for object classification and localization. Recently, a convolution neural network (CNN) has been widely used for object detection and classification. However, CNN is primarily divided into two types such as you only look once (YOLO) and mask region-based convolution neural network (R-CNN).

This research proposes using a dual-arm collaborative robot, "YuMi", and a red, green, blue-depth (RGB-D) camera with YOLOv5 for pick-and-place application. The working principle of this research is as the L515 camera will send images to YOLOv5. The position of the object is received and sent to the YuMi robot. The robot will pick the object and place it in a specified position. To resolve the problem regarding the uncertainty of light and environment. Object detection and classification by using AI is used. The ability of

AI to resist uncertainty comes from training with a lot of data. Images with various conditions of light and environment are used to train. YOLOv5 is faster in comparison to other YOLOs. YOLOv5 could reduce training time and detection time for real-time operation.

The rest of this paper is organized as follows. Section 2 discusses a literature review about robots with vision systems and vision systems with CNN. Section 3 provides a methodology of the system setup with each component, system overview, camera, and coordinate system. Section 4 presents the methodology of object detection with YOLO, including the principle of YOLO, dataset selection, and training set. Section 5 presents the experiment setup, result, and discussion. Finally, section 6 are discussion and conclusion.

## 2. LITERATURE REVIEW

### 2.1. Review robot with vision system

Currently, a lot of research works implement robot arms with vision systems to perform specific tasks, e.g. [1], [2]. Zakhama *et al.* [3] used a selective compliance assembly robot arm (SCARA) robot with image processing. Red, green, and blue (RGB) image is converted to hue-saturation-value (HSV) image and then the histogram is analyzed to find the object. Kirschner *et al.* [4] used a dual-arm collaborative robot, YuMi, with an RGB-D camera to detect and grasp the object. Liang *et al.* [5] proposed using dual quaternion-based kinematic control to control the YuMi robot. Due to the safety issue, collaborative robots are suitable for applications that work alongside with human operators. Wu *et al.* [6] used robots with YOLO for detecting objects. The system was implemented based on the robot operating system (ROS) and Python. Yang *et al.* [7] use the UR5 robot arm and intel RealSense D435 camera with YOLOv3 to detect objects and grasp them. YOLO was great for recognizing and localizing the objects in the image. However, some more work is required to obtain the orientation data. Opaspilai *et al.* [8] used YOLOv3 with SCARA robot to depalletization of pharmaceutical products and used an RGB camera to collect data, while in [9], they changed from an RGB camera to a 3D camera which improved the localization accuracy. Lelachaicharoeanpan and Vongbunyong [10] used YOLOv5 with UR5 robot to the classification of surgical devices, which surgical devices have very similar. Using only image processing is not enough.

In regard to the vision system, RGB-D cameras are used for object detection applications. Hu *et al.* [11] used an Intel RealSense R200 camera with image processing to do segmentation in-depth images. Pohl *et al.* [12] used Intel NUC7i7BNH and intel RealSense D435i with Ubuntu 18.04 and OpenCV 3.4.5 to create depth maps for being used in drones. Francisco and Araujo [13] used intel RealSense SR305, D415, and L515 in experimental evaluation and comparison of depth images and found that the resolution of D415 and L515 were superior yet using different techniques. The D415 camera uses stereoscopic depth technology while the L515 camera uses light detection and ranging (LiDAR) depth technology. Therefore, the L515 camera has a more compact size, lower noise, and high resolution in both RGB and depth images than the others.

### 2.2. Review vision system with YOLO

A number of object detection techniques are considered in this research. A number of research works studied about the performance of each object detection technique and various versions of YOLO from YOLOv1 to YOLOv5. Fang *et al.* [14] needed real-time objection, so tinier YOLO was a good choice due to it having a tinier model than YOLO. However, a decrease in precision is a major drawback. Du [15] compared each object detection based on the CNN family and YOLO. Dataset training time has decreased and the frame rate in detection has increased as the version is more updated. Zhang *et al.* [16] used YOLOv2 for real-time detection in Chinese traffic signs. It shows fastest detection speed was 0.017 seconds per image. Sang *et al.* [17] used YOLOv2 to improve vehicle detection and achieve the fastest detection speed at 0.038 seconds per image. Mahto *et al.* [18] used YOLOv4 for vehicle detection in other specific situations. Tian *et al.* [19] used improved YOLOv3 for apple detection in orchards. The different growth stages of the apple created more complexity in the model. Zhao and Li [20] improved YOLOv3 for achieving better mean average precision (mAP) of object detection.

YOLO is faster than faster R-CNN [21], [22] because YOLO detects the objects from feature maps while faster R-CNN is region-based detection. Therefore, faster R-CNN takes more computation time than YOLO although these techniques are based on CNN as well. The performance of YOLO has improved a lot recently. YOLOv5 was proposed a few months after YOLOv4 and showed significant improvement in speed [23], [24]. Applications of these detection techniques are presented in many research works. For example, Benjdira *et al.* [21] compared faster R-CNN and YOLOv3 in car detection in an unmanned aerial vehicle application. YOLOv3 performed better than R-CNN due to the above reason. Tekin *et al.* [25] presented a development of YOLOv2 from 2D prediction to 6D prediction in real-time.

### 2.3. YOLOv5 principle

YOLOv5 is based on CNN (structure shown in Figure 1) consisting of convolution (Conv) layers, batch normalization (BN) layers, and leaky rectified linear unit (ReLU) in many layers. YOLOv5 neural networks provide the different models with different configurations and each parameter is shown in Figure 2, where FP16 stands for the half floating-point precision, V100 is an inference time in milliseconds on the Nvidia V100 GPU, and mAP is based on the original COCO dataset. YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x are the structure of the neural network model. Each structure of a neural network with higher complexity uses a longer time for the training process but offers higher accuracy for the detection process respectively. The size of the models varies from 14 to 168 MB. The mAP values for the COCO dataset are from 36.8 to 50.1 and the inference time on the Nvidia V100 GPU is from 2.2 to 6.0 milliseconds. In this research, YOLOv5l neural network model is used for the experiment.

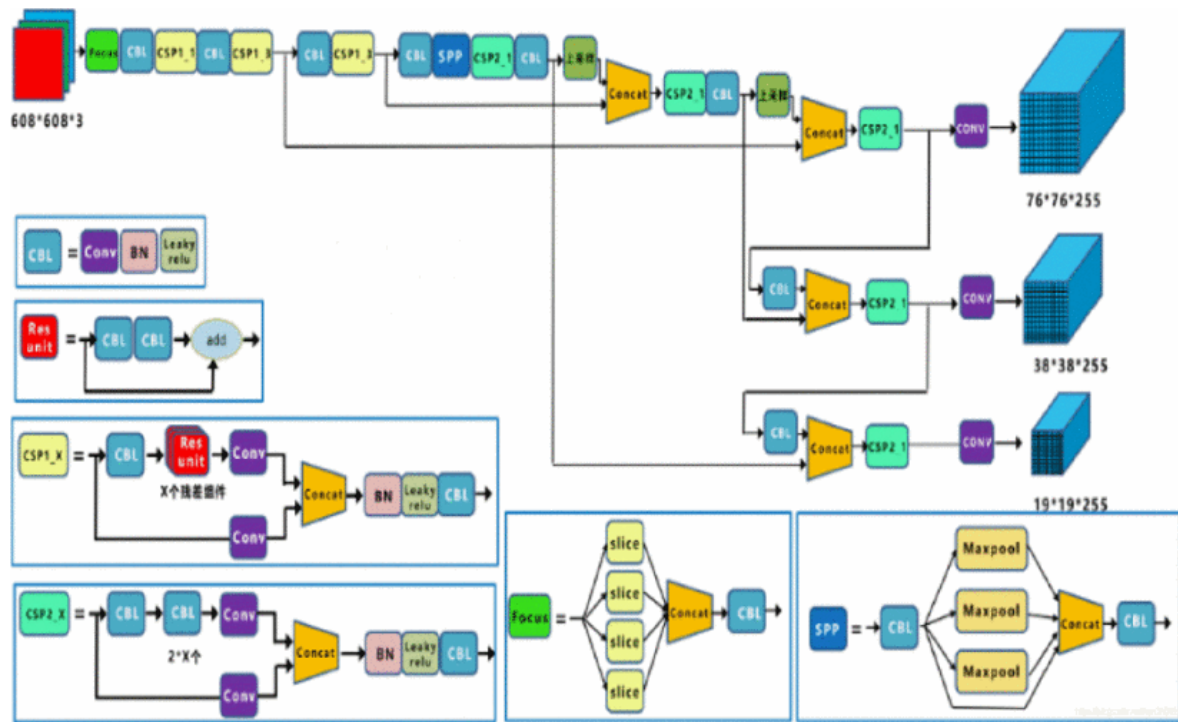


Figure 1. Structure of YOLOv5 [24]

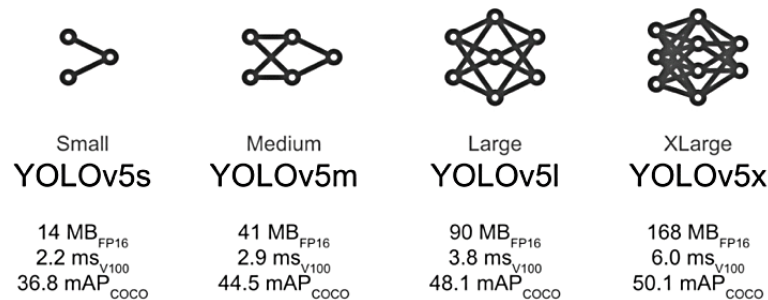


Figure 2. YOLOv5 different model sizes [26]

## 3. METHODOLOGY: SYSTEM SETUP

### 3.1. Description of each component

The configuration of the pick-and-place system is shown in Figure 3. In this research, ABB IRB-14000 or “YuMi” robot is the main dual arms robot controlled by an IRC5 controller. It is a collaborative robot designed for working side-by-side with human operators. Each arm has 7 degree-of-freedom (DoF).

YuMi is suitable for assembling small parts with a payload of 500 g for each arm. The workspace of this system is 560×320 mm on XY-plane in front of the robot as shown in Figure 3(a) and (b).

Each robot arm is equipped with a gripper modified from the standard YuMi gripper. Originally, the integrated 2-finger friction gripper with vision and vacuum is equipped as a standard version. The new friction gripper is specifically designed for grasping bottles and other objects as shown in Figure 3(b).

In order to sense the depth information in the z-direction, RealSense L515 is used as an RGB-D camera equipped above the robot as shown in Figure 3. This camera is selected according to its high resolution and its compact size. The specification as the minimum depth distance is 250 mm and the depth accuracy is 5 to 14 mm. Depth image resolution is 1024×768 pixels at the frame rate of 30 fps. The RGB color image resolution is 192×1080 pixels at the frame rate of 30 fps.

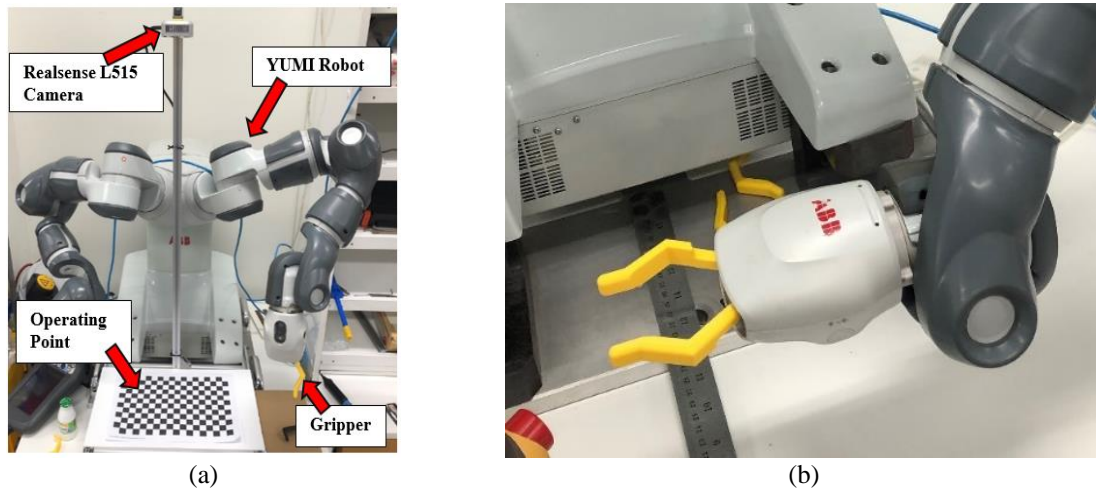


Figure 3. Configuration of pick and place system of (a) YuMi and (b) gripper

### 3.2. System overview

A diagram of the pick-and-place system of this research is shown in Figure 4. Real-time RGB and depth images are sent from the camera to the computer. Object detection and classification are computed on the computer. The location of the object ( $x, y, z, q_1, q_2, q_3, q_4$ ) is sent from the computer to the robot by a LAN cable (TCP/IP protocol).

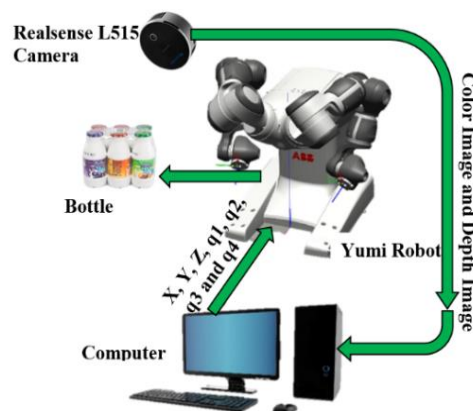


Figure 4. Diagram of pick and place system of this research

The overview of the system is shown as a diagram in Figure 5. The process consists of three sections i) object detection algorithm, ii) grasp planning and coordination conversion, and iii) process within the robot. The details are explained below.

In the object detection process, a color image from the camera is processed by YOLOv5. The output from YOLOv5 is the class and the location of the object. The class of the object is displayed on the computer screen for validation purposes. The location of the center of the object on the XY-plane is obtained from the color image while the position on Z-axis is obtained from the depth image at the associated XY position.

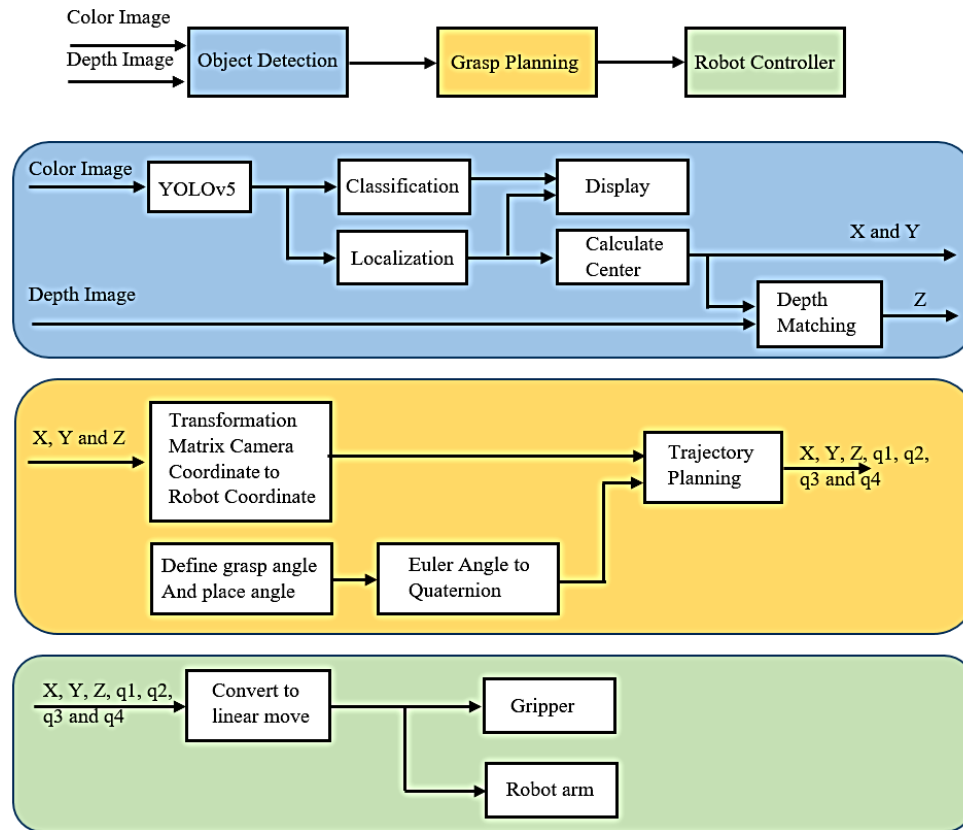


Figure 5. System overview of this research

For the grasp planning algorithm, the position XYZ of the object in the camera coordinate obtained from the previous process is transformed into the robot coordinate. The Euler angle roll, pitch, and yaw are converted to quaternion  $q_1$ ,  $q_2$ ,  $q_3$ , and  $q_4$ . The output of the object location will be managed by the trajectory planning system afterward. For trajectory planning, the robot will grasp the object from above along the z direction. Planning of pick and place starts from the standby posture, moves to the pre-grasp position, grasps the object, moves back to pre-grasp position, moves back to standby posture, moves to pre-place, place position, moves back to pre-place position, and then moves back to standby posture. Many via points are created in order to avoid the singularity of the robot arm. Trajectory planning is done on the computer with a Python program. The set of commands, such as  $x$ ,  $y$ ,  $z$ ,  $q_1$ ,  $q_2$ ,  $q_3$ ,  $q_4$ , and the state of the gripper, is sent to the robot via TCP/IP protocol with socket messaging.

### 3.3. Coordinate system

The transformation matrix is used to transform the location of the object in the camera coordinate to the robot coordinate. Therefore, the position of the object from the image coordinates will be transformed and used for the robot to pick up. The reference coordinate of the YuMi robot and the camera are shown in Figures 6(b) and 6(a), respectively. The position of the object in the camera coordinate is shown as (1), where  $x_c$ ,  $y_c$ , and  $z_c$  are the distance from the center of the camera to the object in the  $x$ ,  $y$ , and  $z$  directions. The position of the object in the robot coordinate is shown as (2), where  $x_b$ ,  $y_b$ , and  $z_b$  are the distance from the center of the robot base to the object in the  $x$ ,  $y$ , and  $z$  directions. In (3), the distance from the camera to the robot in the  $x$ ,  $y$ , and  $z$ -axis of the robot is designated in column 4 of the matrix as shown in Figure 7. The distances in  $x$ ,  $y$ , and  $z$  are 232, 0, and 780 mm, respectively. The  $3 \times 3$  rotation matrix in the transformation matrix is calculated according to the relative angles between two coordinates. The result of (3) is (4).

$$p^c = \begin{bmatrix} x^c \\ y^c \\ z^c \\ 1 \end{bmatrix} \quad (1)$$

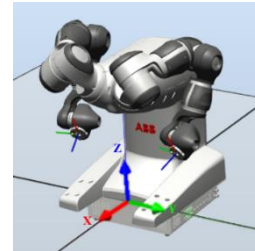
$$p^b = \begin{bmatrix} x^b \\ y^b \\ z^b \\ 1 \end{bmatrix} \quad (2)$$

$$T_c^b = \begin{bmatrix} \cos 180^\circ & \cos 90^\circ & \cos -90^\circ & 232 \\ \cos 90^\circ & \cos 0^\circ & \cos 90^\circ & 0 \\ \cos 90^\circ & \cos 90^\circ & \cos 180^\circ & 780 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$$T_c^b = \begin{bmatrix} -1 & 0 & 0 & 232 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 780 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$



(a)



(b)

Figure 6. Reference coordinate of YuMi robot and RealSense L515 camera (a) RealSense L515 camera and (b) YuMi robot

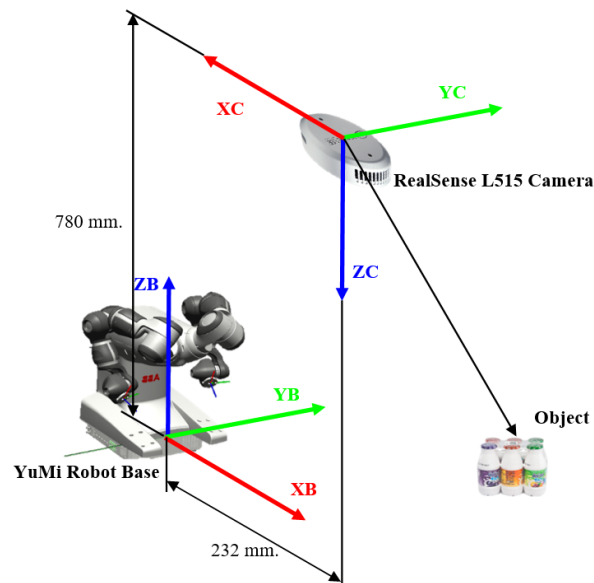


Figure 7. Coordinate of YuMi robot and coordinate of RealSense camera

After that, (1), (2), and (4) would be represented in (5). Eventually, the position of the object in the robot coordinate can be obtained from (6) to (7).



$$p^b = T_c^b p^c \quad (5)$$

$$\begin{bmatrix} x_b \\ y_b \\ z_b \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 232 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 780 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} x_b \\ y_b \\ z_b \\ 1 \end{bmatrix} = \begin{bmatrix} -x_c + 232 \\ y_c \\ -z_c + 459 \\ 1 \end{bmatrix} \quad (7)$$

### 3.4. Camera calculation

According to Figure 1, the camera is installed above the robot and points downward to the working area. The full frame of the color image from the camera is shown in Figure 8(a) and only the region of interest (ROI), marked as a red frame, is used for further processing as shown in Figure 8(b). Only the ROI is concerned to reduce computation time and disturbance. The origin points of the image coordinates refer to the line of sight of the camera. It is at  $x=160$  mm and  $y=300$  mm when measured according to the ROI frame. The depth image and color image are captured at the same time but with different resolutions. Therefore, the resolution is reduced to  $700 \times 370$  pixels and both images can be aligned with position offset.

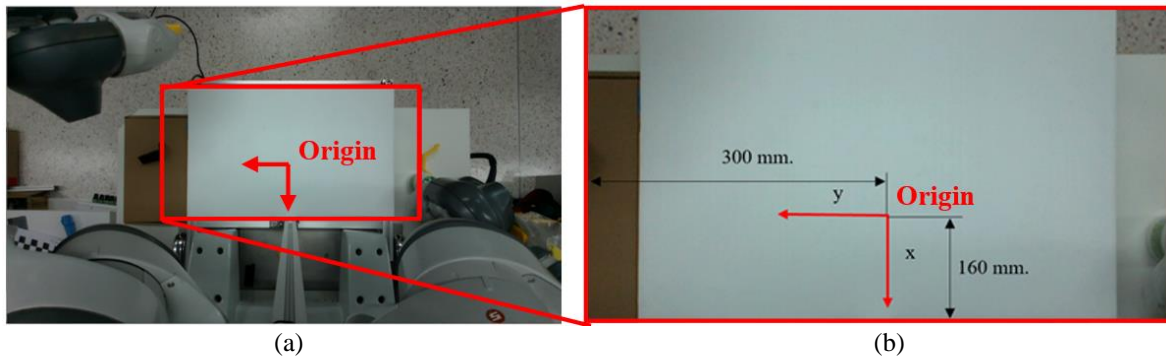


Figure 8. Full image and cropped image, distance, and center of camera on image (a) full image and cropped image within red edge and (b) distance and center of camera on image

## 4. METHODOLOGY: OBJECT DETECTION WITH YOLOv5

### 4.1. How YOLOv5 works in this system

The YOLOv5 works in this system as follows. First, the images from the real-time camera are sent to convolution in Figure 1. These images are flattened from the  $n \times m$  matrix to  $nm \times 1$ . After that, the data is sent to the neural networks model (YOLOv5l). Eventually, the output is a bounding box and class of the image.

Many versions of YOLO were preliminarily tested and compared to select the most suitable one for this application. YOLOv5s and YOLOv5m are found to have too small a neural network structure for this application. Three versions of YOLOv5, including YOLOv5m, UOLOv5l, and YOLOv5x, were compared. As a result, YOLOv5m was faster than UOLOv5l for training and detection but mAP is lower. Meanwhile, the training and detecting duration was almost double for YOLOv5x in comparison to YOLOv5l. Eventually, YOLOv5l is most suitable and used in this research.

### 4.2. Dataset selection and usage

The standard size of the color image is  $1280 \times 720$  pixels and the depth of the image is  $1024 \times 768$  pixels. RGB image is resized to  $700 \times 370$  pixels. Only color images are used for training while depth information is used for obtaining the location of the object. The training set data is collected from various postures and backgrounds. LabelImg program is used to label classes and bounding boxes. After that .txt file is the outcome of this program. Eventually, in this research, 563 images are used for training. Figure 9(a) and (b) shows the color and depth images from the camera, and examples of color images with label are shown in Figure 10.



Figure 9. The image in (a) full and cropped and (b) distance and center of camera on image



Figure 10. The dataset with label examples

#### 4.3. Training set

The normal dataset for training is shown in Figure 10. The modified random dataset is prepared to be used for an augmentation which is helpful to train the model that is capable of handling more complexity. The sample object used in this research is a milk bottle with a color label. They are chosen to be testing samples because the size, shape, and weight are suitable for the robot's capacity (500 g maximum payload). The object is around 200 g, and the shape is symmetrical. The color label is different for each bottle, which is an advantage for increasing the mAP as the training set is more varied.

The dataset from the above method is trained by the YOLOv5 structure. The backbone of YOLOv5 is PyTorch. It is operated on a Python program. Google collaborator is used for training data. GPU for training data is random from Google Collaborator. Tesla T4, 15109.75 MB graphic card is used in this research. In summary, this model is 499 layers and the number of YOLOv5l parameters is 46,642,120. The performance of GPU is 114.3 giga-floating point operations per second (GFLOPS). As a result, training 200 epochs can be completed in 0.642 hours. The testing was done with various settings of epochs: 50,100,150, and 200. Consequently, 200 epoch is selected as it provides the highest mAP.

### 5. EXPERIMENT

Three experiments were conducted in order to evaluate the performance of the system. It focused on the validation of the training set, the performance of the detector, and the robot operations.

#### 5.1. Validation of training set

The experiment is to evaluate the performance of the detector constructed by the training process. The testing set consists of an image that was used for training and images that were not. The results are



shown as graphs illustrating the following data: box, objectness, classification, precision, recall, val box, val classification, and mAP.

- Box is the error of the bounding box in training.
- Objectness is essentially a measure of the probability that an object exists in a proposed region of interest (ROI).
- Classification is the problem of determining the category or goal label in training.
- Precision is referred to the number of true positives divided by the total number of positive predictions.
- Recall is referred to the number of correctly classified positives divided by the total number of positives.
- Val box is the error of the bounding box in validation.
- Val classification is the problem of determining the category or goal label in validation.
- mAP is calculated by finding average precision (AP) for each class and then average over a number of classes as shown in (8).

$$mAP = \frac{1}{N} \sum_{i=0}^N AP_i \quad (8)$$

The result of training and validation is shown in Figure 11 (the y-axis is the percentage, and the x-axis is the number of epochs). These results are considered only at epoch 200. The error of the box from the training was 0.5%, the error of the objectness was 0.05%, the error of the classification was 0.1%, the precision was 100% and the recall was 100%. For validation, the error of the val box from the training still was 0.05% yet, the error of the val objectness was 0% and the error of the classification was 0%. Finally, the mAP is the result of all the classes. The mAP@0.5 was 100% and mAP@0.5:0.95 was 90%.

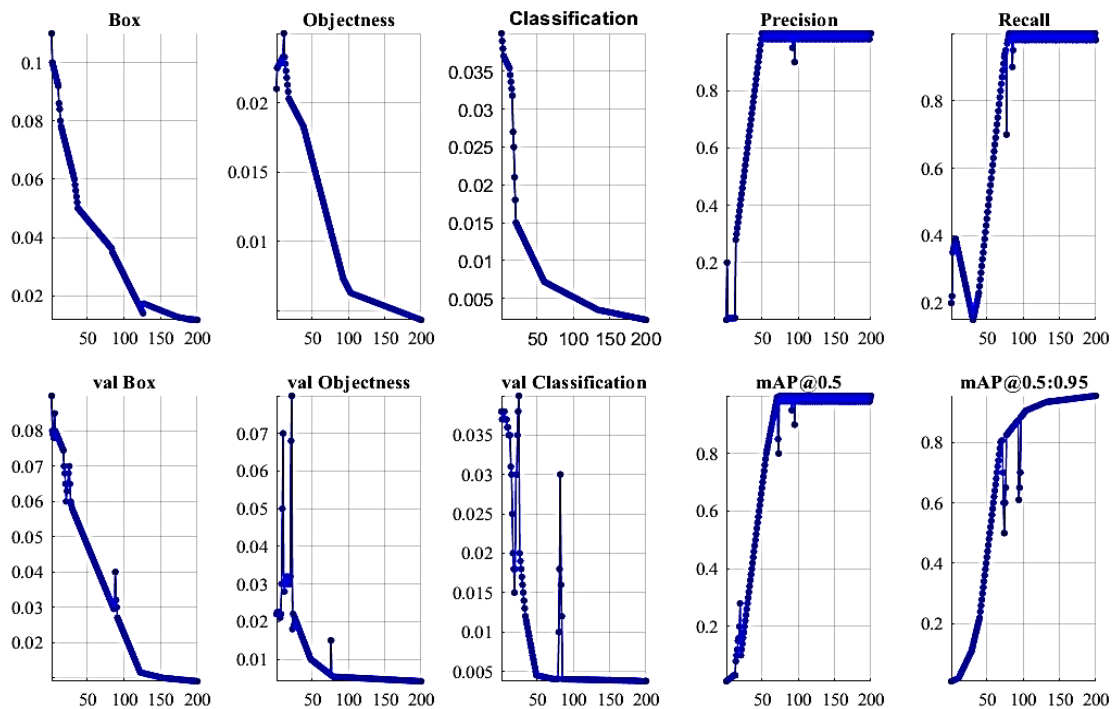


Figure 11. The result of training and validation

## 5.2. Performance of the detector

Precision is a value compared between true positive and false positive as shown in (9). Confidence is the confidence score of prediction, where 1 is full confidence and 0 is not confidence. The comparison between precision and confidence is shown in Figure 12. When this model had given a confidence score of less than 5%, precision is very low; otherwise, precision is high.

The recall is a correctness value that compares true positives and false positives as shown in (10). When the recall is 1 that the predicted data is 100% correct vice versa when the recall is 0 the predicted data is not correct. The comparison between recall and confidence is shown in Figure 13. When confidence is less than 90%, recall is 100%.

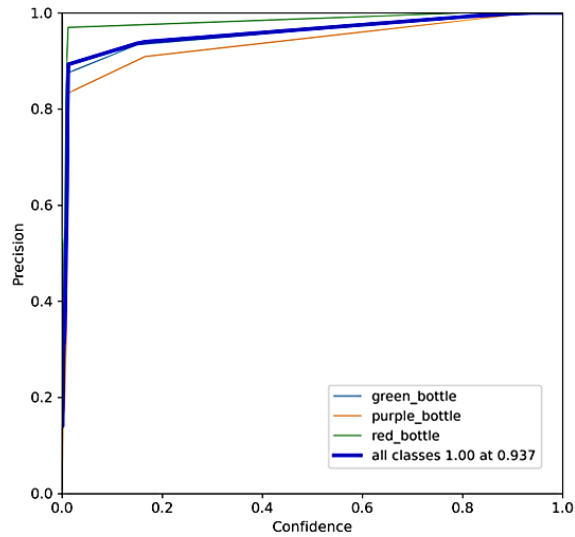


Figure 12. The comparison between precision and confidence

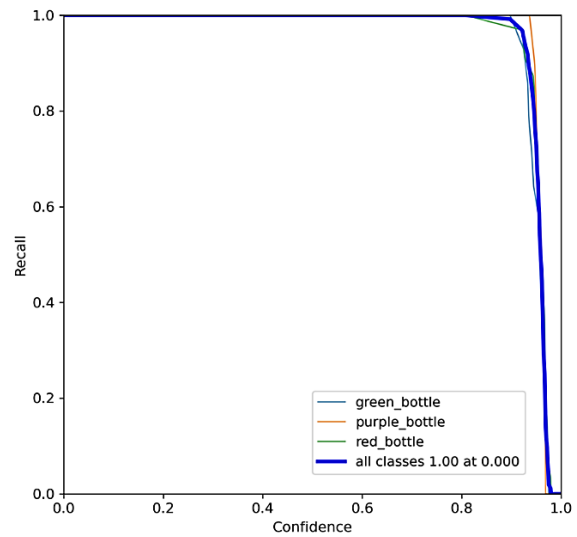


Figure 13. The comparison between recall and confidence

The comparison between precision and recall is shown in Figure 14. For all classes, every value of recall has high precision. The F1 score is the average value of precision and recall as shown (11). The comparison between F1 and confidence is shown in Figure 15 when predicted data has confidence between 5% and 95% which gives F1 more than 90%. However, precision can be seen as a measure of quality and recall as a measure of quantity. F1 score is the harmonic mean that combines recall and precision. It maintains a balance between recall and precision.

$$\text{precision} = \frac{tp}{tp+fp} \quad (9)$$

$$\text{recall} = \frac{tp}{tp+fn} \quad (10)$$

$$F_1 = \left( \frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} \quad (11)$$

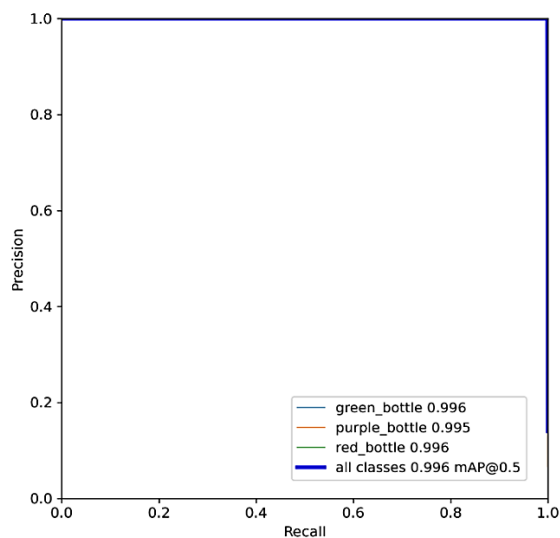


Figure 14. The comparison between precision and recall

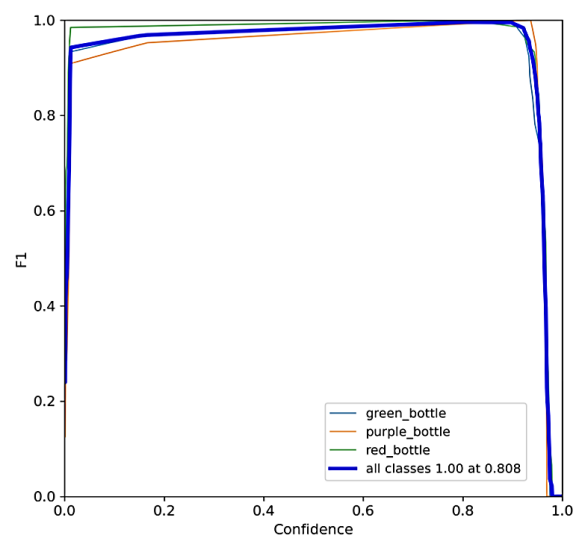


Figure 15. The comparison between F1 and confidence

### 5.3. Robot operation

The operation includes that step involving the robot and vision system. Figure 16 shows the operation of the robot grasping the bottles and moving them to the target location. The steps are related to the postures that some of them are required in order to resolve singularity as follows.

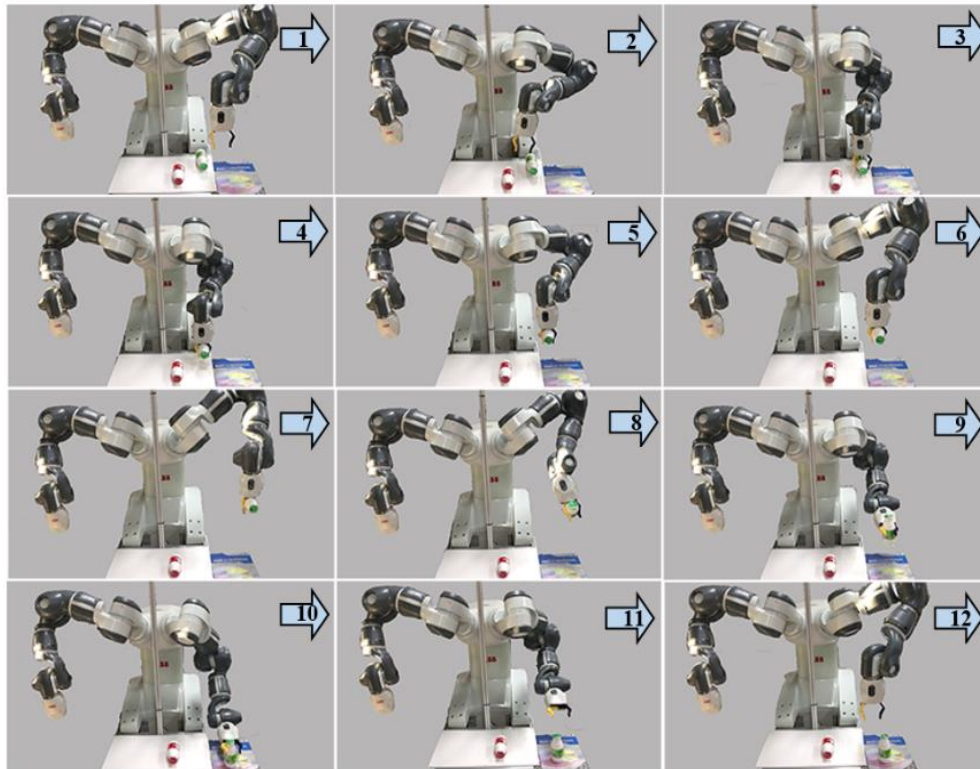


Figure 16. The posture of YuMi robot for grasping object

The experiment was done with 10 images in two conditions, with and without robot operation, under the assumption that YOLOv5 was able to detect the object with 100% accuracy. For the result as shown in Table 1, focusing on the image processing part alone, the average time for classification and localization is 0.014 seconds. Therefore, the framerate is 71.94 fps which is fast enough for real-time estimation. However, when realizing the robot's operation, the average time is 17.32 seconds. The detection process could be a real-time detection. Little time difference occurred due to the different colors and postures of the objects. In Table 2, robot operation took a lot of time due to the physical movement from one position to another. The processing time includes the time for image processing for detection and other computation processes.

Table 1. Usage time of classification and localization without robot in the loop and with robot in the loop

Image No.	Robot operation Time (s)	Processing Time (s)	Total (s)
1	17.00	0.021	17.021
2	17.02	0.019	17.039
3	16.88	0.016	17.04
4	16.98	0.016	17.14
5	17.08	0.016	17.024
6	18.02	0.012	18.032
7	18.01	0.012	18.022
8	17.54	0.012	17.552
9	17.68	0.011	17.79
10	16.95	0.011	16.961
Average	17.32	0.014	17.334
S.D.	0.429	0.0034	0.419

Table 2. The steps of each posture

Step (Posture) No.	Description	Component involved (e.g. arm, gripper, and vision)	End point (x [mm], y [mm], z [mm], Roll [deg], pitch [deg], yaw [deg] )	angle of joint (j1, j2, j3, j4, j5, j6, j7) [deg]
1	Standby posture before grasp	Arm in standby posture before grasp, vision in real-time detect	-	-30, -52, 50, -257, 49, 31, 59
2	Pre-grasp posture	Arm on the object, gripper open, object is detected	x object, y object, z object + 90, 180, 0, 90	-
3	Grasp posture	Arm on the object, gripper close	x object, y object, z object, 180, 0, 90	-
4	Pre-grasp posture	Arm on the object	x object, y object, z object + 90, 180, 0, 90	-
5	Standby posture before grasp	Arm in standby posture before grasp,	-	-30, -52, 50, -257, 49, 31, 59
6	Standby posture before place	Arm in standby posture before place,	-	0, -80, 51, -270, 42, 83, 65
7, 8, 9	Pre-place posture	Arm in pre-place posture	320 - offset, 268, 40 + 120, 90, 0, 90	-
10	place posture	Arm in place posture, gripper open	320 - offset, 268, 40, 90, 0, 90	-
11	Pre-place posture	Arm in pre-place posture	320 - offset, 268, 40 + 120, 90, 0, 90	-
12	Standby posture before grasp	Arm in standby posture before grasp, vision in real-time detect	-	-30, -52, 50, -257, 49, 31, 59
1	Standby posture before grasp	Arm in standby posture before grasp, vision in real-time detect	-	-30, -52, 50, -257, 49, 31, 59
2	Pre-grasp posture	Arm on the object, gripper open, object is detected	x object, y object, z object + 90, 180, 0, 90	-

#### 5.4. Discussion

The validation of the training set shows the performance during the training and validation process of the captured images. The result shows high precision and low error. Meanwhile, the experimental result shows the performance of the detector. The results are compared between precision confidence, recall confidence, and F1 confidence. The comparison shows the performance is high from every perspective. In regard to robot operation, real-time detection is possible due to the short operation time required. However, the robot operation takes a long time in comparison to the detection process alone. Regarding the robot's movement, the endpoints are set as the target position where the robot will grasp the object. However, the singularity can be presented along the way when moving from certain starting points. Therefore, via points are set as the standby posture in which the movement is done with joint coordination.

#### 6. CONCLUSION

A lot of research works currently have implemented industrial robots with vision systems using 3D cameras. AI has increased the application in vision systems for handling more complex situations, such as the detection of objects with variations. YOLO is one of the most popular models for object detection. In this research, YOLOv5 is implemented with the YuMi dual-arm collaborative robot and RealSense RGB-D camera. The color images are used for training and detection while the depth image is used for localization.

From the experiment, the error of the box, objectness, and classification of training and validation were less than 1%, while the precision and recall of training and validation were 100%. This model shows very high performance in object detection. However, the performance can be marginally lower if the ambient conditions, e.g., lighting and shade, affect the objects.

To validate the performance due to the time consumption, processing time was 0.014 seconds on average (~71 frames per second), which is sufficient for real-time detection. The time for robot operation was 17.32 seconds on average. It takes a lot of time as it needs to avoid the singularity by moving to predefined via points as shown in Table 2 in steps 1, 5, 6, and 12.

In conclusion, using YOLOv5 with the YuMi robot could be efficient. The machine learning approach is more flexible than traditional machine vision methods. It achieved high precision, lower error, and high endurance to light variation. This has proved to be sufficient for most industrial applications.

In future work, we would be continuing detection. The next object is uncertain shape and asymmetric. X, Y, Z, and the angle of the object are interested. Precision in detection and grasp is to be experimented.

## ACKNOWLEDGEMENTS




We would like to thank The Royal Golden Jubilee Ph.D. Programme (RGJ Ph.D.) contract number PHD/0202/2561, which is the scholarship supported by Thailand Research Fund (TRF). We also would like to thank ABB Automation (Thailand) Co., Ltd. who support the YuMi robot for being used in this research.

## REFERENCES




- [1] W. Zhang, C. Wu, S. Yang, B. Huang, and D. Wu, "Study on the characteristics of the nuclear reactor coolant pump during the process-divided start-up period," *Annals of Nuclear Energy*, vol. 178, Dec. 2022, doi: 10.1016/j.anucene.2022.109381.
- [2] C. Choi and H. I. Christensen, "3D pose estimation of daily objects using an RGB-D camera," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2012, pp. 3342–3349. doi: 10.1109/IROS.2012.6386067.
- [3] A. Zakhama, L. Charrabi, and K. Jelassi, "Intelligent selective compliance articulated robot arm robot with object recognition in a multi-agent manufacturing system," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, Mar. 2019, doi: 10.1177/1729881419841145.
- [4] D. Kirschner, R. Velik, S. Yahyanejad, M. Brandstötter, and M. Hofbaur, "YuMi, come and play with me! A collaborative robot for piecing together a tangram puzzle," in *Lecture Notes in Computer Science*, Springer International Publishing, 2016, pp. 243–251. doi: 10.1007/978-3-319-43955-6\_29.
- [5] J. Liang *et al.*, "Dual quaternion based kinematic control for Yumi dual arm robot," in *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, Jun. 2017, pp. 114–118. doi: 10.1109/URAI.2017.7992899.
- [6] S.-H. Wu and X.-S. Hong, "Integrating computer vision and natural language instruction for collaborative robot human-robot interaction," in *2020 International Automatic Control Conference (CACS)*, Nov. 2020, pp. 1–5. doi: 10.1109/CACS50047.2020.9289768.
- [7] R. Yang, T. P. Nguyen, S. H. Park, and J. Yoon, "Automated picking-sorting system for assembling components in an IKEA chair based on the robotic vision system," *International Journal of Computer Integrated Manufacturing*, vol. 35, no. 6, pp. 583–597, Jun. 2022, doi: 10.1080/0951192X.2021.1992658.
- [8] P. Opaspirai, S. Vongbunyong, and A. Dheeravongkit, "Robotic system for depalletization of pharmaceutical products," in *2021 7th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*, Apr. 2021, pp. 133–138. doi: 10.1109/ICEAST52143.2021.9426302.
- [9] P. Opaspirai, S. Vongbunyong, and A. Dheeravongkit, "Robotic system for depalletization of pharmaceutical products with 3D camera," in *2021 25th International Computer Science and Engineering Conference (ICSEC)*, Nov. 2021, pp. 422–427. doi: 10.1109/ICSEC53205.2021.9684575.
- [10] J. Lelachacharoeanpan and S. Vongbunyong, "Classification of surgical devices with artificial neural network approach," in *2021 7th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*, Apr. 2021, pp. 154–159. doi: 10.1109/ICEAST52143.2021.9426258.
- [11] J. Hu, Y. Niu, and Z. Wang, "Obstacle avoidance methods for rotor UAVs using RealSense camera," in *2017 Chinese Automation Congress (CAC)*, Oct. 2017, pp. 7151–7155. doi: 10.1109/CAC.2017.8244068.
- [12] D. Pohl, S. Dorodnicov, and M. Achtelik, "Depth map improvements for stereo-based depth cameras on drones," in *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems*, Sep. 2019, pp. 341–348. doi: 10.15439/2019F66.
- [13] L. Francisco and H. Araujo, "Intel RealSense SR305, D415 and L515: experimental evaluation and comparison of depth estimation," in *VISIGRAPP (4: VISAPP)*, 2021, pp. 362–369.
- [14] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: a real-time object detection method for constrained environments," *IEEE Access*, vol. 8, pp. 1935–1944, 2020, doi: 10.1109/ACCESS.2019.2961959.
- [15] J. Du, "Understanding of object detection based on CNN family and YOLO," *Journal of Physics: Conference Series*, vol. 1004, Apr. 2018, doi: 10.1088/1742-6596/1004/1/012029.
- [16] J. Zhang, M. Huang, X. Jin, and X. Li, "A real-time chinese traffic sign detection algorithm based on modified YOLOv2," *Algorithms*, vol. 10, no. 4, Nov. 2017, doi: 10.3390/a10040127.
- [17] J. Sang *et al.*, "An improved YOLOv2 for vehicle detection," *Sensors*, vol. 18, no. 12, Dec. 2018, doi: 10.3390/s18124272.
- [18] P. Mahto, P. Garg, P. Seth, and J. Panda, "Refining Yolov4 for vehicle detection," *International Journal of Advanced Research in Engineering and Technology (IJARET)*, vol. 11, no. 5, 2020.
- [19] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang, "Apple detection during different growth stages in orchards using the improved YOLO-V3 model," *Computers and Electronics in Agriculture*, vol. 157, pp. 417–426, Feb. 2019, doi: 10.1016/j.compag.2019.01.012.
- [20] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," *Electronics*, vol. 9, no. 3, Mar. 2020, doi: 10.3390/electronics9030537.
- [21] B. Benjdira, T. Khurshed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: comparison between faster R-CNN and YOLOv3," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, Feb. 2019, pp. 1–6. doi: 10.1109/UVS.2019.8658300.
- [22] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 1440–1448. doi: 10.1109/ICCV.2015.169.
- [23] A. Kuznetsova, T. Maleva, and V. Soloviev, "Detecting apples in orchards using YOLOv3 and YOLOv5 in general and close-up images," in *Advances in Neural Networks ISNN 2020*, Springer International Publishing, 2020, pp. 233–243. doi: 10.1007/978-3-030-64221-1\_20.
- [24] G. Yang *et al.*, "Face mask recognition system with YOLOV5 based on image recognition," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, Dec. 2020, pp. 1398–1404. doi: 10.1109/ICCC51575.2020.9345042.
- [25] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 292–301. doi: 10.1109/CVPR.2018.00038.
- [26] D. Dlužnevskij, P. Stefanović, and S. Ramanauskaitė, "Investigation of YOLOv5 efficiency in iPhone supported systems," *Baltic Journal of Modern Computing*, vol. 9, no. 3, 2021, doi: 10.22364/bjmc.2021.9.3.07.



**BIOGRAPHIES OF AUTHORS**

**Dumrongsak Kijdech**    is a lecturer at Mechanical Engineering, Southeast Asia University, Bangkok, Thailand. He holds a Bachelor of Engineering in Mechanical Engineering from Rajamangala University of Technology Phra Nakhon, Thailand; a Master of Engineering in Mechanical Engineering from Kasetsart University, Thailand; currently a Ph.D. student at the Institute of Field Robotics, King Mongkut's University of Technology Thonburi, Thailand. His main research directions include artificial intelligence (YOLOv3, YOLOv5, CNN, Mask R-CNN), Automatic control, smart farm, image processing, and robot control. He can be contacted at [dumrongsakk@sau.ac.th](mailto:dumrongsakk@sau.ac.th).



**Supachai Vongbunyong**    is an assistant professor at the Institute of Field Robotics (FIBO), King Mongkut's University of Technology Thonburi, Bangkok, Thailand. He holds a Bachelor of Engineering and Master of Engineering in Mechanical Engineering from Chulalongkorn University, Thailand, and a Ph.D. in Manufacturing Engineering from UNSW. He started his post-doc research as a research associate in 2013 at the University of New South Wales (UNSW) in the School of Mechanical and Manufacturing Engineering. His expertise is in robotics and automation. He is a co-founder of the Innovation and Advanced Manufacturing Research Group and a founder of the Hospital Automation Research Center at FIBO. He can be contacted at [supachai.von@kmutt.ac.th](mailto:supachai.von@kmutt.ac.th).