Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 50-1 (2017) 11447–11452

Vision-based Tracking by a Quadrotor on ROS *

Yusheng Wei [*] Zongli Lin [*]

[*] *Charles L. Brown Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904-4743, USA (e-mail: yw4dp@virginia.edu, zl5y@virginia.edu)*

Abstract: Real-time following of a moving target by a flying tracker without radar detection and GPS reception inspires vision-based tracking through the use of surveillance camera, laser-scanned lidar, and SLAM (simultaneous localization and mapping). In this paper, we propose vision-based tracking of a moving target in three consecutive steps, image processing, coordinate transformation and trajectory following. A testing platform containing two spawned quadrotors as target and tracker is developed on ROS, from which sequence of images are generated by an on-board camera of the tracker while the target moves along a rectangular path on the ground. An image processing algorithm based on target recognition is proposed to achieve visual tracking on image frames. Coordinate transformation is established to recover the original target trajectory as the desired path of the tracker in the real world from the image processing algorithm. The target following control law then moves the tracker along the desired trajectory point-by-point through velocity control.

Keywords: Tracking, quadrotor, vision-based, ROS

1. INTRODUCTION

The past few decades have witnessed rapid increase in automated robots taking tedious or dangerous tasks such as surveillance, rescuing, geological exploration or delivery. Among these tasks, target tracking by using unmanned aerial vehicles (UAVs) has found many applications in engineering practice with both research and commercial background (see, for example, Dobrokhodov et al. (2008), Guenard et al. (2008), Li et al. (2010), Santamaria-Navarro and Andrade-Cetto (2013)). Equipped with intelligent control units and highly accurate sensors, UAVs have the advantages of small size and hence more accessible manipulation compared with traditional aerial vehicles. In particular, miniature UAVs such as quadrotors diminish the cost of accidental crashes while maintaining performance such as flight stability, adaptation to unexpected scenarios and maneuverability. However, the lack of simulation platform in the testing phase of the quadrotor development leads to recurrent failures caused by sensor saturation or motor malfunction. This results in the development of the Robot Operating System (ROS), which is an open-source software package as a testing platform for various automated robots and has become one of the most popular tools for developing and implementing control algorithms to achieve a given task.

Traditional approach to target tracking by a UAV involves radar detection of the shape, size and velocity of the moving target. On the other hand, GPS provides the global coordinates of the tracker to facilitate navigation. However, in some applications, radar signals in the form of electromagnetic waves can be easily interfered or even completely blocked. Moreover, some GPS-denied environments limit the access of precise information of the tracker's global coordinates, which leads to considerable tracking errors, or even instability of the tracker itself. Considering these difficulties, researchers implemented surveillance camera and laser-scanned lidar to substitute the radar detector and the GPS navigation system, respectively. In particular, with the help of the image processing technology, image trajectory of a moving target can be obtained by the use of a surveillance camera. Moreover, real-time tracker positions are estimated by a SLAM algorithm at the scanning rate of lidar. These two sets of information can be considered substitutes for the radar detecting data of the target and the GPS global coordinates of the tracker. They jointly provide sufficient feedback for the tracker to achieve a tracking task.

Vision-based tracking by a UAV basically consists of three consecutive steps, image processing, coordinate transformation and trajectory following. By applying an image processing algorithm to a sequence of image frames taken by the surveillance camera on the tracker, visual tracking of the target on these image frames can be achieved. Coordinate transformation between the image frame and the real world frame then generates the desired trajectory for the tracker in the real world frame such that the projection of the desired trajectory on the ground plane imitates the original target movement. Based on the real-time estimate of the current tracker position obtained from the SLAM algorithm with the help of lidar detection, a trajectory following law propels the UAV to follow the desired trajectory such that target tracking can be accomplished.

* This work was supported in part by the U.S. Army Research Office under grant W911NF1510275.

2405-8963 © 2017, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2017.08.1814

Various image processing algorithms and navigation laws have been proposed to achieve vision-based tracking. Liu et al. (2012) introduced a new image processing method, referred to as the grid-based Bayesian approach, whose robustness to image jitter or target occlusion is improved and computational complexity is reduced compared with other algorithms such as the snake tracker in Xu and Prince (1998) and the Monte Carlo tracker in Cui et al. (2006). This improvement in computational efficiency made real-time tracking of a moving target possible provided that the actuation of navigation law is fast enough. Robustness of the method to erratic movement of the target was further improved in Qian et al (2015). In Dobrokhodov et al. (2008), tracking is achieved by collecting the information of the horizontal distance between the target and the tracker from an image frame as feedback to navigation law that keeps the distance within a reasonable range in the real world frame. The centroid of the moving target in an image frame is then guaranteed to be inside a given area, and the navigation law guides the tracker to draw a rounded orbit above the target. Despite these results, the majority of the literature only focuses either on the image processing algorithm or the navigation law, and the effort to design an integrated vision-based tracking approach that combines these two separate parts is lacking, let alone the development of a simulation platform for testing purposes of such an approach.

In this paper, we propose a comprehensive approach to achieving vision-based tracking by a quadrotor, where image processing and trajectory following are combined through a coordinate transformation. A testing platform including two spawned quadrotors as the target and the tracker in the vision-based tracking simulation is built based on the hector-quadrotor package of ROS. To track a rectangular movement of the target on the ground, a sequence of image frames are obtained from a surveillance camera on the tracker, and, with an image processing algorithm based on target recognition, the target trajectory on the image frames is identified. A coordinate transformation then recovers an estimated target trajectory in the real world frame as the desired tracker trajectory in the same frame from the image processing algorithm. With the information of the real tracker position, we achieve point-to-point following of the desired trajectory by the tracker through velocity control. Comparison between the target trajectory and the tracker trajectory in the real world frame shows that the proposed tracking approach achieves satisfactory accuracy.

The remainder of this paper is organized as follows. Section 2 develops a testing platform for our vision-based tracking simulation by using the hector-quadrotor package of ROS. Two quadrotor models are spawned as the target and the tracker for the simulation purpose. An image processing algorithm based on target recognition, and a coordinate transformation are proposed in Sections 3 and 4, respectively. Section 5 presents a trajectory following law for the tracker based on velocity control. Comparison between the tracker trajectory and the target trajectory shows satisfactory tracking performance. Section 6 concludes the paper.

2. DEVELOPMENT OF A TESTING PLATFORM USING ROS

Frequent accidental crashes of quadrotors in testing environments lead to the development of an open-source hector-quadrotor package in ROS. Considering simple aerial dynamics and accessible maneuverability of the quadrotor model through keyboard control, we build our testing platform for vision-based tracking simulation by using the hector-quadrotor package.

2.1 Components of the Testing Platform

The testing platform consists of a ground plane, a willow garage map and two quadrotors with the same mechanical structure and aerial dynamics. The ground plane can be considered as the benchmark for zero altitude in the real world frame, and also as the landing and taking-off surface for the two quadrotors. The willow garage map is a typical indoor scenario that can be readily implemented from the Gazebo simulator. Note that the choice of map is highly non-unique, and even the target model can have multiple choices such as Turtlebot or PR2 as long as the robot is maneuverable through keyboard control to move freely on the ground. In our simulation, the testing platform is developed by using the ROS Indigo distribution installed on the Ubuntu 14.04.4 LTS (Trusty) operating system.

2.2 Quadrotor Model

We spawn two quadrotors based on the same model described by quadrotor_hokuyo_utm30lx.urdf.xacro in the hector_quadrotor_urdf package of ROS (Meyer et al. (2012)). The model consists of a mechanical base with two additional sensors, namely, a Hokuyo UTM-30LX LIDAR and a forward facing camera. The base consists of an inertial measurement unit (IMU), a barometric sensor and an ultrasonic sensor in addition to a typical propelling system. As pointed out in Meyer et al. (2012), IMU provides speed and orientation information to the control unit of the quadrotor that guarantees flight stability, the barometric sensor measures the altitude of the quadrotor by sensing atmospheric pressure, and the ultrasonic sensor guarantees safe landing by measuring the distance between the quadrotor and the landing surface, which determines the instance of switching on the decelerator at the landing phase.

Through implementation of a SLAM algorithm proposed in Kohlbrecher et al. (2011), the laser-scanned lidar estimates the 2D coordinates of the tracker in the real world frame and simultaneously constructs an estimated map for the unknown scenario. With the altitude measured by the barometric sensor, complete 3D coordinates of the tracker in the real world frame can be obtained. Note that the SLAM is achieved through tracker emitting laser light and receiving echo reflected from surrounding surface. Indoor scenarios such as the willow garage map provides sufficient reflective surface such as walls and furnitures. However, some of the outdoor scenarios lacking these surfaces lead to SLAM failure. This can be readily verified by flying the tracker in our testing platform sufficiently high. Some effort has been made in Caballero et al. (2008) to improve the adaptation of the SLAM algorithm to the tracker with medium and high aerial altitude. Since the tracker in our testing platform is built based on the quadrotor_hokuyo_utm30lx.urdf.xacro model, we retain the use of Hokuyo UTM-30LX LIDAR implemented on the model and the SLAM algorithm, which was from Kohlbrecher et al. (2011).

The surveillance camera on the tracker is manually set to face downward, which provides the widest tracking prospective regardless of any movement of the target on the ground. According to ROS, image frames generated by the camera have 640 pixels in width and 480 pixels in height, and have a resolution of 3780 pixels/m both horizontally and vertically.

2.3 Initial Conditions

The initial conditions for the vision-based tracking approach are defined as follows. In order for the target to simulate a moving object on the ground, we fly it with a small constant altitude in the real world frame and move it along a rectangular path. Target movement is achieved through velocity control via keyboard. During a period of target moving, the tracker stays static in the air with an altitude a little bit lower than that of the side walls. This provides lidar on the tracker with sufficient reflective surface in order to achieve full potential of the SLAM algorithm and also prevents target deformation on a sequence of image frames caused by flight vibration of the tracker. Considering that surveillance camera is fixed on board, we initialize the tracker orientation such that an alignment between the image frame and the real world frame is satisfied. This requires that the x and y axes of the image frame align with those of the real world frame, both of which are compared in the image frame or the real world frame. This requirement brings considerable convenience to the coordinate transformation to be presented in Section 4. Note that the target trajectory in the real world frame can be considered a benchmark for determining the performance of the integrated vision-based tracking approach when compared with the real tracker trajectory. Hence, we record this information in the Gazebo simulator for the purpose of the comparison to be presented in Section 5.

Visualizations of our testing platform in rviz of ROS and in the Gazebo simulator are shown in Fig. 1 and Fig. 2, respectively. The layout of the willow garage map is illustrated in the Gazebo simulator. It can be readily observed in Fig. 1 that the target trajectory is set to be green and the tracker trajectory is set to be red for the purpose of distinguishing one from the other. Moreover, the laser-scanned lidars on both quadrotors detect the side walls whose edges are shown in purple and orange lines in rviz. An estimated map of the unknown scenario, namely, the willow garage map in our testing platform, generated by the SLAM algorithm, is illustrated as a light grey surface with black boundaries. The initial conditions for vision-based tracking approach can be verified in both rviz and in the Gazebo simulator.

3. IMAGE PROCESSING ALGORITHM

Visual tracking of the target in a sequence of images obtains accurate estimate of its 2D coordinates along the

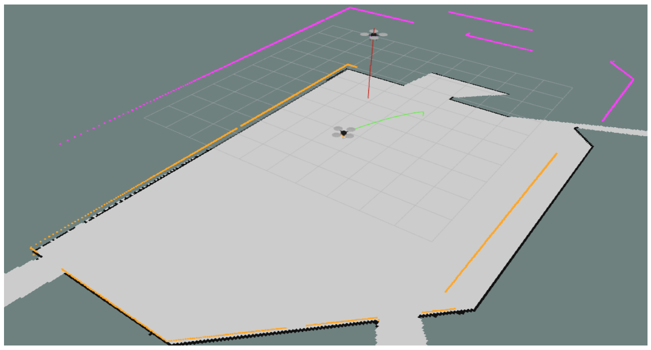


Fig. 1. Testing platform visualized in rviz.

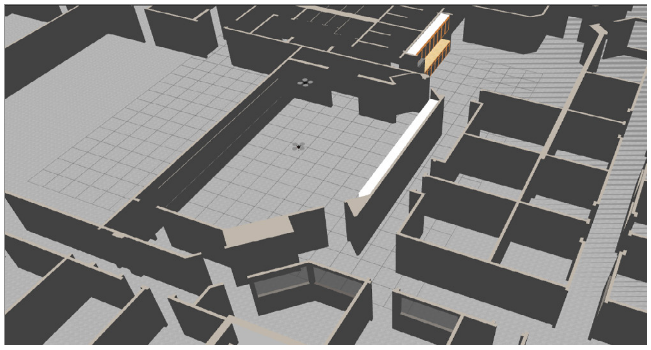


Fig. 2. Testing platform visualized in the Gazebo.

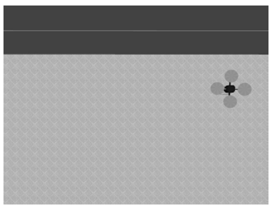
image trajectory. Considering the simple background of the willow garage scenario and the distinguishable shape of the target, we propose an image processing algorithm, Algorithm 1, based on target recognition to achieve visual tracking.

Algorithm 1 Image processing algorithm

```
(1) Import image data of a sequence of $N$ consecutive image frames;
(2) For the $n$th frame, convert the image data into binary matrix $K_n$ of dimensions $h \times w$ with threshold equalling $level$, where $h$ and $w$ represent the height and the width of the image frame, expressed in pixels;
(3) Find the logical negation of $K_n$ and denote it as $F_n$;
(4) For any area of adjoining elements 1 in $F_n$ with size not smaller than $connectivity$, distribute an area number starting from one to each of these areas, and label every element within the area with its area number, then denote the labelled matrix as $L_n$ with the same dimensions of $F_n$;
(5) Determine the largest element $AreaNumber\_n$ and the sizes of all areas in $L_n$, and generate a vector $AreaSize\_n$ with dimension $AreaNumber\_n$, whose elements are the area sizes;
(6) Determine the $R$th largest number $S_n$ in $AreaSize\_n$, and its corresponding area number $M_n$;
(7) For those elements of $L_n$ that equal $M_n$, determine each of its horizontal and vertical coordinates in $L_n$, namely, $x_{ni}$ and $y_{ni}$ for $i \in \mathbf{I}[1, S_n]$, where $\mathbf{I}[a, b]$ with $a \in \mathbb{Z}$, $b \in \mathbb{Z}$, $a \leq b$, denotes the set of all integers between $a$ and $b$;
(8) Compute $x_n = \frac{\sum_{i=1}^{S_n} x_{ni}}{S_n}$ and $y_n = \frac{\sum_{i=1}^{S_n} y_{ni}}{S_n}$.
```

The parameters in Algorithm 1 include the number of image frames $N \in \mathbb{N}$ in a sequence of images, the binary threshold $level \in [0,1]$, the number of horizontal pixels of image $h \in \mathbb{N}$, the number of vertical pixels of image $w \in \mathbb{N}$, the size threshold $connectivity \in \mathbb{N}$ for qualifying an area of adjoining elements 1 in $F_n, n \in \mathbf{I}[1,N]$, and $R \in \mathbb{N}$, which is such that the target area in F_n, $n \in \mathbf{I}[1,N]$, has the Rth largest area size among all areas. It is assumed here that R is constant for all $n \in \mathbf{I}[1,N]$ considering the fact that the composition of each image remains unchanged during the target movement.

Simulation of visual tracking is done by using MATLAB 2013a on a PC with an Intel Core i7 CPU (2.0 GHz and 8 GB RAM). A sequence of $N = 23$ image frames recording a rectangular target movement with constant speed are generated by the surveillance camera on the tracker. The first image frame of the sequence is shown in Fig. 3(a), and its binary counterpart with threshold $level$ equalling 0.5 is presented in Fig. 3(b). Note in Fig. 3(b) that there are two connected black areas corresponding to two different objects, namely, a side wall and the target, under $connectivity = 8$. Moreover, it can be readily observed that the target area is the second largest among all visible areas. Hence, $R = 2$ in this simulation. Other parameters of Algorithm 1 are recalled from Section 2 as $h = 640$ and $w = 480$. Successful visual tracking of the target moving along a rectangular path is achieved on all image frames. Fig. 4 shows Frames #$1, 5, 7, 9, 13, 17, 21$ and 23 of the processed image sequence, where the estimated target position, (x_n, y_n), on each image frame is marked as a red cross.


(a) First image frame under a rectangular target movement.

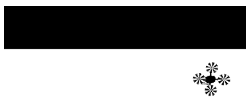

(b) Binary image of Fig. 3(a).

Fig. 3. First image frame under a rectangular target movement and its binary counterpart with $level = 0.5$.

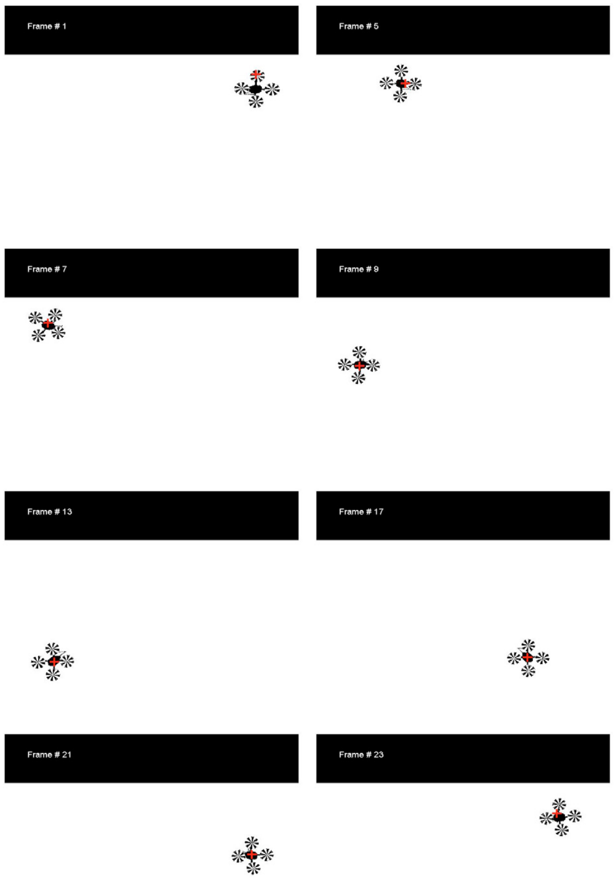


Fig. 4. Visual tracking of the target.

4. COORDINATE TRANSFORMATION

The image processing algorithm only provides the tracker with an estimated 2D target trajectory in the image frame. To achieve target following, we consider moving the tracker from its static position as given by the initial condition in Section 2 along a desired trajectory in the plane that parallels the ground such that the projection of this trajectory on the ground plane imitates the target trajectory in the real world frame. This provides a necessary condition for the tracker to achieve real-time target following as long as the actuation of the trajectory following law is fast enough.

The desired tracker displacements and the initial tracker position together constitute a desired tracker trajectory in the real world frame. Considering that the projection of the desired tracker trajectory imitates the original target trajectory, it suffices to obtain the target displacement along the original target trajectory in the real world frame. With the alignment between the image frame and the real world frame, which was determined in Section 2 as the initial condition, a coordinate transformation recovers the target displacement in the real world frame from the target displacement in the image frame. Basically, such a coordinate transformation consists of two consecutive steps. Considering the image resolution, we first transfer the target displacement expressed in pixels to the target displacement in meters, both in the image frame. Next, by using the image scale between the image frame and the real world frame, we further transfer the target displacement in the image frame expressed in meters to the target displacement in the real world frame, also expressed in meters. Then, with the initial position of the tracker, a complete desired tracker trajectory would be constructed.

Both the horizontal and the vertical resolutions of the image are recalled from Section 2 as 3780 pixels/m. Considering the symmetric structure of the quadrotor model, we measure the diagonal distance of a quadrotor as 2.5 cm in the image frame and 0.8 m in the real world frame, which implies that image scale is 32 times. Now we decompose both the target displacement $\boldsymbol{d}_p$, expressed in pixels, in the image frame, and the target displacement $\boldsymbol{d}_r$, expressed in meters, in real world frame as,

$$\boldsymbol{d}_p = [d_{px}\ d_{py}]^{\mathrm{T}}\,,\ \boldsymbol{d}_r = [d_{rx}\ d_{ry}]^{\mathrm{T}}\,,$$

where d_{px} and d_{py} are the projections of $\boldsymbol{d}_p$ to the x axis and the y axis of the image frame, d_{rx} and d_{ry} are the projections of $\boldsymbol{d}_r$ to the x axis and the y axis of the real world frame. Positive directions of the x axes and y axes of the image frame and the real world frame in a typical image from Section 3 is shown in Fig. 5, then, the coordinate transformation between $\boldsymbol{d}_p$ and $\boldsymbol{d}_r$ can be written as follows,

$$d_{rx} = d_{px}\frac{TF_{\text{scale}}}{TF_{\text{resolution}}},\ \ d_{ry} = -d_{py}\frac{TF_{\text{scale}}}{TF_{\text{resolution}}}. \qquad (1)$$

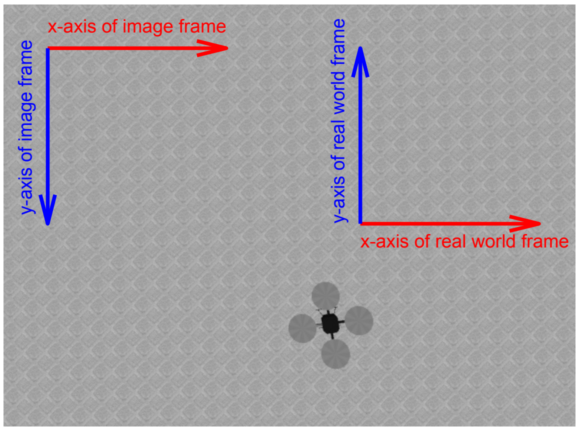


Fig. 5. The x axes and y axes of the image frame and the real world frame in our simulation.

For the nth and $(n+1)$th image frames in Section 3, the target displacement on these two consecutive images can be calculated as,

$$d_{px} = x_{n+1} - x_n,\ \ d_{py} = y_{n+1} - y_n,$$

where x_n and y_n are as defined in Step 8 of Algorithm 1. Then, a pair (d_{rx}, d_{ry}) can be obtained by using (1). With the initial tracker position given by

$$X_{\text{int}} = -2.3744, Y_{\text{int}} = -0.6065, Z_{\text{int}} = 2.68734,$$

a desired tracker trajectory whose projection to the ground plane imitates the target trajectory in the real world frame is established. Here, we assume that the tracker remains at a constant altitude in the real world frame while tracking is in process.

5. TRAJECTORY FOLLOWING LAW

In order for the tracker to follow the desired trajectory in the real world frame as presented in Section 4, we design a point-to-point trajectory following law, where the tracker would leave for the next desired position along the desired trajectory only if its current position is within a ball of its current desired position. The radius of this ball can be determined by the requirement for tracking accuracy, and hence is referred to as tolerant radius. On the one hand, a large tolerant radius results in the tracker deviating from the desired trajectory significantly. On the other hand, a small tolerant radius leads to the tracker following the desired trajectory with a low speed, and hence fails the real-time tracking task.

By using the error information between the desired position and the current position of the tracker, we construct a PID controller whose input is the error and whose output is a velocity control signal. The closed-loop system illustrating the velocity control of the tracker along the x axis or the y axis of the real world frame is shown in Fig. 6, where dp denotes the desired tracker position along its desired trajectory, cp denotes the current tracker position produced by the SLAM algorithm, $e = dp - cp$ represents the tracking error of the trajectory following law, and vel denotes the velocity control signal. Also, the PID controller can be expressed as follows,

$$vel(t) = k_p e(t) + k_i \int_0^t e(t)\mathrm{d}t + k_d \frac{\mathrm{d}}{\mathrm{d}t}e(t),\ t \geq 0,$$

where $k_p > 0$, $k_i > 0$ and $k_d > 0$ represent the proportional, integral and derivative gains, respectively.

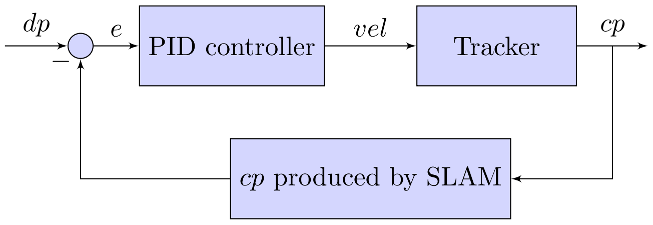


Fig. 6. Closed-loop diagram for the velocity control of the tracker

The velocity control scheme as shown in Fig. 6 asymptotically eliminates the tracking error e for any given initial error. This can be proven as follows. Without loss of generality, we assume that the velocity control scheme is implemented on the tracker along the x axis of the real world frame. From $e = dp - cp$ we readily obtain

$$\dot{e}(t) = -\dot{cp}(t) = -vel(t),$$

where $cp(t)$ and $vel(t)$ are the projections of the current tracker position and velocity to the x axis of the real world frame. By virtue of the PID controller expression, we derive

$$(1 + k_d)\ddot{e}(t) = -k_p\dot{e}(t) - k_i e(t),$$

which implies that $\lim_{t\to\infty} e(t) = 0$.

Then, in view of the notion of the tolerant radius, the target is allowed to head to the next desired position along the desired trajectory as long as the current e is smaller than the tolerant radius. In this case, velocity control of the tracker is implemented $N-1$ times for $N-1$ number of desired positions along the desired trajectory, where N is the number of image frames as defined in Section 3. For the simulation purpose, we choose the tolerant radius to be 0.1 and

$$k_p = 0.5,\ k_i = 0.002,\ k_d = 0.0005,$$

which would guarantee the tracker to follow smoothly the desired trajectory while maintaining flight stability.

Tracking simulation is run under the trajectory following law. The actual tracker trajectory in the real world frame is then recorded by the Gazebo simulator to examine two aspects of tracking accuracy. First, comparison between the tracker trajectory and the desired tracker trajectory shows the performance of the trajectory following law. Second, comparison between the tracker trajectory and the target trajectory shows the performance of the integrated vision-based tracking approach, which incorporates an image processing algorithm, a coordinate transformation and a trajectory following control algorithm.

Figs. 7 and 8 illustrate 3D rectangular trajectories of the target and the tracker in the real world frame and their 2D projections on the ground plane, respectively. Satisfactory trajectory following can be verified by comparing the red solid path in Fig. 8 and the desired tracker trajectory as computed in Section 4. Also, satisfactory vision-based tracking can be readily verified by comparing the red solid path and the blue dashed path both in Fig. 8. It is worth mentioning here that the implementation of the integrated vision-based tracking approach in the case of circular target movement is also performed to demonstrate the robustness of the approach to erratically moving target on the ground, and satisfactory tracking accuracy is also achieved as in the case of rectangular target movement. Illustration of the circular case is omitted due to space limitation of the paper.

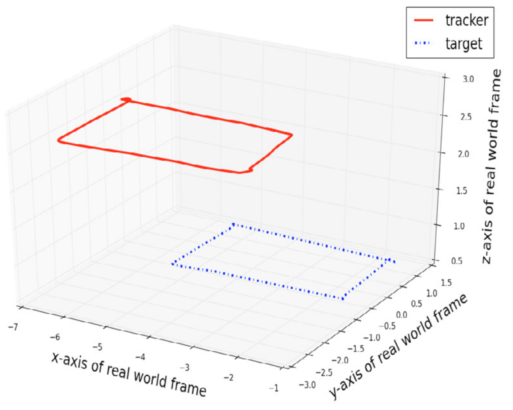


Fig. 7. 3D rectangular trajectories of the target and the tracker in the real world frame.

6. CONCLUSIONS

We considered vision-based tracking by a UAV of a moving target on the ground through simulation. A testing platform containing two independent quadrotors as the target and the tracker was developed on ROS. Through three consecutive steps, image processing, coordinate transformation and trajectory following, the tracker achieves satisfactory tracking performance of a moving target along a rectangular path on the ground. Performance of the integrated tracking approach is illustrated by comparing the tracker trajectory with the target trajectory, both in the real world frame.

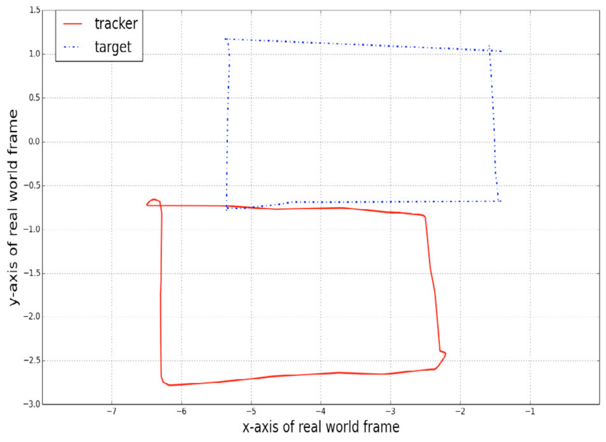


Fig. 8. 2D rectangular trajectories of the target and the tracker in the real world frame.

REFERENCES

F. Caballero, L. Merino, J. Ferruz, and A. Ollero. Vision-based odometry and SLAM for medium and high altitude flying UAVs. *Unmanned Aircraft Systems*, Springer Netherlands, pp. 137-161, 2008.

J. Cui, S. T. Acton, and Z. Lin. A Monte Carlo approach to rolling leukocyte tracking in vivo. *Medical Image Analysis*, Vol. 10, No. 4, pp. 598-610, 2006.

V. N. Dobrokhodov, I. I. Kaminer, K. D. Jones, and R. Ghabcheloo, Vision-based tracking and motion estimation for moving targets using unmanned air vehicles. *Journal of guidance, control, and dynamics*, Vol. 31, No. 4, pp. 907-917, 2008.

N. Guenard, T. Hamel, and R. Mahony. A practical visual servo control for an unmanned aerial vehicle. *IEEE Transactions on Robotics*, Vol. 24, No. 2, pp. 331-340, 2008.

S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingau. A flexible and scalable slam system with full 3d motion estimation. *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pp. 155-160, 2011.

Z. Li, N. Hovakimyan, V. Dobrokhodov, and I. Kaminer. Vision-based target tracking and motion estimation using a small UAV. *49th IEEE Conference on Decision and Control (CDC)*, Atlanta, USA, pp. 2505-2510, 2010.

X. Liu, Z. Lin, and S. T. Acton. A grid-based Bayesian approach to robust visual tracking. *Digital Signal Processings*, Vol. 22, No. 1, pp. 54-65, 2012.

J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. Von Stryk. Comprehensive simulation of quadrotor UAVs using ros and gazebo. *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, Springer Berlin Heidelberg, pp. 400-411, 2012.

S. Qian, Z. Lin, and S. T. Acton. A grid-based tracker for erratic targets. *Pattern Recognitions*, Vol. 48, No. 11, pp. 3527-3541, 2015.

A. Santamaria-Navarro, and J. Andrade-Cetto. Uncalibrated image-based visual servoing. *2013 IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, pp. 5247-5252, 2013.

C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, Vol. 7, No. 3, pp. 359-369, 1998.