2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

A Moving Target Tracking System of Quadrotors with Visual-Inertial Localization

Ziyue Lin[1,2],Wenbo Xu[1,2]and Wei Wang[1,2]

Abstract— **This paper implements a vision-based moving target tracking system of quadrotors with visual-inertial localization in GNSS-denied indoor environments. We use the visual-inertial odometry to estimate the states of the UAV by minimizing visual and inertial residuals, and estimate the states of the target with extended Kalman Filter from visual detection. This research formulates the target tracking problem as optimization-based trajectory generation where a weighted sum cost function jointly penalizes the tracking error, the control cost of the trajectory and the trajectory length, while enforcing the safety and feasibility constraints. We present a strategy that represents the trajectory as piecewise Bézier curves using Bernstein polynomial basis. Due to the special properties of Bézier curves, the position of the entire trajectory and its derivatives can be directly bounded within the safe spaces, thus this facilitating the dynamics of the quadrotor. The proposed strategy can generate smooth and collision-free tracking trajectories and is time and space efficient. We conduct simulations and real-world experiments to validate the effectiveness of our system.**

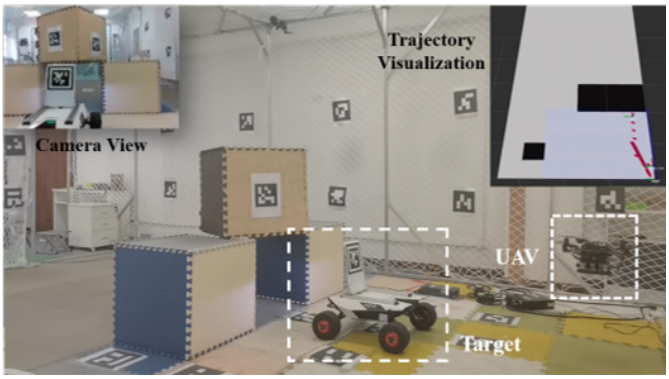

Fig. 1. Demonstration of the UAV and the indoor environment in the experiments. On the upper left corner is an image of the camera view, and on the right corner the visualization of the generated trajectory.

I. Introduction

Recently, unmanned aerial vehicles (UAVs) have drawn increasing interest of research in the field of robotics. UAV platforms equipped with visual sensors can perform a wide range of tasks in agriculture, surveillance, mapping, search and rescue [1]–[4]. Many of these tasks require UAVs to autonomously track a maneuvering target in complex environments where GNSS is unavailable. Tracking in these environments meets other challenges except for minimizing the tracking error between the UAV and the target. For the sake of safety, it is necessary for the UAV to plan feasible and collision-free motions under the velocity and acceleration constraints. The trajectory of the UAV should be smooth for stable flights and reduce camera blur. In addition, the onboard computation has to be done with limited time and resources.

To address these issues above, this work implements a vision-based system that enables a UAV to track a moving target while avoiding static obstacles. Our system detects the target and estimates the states of the target in the world frame using computer vision from the onboard camera. Based on the feedback obtained by the vision system, the autonomous tracking control is achieved by optimization-based motion planning techniques. We formulate the tracking problem as a trajectory optimization problem where the cost function jointly considers the tracking error, the control cost of the trajectory, and the trajectory length. We utilize the piecewise Bézier curve to represent the tracking trajectory. Due to the special property of Bézier curve, the motion constraints on safety and feasibility are directly imposed in the form of linear inequalities.

We implement the system on a UAV equipped with cameras and an onboard computer and perform experiments where the UAV attempts to follow a UGV as the moving target. Fig. 1 illustrates the real-world tracking experiment in the indoor environments.

[1]School of Artificial Intelligence, University of Chinese Academy of Sciences, 19 A Yuquan Road, Beijing 100049, China.
[2]Institute of Automation, Chinese Academy of Sciences, 95 Zhongguancun East Road, Beijing 100190, China.
Corresponding author: Ziyue Lin(linziyue2020@ia.ac.cn)

II. Related Work

A. Aerial Tracking

Autonomous UAV tracking of a moving target has been investigated during the last decades. Many research efforts are devoted to vision-based control methods, as cameras have become a popular option for UAVs. [5]–[7] have incorporated visual measurements in the control loop as feedback to compute the tracking error of a set of features available in the image. In [8], [9], vision-based methods are proposed to simultaneously control the poses of the UAV and the camera, which keeps the target in the center of the frame while performing tracking. However, in these cases, several constraints such as obstacle avoidance or dynamics limits are not taken into account. Coupling the visual processing system with the trajectory planning approach allows the UAV to reach its desired state while satisfying the constraints [10]. [11] proposes a trajectory planning method for real-time tracking of a moving target. The tracking trajectory is generated by

solving an optimization problem where the cost function embeds the tracking error and the control cost, and the motion constraints are described as linear inequalities. [12], [13] adopt multi-objective programming and use a weighted sum cost function for trajectory generation. [14] uses a receding horizon planner to autonomously record scenes with moving targets. The planner utilizes model predictive control (MPC) with non-linear constraints for generating trajectories from set-points defined in image space. [15] proposes an optimization-based method for tracking a target moving at an average speed of 1.3m/s. In the front end, the kinodynamic search is used to search for a safe trajectory, while in the back end the trajectory is optimized to be safe and feasible by a spatial-temporal planning method.

B. Visual-Inertial Odometry

Visual-inertial odometry (VIO) for state estimation is becoming increasingly prevalent in various applications due to its accuracy and robustness. Assisting visual odometry with inertial measurements, high-frequency measurements of acceleration and angular velocity from the IMU bridge the gap between sequences of measurements from visual sensors. VIO has shown more robustness in textureless scenes and in varying lighting conditions.

There are generally two trends of ways to fuse visual measurements with inertial measurements. The first is filter-based sensor fusion, generally achieved by the Kalman Filter-based methods [16], [17]. Based on the Kalman Filter, Mourikis and Roumeliotis propose the multi-state constraint filter [18] to impose constraints between multiple camera poses. OpenVINS [19] implements MSCKF in a modern way and presents an open-source codebase for the research of the VIO. The other approach is optimization-based sensor fusion, also known as Bundle Adjustment (BA), which jointly optimizes visual and inertial measurements maintained in a sliding window. PTAM [20] is one of the representative works based on parallel threads and sliding-window methods, which achieves real-time state estimation. ORB-SLAM3 [21] proposes a multiple map system that enables VIO to survive in long periods of textureless conditions. Qin proposes a tightly-coupled, monocular VIO based on BA with camera-IMU extrinsic calibration and IMU bias estimation [22]. Compared with the approaches of filter-based sensor fusion, the approaches of optimization-based sensor fusion have shown higher accuracy while suffering from more computational complexity.

III. Vision-Based Target Tracking

Fig. 2 outlines the overall architecture of the system. The target pose estimator detects the target from the vision system and obtains the filtered position and orientation of the target in the 3D space. Collecting observations from both the visual and inertial sensors, the state of the UAV itself is estimated by the visual-inertial odometry. After obtaining the target's state in the world frame, the trajectory planner generates a smooth and dynamically feasible trajectory with a pre-built map of the indoor environments. The high-level controller will forward

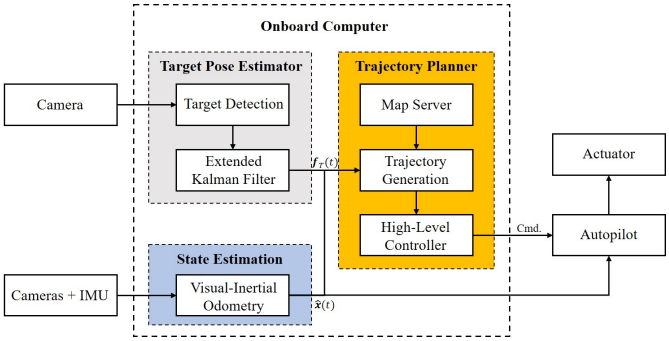

Fig. 2. The diagram of the system architecture.

the control command to the autopilot to track the generated trajectory.

A. Target State Estimation

We use a UGV equipped with the AprilTag as the moving target. The AprilTag visual fiducial [23], [24] mounted at the front of the UGV is used to identify and localize the target. The detection algorithm based on [24] extracts the information of the tag through visual measurement and detects the position and orientation of the target.

To estimate the state of the moving target from the visual detection, we use extended Kalman Filter (EKF). The state vector of the moving target is defined as $\mathbf{x}_T = [p_x, p_y, v_T, \theta, \dot{\theta}]^T$, where the p_x, p_y are the x, y position of the target, v_T the velocity of the target, θ is the angle between the forward direction of the target and x-axis, and $\dot{\theta}$ is the angular velocity. The state dynamics of the target is given as:

$$\begin{aligned} \dot{\mathbf{x}}_T(t) &= f_T(\mathbf{x}_T(t)) + \mathbf{w}(t) \\ \mathbf{z}_T(t) &= h_T(\mathbf{x}_T(t)) + \mathbf{v}(t) \end{aligned} \tag{1}$$

where $f_T(\cdot)$ is the system's dynamics function, $h_T(\cdot)$ the observation function, and $\mathbf{w}(t), \mathbf{v}(t)$ the noise process, which is modelled as zero-mean white Gaussian noise. The dynamics of the target $f_T(\mathbf{x}_T(t), \mathbf{u}(t))$ is represented as a constant velocity and angular velocity model:

$$\begin{aligned} \dot{p_x} &= v_T \cos(\theta), \dot{p_y} = v_T \sin(\theta) \\ \ddot{\theta} &= 0, \dot{v_T} = 0 \end{aligned} \tag{2}$$

The observation vector is defined as $\mathbf{z}_T = [p_x, p_y, \theta]^T$, which is obtained from the tag detection. The EKF will update the state estimate by applying the correction equation:

$$\hat{\dot{\mathbf{x}}}_T(t) = f_T(\hat{\mathbf{x}}_T(t)) + \mathbf{K}(t)(\mathbf{z}_T(t) - h_T(\hat{\mathbf{x}}_T(t))) \tag{3}$$

where $\mathbf{K}(t)$ is the Kalman gain.

B. Visual-Inertial Odometry

Equipped with stereo cameras and IMU, the quadrotors estimate the state using visual-inertial odometry. The states to be estimated involve the body position $\mathbf{p}_i \in \mathbb{R}^3$ and orientation $\mathbf{R}_i \in SO(3)$ in the world frame, the body velocity $\mathbf{v}_i$, the inverse depth of landmarks λ, and the acceleration bias and

gyroscope bias of the IMU, denoted by $\mathbf{b}_i^a$ and $\mathbf{b}_i^g$. The full state vector $\mathcal{X}_r$ is defined as:

$$\begin{aligned}\mathcal{X}_r &= [\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_n, \lambda_0, \lambda_1, \cdots, \lambda_m] \\ \mathbf{x}_i &= [\mathbf{p}_i, \mathbf{R}_i, \mathbf{v}_i, \mathbf{b}_i^g, \mathbf{b}_i^a], i \in [0, n]\end{aligned} \tag{4}$$

where n denotes the size of the sliding window, $\mathbf{x}_i$ denotes the body state vector, and m is the size of the landmarks observed in the sliding window. The inverse depth of landmark λ_j is registered to the maintained state when observed for the first time in the keyframe.

Given the measurements from the stereo cameras and IMU, the VIO problem is formulated as the following nonlinear least-square problem:

$$\begin{aligned}\min_{\mathcal{X}_r} \|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\|^2 + \sum_{i \in \mathcal{B}} \|\mathbf{r}_{I_{i-1,i}}\|^2_{\mathbf{\Sigma}_{I_{i-1,i}}} \\ + \sum_{j \in \mathcal{C}} \sum_{i \in \mathcal{K}^j} \rho(\|\mathbf{r}_{ij}\|_{\mathbf{\Sigma}_{ij}})\end{aligned} \tag{5}$$

where $\mathbf{r}_{I_{i-1,i}}$ is the inertial residual, $\mathbf{r}_{ij}$ denotes the visual residual, $\mathcal{B}$ is the set of keyframes maintained in the sliding window, $\mathcal{C}$ is the set of landmarks, and $\mathcal{K}^j$ denotes the set of keyframes observing the landmark j. The Mahalanobis norm is defined as $\|\mathbf{r}\|_{\mathbf{\Sigma}} = \mathbf{r}^T \mathbf{\Sigma}^{-1} \mathbf{r}$, where $\mathbf{\Sigma}_{I_{i-1,i}}$ and $\mathbf{\Sigma}_{ij}$ denote the covariance of inertial residual and visual residual respectively. $\{\mathbf{r}_p, \mathbf{H}_p\}$ is the prior information left by marginalization [25]. In addition, the Huber norm $\rho(\cdot)$is used for reducing the effects of outliers [26]. Gauss-Newton and Levenberg-Marquadt approaches can be utilized to solve the nonlinear least-squares problem. Now, the details of the inertial residual and visual residual are stated as follows.

Following the theory proposed in [22], [27], we pre-integrate consecutive IMU measurements between two keyframes and represent the relative position, orientation, and velocity of two keyframes as $\Delta\mathbf{p}_{i,i+1}$, $\Delta\mathbf{R}_{i,i+1}$ and $\Delta\mathbf{v}_{i,i+1}$. In addition, we take the IMU's acceleration bias residual $\Delta\mathbf{b}^a_{i,i+1}$ and gyroscope bias residual $\Delta\mathbf{b}^g_{i,i+1}$ into state estimation to achieve higher accuracy. Therefore, the inertial residual is formulated as:

$$\mathbf{r}_{I_{i,i+1}} = \begin{bmatrix} \mathbf{R}_i^T(\mathbf{p}_{i+1} - \mathbf{p}_i - \mathbf{v}_i \Delta t - \frac{1}{2}\mathbf{g}\Delta t^2) - \Delta\mathbf{p}_{i,i+1} \\ \mathrm{Log}(\Delta\mathbf{R}_{i,i+1}^T \mathbf{R}_i^T \mathbf{R}_{i+1}) \\ \mathbf{R}_i^T(\mathbf{v}_{i+1} - \mathbf{v}_i - \mathbf{g}\Delta t) - \Delta\mathbf{v}_{i,i+1} \\ \mathbf{b}^g_{i+1} - \mathbf{b}^g_i \\ \mathbf{b}^a_{i+1} - \mathbf{b}^a_i \end{bmatrix} \tag{6}$$

where $\mathrm{Log} : SO(3) \rightarrow \mathbb{R}^3$ converts rotation matrices into the corresponding rotation vectors, and $\mathbf{g}$ represents the gravity vector.

The reprojection error $\mathbf{r}_{ij}$, between the landmark $\mathbf{l}_j$ and the keyframe i observing $\mathbf{l}_j$ is formulated as:

$$\mathbf{r}_{ij} = \mathbf{z}_{ij} - \pi_c \left(\mathbf{T}_{BC}^{-1} \mathbf{T}_i^{-1} \mathbf{T}_h \mathbf{T}_{BC} \frac{1}{\lambda_j} \pi_c^{-1}(u, v) \right) \tag{7}$$

where $(u, v)^T$ is the observation of the landmark from i^{th} image, $\pi_c(\cdot)$ and $\pi_c^{-1}(\cdot)$ represent the projection and back-projection model of the camera, the pose of the target frame i is denoted as $\mathbf{T}_i$, and $\mathbf{T}_h$ the pose of the host frame, $\mathbf{T}_{BC}$, obtained from offline calibration, denotes the extrinsic transformation matrix from the camera frame to the body frame (IMU frame). The feature observation $\mathbf{z}_{ij}$ is obtained from camera measurement in the reference frame.

C. Flight Corridor Generation

Generating a trajectory for tracking a moving target consists of two phases: path searching at the front end and trajectory optimization at the back end. We use the A* algorithm to search for a discrete and collision-free path with a pre-built grid map of the environment. The nodes in the optimal path obtained from the A* algorithm provide waypoints to generate the flight corridors for further trajectory optimization [28], [29]. The safe flight corridor is comprised of a sequence of connected and convex cubes in the free space. To obtain the flight corridor, first we initialize the flight corridor as a series of nodes on the path searched by the A* algorithm. Then each node in the flight corridor is inflated into a cube by containing the free node in its neighbor on the axis-aligned direction. Each of these cubes will be inflated to its maximum size until it contacts with any obstacle in the map. The procedure of generating the flight corridor is demonstrated in Fig. 3.

(a) (b) (c)

Fig. 3. The procedure of generating a trajectory. The nodes on the path search by A* algorithm is marked red in 3(a). Then the safe flight corridor is obtained by inflation in 3(b) marked blue. The optimal trajectory is generated based on the safe flight corridor marked red in 3(c).

D. Trajectory Optimization

We leverage the differential flatness property to facilitate the trajectory generation for the underactuated quadrotors [30]. Due to this property, smooth trajectories such as polynomial functions can be used to specify trajectories in the space on $[x, y, z, \psi]$, where the x, y, z are 3D coordinates and ψ the yaw angle. In this paper, we parametrize the quadrotor trajetory as piecewise Bézier curves for 3D postition. In addition, since the control effort for the yaw angle is independent of the 3D position, we use a PD controller to control the yaw angle.

The Bernstein basis polynomials of degree n are defined as:

$$b_n^i(t) = \binom{n}{i} t^i (1-t)^{n-i}, i = 0, 1, \cdots, n \tag{8}$$

where $\binom{n}{i}$ is the binomial coefficient. For each dimension $l \in$

$\{x, y, z\}$, the trajectory is represented as:

$$f_l(t) = \begin{cases} s_1 \sum_{i=0}^{n} c_{i,1} b_n^i \left(\frac{t-T_0}{s_1}\right) & T_0 \le t \le T_1 \\ s_2 \sum_{i=0}^{n} c_{i,2} b_n^i \left(\frac{t-T_1}{s_2}\right) & T_1 \le t \le T_2 \\ \vdots & \vdots \\ s_M \sum_{i=0}^{n} c_{i,M} b_n^i \left(\frac{t-T_{M-1}}{s_M}\right) & T_{M-1} \le t \le T_M \end{cases} \tag{9}$$

where M is the number of segments of the trajectory, n is the order of the polynomial in each segment, c_{ij} is the i^{th} coefficient of the j^{th} trajectory, known as the control point, $T_1, \cdots, T_M$ are the end times of the segments, s_j is the time scale factor to map the j^{th} time interval from $[0, 1]$ to $[T_{j-1}, T_j]$. Now, let $\mathbf{c}_j = [c_{0j}, c_{1j}, \cdots, c_{nj}]^T$ and $\mathbf{b}_j(t) = \left[s_j b_n^0 \left(\frac{t-T_{j-1}}{s_j}\right), s_j b_n^1 \left(\frac{t-T_{j-1}}{s_j}\right), \cdots, s_j b_n^n \left(\frac{t-T_{j-1}}{s_j}\right)\right]^T$, the j^{th} segment of the trajectory in Equation (9) can be written as $f_j(t) = \mathbf{c}_j^T \mathbf{b}_j(t)$.

We formulate the tracking problem as trajectory optimization, which considers the objective of minimizing the tracking cost including the tracking error, the control cost and the trajectory length, together with the constraints on the safety and feasibility. The weighted sum cost function with weights λ_1, λ_2 is stated as:

$$\min_{\mathbf{f}(\cdot)} J = \frac{1}{2} \int_{T_0}^{T_M} \|\mathbf{f}^{(4)}(t)\|_2^2 + \lambda_1 \|\mathbf{f}(t) - \mathbf{f}_T(t)\|_2^2 + \lambda_2 \|\mathbf{f}'(t)\|_2^2 \mathrm{d}t \tag{10}$$

The cost function (10) is divided into three parts. The first term minimizes the integral of the square of the norm of the snap. The second term penalizes the position error between the quadrotor and the target. To simplify the optimization problem, we fit the trajectory of the target using the Bernstein basis polynomials, denoted by $f_T(t) = \mathbf{c}_0^T \mathbf{b}(t)$. Then the tracking error is defined as $\mathbf{e}(t) = \mathbf{f}(t) - \mathbf{f}_T(t)$. The third term minimizes the total length of the trajectory. The cost function (10) can now be written as:

$$J = \frac{1}{2} \sum_{x,y,z} \sum_{j=1}^{M} \mathbf{c}_j^T \left(\mathbf{Q}_j^{(4)} + \lambda_1 \mathbf{Q}_j^{(0)} + \lambda_2 \mathbf{Q}_j^{(1)}\right) \mathbf{c}_j - 2\lambda_1 \mathbf{c}_0^T \mathbf{Q}_j^{(0)} \mathbf{c}_j \tag{11}$$

$$\mathbf{Q}_j^{(r)} = \int_{T_{j-1}}^{T_j} \mathbf{b}_j^{(r)}(t) (\mathbf{b}_j^{(r)}(t))^T \mathrm{d}t, r = 0, 1, \cdots \tag{12}$$

Stacking the coefficient $\mathbf{c} = [\mathbf{c}_1^T, \cdots, \mathbf{c}_M^T]^T$ and $\mathbf{Q}^{(r)} = \mathrm{diag}\left\{\mathbf{Q}_1^{(r)}, \cdots, \mathbf{Q}_M^{(r)}\right\}$, we obtain the cost function in (11) as:

$$J = \sum_{x,y,z} \frac{1}{2} \mathbf{c}^T \mathbf{Q} \mathbf{c} + \mathbf{h}^T \mathbf{c} \tag{13}$$

$$\mathbf{Q} = \mathbf{Q}^{(4)} + \lambda_1 \mathbf{Q}^{(0)} + \lambda_2 \mathbf{Q}^{(1)}, \ \mathbf{h} = -\lambda_1 \mathbf{Q}^{(0)} \mathbf{c}_0 \tag{14}$$

Next, we enforce the constraints on safety and feasibility in the form of linear equalities and inequalities. The start and end state provide a boundary constraint on the trajectory as:

$$\mathbf{f}^{(r)}(T_0) = \mathbf{f}_0^{(r)}, \mathbf{f}^{(r)}(T_M) = \mathbf{f}_M^{(r)} \tag{15}$$

The trajectory should be smooth at all the n^{th} derivatives when the quadrotor passes through two segments, which is shown in (16):

$$\lim_{t \to T_j^-} \mathbf{f}^{(r)}(t) = \lim_{t \to T_j^+} \mathbf{f}^{(r)}(t) \tag{16}$$

In addition, the generated trajectory should be safe and collision-free. To achieve this, we utilize the convex hull property of Bézier curves that the entire curve is bounded within the convex hull of the control points. Therefore, the flight corridor described in III-C can be used to impose safety constraints. Since the flight corridor is comprised by a set of collision-free convex cubes, confining all the control points within the flight corridor will lead to the fact that the entire curve is bounded in the flight corridor, thus avoiding any obstacle. So we obtain:

$$\mathbf{c}_j^- \le \mathbf{c}_j \le \mathbf{c}_j^+ \tag{17}$$

where $\mathbf{c}_j^-, \mathbf{c}_j^+$ is the lower and upper bound of the cube in the flight corridor for the j^{th} segment.

For the motion planning to be dynamically feasible, the maximum velocity and acceleration of the quadrotor should be bounded, which is demonstrated as:

$$\begin{aligned} \mathbf{v}_{min} \le \mathbf{f}^{(1)}(t) \le \mathbf{v}_{max} \\ \mathbf{a}_{min} \le \mathbf{f}^{(2)}(t) \le \mathbf{a}_{max} \end{aligned} \tag{18}$$

Equation (15), (16) can be rearranged as linear equality constraints for the coefficient $\mathbf{c}$, denoted by $\mathbf{A}_{eq}\mathbf{c} = \mathbf{b}_{eq}$, while equation (17), (18) as linear inequality constraints, denoted by $\mathbf{A}_{ieq}\mathbf{c} \le \mathbf{b}_{ieq}$. Therefore, the full optimization problem for tracking trajectory generation is formulated by combining these linear constraints with cost function (13):

$$\begin{aligned} \min_{\mathbf{c}} J = \sum_{x,y,z} \frac{1}{2} \mathbf{c}^T \mathbf{Q} \mathbf{c} + \mathbf{h}^T \mathbf{c} \\ \text{s.t.} \quad \mathbf{A}_{eq}\mathbf{c} = \mathbf{b}_{eq} \\ \mathbf{A}_{ieq}\mathbf{c} \le \mathbf{b}_{ieq} \end{aligned} \tag{19}$$

which is a Quadratic Programming (QP) problem, and can be solved by solvers such as [31].

IV. Simulation

Our simulation demonstrates the effect of adjusting the weights λ_1, λ_2 in cost function (10) on the optimal trajectory. The simulation results are illustrated in Fig. 4. The values of the weights range from 10^{-4} to 2.

The simulation results shows that tuning the weights λ_1, λ_2 in cost function (10) accordingly changes the performance of the generated trajectory. From Fig. 4 (a), (b) and (c), increasing the parameter λ_2 tends to reduce the length of the trajectory. From Fig. 4 (a), (d), (e) and (f), the generated trajectory tends to converge faster and shift closer to the target if the parameter λ_1 increases. However, all these trajectories are still smooth and collision-free when the weights are adjusted.

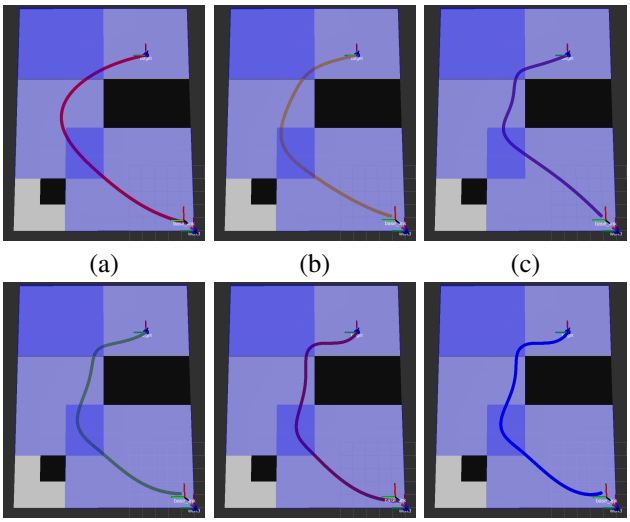

Fig. 4. Different generated trajectories marked with different colors when adjusting the weights λ_1, λ_2. The UAV is marked with the text "base link" on the right bottom corner, and the UGV is marked with "target". The flight corridor is marked as a series of blue cubes. The values of the weights are: (a)$\lambda_1 = 0, \lambda_2 = 0$, (b)$\lambda_1 = 10^{-4}, \lambda_2 = 10^{-2}$, (c) $\lambda_1 = 10^{-4}, \lambda_2 = 1$, (d)$\lambda_1 = 10^{-3}, \lambda_2 = 10^{-4}$, (e) $\lambda_1 = 10^{-1}, \lambda_2 = 10^{-2}$, (f)$\lambda_1 = 1, \lambda_2 = 2$.

V. Experiments

A. Experiment setup

To verify the effectiveness of the system, we implement the localization and tracking algorithms on the UAV, illustrated in Fig. 5. The UAV is equipped with an onboard computing device, Intel NUC (CPU: i5-1135, 16GB RAM). Stereo cameras and IMU (Intel RealSense D435i) are rigidly mounted to the front of the UAV, publishing images at 30 Hz and inertial measurements at 200 Hz, respectively. A RGB camera is used for detecting the target UGV with a tag of the size of 0.218m. We use the PX4 flight controller for stabilizing the UAV.

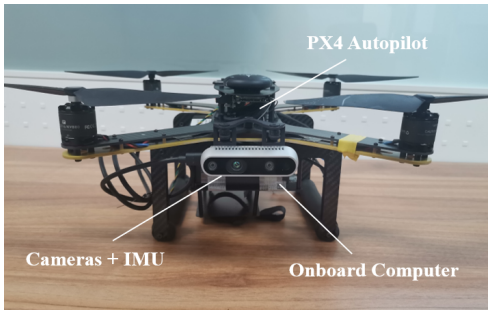

Fig. 5. Hardware of the UAV platform.

B. Indoor Experiments Without Obstacles

First we validate our approach in indoor environments without obstacles, as illustrated in Fig. 6. In this experiment, the UAV starts at $\mathbf{x} = (0.1, 0.0, 0.5)^T$, while the target UGV starts moving at $\mathbf{x}_T = (1.5, 0.5, 0.2)^T$. The target UGV moves at an average speed of 0.17 m/s.

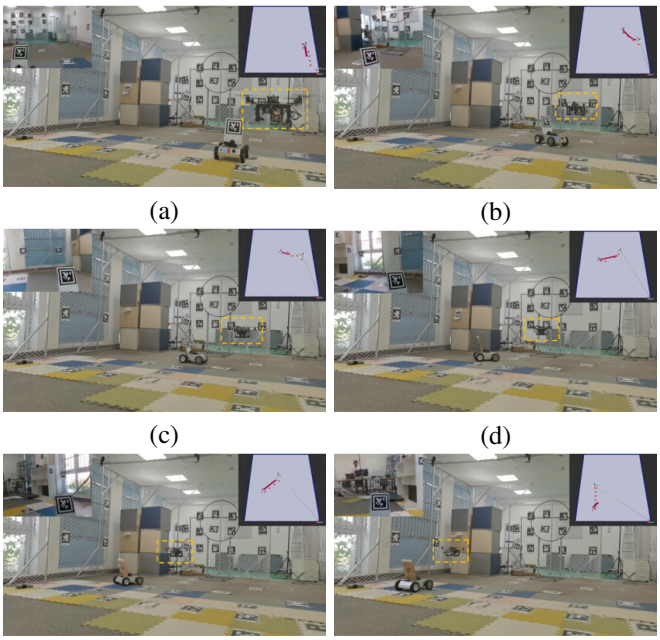

Fig. 6. Experiments in indoor environments without obstacles. The camera view is shown on the left upper corner, and the visualization of the generated trajectory is on the right upper corner. The UAV is marked with an orange bounding box.

Fig. 7 illustrates the 3D trajectory of the UAV and the UGV and the tracking error $\|\mathrm{e}_{xy}(t)\|_2$. We observe that the UAV can follow the trajectory of the target UGV while planning smooth trajectories. The vision observations are filtered to reduce noise from the environment, and obtain more accurate prediction of the target trajectory. The UAV can re-plan feasible trajectories to reduce the tracking error between the target, and the average tracking error is $1.03m$ during the expriment.

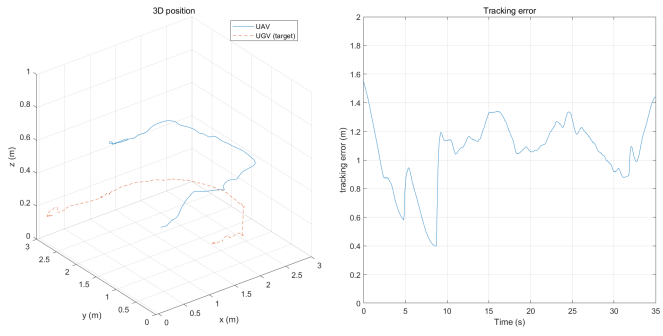

Fig. 7. 3D trajectories of the UAV and the UGV as the target and the tracking error between them. (In section V-B)

C. Indoor Experiments With Obstacles

We also perform experiments in indoor environments with static obstacles using a pre-built grid map, as shown in Fig. 8. The traget UGV passes through a hole in the obstacle area, forcing the UAV to perform tracking while avoiding the

obstacles. The 3D trajectory of the UAV and the UGV and the tracking error are illustrated in Fig. 9.

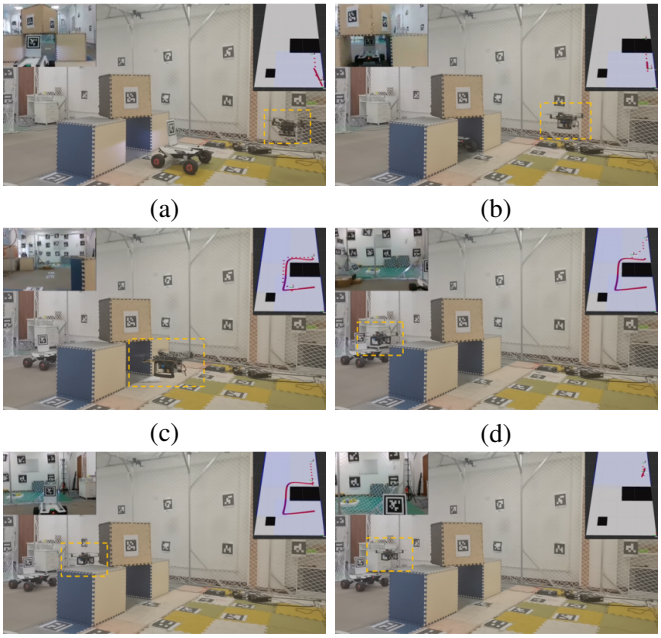

Fig. 8. Experiments in indoor environments with static obstacles. The target UGV passes through a hole inside the obstacles, forcing the UAV to perform tracking while avoiding the obstacles. The UAV keeps tracking when the target is regained in the camera view.

Fig. 10 plots the setpoints and the estimated position of the UAV. We observe that at $t = 7.9s$, as the target UGV passes through the hole inside the obstacles, the UAV re-generated its trajectory due to the safe constraints. The target UGV is regained in the camera view when the UAV arrives at the other side of the hole at $t = 29.0s$. Then the UAV generates a new trajectory to tracking the target at $t = 33.8s$. Fig. 10 shows that the smooth and feasible motions are planned by the trajectory generator, but there exists time delay from generating the desired state to reaching the state, which is possibly caused by the response time of the actuators. These results show that the UAV can perform the tracking task while satisfying the safety and feasibility constraints.

VI. Conclusion

In this paper, we implement a vision-based system of quadrotors for tracking a moving target with visual-inertial localization. The localization of the UAV is obtained by fusing visual and inertial measurements. The target tracking problem is achieved by optimization-based trajectory generation. The experiments show that our UAV has successfully finished the tasks of tracking a moving target while avoiding static obstacles.

For the future work, we will focus on sensing and avoiding dynamic obstacles for the UAV to autonomously navigate in unknown environments. Additionally, we plan to apply object detection algorithms for target perception without visual fiducial markers.

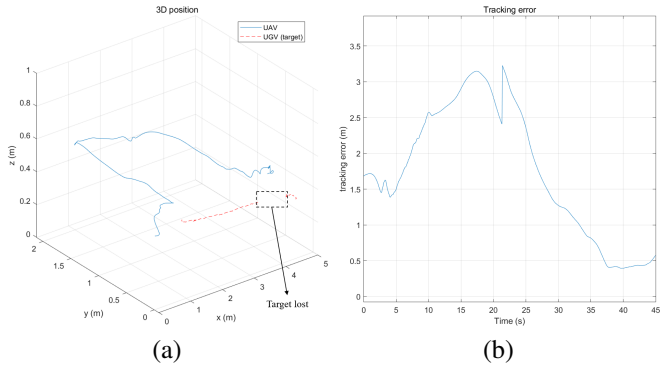

Fig. 9. 3D trajectories of the UAV and the UGV as the target and the tracking error between them. (In section V-C)

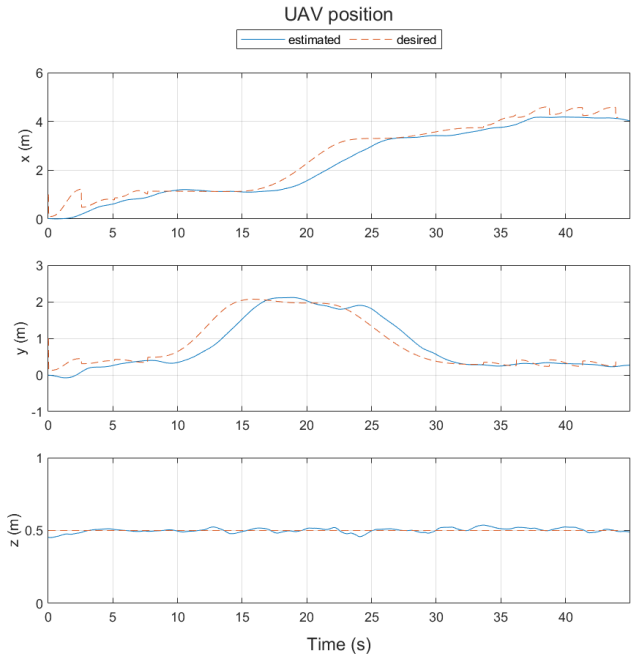

Fig. 10. The desired and estimated position of the UAV. The desired position is computed from the generated trajectory while the estimated position is obtained from VIO.

References

[1] X. Liu, S. W. Chen, S. Aditya, N. Sivakumar, S. Dcunha, C. Qu, C. J. Taylor, J. Das, and V. Kumar, "Robust fruit counting: Combining deep learning, tracking, and structure from motion," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1045–1052, 2018.

[2] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[3] J. He, Y. Zhou, L. Huang, Y. Kong, and H. Cheng, "Ground and aerial collaborative mapping in urban environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 95–102, 2020.

[4] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, *et al.*, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022.

[5] C. Teuliere, L. Eck, and E. Marchand, "Chasing a moving target from a flying UAV," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4929–4934, 2011.

[6] S. Azrad, F. Kendoul, and K. Nonami, "Visual servoing of quadrotor micro-air vehicle using color-based tracking algorithm," *Journal of System Design and Dynamics*, vol. 4, no. 2, pp. 255–268, 2010.

[7] J. Kim and D. H. Shim, "A vision-based target tracking control system of a quadrotor by using a tablet computer," in *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1165–1172, 2013.

[8] F. Lin, X. Dong, B. M. Chen, K.-Y. Lum, and T. H. Lee, "A robust real-time embedded vision system on an unmanned rotorcraft for ground target following," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 2, pp. 1038–1049, 2012.

[9] H. Cheng, L. Lin, Z. Zheng, Y. Guan, and Z. Liu, "An autonomous vision-based target tracking system for rotorcraft unmanned aerial vehicles," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1732–1738, 2017.

[10] Y. Mezouar and F. Chaumette, "Path planning for robust image-based control," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 4, pp. 534–549, 2002.

[11] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 446–453, 2016.

[12] J. Thomas, J. Welde, G. Loianno, K. Daniilidis, and V. Kumar, "Autonomous flight for detection, localization, and tracking of moving targets with a small quadrotor," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1762–1769, 2017.

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

[14] T. Nägeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges, "Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1696–1703, 2017.

[15] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, "Fast-tracker: A robust aerial system for tracking agile target in cluttered environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 328–334, 2021.

[16] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 298–304, 2015.

[17] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4974–4981, 2014.

[18] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, 2007.

[19] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4666–4672, 2020.

[20] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 225–234, 2007.

[21] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[22] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[23] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3400–3407, 2011.

[24] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4193–4198, 2016.

[25] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.

[26] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in Statistics*, pp. 492–518, Springer, 1992.

[27] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[28] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1476–1483, 2016.

[29] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[30] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2520–2525, 2011.

[31] E. M. Gertz and S. J. Wright, "Object-oriented software for quadratic programming," *ACM Transactions on Mathematical Software (TOMS)*, vol. 29, no. 1, pp. 58–81, 2003.