Journal of
Marine Science and Technology

Volume 26 | Issue 3 Article 10

AUTOMATIC CLASSIFICATION OF MOVING OBJECTS ON AN UNKNOWN SPEED PRODUCTION LINE WITH AN EYE-IN-HAND ROBOT MANIPULATOR

Jinsiang Shaw
Department of Mechanical Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C., jshaw@ntut.edu.tw

Wen-Lin Chi
Institute of Mechatronic Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C

Recommended Citation
Shaw, Jinsiang and Chi, Wen-Lin (2018) "AUTOMATIC CLASSIFICATION OF MOVING OBJECTS ON AN UNKNOWN SPEED PRODUCTION LINE WITH AN EYE-IN-HAND ROBOT MANIPULATOR," *Journal of Marine Science and Technology*: Vol. 26: Iss. 3, Article 10.
DOI: DOI: 10.6119/JMST.201806_26(3).0010
Available at: https://jmstt.ntou.edu.tw/journal/vol26/iss3/10

Journal of Marine Science and Technology, Vol. 26, No. 3, pp. 387-396 (2018)
DOI: 10.6119/JMST.2018.06_(3).0010

AUTOMATIC CLASSIFICATION OF MOVING OBJECTS ON AN UNKNOWN SPEED PRODUCTION LINE WITH AN EYE-IN-HAND ROBOT MANIPULATOR

Jinsiang Shaw[1] and Wen-Lin Chi[2]

Key words: image-based visual servoing (IBVS), CAMshift, Kalman filter.

ABSTRACT

In this paper, techniques for tracking and grasping moving objects with an unknown speed on a conveyor using an eye-in-hand robot arm are presented, which are useful in a production line for automatic object classification. First of all, the CAMshift (Continuously Adaptive Meanshift) algorithm is employed to continuously track a moving object in the image plane. Then, the minimum area rectangle method is integrated for correctly identifying a rectangle enclosing the target object. Object features for tracking purposes can be extracted from this rectangle. Next, through the application of an image Jacobian matrix, the tracking error in the image plane can be transformed to be the displacements of the robot's end effector. Accordingly, the robot arm can be controlled for tracking this object. However, because of sensor noise and the fact that the object is moving, tracking errors cannot be eliminated at this stage. Therefore, the Kalman filter is used to estimate the state of the moving object, especially the moving speed. Finally, on the basis of the estimated speed, the robot gripper can thus be controlled to the point on the conveyor for accurately grasping and placing the moving object to a specified location. Experimental results showed the effectiveness of the techniques for grasping different target objects with different moving speeds and at any orientations.

I. INTRODUCTION

The eye-in-hand architecture of a robot arm is adopted primarily to guide the end effector and to ensure that the tool can properly engage the intended target. The advantages of an eye-in-hand configuration include higher flexibility, higher accuracy, and occlusion avoidance. Substantial research studies have been done recently for tracking static objects via an eye-in-hand configuration (Fang et al., 2009; Lazar and Burlacu, 2009; Huang et al., 2013; Hajiloo et al., 2016; Van et al., 2016). On the other hand, an eye-to-hand configuration has been used in some research studies for grasping moving objects through binocular vision (Allen et al., 1993; Fuentes-Pacheco et al. 2009; Zhang and Shen, 2014) or Kinect vision sensors (Husain et al., 2014; Suzuki et al., 2015) to acquire the depth information between the camera and the object. The aim of this study was to apply an eye-in-hand robot arm in a production line with an unknown speed to automatically grasp and place the coming objects into groups for classification purposes. A two-finger gripper module with a camera installed inside is attached to the robot's end effector for the tasks mentioned here.

Visual servoing plays an important role in robot applications. It makes robots more intelligent and flexible. The goal of this task is to calculate the control input to the robot system so that the error between the desired signal and the feedback signal, which is extracted from a vision sensor, can converge to zero. Generally, visual servoing can be classified into three types: position-based visual servoing (PBVS), image-based visual servoing (IBVS), and hybrid visual servoing (Malis et al., 1999; Corke and Hutchinson, 2001; Nobakht and Liu, 2015). Among them, IBVS is an important control technique often used for solving complex control problems of a six-degrees-of-freedom robot arm for object tracking. With the use of an image Jacobian matrix, the image plane tracking errors can be transformed into errors in the Cartesian space. However, IBVS has problems in tracking objects with large angles and displacement motions (Chaumette, 1998). In this study, the angle of an object was compensated first by aligning the robot's gripper with the object's long edge. Then, IBVS was initiated to keep tracking the moving object using an eye-in-hand configuration so that the object could be kept in the field of view.

For the robot arm to grasp moving objects on a conveyor,

Paper submitted 12/08/17; revised 03/05/18; accepted 05/08/18. Author for correspondence: Jinsiang Shaw (e-mail: jshaw@ntut.edu.tw).
[1] Department of Mechanical Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C.
[2] Institute of Mechatronic Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C.

encoder information from the conveyor motor is usually used to acquire the velocity of the objects (Wang et al., 2015; Lin et al., 2016). Furthermore, an eye-to-hand configuration is often adopted to track the trajectories of the object (Husain et al., 2014; Zhang and Shen, 2014). In this study, an eye-in-hand configuration was used and the space position of the object was estimated through the image information. Moreover, the Kalman filter was employed to acquire the velocity of the moving object. In image processing for extraction of important image features, algorithms such as Speeded-Up Robust Features (SURF) (Wang et al., 2015), support vector machines (Zhang and Shen, 2014), or optical flow (Allen et al., 1993) are often used. In this study, an object tracking algorithm (i.e., CAMshift) proposed by Bradski (1998) was employed, which uses a color histogram as its target model. This algorithm may not be easily influenced by the changes in the target shape. It uses color probability so that it can efficiently solve the problem of the target being in motion or partly sheltered. Although CAMshift can rapidly achieve the purpose of the target tracking, it only relies on back-projection. Therefore, it would fail in some cases, e.g., the object's color is changed by the environment or similar-color objects are detected, which can influence the result. In addition, it would lose the target. Therefore, Joshi et al. (2004) improved CAMshift with the SURF algorithm to make it more robust. In this study, for the sake of robustness, the minimum area rectangle method in EmguCV was integrated with CAMshift owing to its simplicity and easy implementation.

The rest of the paper is organized as follows. In Section II, digital image processing methods are introduced. Then, both visual servoing and the Kalman filter are described in Section III. In Section IV, experimental results of grasping moving objects on a conveyor are shown. Finally, the conclusion and future work are given in Section V.

II. IMAGE PROCESSING

A six-axis robot arm (TX60L) from Staubli, with a gripper module having a camera inside attached to its end effector, was used as the eye-in-hand robot manipulator in the study. For the Staubli robot manipulator, only the point-to-point control command with blending move is available to the user, and, hence, it will be used here to track moving objects on a conveyor. From the captured images, digital image processing techniques are applied for detecting the object and extracting the object's features for tracking purposes.

In this study, two major methods were used in the digital image processing. The first one was the CAMshift algorithm, and the other was the minimum area rectangle method. To realize the automatic tracking of a moving object on a conveyor, it is imperative to detect whether an object is on the conveyor. Note that background subtraction is a technique in the field of image processing for extracting the foreground (the target object). It is a widely used approach for detecting moving objects in videos from static cameras. Therefore, in the beginning of the process when the robot arm is at the start point, background subtraction is first used to detect whether an object is moving on the conveyor. If so, the resulting image of the object from the background subtraction further goes through a binarization process and then a gravity method is employed to acquire the center coordinates of the moving object. The detected object is then tracked continuously using both the CAMshift algorithm and the minimum area rectangle method when moving on the conveyor.

1. CAMshift Algorithm

CAMshift is a well-known object-tracking algorithm. When a target is chosen, the tracking window would keep following it. CAMshift uses color information to track the moving target. It is mainly based on back-projection and the mean shift algorithm (or called Meanshift). Meanshift is a method which can find the local maximum of the probability function using iteration approach for one picture, but CAMshift can work in a sequence of images. CAMshift uses a color histogram as its target model. The extracted image contains many numbers of pixels, and each pixel has associates with a set of values for the hue, saturation, and value (HSV) components. Through back- projection, CAMshift can generate a color probability of this image from the hue histogram distribution. Then, it would produce a grayscale image, which means that the lighter pixels most possibly belong to the target model. In addition, CAMshift uses a rectangular tracking window in EmguCV, which is a cross-platform .NET wrapper for the OpenCV image processing library, to highlight the target object. Calculation of the window's location is an iterative and converging process. Each time, the new window is computed according to the window of the last frame. The location of the new window is determined by the difference between the two windows, and it needs to be smaller than the preset threshold. Fig. 1 shows a block diagram of the CAMshift algorithm.

The following are some formulas that show how to calculate the center's coordinates of the tracking window:

0-order moment:

$$M_{00} = \sum_x \sum_y I(x, y)$$

1-order moment:

$$M_{01} = \sum_x \sum_y yI(x, y)$$
$$M_{10} = \sum_x \sum_y xI(x, y)$$

where $I(x, y)$ is the value of each pixel in the back-projection. Then, the center's coordinates (x_c, y_c) are given as

$$x_c = M_{10} / M_{00},\ y_c = M_{01} / M_{00} \tag{1}$$

The coordinates of the center point are used to build a rectangle as the tracking window, with length L and width W given as

$$W = 2\sqrt{M_{00} / 256} \tag{2}$$

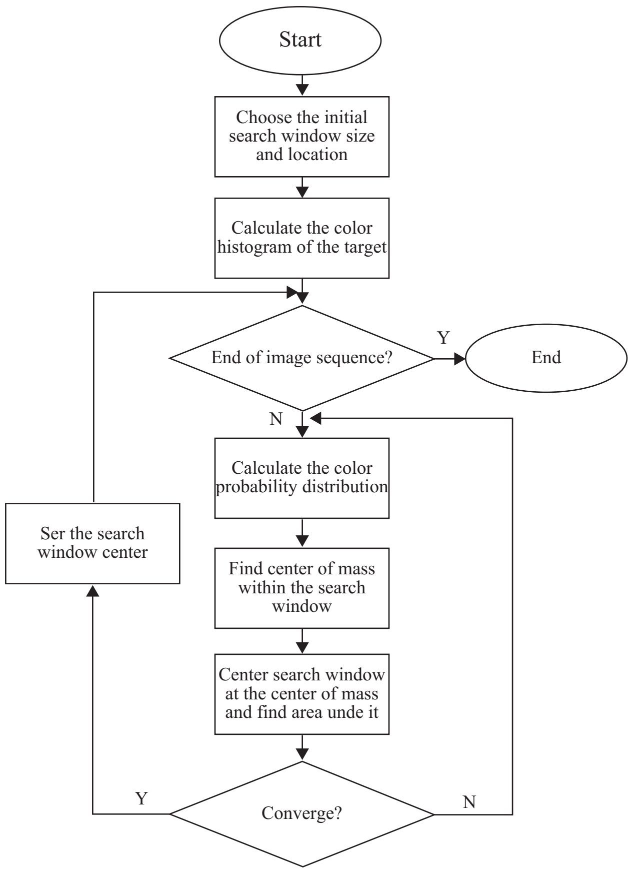
Start
Choose the initial search window size and location
Calculate the color histogram of the target
End of image sequence?
Y
End
N
Calculate the color probability distribution
Find center of mass within the search window
Center search window at the center of mass and find area unde it
Converge?
Y
N
Ser the search window center

Fig. 1. Block diagram of the CAMshift algorithm.

$$L = 1.2\,W \tag{3}$$

As mentioned earlier, background subtraction can detect an initial target object on the conveyor. With the location of the target object and a suitable size for the initial search window, CAMshift is then applied for tracking the object moving with the conveyor.

2. Minimum Area Rectangle Method

Once CAMshift is working, a tracking window, which is a rectangle, can be acquired. However, this tracking window cannot represent the target itself well, especially when the object has a certain inclination. But we do know where the target is (inside the tracking window), and it can keep track of the target in the image plane. The minimum area rectangle method which is a method from OpenCV library (opencv.org, 2018) can generate another bounding rectangle with a minimum rectangle area and any inclination by a set of points. However, using only back-projection, it may have some noises, which is the problem with color probability. Therefore, we combine the tracking windows from the CAMshift algorithm and the minimum area rectangle using the intersection operation to generate a new rectangle that is more robust and has useful information about the object. More specifically, any object point (x, y) inside the minimum area rectangle is a true point of the object as long as

$$(x - x_c)^2 + (y - y_c)^2 \le (\frac{L}{2})^2 + (\frac{W}{2})^2 \tag{4}$$

Fig. 2 illustrates results the application of the CAMshift and minimum area rectangle algorithms to a target object. It can be seen clearly that, if only the CAMshift algorithm is used, the tracking window is not good enough to represent the object (red rectangle in Fig. 2(a)). If only the minimum area rectangle method is applied, the rectangle would be easily influenced by noises. However, if both methods are combined, the result is better (the red rectangle in Fig. 2(b)).

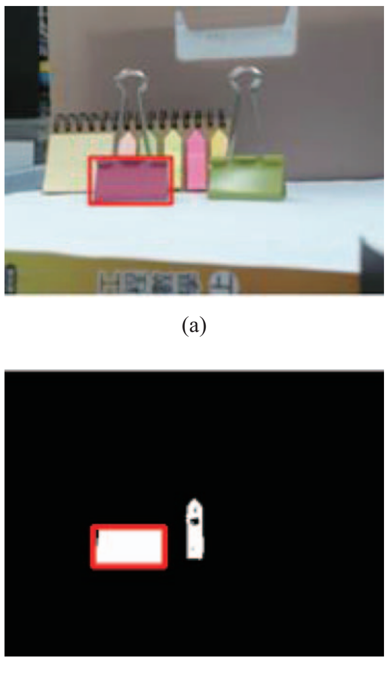

Fig. 2. Tracking of an object: (a) tracking window from CAMshift; (b) combined results of CAMshift and the minimum area rectangle method.

III. CONTROLLER DESIGN

1. Visual Servoing

In this study, image-based visual servoing was employed, which is based on the error between the current and the desired feature points on the image plane. The goal of the visual servoing process is to minimize the image error e(k), so that the current image feature $f(k)$ can reach the desired feature points $f_d(k)$:

$$e(k) = f_d(k) - f(k) \tag{5}$$

In the IBVS structure, there is a feature space control law, which uses the image error $e(k)$ as input and generates the corresponding robot control command. In the following compu-

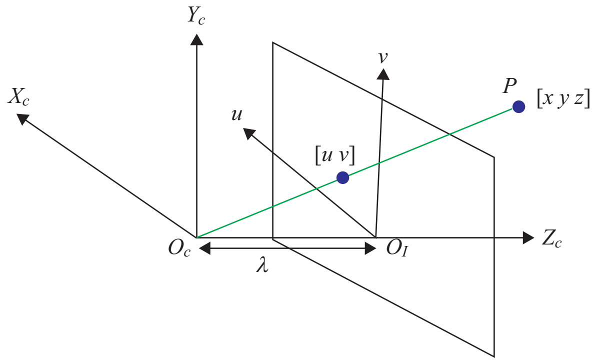

Fig. 3. Perspective projection model (O_c is the origin of the camera frame and O_I is the origin of the image plane).

tation, there is a relationship between the changing rate of the feature point in the image plane and the end effector velocity in the task space. First, assume the camera is in a static environment. When the camera moves in the task space with a linear velocity $T = [T_x, T_y, T_z]^T$ and angular velocity $\Omega = [\omega_x, \omega_y, \omega_z]^T$, the velocity of a point $P = [x, y, z]^T$ relative to the camera frame can be expressed as

$$\frac{dP}{dt} = -\Omega \times P - T \tag{6}$$

Using the classical perspective projection model in Fig. 3, one obtains

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{\lambda}{z}\begin{bmatrix} x \\ y \end{bmatrix} \tag{7}$$

where λ is the focus distance, (u, v) is the feature point coordinate in the image plane, and (x, y) is the coordinate in space. In Eq. (7), the unit of (u, v) is length, but we prefer (u, v) in the unit of pixel. To transform the former to pixel units, we rewrite Eq. (7) as

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{\lambda}{z}\begin{bmatrix} x/s_x \\ y/s_y \end{bmatrix} \tag{8}$$

where s_x and s_y are the transformation coefficients of the x and y axis in the image plane, respectively. s_x, s_y, and λ are associated camera parameters, which can be measured by experiments. To simplify Eq. (8), we define

$$\lambda_x = \lambda / s_x,\ \lambda_y = \lambda / s_y \tag{9}$$

Eq. (8) now becomes

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{z}\begin{bmatrix} x\lambda_x \\ y\lambda_y \end{bmatrix} \tag{10}$$

The relationship between an image point and the velocity of P can be calculated according to Eq. (6) and Eq. (10). The derivative of the coordinate of P in terms of image feature coordinate (u, v) is

$$\begin{aligned} \dot{x} &= -z\omega_y + \frac{zv}{\lambda_y}\omega_z - T_x \\ \dot{y} &= -\frac{zu}{\lambda_x}\omega_z + z\omega_x - T_y \\ \dot{z} &= -\frac{zv}{\lambda_y}\omega_x + \frac{zu}{\lambda_x}\omega_y - T_z \end{aligned} \tag{11}$$

On the other hand, taking the derivative of Eq. (10) and assuming z is constant (as the case in our application), one obtains the derivative of the coordinates of the feature parameters using Eq. (11):

$$\begin{aligned} \dot{u} &= \frac{-\lambda_x}{z}T_x + \frac{u}{z}T_z + \frac{uv}{\lambda_y}\omega_x - \frac{\lambda_x^2 + u^2}{\lambda_x}\omega_y + \frac{\lambda_x}{\lambda_y}v\omega_z \\ \dot{v} &= \frac{-\lambda_y}{z}T_y + \frac{v}{z}T_z + \frac{\lambda_y^2 + v^2}{\lambda_y}\omega_y - \frac{uv}{\lambda_x}\omega_y - \frac{\lambda_y}{\lambda_x}u\omega_z \end{aligned} \tag{12}$$

We can rewrite Eq. (12) in vector-matrix form as

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{-\lambda_x}{z} & 0 & \frac{u}{z} & \frac{uv}{\lambda_y} & \frac{-\lambda_x^2 - u^2}{\lambda_x} & \frac{\lambda_x}{\lambda_y}v \\ 0 & \frac{-\lambda_y}{z} & \frac{v}{z} & \frac{\lambda_y^2 + v^2}{\lambda_y} & -\frac{uv}{\lambda_x} & -\frac{\lambda_y}{\lambda_x}u \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{13}$$

In this paper, we need to control the robot manipulator to grasp the target object in three-dimensional space; hence, at least two feature points are required in the image plane. Here, the two feature points at the middle points in the two long sides of the rectangle are chosen. In addition, we further assume that $T_z = \omega_x = \omega_y = 0$ for in-plane object tracking. Consequently, the final feature points in the image plane can be obtained as

$$\begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \dot{u}_2 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} \frac{-\lambda_x}{z} & 0 & \frac{\lambda_x}{\lambda_y}v_1 \\ 0 & \frac{-\lambda_y}{z} & -\frac{\lambda_y}{\lambda_x}u_1 \\ \frac{-\lambda_x}{z} & 0 & \frac{\lambda_x}{\lambda_y}v_2 \\ 0 & \frac{-\lambda_y}{z} & -\frac{\lambda_y}{\lambda_x}u_2 \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ \omega_z \end{bmatrix} \tag{14}$$

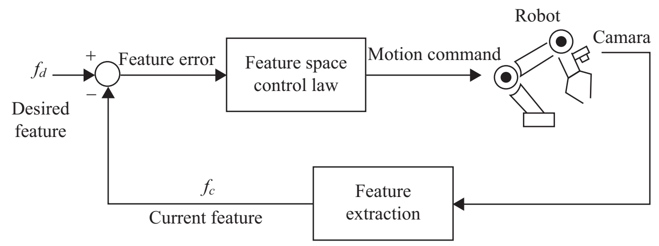

Fig. 4. Image-based visual servoing control.

Eq. (14) can be represented as

$$\dot{f} = J_{img}V_c \tag{15}$$

where $\dot{f}$ is the velocity vector of the feature points, J_{img} is the image Jacobian matrix, and V_c is the velocity of the camera. In this study, to compensate for the image error $e(k)$ in Eq. (5), we set $\dot{f} \doteq \Delta f / \Delta t = e(k)/\Delta t$ and solve for the camera velocity in Eq. (15):

$$V_c = J_{img}^{+}\dot{f} \tag{16}$$

where

$$J_{img}^{+} = (J_{img}^{T}J_{img})^{-1}J_{img}^{T} \tag{17}$$

It is noted that the velocity of the camera is equal to the velocity of the end effector because the tool frame and the camera frame are moving together. Next, the velocity of the end effector can be transformed to be the displacement of the end effector during a sample period. Then, on the basis of the forward kinematics of the robot arm, the displacement of the end effector can be transformed to be the displacement of the end effector in the robot base frame. Finally, with the use of the inverse kinematics, the robot motion control command can be issued to compensate for the image error for continuous object tracking. Fig. 4 shows a block diagram for the image-based visual servoing control.

2. Kalman Filter

The abovementioned object tracking algorithm using IBVS will not perform well for two reasons: (1) The object is not stationary, but is moving with the conveyor at a certain speed. When the robot controller gets the motion control command and moves the robot arm to the desired location, the object has already moved to a new location. (2) For the Staubli robot manipulator used in the study, only the point-to-point position control is available. Consequently, the robot manipulator tries to follow the moving object, but is always left behind with a constant stop-and-go maneuvering. To solve these problems, we require that the speed of the moving object be known for predicting the future object position and that blending move be included in the point-to-point robot arm control for a smooth and continuous tracking.

To estimate the moving object's speed, we adopted the Kalman filter here because of its robustness against sensor noise and disturbance, e.g., the noises in the captured image and the inaccuracies in the camera parameters and in the pseudo-inverse of the Jacobian matrix in (17). This algorithm uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone. The Kalman filter is also a recursive estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. Unlike in batch estimation techniques, no history of observations or estimates is required for the Kalman filter. The state of the process is represented by two variables. One is $\hat{x}_{k|k}$, which is the state estimate at time k given the observations up to and including time k. The other is $P_{k|k}$, which is the error covariance matrix. The Kalman filter works in a two-step process: "predict" and "update."

Predict: The predict process uses the state estimate from the previous time step to produce an estimate of the state at the current time step.

Predicted state estimate:

$$\hat{x}_{k|k-1} = F_k\hat{x}_{k-1|k-1} + B_k u_k \tag{18}$$

Predicted covariance estimate:

$$P_{k|k-1} = F_k P_{k-1|k-1}F_k^T + Q_k \tag{19}$$

where F_k is the state transition model that is applied to the previous state $\hat{x}_{k-1}$, B_k is the control-input model that is applied to the control vector u_k, and Q_k is the covariance of the process noise.

Update: In the update process, the current prediction is combined with the current observation information to refine the state estimate. This improved estimate is termed the updated state estimate.

Measurement residual:

$$\tilde{y}_k = z_k - H_k\hat{x}_{k|k-1} \tag{20}$$

Innovation covariance:

$$S_k = H_k P_{k|k-1}H_k^T + R_k \tag{21}$$

Optimal Kalman gain:

$$K_k = P_{k|k-1}H_k^T + S_k^{-1} \tag{22}$$

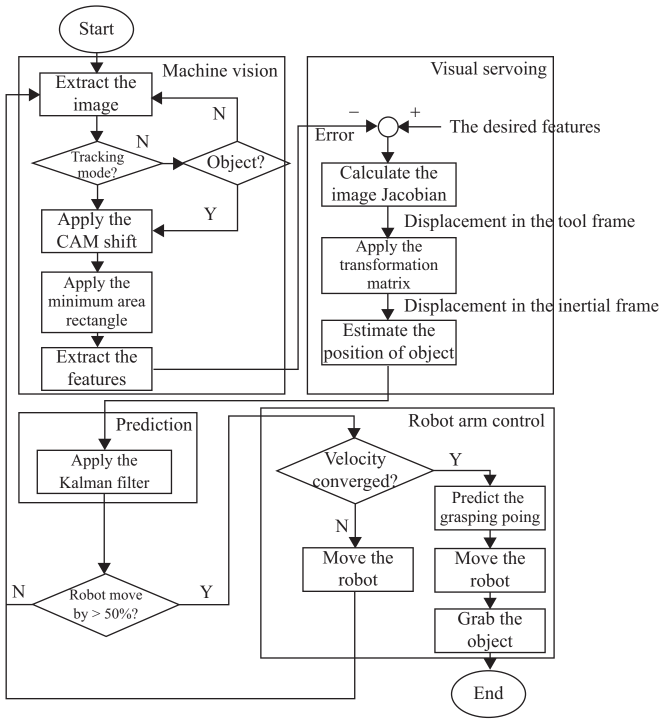

Fig. 5. Block diagram of the grasping system.

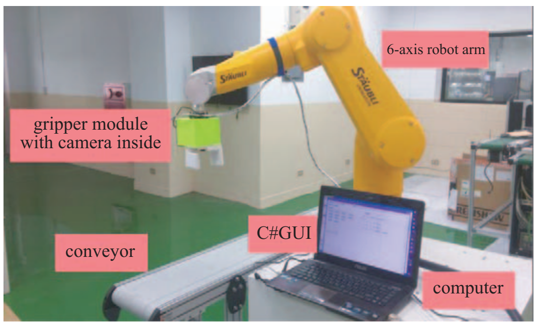

Fig. 6. The experimental setup for object tracking and grasping.

Updated state estimate:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k \tag{23}$$

Updated covariance estimate:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{24}$$

where z_k is the observation at time k, H_k is the observation model, R_k is the covariance of the observation noise, and I is the identity matrix.

In this study, the measurement for the Kalman filter was the coordinates of the object from IBVS. After the iterations of the Kalman filter, the state of the object converges to a steady state, which has information about the object position and velocity. By using the information of the estimated velocity, we can predict the object motion and, thus, the robot arm can be controlled in advance for tracking and grasping the moving object.

For a smooth and continuous tracking of the moving object, blending move will be included also in the point-to-point robot arm control. In this way, non-smooth stop-and-go robot movement can be avoided. In addition, because the camera extracts the image much faster than the robot movement, only when the current movement of the robot arm is more than 50% finished will the new robot command control be issued to the robot arm. Finally, as long as the robot manipulator is catching up with the object speed and they move synchronously, the robot manipulator will start to approach the object toward the conveyor in the z direction. On the basis of the estimated object speed, the robot manipulator can reach the point on the conveyor at just the right time that the object gets there with its two-finger gripper open wide enough to grasp the object. The grabbed object can be placed at a designated location according to the type of the object for classification purposes. After the introductions of machine vision-based digital image processing, visual servoing, Kalman filter, and the grasping procedure, the overall flowchart for object tracking and grasping on a conveyor using an eye-in-hand robot manipulator is shown in Fig. 5. The integrated control algorithms based on this flowchart were implemented on a laptop computer using the C# programming language.

IV. EXPERIMENTAL RESULTS

The experimental setup for the tracking and grasping of moving objects on an unknown speed conveyor using an eye-in-hand robot manipulator is shown in Fig. 6. At the robot arm end effector, a two-finger gripper module with a camera inside is attached for tracking and grasping the moving object. An unknown target object is placed on the left end of the conveyor and moves to the right at an unknown conveyor speed. The conveyor is in the x-y plane of the robot base frame with the x axis parallel to the conveyor. Initially, the robot arm is stretched to the left end of the conveyor at its center with a height of 390 mm in the z axis from the camera to the conveyor. The camera will keep monitoring the conveyor for object detection using background subtraction. To avoid the drawback of IBVS when the target object is randomly placed with a large angle orientation, the gripper module will rotate first about the z axis to align with the long edges of the object before it tracks the target. Then, the following whole tracking process is in the x-y plane. The final grasping and placement of the object have to be done within the workspace of the robot arm.

In the IBVS setting, the image has a size of 320×240 pixels and is captured and processed every 110 ms. The camera parameters λ_x, λ_y in Eq. (10) can be experimentally obtained by calibration: first, an object with 56 mm long is placed in front of the camera at a distance of 380 mm. Then the object images are captured respectively with the object horizontally and vertically placed. Finally, measure the corresponding pixels of the object in the image plane in x and y axis respectively, and are found to be 50 pixels for both directions. By Eq. (10), λ_x, λ_y

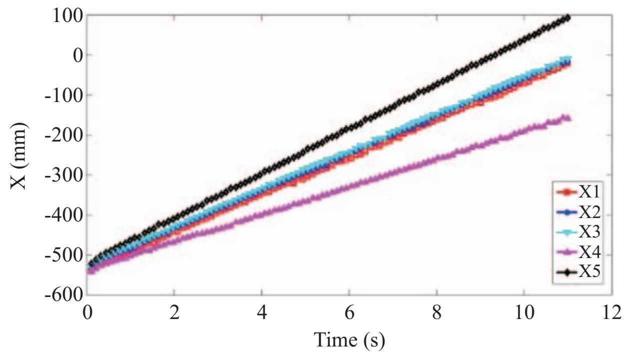

Fig. 7. Estimation of the x coordinate of an object.

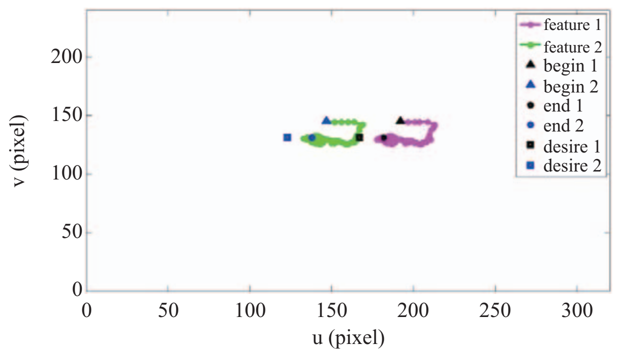

Fig. 9. Loci of the feature points in the image plane.

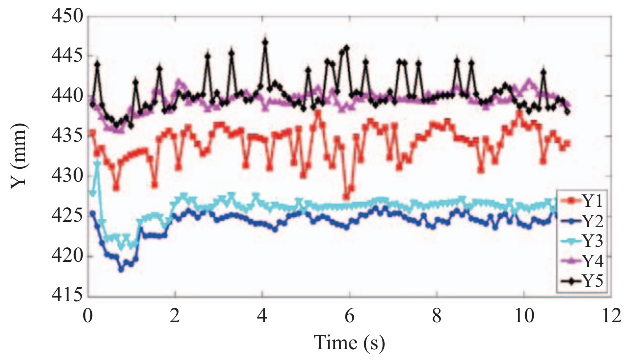

Fig. 8. Estimation of the y coordinate of an object.

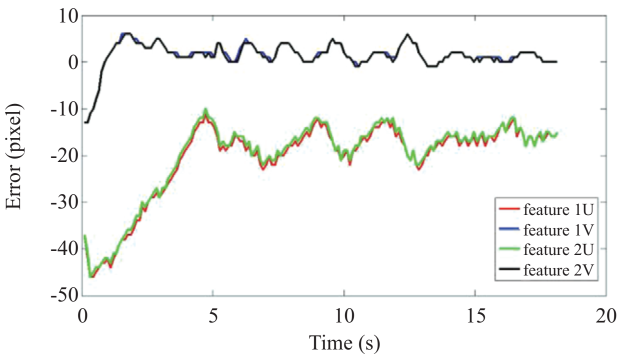

Fig. 10. Time response of the image error.

can be solved to be 339.29 pixels, respectively. The two desired feature points in the image plane $f_d(k)$, which are also the desired locations of the robot's two fingertips so that the gripper can grasp the target, are set near the image center point at coordinates $(145 \pm \frac{d}{2}, 131)$, where d is the width of the enclosing rectangle obtained online using both the CAMshift algorithm and the minimum area rectangle method. The convergent condition of CAMshift algorithm for continuously tracking a moving object is met if the distance between previous window and new window in each iteration is smaller than 1 pixel or the number of iterations reaches 10.

In the following experiments, four different cases are discussed. For the first case, only the IBVS method was applied to track a moving object. In the other three cases, the Kalman filter was included for predicting the velocity of the target so that tracking and grasping of the object could be successfully achieved within the workspace of the robot manipulator.

Case 1: Tracking Using Only the IBVS Method

In this case, there were five experiments conducted. Experiments 1, 2, and 3 had the same object speed, but with different y locations to start with. The purpose was to evaluate the tracking performance of the IBVS method. In the meantime, experiments 4 and 5 had a slower and faster object speed, respectively, compared to experiments 1, 2, and 3. The x-y coordinates of the moving object during the tracking process were calculated on the basis of the IBVS result and the location of the robot end effector, as shown in Figs. 7 and 8, respectively. In Fig. 7, experiments 1, 2, and 3 indeed had the same object speed in the x direction (the same slopes), whereas experiments 4 and 5 had a slower (smaller slope) and faster (larger slope) object speed, respectively. Likewise, the y coordinate of the moving object is illustrated in Fig. 8, where a small oscillation about the true y coordinate can be clearly seen for each experiment owing to the noises in the image processing and to the computations of the inverse Jacobian matrix and robot kinematics. These small oscillations in the y coordinate can be neglected by verifying that the IBVS method can indeed track the moving object quite well in both the x and the y direction without losing the object and divergence.

However, a drawback of the IBVS method can be easily noticed by looking at the responses in the image coordinate for one of the experiments, say, experiment 1, as an example. Figs. 9 and 10 show the responses in the u-v image plane and the time response of the image errors, respectively. In Fig. 9, the loci of the two feature points of the object in the image plane during the whole tracking process clearly indicate that the robot arm kept tracking the object well and did not lose it (without divergence). Specifically, the final tracking errors were within a small bound of 15 pixels in the u direction and nearly zero pixels in the v direction. However, the final tracking error in u was constant and not converging to zero, meaning the robot arm could not catch up with the moving object and was always left behind the object. The same tracking errors in the u and v directions in

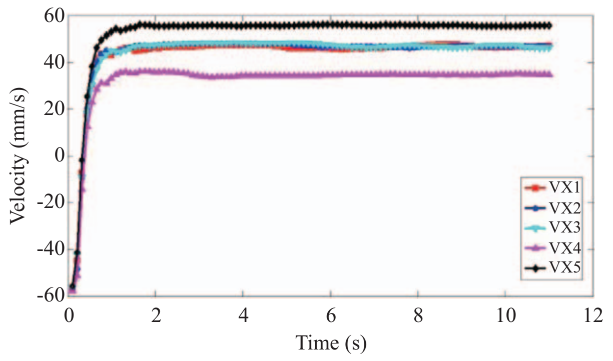

Fig. 11. Estimation of the *x* velocity of an object by the Kalman filter.

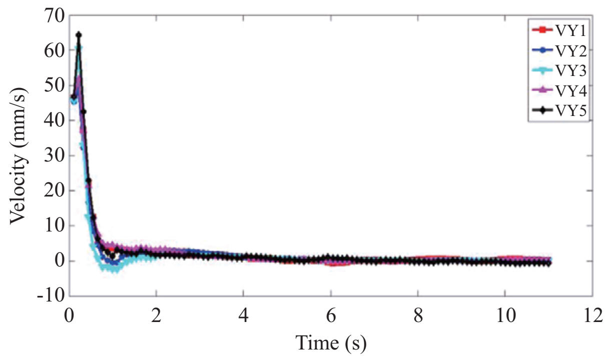

Fig. 12. Estimation of the *y* velocity of an object by the Kalman filter.

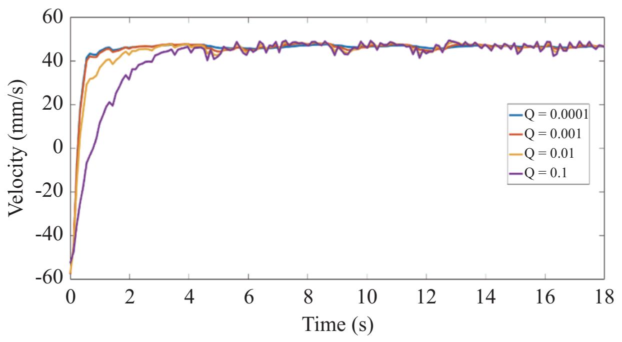

Fig. 13. Estimation of the *x* velocity of an object using different process noise parameter *Q* by the Kalman filter.

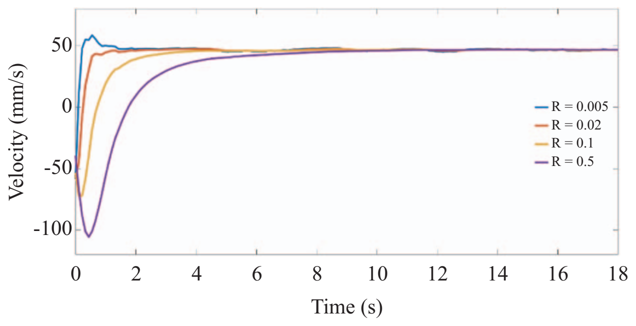

Fig. 14. Estimation of the *x* velocity of an object using different observation noise parameter R_k by the Kalman filter.

the time domain of the two feature points are shown in Fig. 10, where error convergence to zero in the *v* direction and a constant final error bound in the *u* direction can again be easily observed. Therefore, the robot manipulator could not track and grasp the moving object on the conveyor using only the IBVS method.

For this case, if the Kalman filter was included following the IBVS method to estimate the state of the moving object against all types of noise inherent in this application, the resulting estimated *x* and *y* velocities for each experiment are shown in Figs. 11 and 12, respectively. Although there did exist certain noises in the measurement or computation, the estimated velocity still converged to a constant value in both the *x* and the *y* direction (indeed, the object was moving on the conveyor with a constant speed in the *x* direction only). Moreover, the estimated velocity was quick to converge and, thus, can be used for designing the ensuing tracking and grasping algorithms for the moving object when it is still within the robot arm workspace.

The Kalman filter having the performances in Figs. 11 and 12 are tuned experimentally. The following parameters are used for this filter:

The initial state estimate and the error covariance matrix are chosen for simplicity as

$$\hat{x}_{0|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; P_{0|0} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

In fact, the values used are not important because they would converge to real values after a few iterations. Moreover, because the object is moving with the conveyor, thus both having the same velocity, the control vector u_k is therefore identically zero. As for the covariance of the process noise Q_k and the covariance of the observation noise R_k, different values are tried experimentally. The covariance of the process noise Q_k is not measurable. We assume that there is a small noise in the process but it cannot be zero. Assuming $Q_k = Q_I$ be a diagonal matrix with diagonal element Q, Fig. 13 shows convergences of the estimated x velocity using different diagonal element Q. The case with $Q = 0.0001$ has a faster and smoother response as compared to other Q values. For the observation noise R_k, different values are tested. Again we assume that there is a small noise in the observation but it cannot be zero. Fig. 14 depicts convergences of the estimated x velocity using different observation noise parameter. The case with $R_k = 0.02$ has the best result with no overshoot and fastest settling time and the least SAE (sum of absolute error) from the true velocity value. Consequently,

$$Q_k = \begin{bmatrix} 0.0001 & 0 \\ 0 & 0.0001 \end{bmatrix}, \; R_k = 0.02$$

are chosen for the designed Kalman filter, which produces quite satisfactory performances in estimating the x and y axis velocities as shown in Figs. 11 and 12, respectively.

Table 1. Experiment results for grasping two different objects.

Experiment number	Conveyor speed (mm/s)	Kalman Speed (mm/s)	Error of speed	Total time (s)
1 (eraser)	46.92	48.28	2.90%	4.56
2 (poker box)	46.92	48.12	2.56%	4.54

Table 2. Experiment results with different object speeds.

Experiment number	Conveyor speed (mm/s)	Kalman Speed (mm/s)	Error of speed	Total time (s)
1	56.30	56.11	0.34%	5.25
2	46.92	49.1	4.65%	4.67
3	34.40	36.38	5.76%	4.48
4	15.64	15.23	2.62%	7.66

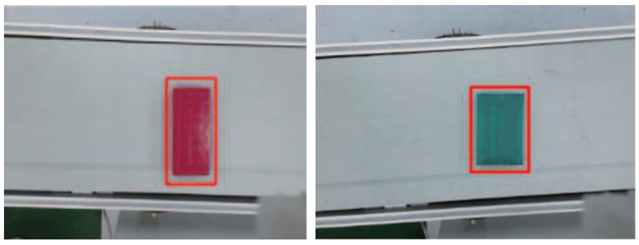
Fig. 15. Different objects to be grasped: blackboard eraser (left) and poker box (right).

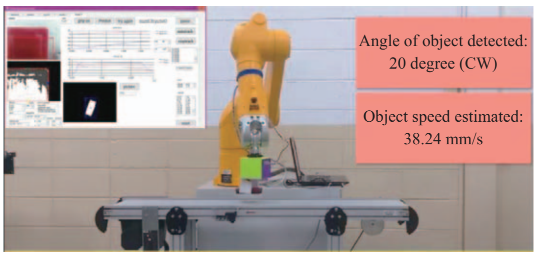

Fig. 16. Snapshot photo during experiment.

Case 2: Grasping Different Objects for Classification

In this case, the robot arm tried to grasp different objects. On the basis of the data base of the color histogram model for each object, the system can automatically identify the moving objects by back-projection and grasp them to corresponding locations for classification purposes in a conveyor production line. Two experiments were conducted here: one was a blackboard eraser and another was a poker box with a green cover, as shown in Fig. 15. A video clip showing the experiment results of the tracking and grasping algorithms for the two target objects was uploaded to the website given in Shaw and Chi (2017) for the reader's reference. There were two experiments in the video involving the two target objects with different speeds and different orientation angles. A snapshot photo taken from the video is illustrated in Fig. 16, verifying the target object had been successfully tracked and grabbed to a designated place. The estimated speed using the Kalman filter, along with the actual conveyor speed from the motor encoder, and the time spent by the system to complete the task are given in Table 1. It was readily observed that the object moving speed was estimated close enough (error $\leq 3\%$) and quickly enough so that the robot gripper could be controlled in time to catch up with and grasp the moving target on the conveyor.

Case 3: Grasping an Object with Different Unknown Speeds

In this case, the robot arm tried to grasp the target object with different moving velocities, from low to high conveyor speed, for evaluating both the system robustness in speed variation and the response time (Shaw and Chi, 2017). Four different velocities were tested, and the results are summarized in Table 2. Note that the Kalman filter estimated all four speeds well, and, hence, the robot was successful in tracking and grabbing each object within the workspace. The reason why the spent times of experiments 1 and 4 were longer than those of other experiments was that the estimated speed was slower than the actual one; hence, the robot arm spent more time to catch up with the target.

Case 4: Grasping an Object with Different Orientation Angles

In this case, there were five different orientation angles of the object that the robot had to work with, indicating that the object can be randomly placed on a conveyor. As mentioned earlier, the IBVS method usually does not have a good performance in tracking an object with a large orientation angle. Therefore, the robot arm will be controlled to rotate about the z axis in the first move to align with the long edges of the object before it tracks the target. This move can make the object angle small enough for the IBVS method to begin with. In this way, the system can grasp a moving object at any orientation angle it is placed no matter how large the angle is. Table 3 summarizes the performance of the experiments (Shaw and Chi, 2017). It was noted that the objects in experiments 1 and 2 had counterclockwise angles, whereas the objects in experiments 4 and 5 had clockwise angles. The total times for the task completion of the counterclockwise objects were shorter than those of the clockwise objects. The reason why the clockwise objects were slower to catch up was that the camera was installed side by side and in front of the gripper. After the first gripper rotation movement, the distance between the object and the gripper module

Table 3. Experiment results with different angles of the object.

Experiment number	Orientation angle	Conveyor speed (mm/s)	Kalman speed (mm/s)	Error of speed	Total time (s)
1	-68°	46.92	49.23	4.92%	5.02
2	-28°	46.92	49.11	4.67%	4.90
3	1°	46.92	49.10	4.65%	4.67
4	20°	46.92	47.86	2.00%	5.11
5	58°	46.92	46.7	0.47%	5.68

of the clockwise objects was longer, hence costing more time to track the target. In addition, the bigger the angle of the object was, the more time was needed to finish the task.

V. CONCLUSIONS AND FUTURE WORKS

In this study, we combined the CAMshift algorithm, minimum area rectangle method, visual servoing method, and Kalman filter for grasping moving objects on a conveyor with an unknown speed using an eye-in-hand robot manipulator having a two-finger gripper module attached to it. The developed tracking and grasping algorithm was applied experimentally to validate its performance. According to the obtained results, it was readily observed that the robot arm was capable of grasping a moving object on the conveyor, no matter at what speed the object was moving and how large the orientation angle it was placed, to a desired location on the basis of its color histogram model. The robot arm would approach toward the conveyor and grasp the target if the estimated velocity by the Kalman filter was convergent.

It is noted that the rectangle bounding the object is computed by combining methods of the CAMshift and the minimum area rectangle for robustly tracking the moving target. Hence objects with much-like rectangular shapes are more likely to be successfully grasped by the designed two-finger gripper. More study and experiments are needed if objects other than rectangular shape are used. In addition, the SURF algorithm (Joshi et al., 2004; Shaw and Cheng, 2016) and/or shape-related features will be included to help identify more general objects, instead of relying only on the color information, as used in this study. Finally, a gripper with a force sensor installed can be employed for the robot arm for grasping an object using the right grabbing force without deforming it too much or crushing it (Shaw and Dubey, 2016).

REFERENCES

Allen, P. K., A. Timcenko, B. Yoshimi and P. Michelman (1993). Automated tracking and grasping of a moving object with a robotic hand-eye system. IEEE Transactions on Robotics and Automation 9(2), 152-165.

Bradski, G. R. (1998). Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal, 1-15.

Chaumette, F. (1998). Potential problems of stability and convergence in image-based and position-based visual servoing. In: The confluence of vision and control, volume 237 of Lecture Notes in Control and Information Sciences, Springer-Verlag, 66-78.

Corke, P. I. and S. A. Hutchinson (2001). A new partitioned approach to image-based visual servo control. IEEE Transactions on Robotics and Automation 17(4), 507-515.

Fang, W., Z. Li, X. W. Tu and C. Perron (2009). Switch control of image-based visual servoing with laser pointer in robotic manufacturing system. IEEE Transactions on Industrial Electronics 56(2), 520-529.

Fuentes-Pacheco, J., J. Ruiz-Ascencio and J. M. Rendon-Mancha (2009). Binocular visual tracking and grasping of a moving object with a 3D trajectory predictor. Journal of Applied Research and Technology 7(3), 259-274.

Hajiloo, A., M. Keshmiri, W. F. Xie and T. T. Wang (2016). Robust online model predictive control for a constrained image-based visual servoing. IEEE Transactions on Electronics 63(4), 2242-2250.

Huang S., Y. Yamakawa, T. Senoo and M. Ishikawa (2013). Realizing peg-and-hole alignment with one eye-in-hand high-speed camera. Proceeding of IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 1127-1132.

Husain, F., A. Colome, B. Dellen, G. Alenya and C. Torras (2014). Realtime tracking and grasping of a moving object from range video. Proceeding of IEEE International Conference on Robotics and Automation, Hong Kong, China.

Joshi, S., S. Gujarathi and A. Mirge (2004). Moving object tracking method using improved CAMshift with SURF algorithm. International Journal of Advances in Science Engineering and Technology, 14-18.

Lazar C. and A. Burlacu (2009). Visual servoing of robot manipulators using model-based predictive control. Proceeding of the 7th IEEE International Conference on Industrial Informatics.

Lin, C. J., J. Shaw, P. C. Tsou and C. C. Liu (2016). Vision servo based Delta robot to pick-and-place moving parts. Proceeding of IEEE International Conference on Industrial Technology, Taipei, Taiwan.

Malis, E., F. Chaumette and S. Boudet (1999). 2-1/2-D visual servoing, IEEE Transactions on robotics and automation 15(2), 238-250.

Nobakht H. and Y. Liu (2015). A hybrid positioning method for eye-in-hand industrial robot by using 3D reconstruction and IBVS. Proceeding of IEEE International Conference on Robotics and Biomimetics, Zhuhai, China.

Opencv.org (2018). Website: https://www.docs.opencv.org/3.3.0/dd/d49/tutorial_py_contour_features.html.

Shaw, J. and K. Y. Cheng (2016). Object identification and 3D position calculation using eye-in-hand single camera for robot gripper. Proceeding of IEEE International Conference on Industrial Technology, Taipei, Taiwan.

Shaw, J. and V. Dubey (2016). Design of servo actuated robotic gripper using force control for range of objects. Proceeding of the International Conference on Advanced Robotics and Intelligent Systems, Taipei, Taiwan.

Shaw, J. and W. L. Chi (2017). Video website: https://www.youtube.com/watch?v=22GgnODN_2k.

Suzuki, Y., K. Koyama, A. Ming and M. Shimojo (2015). Grasping strategy for moving object using net-structure proximity sensor and vision sensor. Proceeding of IEEE International Conference on Robotics and Automation, Seattle, Washington.

Van, M., D. Wu and S. S. Ge (2016). Fault diagnosis in image-based visual servoing with eye-in-hand configurations using Kalman filter. IEEE Transactions on Industrial Informatics 12(6), 1998-2007.

Wang, L., Z. Wang, H. Y. Yu, H. M. Ha, Y. K. Kim and J. M. Lee (2015). Vision based robot for recognizing and grasping fast moving conveyor products. Proceeding of the 15th International Conference on Control, Automation and Systems, Busan, South Korea.

Zhang J. and L. Shen (2014). Clustering and recognition for automated tracking and grasping of moving object. Proceeding of IEEE Workshop on Electronics, Computer and Applications, Ottawa, Canada.