Motion Prediction and Robust Tracking of a Dynamic and Temporarily-Occluded Target by an Unmanned Aerial Vehicle

Jun-Ming Li, Ching-Wen Chen, and Teng-Hu Cheng

***Abstract*— A tracking controller for unmanned aerial vehicles (UAVs) is developed to track moving targets in the presence of occlusion. The controller can track moving targets based on a bounding box of the target detected by a deep neural network using the you-only-look-once (YOLO) method. The features generated from the YOLO approach relaxes the assumption of continuous availability of the feature points for applications, which facilitates estimation using an unscented Kalman filter (UKF) and the design of image-based tracking controller in this work. The challenge is that when occlusion is present, the bounding box of the moving target becomes unobtainable and makes the estimation diverge. To solve this, a motion model derived by quadratic programming is employed as a process model in the UKF, wherein the estimated velocity is implemented as a feedforward term in the developed tracking controller in order to enhance the tracking performance. Since no motion constraint is assumed for the target, the developed controller can be applied to track various moving targets. Simulations are used to demonstrate the performance of the developed estimator and controller in the presence of occlusion. Experiments are also conducted to verify the efficacy of the developed estimator and controller.**

***Index Terms*— Estimation, quadratic programming (QP), tracking of moving targets, unmanned aerial vehicle (UAV), unscented Kalman filter (UKF).**

I. Introduction

RESEARCH into unmanned aerial vehicles (UAVs) has received considerable attention in recent years. Their high maneuverability makes UAVs able to execute various missions in diverse terrains including surveillance, aerial photography, search and rescue, and object tracking. Many studies have focused on recovering the position of the target from sensing data obtained from cameras [1]–[3], radar devices [4], and sensor networks [5]–[7]. Since cameras are generally inexpensive and easy to obtain, many approaches rely on the visual features of 2-D images to reconstruct the environment. For example, a structure from motion (SfM) algorithm utilizes feature points in 2-D images to recover the 3-D Euclidean position of the features. A nonlinear algorithm has been used to estimate the positions of static features in 2-D images captured by a camera [8]. The position of a stationary target has been estimated based on the linear and angular velocities of a camera mounted on a moving platform [9]. A perspective camera has been used for the 3-D reconstruction of a deforming object [10], [11]. Artificial patterns (i.e., QR codes) have been used as *a priori* knowledge to reconstruct depth information from the target and to deal with occlusion problems [12]. Several feature types can also be applied to SfM, such as lines [13], cylinders and spheres [14], or planes from measured image moments [15], [16]. However, since SfM relies on triangulation, it is challenging to estimate the motion of a moving target.

When targets are not stationary, knowledge of the target motion is crucial for performing tracking control with UAVs. However, the motion of a target is typically unknown and difficult to predict. Approaches for addressing these issues use visual errors for feedback control to keep the target within the field of view (FOV). For example, [17] and [18] determined the position of a target in an image frame based on its specific color and took the centroid, diameter, or the contour of the target as feature vectors for feedback control of the camera. Bounding boxes derived from an image tracking system (e.g., KCF-TLD) or a deep neural network (DNN) have also been utilized as a feature for feedback control [19]–[21]. However, these studies did not consider the kinematics between the target and the camera, which can degrade the tracking performance.

To solve this issue, the concept of structure and motion (SaM) is introduced to estimate the relative motion between the moving target and the camera. Previous SaM studies have relied on prior knowledge of either the relative motion of the camera or the scene geometry [22]. Visual-based optimization methods have been applied to avoid obstacles and collisions [23], and in the present study, it was assumed that the target position is known in advance. Model-based optimization can be used to maximize the visibility of a point of interest, and simultaneously, to minimize its motion in the image plane for robust sensing [24]. However, the image features were extracted from the environment for visual

Manuscript received February 20, 2020; revised April 28, 2020; accepted July 2, 2020. Date of publication August 6, 2020; date of current version June 10, 2021. Manuscript received in final form July 26, 2020. This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 107-2628-E-009-005-MY3 and in part by the Pervasive Artificial Intelligence Research Labs, Taiwan, under Grant MOST 109-2634-F-009-015. Recommended by Associate Editor A. Serrani. *(Corresponding author: Teng-Hu Cheng.)*

The authors are with the Department of Mechanical Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan (e-mail: michael1874888@gmail.com; amyking90511@gmail.com; tenghu@g2.nctu.edu.tw).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2020.3012619

odometry purposes, rather than with the objective of target tracking. The motion of a target moving at a constant velocity can be estimated by a nonlinear observer given the known camera motion [25]. To relax the assumption of the velocity being constant, a target with a time-varying velocity can be estimated by a state observer [26]–[29]. Prior knowledge is not required of the camera motion or scene geometry [30], [31], but in that case, it is assumed that the movement of the moving object is known. Therefore, some studies [32]–[34] have set up an image pattern (e.g., AprilTag) on the target in advance to obtain the required information for tracking control. However, requiring an artificial pattern on the target for tracking can restrict the applications. Moreover, representing a moving target (e.g., car) by a feature point can be an oversimplification since the orientation of the object is likely to be ignored, which is important information for controlling the relative position in interactions (i.e., the approach or track of the target from a specified direction).

This work describes a method based on applying a DNN for detecting a moving target (e.g., car) undergoing unknown motion and without the use of artificial image patterns or tags. Given an image of the target, a DNN (e.g., one based on the you-only-look-once (YOLO) method [35]) can generate a bounding box in the image to enclose the target and orientation of the target (e.g., car). The bounding box provides the information required for estimating the target motion based on a kinematics model, and the orientation can be leveraged to control the UAV facing the target at a specified angle. Meanwhile, in real applications, the target might not always be within a line of sight due to occlusion, which will lead to intermittent observability and degrade the tracking performance. To deal with this, an unscented Kalman filter (UKF) is developed to estimate the unknown position and velocity of the target. In addition, a high-order polynomial obtained by quadratic programming (QP) is used to emulate the target motion during occlusion so as to facilitate the state estimation in the UKF. The contributions of this study can be summarized as follows.

1) A tracking controller is developed for a UAV to track a moving target undergoing unknown motion.
2) The unknown motion of the target is estimated, and the relative motion between the target and the UAV is modeled and considered in the tracking controller.
3) No artificial image patterns/tags are required to facilitate feature extraction since a well-trained YOLO DNN is applied to generate the bounding box as features.
4) The YOLO DNN is independent of the tracking controller and estimation algorithm, and therefore, it can be replaced by other state-of-the-art machine-learning-based object detectors.
5) The stability of the closed-loop system and the observability of the estimator are proven.

II. Preliminaries

A. Problem Formulation

The mission of tracking a moving target involves the processes of detection, state estimation, controller design,

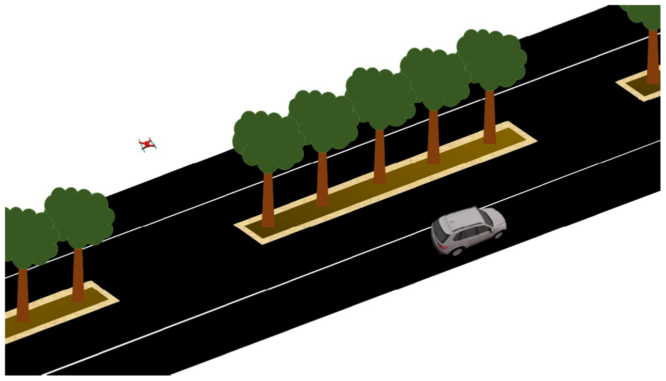
Fig. 1. Tracking of a moving target during occlusion.

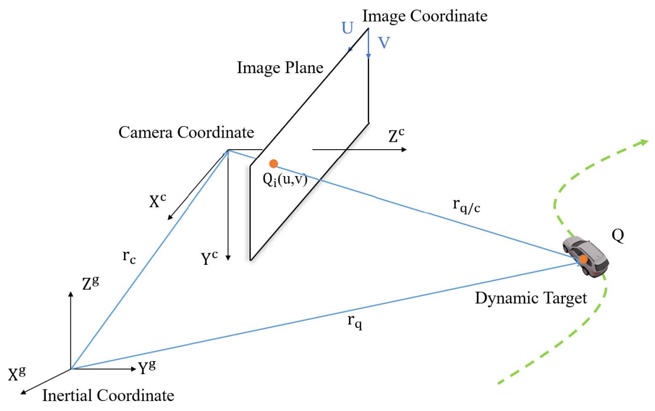

Fig. 2. Kinematics model.

and trajectory planning. Completing these tasks requires knowledge about the target position and velocity, but these parameters are generally not measurable, and the method relies on an image of the target to estimate the target states. Moreover, the target might not always be observable due to the occurrence of occlusion (as shown in Fig. 1), and therefore, continuously and robustly tracking the target is challenging. The goal is to track a moving target whose position and velocity and unknown, and keep tracking it even when occlusion is present.

B. Control Objectives

Fig. 2 shows the relationship between a moving target and an onboard camera fixed to a UAV. The subscript $\mathcal{G}$ denotes the inertial frame with its origin set arbitrarily on the ground, and the subscript C represents the camera frame with the origin fixed at the principal point of the camera, where X^c, Y^c, and Z^c are the axes in the denoted direction. $r_q = [\, x_q \;\, y_q \;\, z_q \,]^T$ denotes the position of the moving target, and $r_c = [\, x_c \;\, y_c \;\, z_c \,]^T$ denotes the position of the camera, which can be measured by the embedded GPS/motion-capture system, all expressed in the camera frame. The relative position between the moving target and the camera is defined as

$$r_{q/c} = r_q - r_c \tag{1}$$

where $r_{q/c} = [\, X \;\, Y \;\, Z \,]$ is defined to facilitate subsequent analysis, where Z can be considered as the depth $d \in \mathbb{R}$ from the target. Taking the time derivative on both sides of (1) yields the relative velocity as

$$\dot{r}_{q/c} = V_q - V_c - \omega_c \times r_{q/c} \tag{2}$$

where $V_c \triangleq [\, v_{cx} \; v_{cy} \; v_{cz} \,]^T$ and $\omega_c \triangleq [\, \omega_{cx} \; \omega_{cy} \; \omega_{cz} \,]^T$ are the translational and angular velocities of the camera, respectively, and both represent control commands that need to be designed. In (2), $V_q = [\, v_{qx} \; v_{qy} \; v_{qz} \,]^T$ is the translational velocity of the dynamic target, which is unknown and needs to be estimated.

The control objective is to design an image-based controller such that

$$r_{q/c} \rightarrow [0, \quad 0, \quad d_{\text{des}}]^T \tag{3}$$

where [0, 0] indicates keeping the target at the center of the image, and $d_{\text{des}} \in \mathbb{R}$ is the desired depth from the target.

To estimate the state of the target, a dynamics model and a measurement model are introduced in Section III.

III. Motion Estimate of the Moving Target

A. Kinematics Model

Since the dynamics of the camera and the target are coupled, the state of the overall system $\mathbf{x} \in \mathbb{R}^9$ is defined as

$$\mathbf{x} = \left[x_1, \quad x_2, \quad x_3, \quad r_q^T, \quad V_q^T\right]^T \tag{4}$$

where

$$[x_1, \quad x_2, \quad x_3] = \left[\frac{X}{Z}, \quad \frac{Y}{Z}, \quad \frac{1}{Z}\right] \tag{5}$$

is the state defined to facilitate the subsequent analysis. By extending our previous work [27], the dynamics model of the overall system can be obtained by taking the time derivative on the both sides of (4)

$$\dot{\mathbf{x}} = \begin{bmatrix} v_{qx}x_3 - v_{qz}x_1x_3 + \zeta_1 + \eta_1 \\ v_{qy}x_3 - v_{qz}x_2x_3 + \zeta_2 + \eta_2 \\ -v_{qz}x_3^2 + v_{cz}x_3^2 - (\omega_{cy}x_1 - \omega_{cx}x_2)x_3 \\ V_q \\ 0_{3\times 1} \end{bmatrix} \tag{6}$$

where (2) is used and $\zeta_1, \zeta_2, \eta_1, \eta_2 \in \mathbb{R}$ are defined as

$$\begin{aligned} \zeta_1 &= \omega_{cz}x_2 - \omega_{cy} - \omega_{cy}x_1^2 + \omega_{cx}x_1x_2 \\ \zeta_2 &= -\omega_{cz}x_1 + \omega_{cx} + \omega_{cx}x_2^2 - \omega_{cy}x_1x_2 \\ \eta_1 &= (v_{cz}x_1 - v_{cx})x_3 \\ \eta_2 &= (v_{cz}x_2 - v_{cy})x_3. \end{aligned}$$

Remark 1: Since the motion of the moving target is unknown, the dynamics defined in (6) is approximated by a constant-velocity model, which is reasonable over a short time period. To prevent error accumulation, the mismatched model is considered as process noise and will be corrected using camera measurements in the UKF defined in Section III-D.

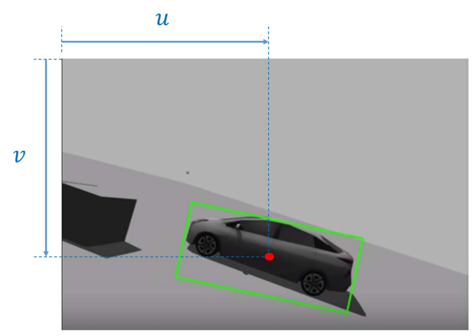

Fig. 3. Bounding box encloses the target by using a YOLO DNN, which is trained to detect the vehicle at various tilt angles on the fly. The red dot indicates the center of the bounding box.

B. Measurement

This section defines two measurements obtained by applying the YOLO approach: bounding box and orientation. The location of the bounding box gives the controller a feedback signal to keep the target in the center of the image, and the size of the box can be used to derive the depth from the target based on the focal length. Therefore, the relative distance from the target can be maintained while keeping the target in the center of the image. On the other hand, tracking the target from a preferred viewing angle requires a signal that indicates the orientation of the target viewed by the camera. To this end, the orientation of the target is also measured by a pretrained YOLO DNN.

1) Bounding Box: A reliable way to measure the depth is directly using stereo cameras, but this is more expensive than using monocular cameras. An alternative method is to derive the depth based on the area of the bounding box. Fig. 3 shows that, given the area of the detected bounding box as a and the side area of the target (i.e., length × height) observed from camera A (e.g., A is 4.6 m × 1.5 m for a sedan), the pinhole model can be applied to derive the depth d as

$$d = \sqrt{\frac{A f_x f_y}{a}} \tag{7}$$

where f_x and f_y are the focal lengths.

Based on pin hole model, the states x_1 and x_2 can be obtained by projecting the position of the target onto the image plane as

$$\begin{aligned} x_1 &= \frac{u - c_u}{f_x} \\ x_2 &= \frac{v - c_v}{f_y} \end{aligned} \tag{8}$$

where c_u and c_v represent the center position of the image. Therefore, based on (1), (5), and (8), the relation between the states and the measurement can be derived as

$$\mathbf{z} = \begin{bmatrix} u \\ v \\ d \\ r_c \end{bmatrix} \tag{9}$$

$$= \begin{bmatrix} f_x x_1 + c_u \\ f_y x_2 + c_v \\ 1/x_3 \\ r_q - r_{q/c} \end{bmatrix} \tag{10}$$

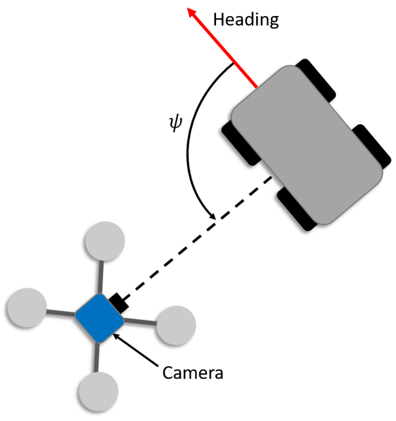

Fig. 4. Orientation of the target ψ is defined as the angle between the heading of the target and the relative position from the target to the UAV.

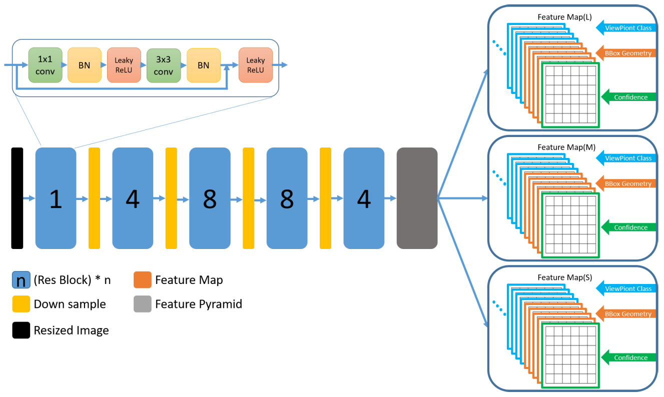

Fig. 5. Network structure of a YOLO DNN, which takes resized images as inputs and can generate three outputs: orientation, bounding box, and detection.

where u and v indicate the center position of the detected bounding box of the moving target along the U and V direction in the image frame, as shown in Fig. 2, and r_c and d are defined in Section II-B.

2) Orientation Measurement: To track the motion of a target, the orientation of the target, denoted as $\psi \in \mathbb{R}$ as shown in Fig. 4, needs to be measurable by a YOLO DNN from an input image captured by the onboard camera; however, a few well-pretrained DNNs are available for our application.

To this end, a modified version of a YOLO DNN is developed in this work for target detection and orientation measurement. The second measurement (the orientation of the target) is required for the UAV to know where it is located relative to the target so that the error between the desired and current orientations can be compensated by the controller developed in Section IV. The detailed network architecture is shown in Fig. 5.

To produce a sufficiently large image data set for training, images are generated using Blender software, in which vehicles of various colors and sizes are synthesized. In addition, each vehicle is enclosed by a bounding box in the image with the ground-truth orientation labeled. Since the YOLO DNN was not the focus of this work, the training process is not described further. The detection and orientation estimation of the target are shown in Fig. 6.

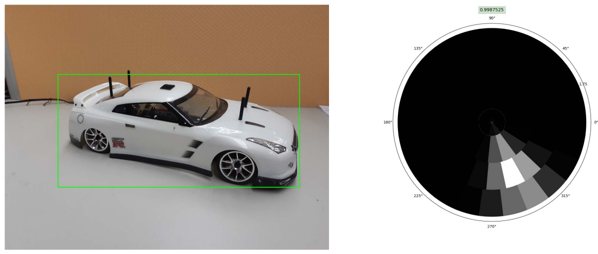

Fig. 6. Demonstration of detection and orientation measurement of the target.

C. Observability

This section discusses the observability of the state defined in (4) and the measurement defined in (10). Since the dynamics of the system described by (6) is nonlinear, the approach developed in [36, eq. (13)] is applied. To this end, Theorem 1 is established.

Theorem 1: The system dynamics defined in (6) with the measurement defined in (10) is observable.

Proof: Consider an observability matrix O defined as

$$O \triangleq \frac{\partial \phi}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{\partial \mathcal{L}_f^0(z(\mathbf{x}))}{\partial x_1} & \cdots & \dfrac{\partial \mathcal{L}_f^0(z(\mathbf{x}))}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial \mathcal{L}_f^{n-1}(z(\mathbf{x}))}{\partial x_1} & \cdots & \dfrac{\partial \mathcal{L}_f^{n-1}(z(\mathbf{x}))}{\partial x_n} \end{bmatrix} \tag{11}$$

where $n = 9$ is the dimension of the state defined in (4), $x_1, \ldots, x_n$ are the elements of state $\mathbf{x}$, and ϕ denoting the differential embedding map is defined as the Lie derivative of $z(\mathbf{x})$

$$\phi \triangleq \begin{bmatrix} \mathcal{L}_f^0(z(\mathbf{x})) \\ \mathcal{L}_f^1(z(\mathbf{x})) \\ \vdots \\ \mathcal{L}_f^{n-1}(z(\mathbf{x})) \end{bmatrix}$$

where the elements are defined as

$$\begin{aligned} \mathcal{L}_f^0(z(\mathbf{x})) &\triangleq z(\mathbf{x}) \\ \mathcal{L}_f^1(z(\mathbf{x})) &\triangleq \frac{\partial z(\mathbf{x})}{\partial \mathbf{x}} \cdot f \\ \mathcal{L}_f^2(z(\mathbf{x})) &\triangleq \frac{\partial}{\partial \mathbf{x}}\left[\mathcal{L}_f^1(z(\mathbf{x}))\right] \cdot f \\ &\vdots \\ \mathcal{L}_f^k(z(\mathbf{x})) &\triangleq \frac{\partial}{\partial \mathbf{x}}\left[\mathcal{L}_f^{k-1}(z(\mathbf{x}))\right] \cdot f \end{aligned}$$

where f is the vector field defined as $f = \dot{\mathbf{x}}$. The submatrix of observability matrix O denoted as $O_s \in \mathbb{R}^{12\times 9}$ is defined as

$$O_s \triangleq \begin{bmatrix} \dfrac{\partial \mathcal{L}_f^0(z(\mathbf{x}))}{\partial \mathbf{x}} \\ \dfrac{\partial \mathcal{L}_f^1(z(\mathbf{x}))}{\partial \mathbf{x}} \end{bmatrix}$$

and after algebraic manipulation can be obtained as

$$O_s = \begin{bmatrix} f_x & 0 & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 & 0 \\ 0 & 0 & -x_3^{-2} & 0 & 0 \\ 0 & 0 & 0 & I_3 & 0_{3\times 3} \\ - & - & - & - & - \\ & & \frac{\partial f_x \dot{x}_1}{\partial \mathbf{x}} & & \\ & & \frac{\partial f_y \dot{x}_2}{\partial \mathbf{x}} & & \\ & & \frac{\partial\left(-x_3^{-2}\dot{x}_3\right)}{\partial \mathbf{x}} & & \\ 0 & 0 & 0 & 0 & I_3 \end{bmatrix}$$

which is a full-rank matrix [i.e., rank$(O_s) = 9$] based on rows 1∼6 and rows 10∼12, which implies O is also a full-rank matrix, and hence, the system is proven to be observable. ■

D. Unscented Kalman Filter

A UKF is applied in this section to estimate the state of dynamic systems with noisy or intermittent measurements, since the UKF is superior to an extended Kalman filter for several reasons [37]–[39].

1) It is accurate to two terms of the Taylor expansion.
2) It has a higher efficiency since it does not require sufficient differentiability of the state dynamics.
3) It provides a derivative-free way to estimate the state parameters of nonlinear systems by introducing an "unscented transformation."

By discretizing (6) and (9) using the Euler method, the nonlinear dynamic system can be expressed as

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k) + w_k \tag{12}$$
$$\mathbf{z}_k = H(\mathbf{x}_k) + v_k \tag{13}$$

where w_k and v_k represent the process and measurement noise, respectively, $F(\cdot)$ is the dynamics model used for prediction, $H(\cdot)$ is the measurement model used for update, and $\mathbf{z}_k$ is the discretized measurement of $\mathbf{z}$ defined in (10).

When the bounding box fails, the state is only predicted by the dynamics model using (12), which is reliable over the short time period before the detection recovers.

E. Estimation of Noise Covariance Matrices

As mentioned in Remark 1, the error of the dynamics model can be considered as a process noise covariance matrix, which is sensitive to the convergence of the estimation. Our simulations confirmed that inaccurate constant covariance matrices could lead to large estimate errors and mission failure. To dynamically estimate the process noise covariance matrices, a previously described method [40] is applied to estimate and update the covariance matrices online, which provides faster and more reliable convergence.

IV. Controller Design

A. Development of Controller

A visual feedback controller is designed in this section to achieve target tracking. Compared with the existing image-based visual servo control methods [41], the controller developed in this work not only uses feedback but also includes a feedforward term to compensate for the target motion and to ensure better tracking performance, where the feedforward term is the state estimated using the UKF. To this end, $s(t) = [x_1, x_2, x_3]^T : [0, \infty) \to \mathbb{R}^3$ defined in (4) is a feature vector, and the tracking error $e: [0, \infty) \to \mathbb{R}^3$ is defined as

$$e(t) = s(t) - s^* \tag{14}$$

where $s^* \triangleq [x_1^*, x_2^*, x_3^*]^T \in \mathbb{R}^3$ is a desired vector prescribed by the user; that is, $s^* \triangleq [0, 0, 1/d_{\text{des}}]^T$ is selected to achieve the control objective defined in (3). Therefore, $e \to 0$ implies that this control objective is achieved.

Taking the time derivative of (14) and using (6) yields the open-loop error system as

$$\dot{e} = \dot{s} = L_e \begin{bmatrix} V_c - V_q \\ \omega_c \end{bmatrix} \tag{15}$$

where V_c and ω_c are the control inputs, V_q is unknown as defined in (2) and can be estimated by the UKF developed in Section III-D, and $L_e \in \mathbb{R}^{3\times 6}$ is the interaction matrix defined as

$$L_e = \begin{bmatrix} -x_3 & 0 & x_1x_3 & x_1x_2 & -\left(x_1^2+1\right) & x_2 \\ 0 & -x_3 & x_2x_3 & \left(x_2^2+1\right) & -x_1x_2 & -x_1 \\ 0 & 0 & x_3^2 & x_2x_3 & -x_1x_3 & 0 \end{bmatrix}. \tag{16}$$

Note that as the error signal e converges to zero, the position of the camera relative to the target is not unique due to the camera control input V_c and ω_c having a higher dimension than e. To approach the target from a desired angle ψ_{des}, the velocity of the camera in the x-direction, v_{cx}, is designed as

$$v_{cx} = -\frac{w_{im} d_{\text{des}} e_\psi}{FOV_u \times f_x} + v_{qx} \tag{17}$$

where $e_\psi \in \mathbb{R}$ is designed as $e_\psi \triangleq \psi - \psi_{\text{des}}$, where ψ is is the orientation of the target defined in Section III-B2. The design of (17) is inspired from [42], w_{im} is the width of the image in pixels, FOV_u is the horizontal FOV of the camera, and v_{qx} is a feedforward term. After designing the controller for v_{cx} defined in (17), the open-loop error system defined in (15) can be modified as

$$\dot{e} = \hat{L}_e \left(\begin{bmatrix} v_{cy} \\ v_{cz} \\ \omega_{cx} \\ \omega_{cy} \\ \omega_{cz} \end{bmatrix} - \begin{bmatrix} v_{qy} \\ v_{qz} \\ 0 \\ 0 \\ 0 \end{bmatrix} \right) \tag{18}$$

where $\hat{L}_e \in \mathbb{R}^{3\times 5}$ is defined as

$$\hat{L}_e = \begin{bmatrix} 0 & x_1x_3 & x_1x_2 & -\left(x_1^2+1\right) & x_2 \\ -x_3 & x_2x_3 & \left(x_2^2+1\right) & -x_1x_2 & -x_1 \\ 0 & x_3^2 & x_2x_3 & -x_1x_3 & 0 \end{bmatrix} \tag{19}$$

which is the submatrix of the interaction matrix L_e, defined in (16), with the corresponding column removed since v_{cx} is specified in (17). To achieve the control objective defined

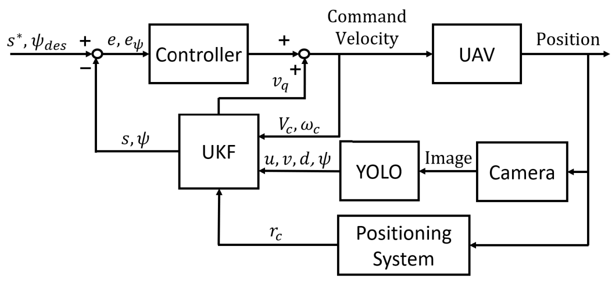

Fig. 7. Block diagram of the controller, where u, v, and d are the measurement to be updated in the UKF.

in (3), the tracking controller for the camera can be designed as

$$\begin{bmatrix} v_{cy} \\ v_{cz} \\ \omega_{cx} \\ \omega_{cy} \\ \omega_{cz} \end{bmatrix} = -\lambda \hat{L}_e^+ e + \begin{bmatrix} v_{qy} \\ v_{qz} \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{20}$$

where $\hat{L}_e^+ \triangleq \hat{L}_e^T(\hat{L}_e\hat{L}_e^T)^{-1} \in \mathbb{R}^{5\times 3}$ is the Moore–Penrose pseudoinverse of matrix $\hat{L}_e$. Substituting the controller defined in (20) into the open-loop dynamics defined in (18) yields the closed-loop error system defined as

$$\dot{e} = -\lambda \hat{L}_e \hat{L}_e^+ e. \tag{21}$$

The block diagram of the controller is shown in Fig. 7.

Remark 2: The control architecture is developed for a general target tracking task that is not limited to vehicles. Therefore, given a bounding box enclosing the target in the image, the measurement of the feature vector and orientation (i.e., s and ψ) can be used to predict the motion of the target v_q, which can facilitate the tracking performance and, in return, enhance the detection of bounding boxes.

B. Stability Analysis

To ensure the stability of the closed-loop system given the controller defined in (20), Theorem 2 is developed.

Theorem 2: Given the controller defined in (20), the closed-loop error system defined in (21) is globally asymptotically stable.

Proof: Consider a candidate Lyapunov function defined as $V_L \triangleq (1/2)e^T e$, and taking the time derivative on both sides yields

$$\dot{V}_L \triangleq e^T \dot{e}. \tag{22}$$

Substituting the closed-loop error system defined in (21) into (22) yields

$$\dot{V}_L = -\lambda e^T \hat{L}_e \hat{L}_e^+ e$$

which implies that $\dot{V}_L$ is negative definite since the following sufficient condition is satisfied:

$$\hat{L}_e \hat{L}_e^+ > 0$$

where $\hat{L}_e \hat{L}_e^+ = I_3$ is an identity matrix of dimension 3 since $\hat{L}_e$ is full rank [i.e., rank($\hat{L}_e$)=3]. Therefore, $\dot{V}_L$ is negative definite, and thus, e is proven to be asymptotically stable. ■

V. Estimation of Target Motion Pattern

The UKF-based method developed in Section III-D continuously estimates the unknown motion of a moving target even when the bounding box is lost for a short time period. However, if this is lost for a longer time period, the method might become infeasible since the dynamics model in (6) is not yet accurate. For example, if the target speeds up when occlusion occurs, the tracking error will gradually increase due to the inaccurate dynamics model utilized in the UKF. As the tracking error increases, the moving target could potentially leave the FOV of the camera altogether, causing a complete tracking failure. A dynamics model should be applied to reduce the tracking error, and therefore, a high-order polynomial is used to emulate the target motion when occlusion occurs. Specifically, the polynomial is found based on a QP method to fit the position and velocity data over the latest time period.

A. Cost Function Design of the Target Trajectory

A polynomial that fits the latest trajectory of the target can be approximated using QP. To this end, an nth-order polynomial $\hat{\Gamma}_i(t) \in \mathbb{R}^3$ defined in the interval $[t_{i-1}, t_i)$ is expressed as

$$\hat{\Gamma}_i(t) = \sum_{j=0}^{n} c_{ij}(t - t_{i-1})^j, \text{ for } t_{i-1} \le t < t_i \tag{23}$$

where $c_{ij} \in \mathbb{R}^3$ denotes the jth-order coefficient. The polynomial of the target trajectory can be derived by fitting the latest estimated position and velocity obtained from the UKF defined in Section III-D. That is, t_i is defined as the instant when the occlusion occurs, and t_{i-1} is determined by the size of m. During the time period $[t_{i-1}, t_i)$, the latest m position and velocity data points are used to obtain the optimal solution for a cost function [32] defined as

$$\min_{\hat{\Gamma}_i} \sum_{k=0}^{m} \|\hat{\Gamma}_i(t_k) - p_k\|_2^2 + \sum_{k=0}^{m} \|\dot{\hat{\Gamma}}_i(t_k) - v_k\|_2^2 \tag{24}$$

where $m > n+1$ is required to find the solution, p_k and $v_k \in \mathbb{R}^3$ are the estimated position and velocity of the moving target at time t_k, and $\dot{\hat{\Gamma}}_i$ denotes the time derivative of the polynomial. Minimizing the cost function will result in polynomial $\hat{\Gamma}_i$ being close to the target trajectory.

Remark 3: To avoid overfit to the target trajectory and generating an aggressive trajectory for the UAV, an acceleration regulator $\ddot{\hat{\Gamma}}_i$ can be added to (24) as

$$\min_{\hat{\Gamma}_i} \sum_{k=0}^{m} \|\hat{\Gamma}_i(t_k) - p_k\|_2^2 + \sum_{k=0}^{L} \|\dot{\hat{\Gamma}}_i(t_k) - v_k\|_2^2 + \lambda \int_{t_{i-1}}^{t_i} \|\ddot{\hat{\Gamma}}_i(t)\|_2^2 dt \tag{25}$$

where λ is the weighting factor of the regulator. Choosing a smaller λ in (25) will fit $\hat{\Gamma}_i$ closer to the estimated target motion, while a larger λ will result in a smoother $\hat{\Gamma}_i$ due to the acceleration regulator.

B. Quadratic Form of the Cost Function

The cost function is then converted into a square form in order to derive its solution by utilizing QP. For the ith segment of the polynomial within the time interval $t = [t_{i-1}, t_i)$, $\hat{\Gamma}_i$ can be expressed as

$$\hat{\Gamma}_i(t) = \sum_{j=0}^{n} c_{ij}(t - t_{i-1})^j = T_i^T C_i \tag{26}$$

where $T_i \in \mathbb{R}^{n+1}$ and $C_i \in \mathbb{R}^{(n+1)\times 3}$ are defined as

$$T_i \triangleq \begin{bmatrix} 1 \\ t - t_{i-1} \\ \vdots \\ (t - t_{i-1})^n \end{bmatrix}, \quad C_i \triangleq \begin{bmatrix} c_{i0}^T \\ c_{i1}^T \\ \vdots \\ c_{in}^T \end{bmatrix}.$$

Substituting (26) back in to (24) yields an alternative form

$$\sum_{k=0}^{m} \|\hat{\Gamma}_i(t_k) - p_k^T\|_2^2 + \sum_{k=0}^{L} \|\dot{\hat{\Gamma}}_i(t_k) - v_k^T\|_2^2 = \sum_{l\in\{x,y,z\}} C_i^{l^T} H_i C_i^l + q_i^l C_i^l + \tau_i^l$$

where C_i^l denotes the data extracted from the lth dimension of C_i, and $H_i \in \mathbb{R}^{(n+1)\times(n+1)}$, $q_i^l \in \mathbb{R}^{1\times(n+1)}$, and $\tau_i \in \mathbb{R}$ are defined as

$$H_i = \sum_{k=0}^{m} \left[T_i(t_k)T_i(t_k)^T + (\dot{T}_i(t_k))(\dot{T}_i(t_k))^T\right]$$
$$q_i^l = -2\sum_{k=0}^{m} \left[p_i^l T_i(t_k)^T + v_i^l(\dot{T}_i(t_k))^T\right]$$
$$\tau_i^l = \sum_{k=0}^{m} \left((p_i^l)^2 + (v_i^l)^2\right).$$

The formula of the regulator can be rewritten as

$$\int_{t_{i-1}}^{t_i} \|\ddot{\hat{\Gamma}}_i(t)\|_2^2 dt = C_i^T G_i C_i$$

where G_i is defined as

$$G_i = \int_{t_{i-1}}^{t_i} (\ddot{T}_i(t))(\ddot{T}_i(t))^T dt.$$

The cost function described in (25) can be rewritten as

$$\min_{C_i} \sum_{l\in\{x,y,z\}} C_i^{l^T}(H_i + \lambda G_i)C_i^l + q_i^{l^T} C_i^l.$$

The QP solver CVXOPT is used to solve the QP problem for trajectory generation. The tradeoff between computational time and the performance of the estimation result also needs to be considered in order to achieve real-time execution.

Remark 4: Increasing the length of the sliding window m not only makes the estimation trajectory smoother but also increases the computational time and might result in overshoot when the motion of the moving target changes dramatically. Conversely, decreasing m can reduce the required computation power, but the motion model $\hat{\Gamma}_i$ generated by QP will become more sensitive to the noise.

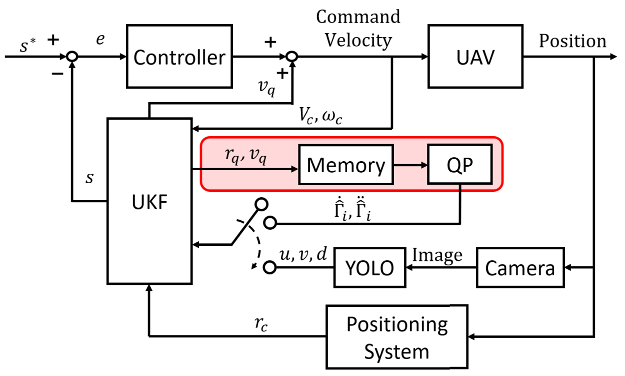

Fig. 8. Block diagram of the controller architecture using the QP-based motion model. Compared with Fig. 7, a switch is added to select one of the inputs to the UKF depending on the observability of the target. The red area indicates the QP algorithm developed in Section V-B, where the memory is a queue that stores all of the r_q and v_q values collected over the latest m samples. During occlusion the measurement data $[u, v, d]$ are not available, and the motion obtained by QP (i.e., $\dot{\hat{\Gamma}}_i$ and $\ddot{\hat{\Gamma}}_i$) is used to emulate the target motion for the UKF as described in (27). Otherwise, the measured u, v, and d values obtained from the detected bounding box are imported into the UKF for updating.

C. QP Augmented With UKF Under Occlusion

The polynomial obtained from QP can be considered as the alternative motion model during occlusion. That is, when occlusion starts to occur at time t_i, $\dot{\hat{\Gamma}}_i$, and $\ddot{\hat{\Gamma}}_i$ are used as alternatives to replace the velocity and acceleration terms in (6) in the UKF

$$\dot{\mathbf{x}} = \begin{bmatrix} \hat{v}_{qx}x_3 - \hat{v}_{qz}x_1x_3 + \zeta_1 + \eta_1 \\ \hat{v}_{qy}x_3 - \hat{v}_{qz}x_2x_3 + \zeta_2 + \eta_2 \\ -\hat{v}_{qz}x_3^2 + v_{cz}x_3^2 - (\omega_{cy}x_1 - \omega_{cx}x_2)x_3 \\ \dot{\hat{\Gamma}}_i^T \\ \ddot{\hat{\Gamma}}_i^T \end{bmatrix} \tag{27}$$

where $\dot{\hat{\Gamma}}_i = [\hat{v}_{qx}, \hat{v}_{qy}, \hat{v}_{qz}]^T$. Using (27) as a process model in the UKF can reduce the tracking error between the estimation and the ground truth and prevent the tracking error from becoming unstable. This approach also increases the probability of the moving target remaining within the FOV of the camera after the occlusion and greatly improves the tracking performance compared to the constant-velocity dynamics model in Section III-D.

VI. Simulations

A. Environment Setup

Simulations[1] were conducted using the Gazebo simulator in the ROS framework (version 16.04, Kinetic). In the environment, the visual-based controller developed in Section IV

[1]https://www.youtube.com/watch?v=w0C-3ljbK7E

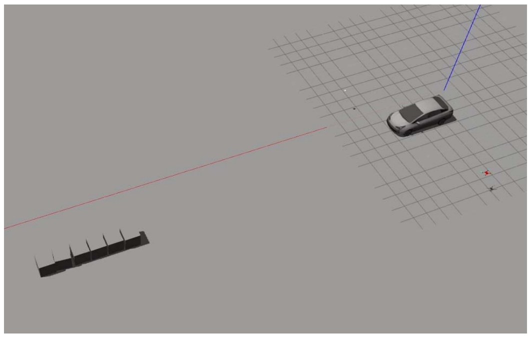

Fig. 9. Initial setup of the car, UAV, and obstacles. When the car begins to move, the UAV keeps tracking the car to estimate its position and velocity.

was implemented in a quadrotor UAV to track a moving vehicle considered to be the moving target. The ground-truth dimensions of the car were 4.6 m $\times$1.8 m $\times$1.5 m (length $\times$ width $\times$ height). The initial locations of the car and UAV were $[0, 0, 0]^T$ and $[0, 5.5, 1.0]^T$, respectively, and their initial orientations were $[0, 0, 0]^T$ and $[0, 0, -(pi/2)]^T$ in the global frame. The desire depth d_{des} was 7 m. The car was free to move in the xy plane, and its velocity was specified based on real-time user commands. To demonstrate the efficacy of the developed controller in the presence of occlusion, six boxes of size 1 m $\times$ 1 m $\times$ 1 m lined up alongside the path were used as an obstacle, as shown in Fig. 9.

Remark 5: Unfortunately, due to the lack of knowledge of the wireless communication delay model, delays were not added to the simulation.

The car was detected by the stereo camera on the UAV at a detection looping rate of 90 Hz. The center position of the detected bounding box u, v in the image frame was generated by a YOLO DNN, and the relative depth d between the target and the camera was directly obtained from the stereo camera. The resolution of the images was 640 $\times$ 480 pixels, and the intrinsic-parameters matrix of the camera

$$K = \begin{bmatrix} 381.36 & 0 & 320.5 \\ 0 & 381.36 & 240.5 \\ 0 & 0 & 1 \end{bmatrix}$$

was obtained by calibration.

The initial process and measurement noise covariance matrices were selected as diag$\{[0.08, 0.08, 0.05, 5, 5, 5, 50, 50, 50] \times 10^{-2}\}$ and diag$\{[100, 100, 0.05, 0.0001, 0.0001, 0.0001]\}$, respectively, and the process noise covariance matrix was estimated online using the method developed in [40].

A third-order polynomial (i.e., $n = 3$) was applied in the QP method to derive the trend of the target trajectory. The window size m was set as 600, and the weighting of the regulator λ was set as 0.1 in (25). The following simulations did not require prior knowledge about the moving target, and the looping rate was set at 80 Hz for the noise estimation, state estimation, and the control process.

The following two simulations compared the estimator performance using different motion models in the UKF for the same scenario shown in Fig. 10. The first and second simulations adopted a constant-velocity model and the QP-based motion model, respectively. In addition, the tracking performances when using the two methods were also compared.

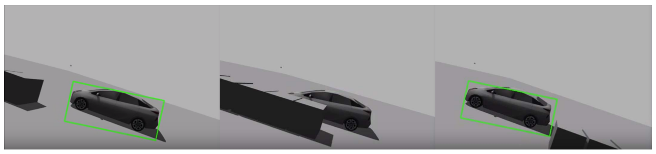

Fig. 10. Setup for the two subsequent simulations. The car started from rest at 2 s (left), then maintained a constant acceleration of 0.1 m^2/s for 30 s (center), and finally maintained a constant velocity of 3 m/s for 3 s (right). The occlusion of the car from 15 to 22 s led to detection failure.

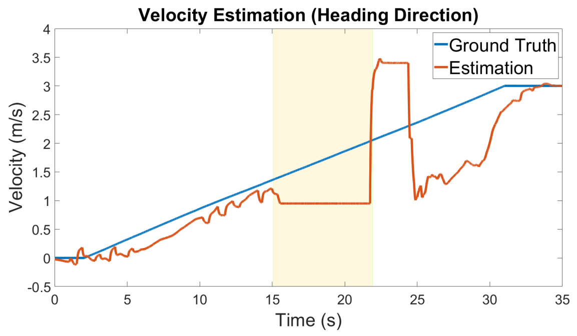

Fig. 11. Ground-truth and estimated velocities of the moving target using the constant-velocity model. The car started from rest at 2 s and approached 3 m/s at 32 s with a constant acceleration of 0.1 m^2/s for 30 s. The yellow zone shows that the car was occluded from 15 to 22 s, which led to detection failure.

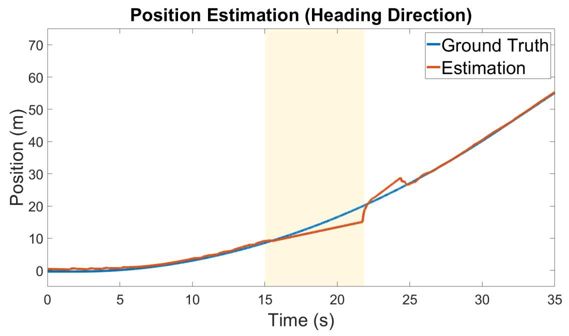

Fig. 12. Ground-truth and estimated positions of the moving target using the constant-velocity model.

B. Simulation 1: Constant-Velocity Motion Model

Fig. 11 shows the velocity estimated using the UKF with the constant-velocity model as defined in (6) (orange line) and the ground-truth velocity (blue line). Under the constant-velocity model (6), the estimated velocity remained constant when the car was occluded. The estimation error of the velocity increased from 15 to 22 s, but then gradually converges after the car was detected again from 22 to 35 s.

The corresponding position trajectories in the heading direction are shown in Fig. 12.

C. Simulation 2: QP-Based Motion Model

Fig. 13 shows the results obtained in the simulations using the combined QP and UKF estimator as defined in (27)

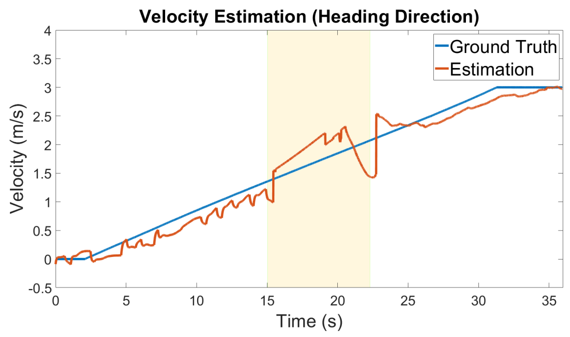

Fig. 13. Ground-truth and estimated velocities of the moving target using the QP-based motion model.

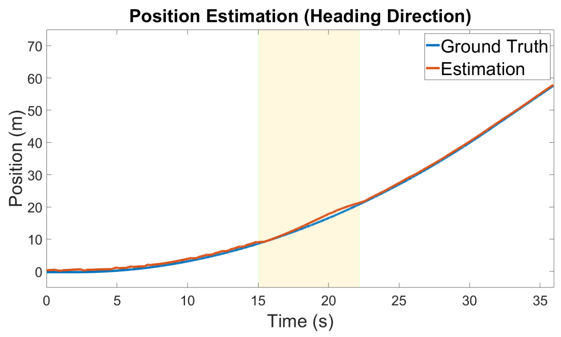

Fig. 14. Ground-truth and estimated positions of the moving target using the QP-based motion model.

(orange line). Fig. 13 shows that using the QP-based motion model for estimation during occlusion provided better velocity estimation compared with using the constant-velocity model in Simulation 1.

The corresponding position trajectories in the heading direction are shown in Fig. 14, which are smoother than those shown in Fig. 12.

D. Comparison: Tracking Errors

This section compares the tracking performance using the two methods defined earlier, where the errors are defined as the difference in image features between the estimated and desired values. The desired values of x_1 and x_2 were both 0, and the estimated depth d is 7 m.

Figs. 15–17 show that at the beginning of the simulation (before occlusion occurred), the tracking errors (e) were close to zero for both the constant-velocity and QP-based motion models. However, these two methods clearly performed differently after occlusion had occurred. That is, from 22 to 35 s, the error obtained from the constant-velocity model fluctuated markedly, and the model took longer to converge. In contrast, the QP-based motion model method showed good tracking performance despite the presence of occlusion, which also reflects what is shown in Fig. 13.

E. Simulation 3: Tracking a Target Exhibiting 2D Motion

The scenario of the simulation performed in this section is shown in Fig. 18, where the controller designed in (17) was implemented to track a target exhibiting 2-D motion, where the trajectory of the target was unknown to the UAV.

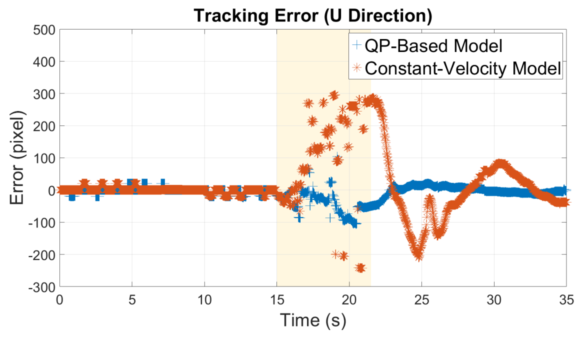

Fig. 15. Image tracking error in the u-direction.

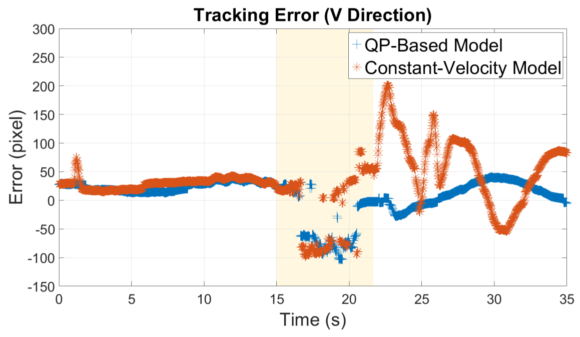

Fig. 16. Image tracking error in the v-direction.

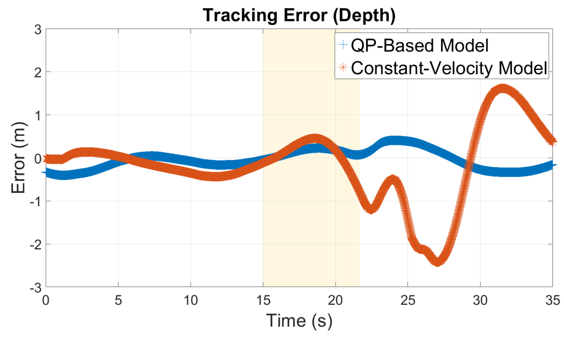

Fig. 17. Image tracking error in the d-direction.

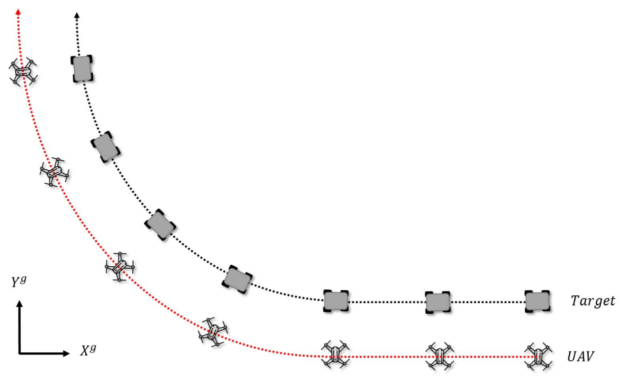

Fig. 18. UAV tracking a target exhibiting 2-D motion.

The trajectories of the UAV and the target are shown in Fig. 19, which indicates that they maintained an almost constant separation (i.e., 7 m).

The orientation ψ obtained using the YOLO DNN is plotted in Fig. 20, where shows that the measured orientation was

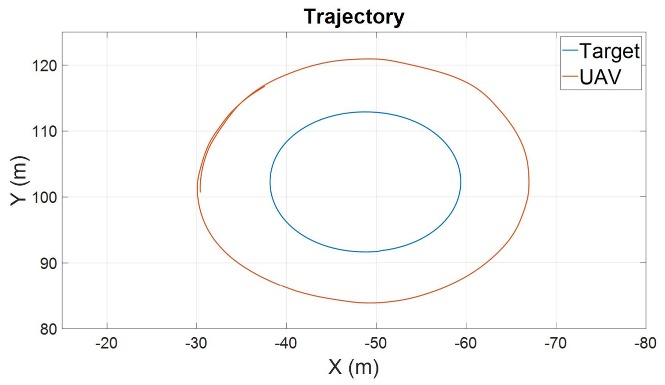

Fig. 19. Top-view trajectories of the UAV and the target moving in near-circular paths.

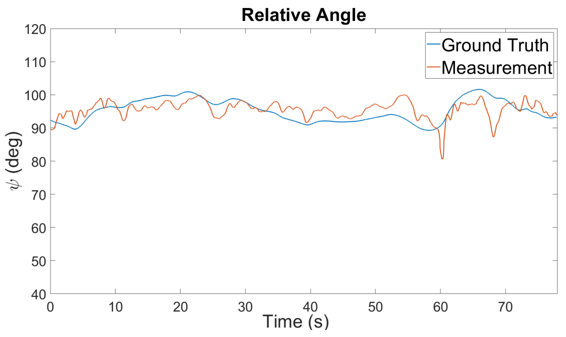

Fig. 20. Trajectories of the measured orientation obtained from the YOLO DNN and the ground-truth orientation.

very close to the ground-truth angle, which implies that the orientation as estimated using the YOLO approach is accurate. Moreover, since ψ is close to the desired orientation ψ_{des}, which was set at 90°, the orientation error e_{ψ} was around zero due to the feedback control v_{cx} defined in (17).

VII. Experiments

Two experiments[2] were conducted in this section to verify the efficacy of the developed controller, and the source code[3] can be found at GitHub. Due to the space constraint, a 1/10 model car was used as the target for the experiments. The ground-truth dimensions of the car were 44 cm $\times$16 cm $\times$ 13 cm (length $\times$ width $\times$ height). The camera used for the experiment was the ZED stereo camera from Stereolabs, Inc., San Francisco, CA, USA. The setup for the two subsequent experiments is as follows. The car started from rest at 12 s, then accelerated for 6 s, and finally decelerated back to zero velocity. The hardware architecture is shown in Fig. 21.

Remark 6: In the simulations and experiments, the same dynamics model defined in (6) was used, but the depths (i.e., Z) were obtained in different ways. Since stereo cameras are more reliable in measuring depth, using a stereo camera to measure the depth is preferable. Therefore, a stereo camera was used in the experiments to directly measure the depth. However, in Gazebo, the stereo camera was not available, and therefore, a monocular camera was used to obtain the depth by applying (7), derived based on the pinhole model.

[2]https://www.youtube.com/watch?v=SX-qqos6×4w

[3]https://github.com/amychen1102/ncrl_motion_prediction_and_robust_tracking

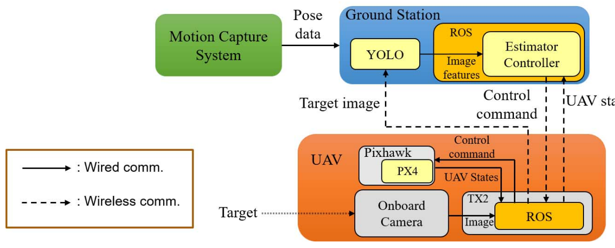

Fig. 21. Hardware architecture for the experiments.

The detailed hardware setup and its data flow of the system shown in Fig. 21 are described as follows.

1) Motion-capture system (green block): OptiTrack
   a) Used to measure the positions and orientations of the UAV and car, and the measurement data of the car are used for comparison, not for feedback control.
2) UAV computation platform (orange block): onboard camera (ZED), TX2, and Pixhawk
   a) Onboard camera.
      i) Used to capture car image for target detection.
   b) TX2.
      i) Runs ROS.
      ii) Used to process and send the captured image to the ground station for processing by the YOLO DNN.
      iii) Communicates the control command and UAV states between Pixhawk and the ground station.
   c) Pixhawk.
      i) Runs PX4 software to execute a low-level control algorithm.
      ii) Computes the UAV states and sends it to the ground station.
3) Ground station (blue block): a laptop with an Intel i7-6700 CPU and an NVIDIA GTX 1080 Ti GPU
   a) Runs ROS.
   b) Runs OptiTrack software to resolve the position and orientation of the UAV and send them to the UAV.
   c) Runs the YOLO DNN to generate detection outcomes.
   d) Executes the tracking control and UKF algorithms developed in this study.
   e) Sends the control command to the UAV via TX2 under the ROS platform.

A. Experiment 1: Constant-Velocity Motion Model

Fig. 22 shows the velocity estimated using the UKF with the constant-velocity model (orange line) and the ground-truth velocity (blue line). Under the constant-velocity model, the estimated velocity remained constant when the car was occluded from 16 to 19 s, as indicated by the light-yellow

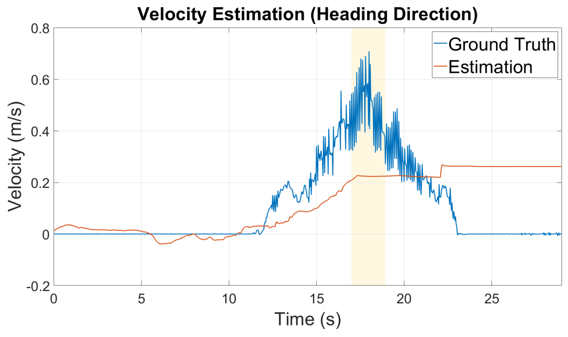

Fig. 22. Ground-truth and estimated velocities of the moving target using the constant-velocity model.

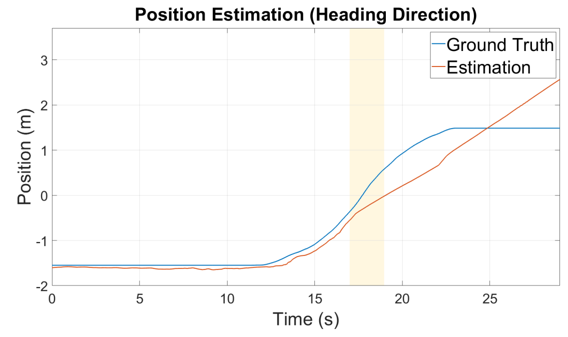

Fig. 23. Ground-truth and estimated positions of the moving target using the constant-velocity model.

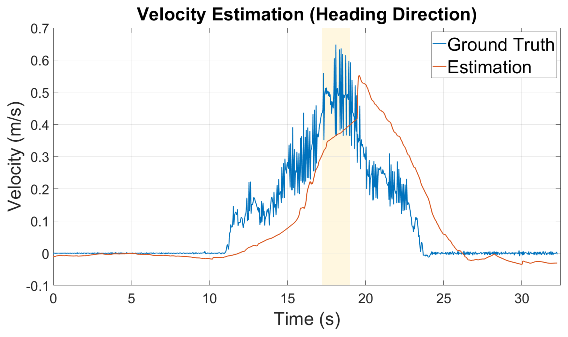

Fig. 24. Ground-truth and estimated velocities of the moving target using the QP-based motion model.

background color in Figs. 22–28. The estimation error of the velocity increased from 12 to 18 s, but then gradually diverged thereafter.

Remark 7: By comparing the simulation and experiment results (e.g., Figs. 13 and 24), it shows that the time delay between the estimation and the ground truth velocities became larger in the experiments, and this divergence might be partly attributable to the computation and communication delays.

The corresponding position trajectories in the heading direction are shown in Fig. 23, where the occlusion of the car from 16 to 19 s led to detection failure.

B. Experiment 2: QP-Based Motion Model

Fig. 24 shows the results obtained in the experiments using the combined QP-based motion model, and it shows that using the QP-based motion model for estimation during occlusion provided a better velocity estimation compared to using the constant-velocity model in Experiment 1. In addition, the gap between the ground truth and the estimation in Fig. 24 can be attributed to the lower detection looping rate of 20 Hz, compared with the looping rate of 90 Hz in the simulation.

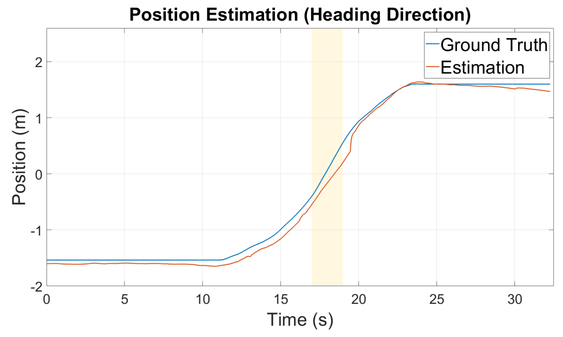

Fig. 25. Ground-truth and estimated positions of the moving target using the QP-based motion model.

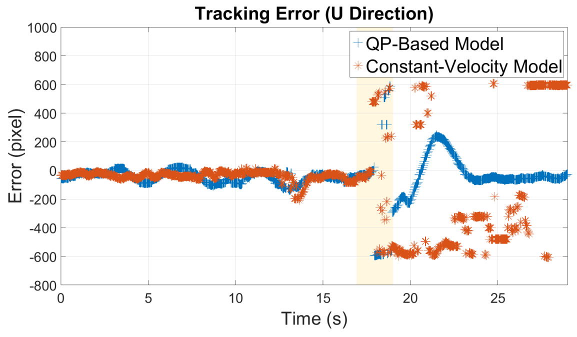

Fig. 26. Image tracking error in the u-direction.

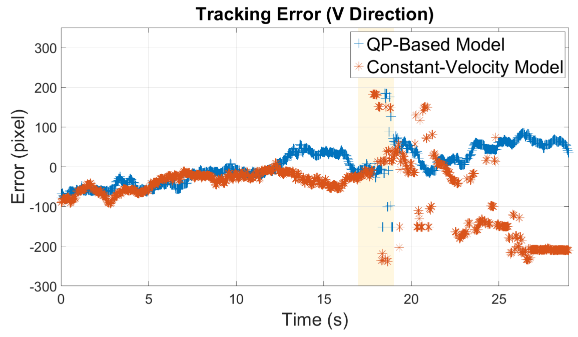

Fig. 27. Image tracking error in the v-direction.

The corresponding position trajectories in the heading direction are shown in Fig. 25, which are smoother than those shown in Fig. 23.

C. Comparison: Tracking Errors

Figs. 26–28 show that at the beginning of the experiment (before occlusion occurred), the tracking errors (e) were close to zero for both the constant-velocity and QP-based motion models. Similar to the observation in the simulation in Section VI, these two methods clearly performed differently after occlusion had occurred. That is, from 16 to 19 s, the error obtained from the constant-velocity model fluctuated markedly, and it diverged thereafter. In contrast, the QP-based motion model method showed good tracking performance

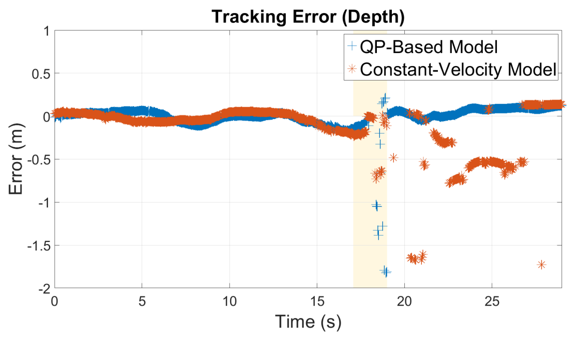

Fig. 28. Image tracking error in the d-direction.

despite the presence of occlusion, which also reflects what is shown in Fig. 25.

VIII. Conclusion

A tracking controller for UAVs has been developed to track moving targets performing the free motion. To the best of our knowledge, this is the first work that has used a DNN (in this case, a YOLO-based one) to generate a bounding box and has included the orientation of the target as one of the features to be estimated and tracked. The tracking during occlusion was enhanced by employing a motion model derived by QP as a process model in the UKF. Since no motion constraint is assumed for the target, the developed controller can be applied to tracking various moving targets provided that they can be detected. Simulations have demonstrated that the developed estimator and controller perform well in the presence of occlusion. Experiments are also conducted to verify the observation in the simulation and to prove the efficacy of the developed estimator and controller.

References

[1] V. K. Chitrakaran, D. M. Dawson, W. E. Dixon, and J. Chen, "Identification of a moving object's velocity with a fixed camera," *Automatica*, vol. 41, no. 3, pp. 553–562, Mar. 2005. [Online]. Available: http://ncr.mae.ufl.edu/papers/auto05.pdf

[2] M. Vrba, D. Hert, and M. Saska, "Onboard marker-less detection and localization of non-cooperating drones for their safe interception by an autonomous aerial system," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3402–3409, Oct. 2019.

[3] M. Vrba and M. Saska, "Marker-less micro aerial vehicle detection and localization using convolutional neural networks," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2459–2466, Apr. 2020.

[4] J. Schlichenmaier, N. Selvaraj, M. Stolz, and C. Waldschmidt, "Template matching for radar-based orientation and position estimation in automotive scenarios," in *Proc. IEEE MTT-S Int. Conf. Microw. Intell. Mobility (ICMIM)*, Nagoya, Japan, Mar. 2017, pp. 95–98.

[5] A. W. Stroupe, M. C. Martin, and T. Balch, "Distributed sensor fusion for object position estimation by multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Autom.*, Seoul, South Korea, May 2001, pp. 1092–1098.

[6] A. Kamthe, L. Jiang, M. Dudys, and A. Cerpa, "SCOPES: Smart cameras object position estimation system," in *Proc. EWSN*, Berlin, Germany, Feb. 2009, pp. 279–295.

[7] H.-S. Ahn and K. H. Ko, "Simple pedestrian localization algorithms based on distributed wireless sensor networks," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4296–4302, Oct. 2009.

[8] V. K. Chitrakaran and D. M. Dawson, "A Lyapunov-based method for estimation of Euclidean position of static features using a single camera," in *Proc. Amer. Control Conf.*, New York, NY, USA, Jul. 2007, pp. 1988–1993.

[9] D. Braganza, D. M. Dawson, and T. Hughes, "Euclidean position estimation of static features using a moving camera with known velocities," in *Proc. 46th IEEE Conf. Decis. Control*, New Orleans, LA, USA, Dec. 2007, pp. 2695–2700.

[10] A. Chhatkuli, D. Pizarro, T. Collins, and A. Bartoli, "Inextensible non-rigid structure-from-motion by second-order cone programming," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2428–2441, Oct. 2018.

[11] A. Agudo and F. Moreno-Noguer, "Simultaneous pose and non-rigid shape with particle dynamics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2179–2187.

[12] H. Shi, G. Sun, Y. Wang, and K.-S. Hwang, "Adaptive image-based visual servoing with temporary loss of the visual signal," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 1956–1965, Apr. 2019.

[13] A. Mateus, O. Tahri, and P. Miraldo, "Active structure-from-motion for 3D straight lines," Jul. 2018, *arXiv:1807.00753*. [Online]. Available: http://arxiv.org/abs/1807.00753

[14] R. Spica, P. R. Giordano, and F. Chaumette, "Active structure from motion: Application to point, sphere, and cylinder," *IEEE Trans. Robot.*, vol. 30, no. 6, pp. 1499–1513, Dec. 2014.

[15] R. Spica, P. R. Giordano, and F. Chaumette, "Plane estimation by active vision from point features and image moments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 6003–6010.

[16] P. R. Giordano, R. Spica, and F. Chaumette, "Learning the shape of image moments for optimal 3D structure estimation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 5990–5996.

[17] I. F. Mondragon, P. Campoy, M. A. Olivares-Mendez, and C. Martinez, "3D object following based on visual information for unmanned aerial vehicles," in *Proc. IEEE IX Latin Amer. Robot. Symp. IEEE Colombian Conf. Autom. Control*, Oct. 2011, pp. 1–7.

[18] H. Wang, B. Yang, J. Wang, X. Liang, W. Chen, and Y.-H. Liu, "Adaptive visual servoing of contour features," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 2, pp. 811–822, Apr. 2018.

[19] D. Hulens and T. Goedemé, "Autonomous flying cameraman with embedded person detection and tracking while applying cinematographic rules," in *Proc. Conf. Comput. Robot Vis.*, Edmonton, AB, Canada, May 2017, pp. 56–63.

[20] Y. Liu, Q. Wang, H. Hu, and Y. He, "A novel real-time moving target tracking and path planning system for a quadrotor UAV in unknown unstructured outdoor scenes," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 11, pp. 2362–2372, Nov. 2019.

[21] J. H. Lee, J. D. Millard, P. C. Lusk, and R. W. Beard, "Autonomous target following with monocular camera on UAS using recursive-RANSAC tracker," in *Proc. Int. Conf. Unmanned Aircr. Syst.*, Dallas, TX, USA, Jun. 2018, pp. 1070–1074.

[22] A. P. Dani, N. R. Fischer, and W. E. Dixon, "Single camera structure and motion," *IEEE Trans. Autom. Control*, vol. 57, no. 1, pp. 238–243, Jan. 2012.

[23] B. Penin, P. R. Giordano, and F. Chaumette, "Vision-based reactive planning for aggressive target tracking while avoiding collisions and occlusions," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3725–3732, Oct. 2018.

[24] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "PAMPC: Perception-aware model predictive control for quadrotors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2018, pp. 1–8.

[25] A. P. Dani, Z. Kan, N. R. Fischer, and W. E. Dixon, "Structure and motion estimation of a moving object using a moving camera," in *Proc. Amer. Control Conf.*, Baltimore, MD, USA, Jun. 2010, pp. 6962–6967.

[26] A. Dani, Z. Kan, N. Fischer, and W. E. Dixon, "Structure estimation of a moving object using a moving camera: An unknown input observer approach," in *Proc. IEEE Conf. Decis. Control*, Orlando, FL, USA, Dec. 2011, pp. 5005–5012.

[27] A. Parikh, T.-H. Cheng, H.-Y. Chen, and W. E. Dixon, "A switched systems framework for guaranteed convergence of image-based observers with intermittent measurements," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 266–280, Apr. 2017.

[28] Z. Mejri, L. Sidhom, and A. Abdelkrim, "Structure recovering of moving object using a real time approach," in *Proc. Int. Conf. Adv. Syst. Electr.*, Mar. 2018, pp. 66–71.

[29] A. Parikh, T.-H. Cheng, R. Licitra, and W. E. Dixon, "A switched systems approach to image-based localization of targets that temporarily leave the camera field of view," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 6, pp. 2149–2156, Nov. 2018.

[30] A. Parikh, T.-H. Cheng, and W. E. Dixon, "A switched systems approach to vision-based localization of a target with intermittent measurements," in *Proc. Amer. Control Conf.*, Jul. 2015, pp. 4443–4448.

[31] D. Chwa, A. P. Dani, and W. E. Dixon, "Range and motion estimation of a monocular camera using static and moving objects," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1174–1183, Jul. 2016.

[32] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, South Korea, Oct. 2016, pp. 446–453.

[33] Z. Cao *et al.*, "Image dynamics-based visual servoing for quadrotors tracking a target with a nonlinear trajectory observer," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 1, pp. 376–384, Jul. 2017.

[34] D. Zheng, H. Wang, and W. Chen, "Image-based visual tracking of a moving target for a quadrotor," in *Proc. 11th Asian Control Conf.*, Gold Coast, QLD, Australia, Dec. 2017, pp. 198–203.

[35] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: https://arxiv.org/abs/1804.02767

[36] A. J. Whalen, S. N. Brennan, T. D. Sauer, and S. J. Schiff, "Observability and controllability of nonlinear networks: The role of symmetry," *Phys. Rev. X*, vol. 5, Jan. 2015, Art. no. 011005. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.5.011005

[37] E. A. Wan and R. van der Menve, "The unscented Kalman filter for nonlinear estimation," in *Proc. IEEE Adapt. Syst. Signal Process., Commun., Control Symp.*, Lake Louise, AB, Canada, Oct. 2000, pp. 153–158.

[38] M. Gupta, L. Behera, V. K. Subramanian, and M. M. Jamshidi, "A robust visual human detection approach with UKF-based motion tracking for a mobile robot," *IEEE Syst. J.*, vol. 9, no. 4, pp. 1363–1375, Dec. 2015.

[39] X. Chen, X. Wang, and J. Xuan, "Tracking multiple moving objects using unscented Kalman filtering techniques," 2018, *arXiv:1802.01235*. [Online]. Available: http://arxiv.org/abs/1802.01235

[40] S. Gao, G. Hu, and Y. Zhong, "Windowing and random weighting-based adaptive unscented Kalman filter," *Int. J. Adapt. Control Signal Process.*, vol. 29, no. 2, pp. 201–223, Feb. 2015.

[41] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[42] J. Pestana, J. L. Sanchez-Lopez, S. Saripalli, and P. Campoy, "Computer vision based general object following for GPS-denied multirotor unmanned vehicles," in *Proc. Amer. Control Conf.*, Portland, OR, USA, Jun. 2014, pp. 1886–1891.

Jun-Ming Li received the M.S. degree from the Department of Mechanical Engineering, National Chiao Tung University, Taiwan, in 2019.

He is currently a Research and Development Engineer of industrial robots with Syntec Technology Company Ltd., Hsinchu, Taiwan. His current research interests include vision-based control, tracking control, model predictive control, and robotics.

Ching-Wen Chen received the B.S. degree from the Department of Mechanical Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 2018, where she is currently pursuing the M.S. degree.

Her current research interests include multi-agent distributed control algorithms, model predictive control, image-based visual servo control, and robotics.

Teng-Hu Cheng received the Ph.D. degree from the Department of Mechanical Engineering, University of Florida, Gainesville, FL, USA, in 2015.

In 2016, he joined the Department of Mechanical Engineering, National Chiao Tung University, Hsinchu, Taiwan. His research interests include robotics, autonomous systems, networked system control, switched control, event-driven control, and nonlinear control.

Prof. Cheng received the O. Hugo Schuck Best Paper Award from the American Automatic Control Council in 2015.