Toward Autonomy of Micro Aerial Vehicles in Unknown and Global Positioning System Denied Environments

Yu Zhou, Shupeng Lai, Huimin Cheng, Abdul Hamid Mohamed Redhwan, Pengfei Wang, Junji Zhu, Zhi Gao, Zhengtian Ma, Yingcai Bi, Feng Lin, and Ben M. Chen, *Fellow, IEEE*

***Abstract*—In this article, we present a comprehensive design and implementation for a micro aerial vehicle (MAV) that is able to perform 3-D autonomous navigation and obstacle avoidance in cluttered and realistic unknown environments without the aid of global positioning system and other external sensors or markers. To achieve these autonomous missions, modularized components are developed for the MAV, including visual–inertial odometry, 3-D occupancy mapping, and motion planning. The proposed system is implemented to run on a small embedded computer in real time. It is demonstrated to be robust in both simulation and real flight experiments. The demonstration video is available at: https://youtu.be/KUKzsnORm-4.**

***Index Terms*—Micro aerial vehicles (MAVs), motion planning, trajectory generation, three-dimensional (3-D) mapping, visual–inertial odometry (VIO).**

I. Introduction

IN RECENT years, micro aerial vehicles (MAVs) have been used in various fields such as agriculture, industrial inspection, civilian disaster management, surveillance, and so on [1]–[3]. Among those applications, one critical capability for an MAV is to navigate in challenging scenarios, especially unknown, global positioning system (GPS) denied and obstacle-cluttered environments. In many applications, it is difficult to deploy external sensors such as radio-frequency identification [4], ultra-wideband [5], or fiducial marker systems [6], and it is necessary to develop fully autonomous navigation systems that rely only on onboard sensing and computation.

Stereo camera-based methods for state estimation in consideration of both efficiency and performance have gained popularity recently. However, with only the visual information, the state may drift significantly especially during aggressive maneuvers. Therefore, fusion with an inertial sensor is necessary to increase its applicability and robustness.

With accurate state estimation, the next step is to build a 3-D map of surrounding obstacles. To this end, a 3-D reconstruction method is proposed, which gives strong support to the motion planning module. This article presents a highly efficient approach to update the Euclidean distance field (EDF) on a 3-D occupancy grid map. The occupancy map is updated with the voxel projection method [7] and EDF is accelerated with dimensionality-reduction Voronoi diagram.

To concatenate with the state estimation and mapping modules, a 3-D motion planning module is designed for real-time path and trajectory generation, where a collision-free and dynamically feasible reference is generated to guide the vehicle to maneuver safely toward its desired target. We represent the trajectory with its boundary state conditions and call it boundary state constrained primitives (BSCPs). The BSCPs are usually achieved by solving a boundary value problem numerically. For real-time responsiveness, we use a neural network (NN) to approximate the BSCP during online optimization and then combine it with particle swarm optimization (PSO) to solve the nonconvex local motion planning problem. NN is used instead of a lookup table because the latter could become prohibitively large for high-dimensional systems [8].

The contributions of this article are listed as follows.

1) The state estimation based on visual–inertial sensor fusion is designed to work with off-board camera-integrated visual odometry data, minimizing the computational complexity. A lightweight velocity-based fault detection is demonstrated to withstand challenging feature-sparse environments.

Manuscript received October 10, 2019; revised January 26, 2020 and May 18, 2020; accepted June 28, 2020. Date of publication July 17, 2020; date of current version April 27, 2021. *(Corresponding author: Shupeng Lai.)*

Yu Zhou, Huimin Cheng, Abdul Hamid Mohamed Redhwan, Pengfei Wang, Junji Zhu, Zhengtian Ma, Yingcai Bi, and Feng Lin are with the Temasek Laboratories, National University of Singapore, Singapore 117411 (e-mail: tslzyu@nus.edu.sg; chenghuimin@nus.edu.sg; tslmor@nus.edu.sg; tslwpf@nus.edu.sg; zhujunji93@hotmail.com; zhengtian@nus.edu.sg; yingcaibi@u.nus.edu; linfeng@nus.edu.sg).

Shupeng Lai is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: eleshla@nus.edu.sg).

Zhi Gao is with the Photogrammetry Department, School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China (e-mail: gaozhinus@gmail.com).

Ben M. Chen is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong, and also with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: bmchen@cuhk.edu.hk).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

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/TIE.2020.3008378

2) We propose to solve the nonconvex local motion planning problem with a gradient-free PSO algorithm. The approach is made real time by using the NN approximated BSCP. With this method, discontinuous and nonconvex problems, such as navigation with a limited field of view, can be solved easily.
3) We implement the proposed approach on a real quadrotor which is capable of navigating in 3-D complex environments and tracking moving references.

This article is organized as follows. We discuss related works in Section II. An overview of our system is given in Section III. Section IV describes the state estimation algorithms. Section V introduces our mapping and motion planning modules. Various experimental results are given in Section VI. Finally, Section VII concludes the article, with discussions on the results and directions of future work that would further improve the present system.

II. Related Works

A. Stereo Visual–Inertial State Estimation

Many different stereo visual–inertial odometry (VIO) implementations have been integrated into MAV platforms and could be broadly classified into four approaches: Tightly coupled optimization [9], tightly coupled filtering [10], loosely coupled filtering [11], and camera-integrated implementations [12].

Optimization approach jointly optimizes all states and measurement. Monocular visual-inertial systems (VINS-Mono) (VINS-Fusion) [9] has achieved real-time performance comparable to filter-based methods, by imposing sliding window and limiting the optimization thread to around 10 Hz. Other implementations such as open keyframe-based visual-inertial SLAM (OKVIS) [13] exist but are generally slower and less suitable to implement on real MAV systems. Tightly coupled filtering approach formulates the estimation problem into a recursive prediction and update process. Multi-state constraint kalman filter (MSCKF) [10] augments the states with a sliding window of camera poses and robust visual inertial odometry (ROVIO) [14] augments the states with visual landmarks. However, tightly coupled methods generally take more effort considering convergency in practice, which makes the tuning more complicated. Loosely coupled filtering treats the whole visual odometry process as a black box and performs Kalman filter-based state estimation based on visual odometry (VO) output and inertial measurement unit (IMU) measurements. In [11], a framework of semi-direct visual odometry (SVO) [15] with unscented kalman filter (UKF) is demonstrated. However, the pose output from visual odometry is treated as the global measurement in UKF; therefore, any drift and failure of the visual odometry process would result in inconsistent filtered state estimation. In multi-sensor fusion (MSF) extended Kalman filter (EKF) framework [16], the drift of the vision pose sensor is explicitly modeled into the states. However, the fully probabilistic formulation regarding drift and jumps relies on the careful tuning of the system and measurement covariance parameters, which may not be intuitive on a real MAV system. A good practice for real application is to use visual odometry as a velocity sensor, instead of an absolute or relative pose sensor.

B. 3-D Mapping

Different map representations can be used for planning. Occupancy map is a typical representation. One of the most popular 3-D occupancy grid maps is Octomap [17], which adopts a hierarchical octree structure to save occupancy probabilities. However, occupancy data alone is inadequate for optimization-based planners, which requires distances to nearest obstacles. It is usually obtained by building an EDF. Previous methods like the one in [18] incrementally build Euclidean signed distance field directly out of truncated signed distance field, which is a surface representation [19] using projective distance and compute these distances within a short truncation radius around the surface. These previous methods [18], [20] are mainly central processing unit (CPU) based as most commercial platforms lack a programmable parallel computing device. These works are largely serialized and rely heavily on thread-based synchronization when parallelized. The synchronization steps limit their efficiency as parallel algorithms [21]. In our work, we explore to perform synchronization-free dense mapping on the MAV with the graphics processing unit (GPU)-enabled TX2 computer.

C. Motion Planning

The problem of motion planning among obstacles is nonconvex by nature. Some previous approaches [22], [23] tried to convexify it by limiting the trajectory to a set of convex subspaces. However, the reduction of the solution space could affect the optimality or even the existence of the solution. Other methods [24] try to solve the nonconvex problem with gradient-based methods directly. However, the quality of the solution relies heavily on the initial guess. Reinitialization is needed from time to time which slows down the overall algorithm. In this article, we take a different approach by representing the trajectory as BSCPs, which can be constructed offline without being limited by solution efficiency [25], motion constraints [26], [27], and optimization targets [28], [29].

The capability of solving nonconvex problems reliably in real time is also suitable for problems that cannot be made convex easily or naturally noncontinuous problems such as safe navigation with limited field of view (FOV). Many previous methods often ignore the FOV limitation [22], [23]. The method in [30] considers this limitation and takes a uniformly random sampling approach, which is proven to be inferior to a proper optimization approach [31].

III. System Overview

A. Hardware Design

The platform has to be as compact as possible while retaining a high thrust-to-weight ratio for sufficient control authority. This led to selecting an off-the-shelf octo-rotor MAV in a coaxial configuration as the research platform which would help increase thrust available in relation to the size of the platform for our attempt at navigating through tight and narrow spaces. The Pixhawk autopilot was chosen for implementing our previous work on the model-based controller [32]. The Nvidia Jetson TX2 was favored as the onboard computer as it is a future-ready processor with its 256 core GPU processing capabilities which

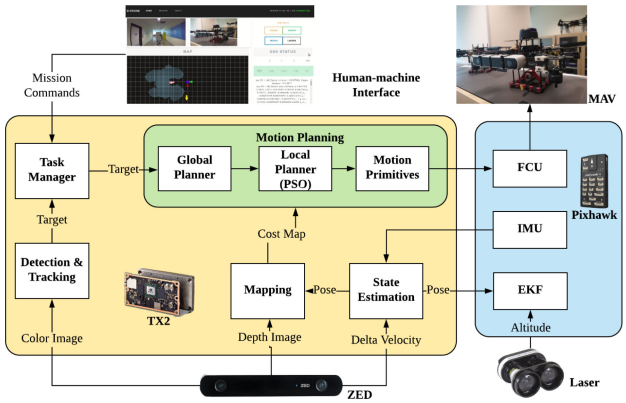


Fig. 1. Block diagram of the software system architecture, showing key components and data flow between different modules.

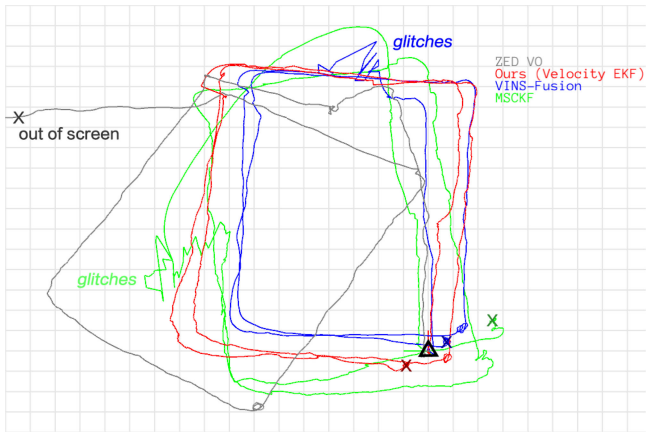


Fig. 2. VIO reliability evaluation in challenging 13.5-by-10 m environments, where the triangle is the starting position, and respective crosses are the estimated loops' end points: Original sensor VO output in gray, our state estimation in red, and VINS and MSCKF frameworks in blue and green, respectively; significant glitches are observed in the latter framework, while VINS, the optimization-based method, is more prone to divergence.

is a good foundation to build upon for optimizing and improving vision-based systems. A ZED stereo camera and a lightweight laser rangefinder were installed to provide information for positioning and navigation.

B. Software Architecture

Fig. 1 shows the high-level block diagram of our system. The modules can be categorized into the following main parts: State estimation, mapping and motion planning, and flight controller unit. Sensor data consists of velocity from the camera, and acceleration and angular velocity from IMU are fused into the error-state (ES) EKF to get state estimation result, which will be fed into the EKF on Pixhawk with height measurement from laser to further improve the result. The depth image is used to generate an occupancy grid map which will be transferred to the EDF as a cost map. Mission targets from either user interface (UI) selection or the target tracking results and the cost map are given to the motion planner to create a valid and obstacle-free trajectory to control the unmanned aerial vehicle (UAV).

IV. State Estimation

Treating visual odometry as a velocity sensor decouples it from the global pose drift inherited in visual odometry. The velocity is obtained by dividing the frame-to-frame pose change by the time lapse between the two frames. Although an optical flow-based velocity estimator would be more computationally efficient than a full-fledged visual odometry estimator, most commercial stereo cameras provide such visual odometry estimator off-the-shelf, which makes visual odometry information readily available.

The sensor fusion algorithm implements an ES-EKF with time delay compensation, adapted from [33]. Reformulation of the filter model has been done to handle body-frame velocity measurements, which makes fault detection straightforward. The motivation is that the failure cases of visual odometry algorithms could hardly be avoided in real missions. Common causes of failure include: Lack of features in the camera's field of view, fast changes in lighting conditions, low texture, high-frequency vibration, and motion blur. Our state estimation system is designed to detect such failures and prevent those erroneous measurements from polluting the state estimation process.

The comparisons with other VIO frameworks are shown in Fig. 2. The proposed approach is not drift-free as the position is in theory not observable from the velocity measurement. The system tolerates drift over time by utilizing only the local map, which is not affected by the drift. The focus of work in this system is to handle the fault in the position estimation under unfavorable vision conditions.

1) *Velocity-Based Visual Odometry and Fault Detection*: For each VO measurement, two fault detection criteria are applied. First, a large discrepancy between the state velocity and the measured velocity would indicate a fault, subject to a scaled (s) sum of standard deviation (σ) of both the measurement and the state. Second, a large difference between the IMU's measured rotation vector and the measured rotation would indicate a fault, subject to a scaled constant. The scale factor s is tunable to obtain an optimal rejection, and c is a constant which is empirically set to 0.1

$$\left|v_{VO} - v_w^i\right| > s * (\sigma_{\text{VO}} + \sigma_{v_w^i}), \tag{1}$$

$$\left|\omega_{\text{VO}} - \omega_w^i\right| > s * c. \tag{2}$$

2) *ES-EKF System States:* The system states include IMU body pose and velocity, IMU biases, VO scale factor, and extrinsic calibrations between sensors (camera-IMU), represented as follows:

$$x = \{p_w^i, v_w^i, q_w^i, b_w, b_a, \lambda, q_i^c, p_i^c\} \in \mathbb{R}^{24\times 1}. \tag{3}$$

The notation p_w^i indicates IMU position in world frame, and similarly for the rest. To improve numerical stability and reduce quaternion representation to its minimum (3 number instead of 4), we represent the states in its error domain [33].

3) *ES-EKF Measurement Model:* The measurement model assumes the visual odometry system to be a body-velocity

observer, decoupling the visual odometry from its past states, which might be corrupted if there are any tracking failures within the visual odometry algorithm. We propose the body-velocity observer's measurement as

$$z = \begin{bmatrix} z_v & z_q \end{bmatrix}^T \in \mathbb{R}^{7\times 1}. \tag{4}$$

The velocity measurement z_v could be obtained from full states, by assuming displacement between IMU and camera mounting negligible

$$z_v = \lambda R^T_{q^c_i} R^T_{q^i_w} v^c_w \approx \lambda R^T_{q^c_i} R^T_{q^i_w} v^i_w. \tag{5}$$

If the displacement is not negligible (camera is mounted far away from IMU sensor), then the expression would become

$$z_v = v^c_c = v^i_c - \omega_c \times p^i_c, \tag{6}$$

$$z_v = \lambda [w_c]_\times R^T_{(q^c_i)} p^c_i + \lambda R^T_{(q^c_i)} R^T_{(q^i_w)} v^i_w. \tag{7}$$

The optional attitude measurement from IMU could be directly obtained from the full states: $z_q = q^i_w$.

4) *Time Delay Compensation:* Since velocity measurement is highly dynamic on a MAV platform, synchronization of measurement and state is required at the time of sensor fusion update step. Our implementation timestamps the velocity to be the mid-time between the two consecutive frames and performs a buffer search for the closest matching state vector, before the actual update step calculation.

V. Mapping and Planning

A. Mapping

To construct an environmental map that is later used for online planning, the mapping process can be categorized into two steps: First, construct a global occupancy grid (OG) map; second, perform Euclidean distance transform (EDT). The result is a grid-based map structure that can be used to query the safety of a given trajectory.

1) Occupancy Grid Map: Each grid in the OG map contains an unsigned 8-b integer which describes the probability that the grid is occupied by an obstacle. In the first step, we update the OG map using the latest sensor pose $\mathcal{P}$ and the captured depth map $\mathcal{D}$. Let $\mathcal{D}_{i,j,w}$ denote the pixel in column i and row j of $\mathcal{D}$ and has depth w, i.e.,

$$\mathcal{G}_{x,y,z} = \mathcal{M}(\mathcal{D}_{i,g,w}, \mathcal{P}) \tag{8}$$

with $\mathcal{G}_{x,y,z}$ being the projected point in the global frame. Traditional methods then perform a ray-cast from the current vehicle's position to $\mathcal{G}_{x,y,z}$. For each grid that is covered by the ray, its occupancy probability is updated based on its distance to $\mathcal{G}_{x,y,z}$. The details can be found in [18]. It is noticed that multiple rays might pass through the same grid and update its occupancy probability with different values. Therefore, a synchronization mechanism is needed during parallel implementation. In our approach, to avoid this problem, a backward projection similar to [34] is used

$$\mathcal{D}_{i,j,\bar{w}} = \mathcal{M}^{-1}(\mathcal{G}_{x,y,z}, \mathcal{P}). \tag{9}$$

Here, $\mathcal{G}_{x,y,z}$ is taken as the center position of the grid $\mathcal{R}_{k_x,k_y,k_z}$, and the output $\mathcal{D}_{i,j,\bar{w}}$ is its corresponding depth pixel in the image frame. We then compare $\mathcal{D}_{i,j,\bar{w}}$ to the true measurement $\mathcal{D}_{i,j,w}$. There are four different cases.

1) $\mathcal{D}_{i,j,\bar{w}}$ is outside of the depth image, which means we receive no measurement on location $\mathcal{G}_{x,y,z}$. Therefore, the occupancy probability in grid $\mathcal{R}_{k_x,k_y,k_z}$ shall not be updated.
2) $\bar{w}$ is larger than the measurement w, which means $\mathcal{G}_{x,y,z}$ locates behind a seen obstacle. Therefore, the occupancy probability in grid $\mathcal{R}_{k_x,k_y,k_z}$ shall not be updated.
3) $\bar{w}$ is smaller than the measurement w, which means $\mathcal{G}_{x,y,z}$ locates in front of a seen obstacle and shall be obstacle free. Therefore, the occupancy probability in grid $\mathcal{R}_{k_x,k_y,k_z}$ shall be reduced.
4) $\bar{w}$ is very close to the measurement w, which means $\mathcal{G}_{x,y,z}$ is occupied by an obstacle. Therefore, the occupancy probability in grid $\mathcal{R}_{k_x,k_y,k_z}$ shall be increased.

In this manner, the occupancy probability of each grid needs to be updated only once, and thread synchronization is no longer needed during parallel implementation.

2) Euclidean Distance Transform: In order to determine the safety of a trajectory, we would like to measure its distance to the closest obstacle. This is achieved by measuring the distance to the closest obstacle for each grid that the trajectory passes through, and such distance can be acquired through the EDT process. Previous work calculates EDT by simulating a wave propagation. It is implemented based on a priority queue and requires per-thread synchronization while performing modification on the queue and its data. In this article, we use the algorithm in [35] to construct EDT by performing 1-D scanning in the interested cuboid region. It first scans along the x-axis to determine each grid's distance to its closest obstacle on the x-axis. The second scan is along the y-axis, and it determines each grid's distance to its closest obstacle on the x–y plane. The final scan is along the z-axis, which calculates the Euclidean distance to the closest obstacle for each grid. Since the computation on each row of the x-axis (or y, z) is independent of the computation of other rows of the x-axis (or y, z), the algorithm is naturally parallelizable on GPU. More details on the algorithm can be found in [35] and omitted in this article due to the page limit.

We implement the algorithm on a GPU with a total of N voxels and p processors. The time complexity of the algorithm is $O(N/p)$. In practice with the onboard TX2 GPU, the EDT over $20 \times 20 \times 6$ m volume with 0.2-m accuracy can be performed at around 10 ms. The real flight indoor testing result is shown in Fig. 3. The comparison of computation efficiency between our method and Voxblox [18] with the same voxel size is shown in Fig. 4.

B. Motion Planning

The local planning is done in a receding horizon way to guide the vehicle to maneuver safely in an unknown environment. Being a local planner, the presented method requires either a higher level global planner (using algorithms like A*) or a human operator to provide it with local targets.

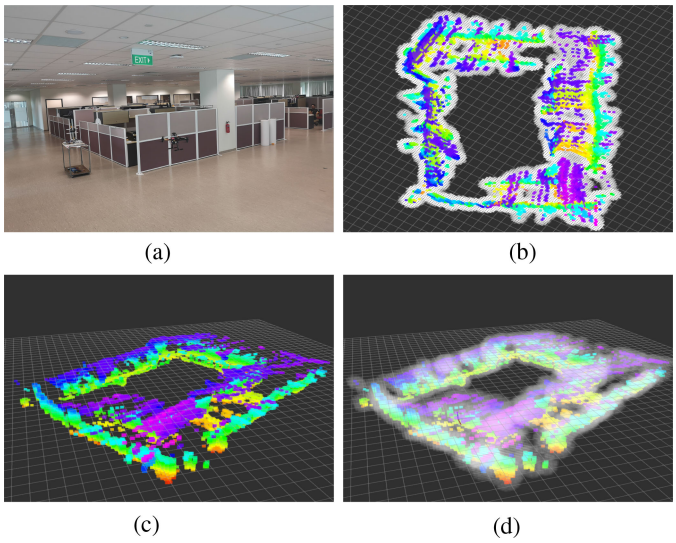


Fig. 3. Map in real indoor flight testing, where the color voxels denote OG map and white voxels are the EDT map. (a) Our indoor 30 m × 18 m testing site. (a) Testing environment. (b) Overhead view. (c) Side view. (d) Side view with EDT.

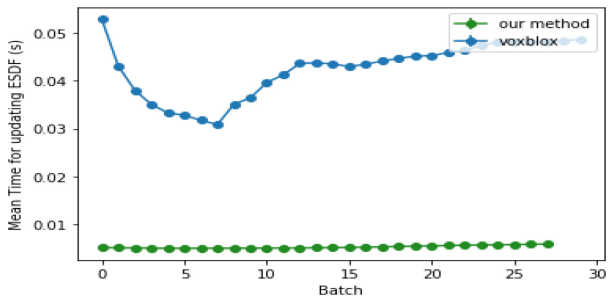


Fig. 4. Mapping runtime comparison between our method and Voxblox. As shown, our method is almost ten times faster than the other one. Batch means the data is sampled in a sequence of a certain time.

1) Problem Description: The local motion can be written as a model predictive control (MPC) problem that finds the state and input to minimize the cost and subject to the system dynamics and certain constraints

$$\begin{aligned} \min_{u(t)} J &= c_f(\mathbf{x}(t_0+T)) + \int_{t_0}^{t_0+T} c(\mathbf{x}(t),\mathbf{u}(t)), \\ \text{s.t. } \dot{x} &= f(\mathbf{x},\mathbf{u}), \\ \mathbf{x}(t_0) &= \mathbf{x}_0, \\ h(\mathbf{x},\mathbf{u}) &= 0, \\ \mathbf{x} &\notin \mathcal{O} \end{aligned} \tag{10}$$

where $\mathbf{x}(t), u(t)$ represents the state and input. c_f and c are terminal and running costs. $\mathbf{x}_0$ is the initial condition. $h(\mathbf{x}, u)$ denotes the invariant constraint. $\mathcal{O}$ denotes the environment-dependent obstacle constraint.

However, searching for valid trajectories directly over maps renders the optimization problem nonconvex and even discontinuous. In this article, we use BSCPs to reformulate the problem. The invariant constraints such as the vehicle dynamics and input limitations are separated from the variant ones such as the obstacles and handled offline during BSCP construction. The online local motion planning then becomes the process of selecting the best BSCP, given the environmental constraints. To handle the nonconvexity and discontinuity, the selection is done through the PSO.

The boundary value problem can be constructed as

$$\min_{\mathbf{x}(t),\mathbf{u}(t),t_f} = G(\mathbf{x}(t), u(t), t_f) \tag{11}$$

subject to all the nonenvironmental-dependent constraints in (10) and an end state constraint $\|g(\mathbf{x}(t,\theta)\| < \epsilon, \forall t > t_f$, where ϵ has a small positive value. G is the cost function and t_f and θ denote the final time and parameters of function g, respectively.

The boundary value problem can be solved through a wide range of approaches. In this article, we choose to design a nonlinear controller. Let $\hat{\mathbf{x}}(t)$ and $\hat{\mathbf{u}}(t), \hat{t}_f$ denote the state and input trajectories and final time generated by regulating the system to $\|g(\mathbf{x}(t,\theta)\| < \epsilon\, \forall t > \hat{t}_f$ with an initial state $\mathbf{x}_0$. We obtain the mapping relationship

$$M : \langle \mathbf{x}_0, \theta \rangle \rightarrow \left\langle \hat{\mathbf{x}}(t), \hat{\mathbf{u}}(t), \hat{t}_f \right\rangle. \tag{12}$$

By fixing the x_0 as the current state of the vehicle, $\hat{\mathbf{x}}(t)$ and $\hat{u}(t)$ are then dependent on θ only. The cost function in (10) can be modified as

$$\min_{\theta} J = \widetilde{c}_f(\theta) + \int_{t_0}^{t_0+T} \widetilde{c}(\theta) dt \tag{13}$$

where $\widetilde{c}$ and $\widetilde{c}_f$ represent running and terminal cost, respectively.

2) BSCPs Generation: We model the MAV as a nine degrees of freedom system, constructing a triple integrator on each of its x-, y-, and z-axes. Assuming $\mathbf{p}, \mathbf{v}, \mathbf{a}$, and $\mathbf{j}$ are all 3×1 vectors representing the position, velocity, acceleration, and jerk of the quadrotor, respectively, the quadrotor can be simplified into the following model: $\dot{\mathbf{p}} = \mathbf{v},\ \dot{\mathbf{v}} = \mathbf{a},\ \dot{\mathbf{a}} = \mathbf{j}$. According to [29], the state and input constraints are

$$\mathbf{v} \in [\mathbf{v}_{\min}, \mathbf{v}_{\max}],\ \mathbf{a} \in [\mathbf{a}_{\min}, \mathbf{a}_{\max}],\ \mathbf{j} \in [\mathbf{j}_{\min}, \mathbf{j}_{\max}]. \tag{14}$$

On each axis, define $\mathbf{x} = [p, v, a]^\intercal$ and $u = j$, where p, v, a, j are the single axis equivalent of $\mathbf{p}, \mathbf{v}, \mathbf{a}, \mathbf{j}$. The discrete time dynamics of the single-axis triple integrator is

$$\mathbf{x}[k+1] = A\mathbf{x}[k] + bu[k] \tag{15}$$

where

$$A = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix},\ b = \begin{bmatrix} \frac{\Delta t^3}{6} \\ \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}.$$

The target of the controller is to regulate the system from any initial state to the desired state $\mathbf{x}_d = [\theta, 0, 0]^\intercal$. Our end state constraint is

$$g(\mathbf{x}, \theta) = \mathbf{x} - \mathbf{x}_d = [p - \theta, v, a]^\intercal. \tag{16}$$

Define the error state dynamic as

$$\delta\mathbf{x}[k+1] = A\delta\mathbf{x}[k] + bu[k] \tag{17}$$

where $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}_d$. Define $V(\mathbf{x})$ as the approximation of the remaining integrated cost from $\delta\mathbf{x}$ to the origin. We have

$$V(\delta\mathbf{x}[n]) = C(\delta\mathbf{x}[n], u[n]) + V(\delta\mathbf{x}[n+1])) \tag{18}$$

where $C(\delta\mathbf{x}[n], u[n])$ is the instantaneous cost. To regulate $\delta\mathbf{x}$ to zero and satisfy the constraint in (14), we define our cost function as

$$C(\delta\mathbf{x}, u) = \delta\mathbf{x}^\intercal R\delta\mathbf{x} + ru^2 + H(\delta\mathbf{x}, u) + C_t(\delta\mathbf{x}) \tag{19}$$

where $\delta\mathbf{x}^\intercal R\delta\mathbf{x}$ represents target deviation; ru^2 denotes the input penalty: We penalize the system from taking aggressive moves

$$\begin{aligned} H(\delta\mathbf{x}, u) = {} & \omega_v \eta^2(s_v, v_{\min}, v_{\max}) \\ & + \omega_a \eta^2(s_a, a_{\min}, a_{\max}) \\ & + \omega_j \eta(j, j_{\min}, j_{\max})^2 \end{aligned} \tag{20}$$

penalizes the violation of the constraints in (14). In which, $\eta(\kappa, \kappa_1, \kappa_2) = \max(\kappa_1 - \kappa, 0) + \max(\kappa - \kappa_2, 0), \kappa, \kappa_1, \kappa_2 \in R$ and $\omega_a, \omega_v, \omega_j$ are weights. Finally, $C_t(\delta\mathbf{x})$ equals a constant C_{time} when $\delta\mathbf{x}$ is not at origin and equals 0 when $\delta\mathbf{x}$ is at origin. It is used to penalize the total time of the trajectory. With the instantaneous cost, we can solve the Bellman equation

$$V^*(\delta\mathbf{x}[n]) = \min_{u[n]\in J} C(\delta\mathbf{x}[n], u[n]) + V^*(\delta\mathbf{x}[n+1])) \tag{21}$$

through the value iteration, where J represents all available input. Once the value iteration has converged, the desired controller is then given as

$$u^* = \underset{u[n]\in J}{\operatorname{argmin}}\, C(\delta\mathbf{x}[n], u[n]) + V^*(\delta\mathbf{x}[n+1])). \tag{22}$$

The details of the value iteration process can be found in [31]. With the controller, we could perform a forward simulation to regulate the system from a given initial state to the origin and record the resulting state trajectory. For an arbitrary state $\delta\mathbf{x}$, the real state trajectory can then be recovered through $\mathbf{x}(t) = \mathbf{x}_d + \delta\mathbf{x}$. The simulation process here is an instance of the mapping relationship M in (12).

We propose to use an NN to approximate the mapping policy M with M_{NN} [36]. With the help of the modern parallel hardware, it accelerates the evaluation of the mapping M and makes algorithms such as PSO work in real time. A four-layer $64 \times 128 \times 128 \times 41$ multilayer perceptron with rectifier linear units is applied and trained with Adam optimizer using PyTorch. The input of this NN only includes the 3×1 vector error position $\delta p(t) = p - \theta$ instead of the full state $\delta\mathbf{x}(t)$, thus limiting the size of the network and making it executable in a GPU-free device. The output contains the future trajectory $\mathbf{p}(t)$ of the system sampled at 2 Hz for 20 s and u, which is a 41×1 vector.

Regarding the runtime for generating one trajectory, we compare the NN approximation with forward simulating for 20 s. The results in Table I show that the NN approximation method helps to increase the efficiency on both CPU and GPU platforms.

The training data can be obtained through forwarding simulation with the controller in (22) by random sampling $\delta\mathbf{x}_0$. The samples should be large enough making the generated trajectories able to cover the state limits in (14). The training and testing set sizes are 800 000 and 200 000, respectively. The batch size is 20 and the epoch is 12. We achieved an average mean squared error (MSE) of 4.72×10^{-4} and a maximum MSE of 0.071. Although a larger and more complex NN could be devised to further improve the accuracy, the simplified network we chose can be evaluated efficiently on a CPU-only platform.

TABLE I
TIME CONSUMPTION COMPARISON

	PyTorch (TX2)	C++ (I7)
NN Approximation	$3\mu s$	$15\mu s$
Forward Simulation	-	$70\mu s$

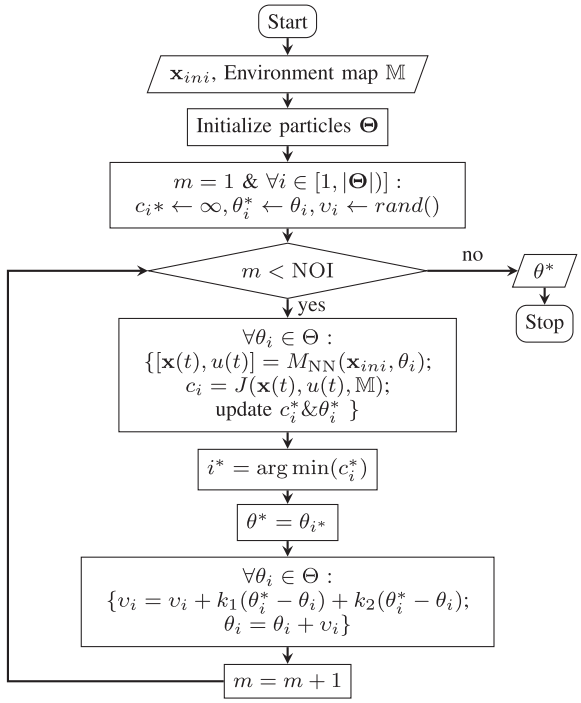


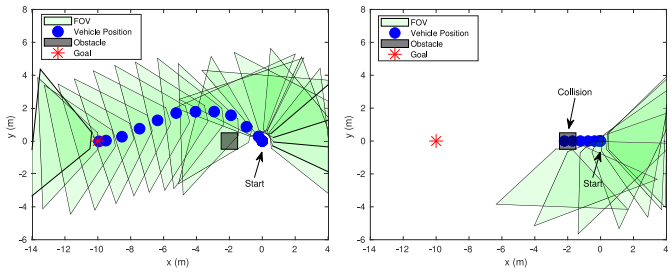
Fig. 5. PSO algorithm.

3) Motion Planning: The BSCPs and the PSO are combined for local motion planning, which is achieved by selecting the best target θ_p with the PSO algorithm. It allows the θ_p to be selected in a continuous space rather than a finite set [28]. The algorithm is shown in Fig. 5. The inputs are vehicle's initial state $\mathbf{x}_{\text{ini}}$ and the environment map $\mathbb{M}$. The particles in the algorithm represent the end state constraint. A finite number of particles Θ are first randomly initialized. Each particle θ_i is also assigned a velocity υ_i either randomly or with a special pattern which decides its future movement in the optimization space. θ_i^* represents each particle's best value during each iteration with its corresponding cost value c_i^*. NOI represents the number of iterations.

The cost function J is designed as follows:

$$J(\mathbf{x}(t), u(t), \mathbb{M}) = \int_{t_0}^{t_0+T} H(\mathbf{x}(t)) + O(\mathbf{x}(t)). \tag{23}$$

$H(\mathbf{x}(t))$ evaluates the progress of the trajectory, which is calculated by measuring the remaining distance to the target. This distance can be the Euclidean distance or calculated by other global planning algorithms [31]. $O(\mathbf{x}(t))$ denotes penalization on the obstacles. We could assign each grid a cost value depending on



(a) (b)

Fig. 6. Flight to a rear target. When considering the limited FOV in (a), the planned future trajectory at each cycle is always constrained in the current $\mathcal{V}_f$, thereby guaranteeing safety. On the other hand, in (b), the FOV limit is ignored, the heading and the translational movement are planned separately, and the future trajectory is no longer guaranteed to stay inside the current $\mathcal{V}_f$. The MAV might end in a volume that has never been observed before, thus possibly leading to a collision.

its distance to the obstacle. The smaller the distance, the higher the cost. $O(\mathbf{x}(t))$ can then be evaluated by adding up the costs of the grids that are occupied by the vehicle while at certain states. The MPC is executed at 5 Hz with a 20-s prediction horizon.

In the particle processing iteration, the state trajectories are obtained through our trained NN mapping M_{NN}. Then the cost of each particle is evaluated by the cost function J with the environment information $\mathbb{M}$. The best target θ^* among all the particles is recorded after each iteration.

4) Safe Navigation With Limited FOV: Safe navigation with limited FOV can be achieved by requiring the planned trajectory to stay inside the volume $\mathcal{V}_f$ that is confirmed to be obstacle-free and inside the camera's FOV cone. The volume $\mathcal{V}_f$ is highly nonconvex due to the sensor limitation and the presence of obstacle occlusion, and it also has a noncontinuous boundary. This implies a noncontinuous and nonconvex problem. The safe navigation constraints are added to (23) as a soft constraint

$$O_{\text{visual}}(x) = \begin{cases} \mu \text{ if } x \notin \mathcal{V}_f \\ 0 \text{ otherwise} \end{cases} \tag{24}$$

where μ is just a large penalty cost.

The resulting nonconvex and noncontinuous optimization can be solved by resorting to the PSO algorithm. With the BSCPs, the problem can be solved in real time with an average computing time of 14 ms on the TX2 computer. In Fig. 6(a) and (b), we compare the results of considering the limited FOV and not to do so. In both figures, the vehicle is initialized at the origin at hover condition with the onboard camera facing in the positive x-direction. The desired goal is behind the vehicle at $[x = -10, y = 0, z = 1.5]$, and the desired heading is to always point toward the desired goal. With the additional FOV constraints, the planned trajectory will always stay inside $\mathcal{V}_f$ and navigate cautiously against unknown volumes.

VI. Experimental Results

The proposed autonomous system has been tested extensively in multiple real-world environments, for example, at an abandoned school shown in Fig. 7. The testing site challenges our

Fig. 7. Testing site of a 30 × 30 m dining hall connected to a 60-m-long corridor with scattered pillars.

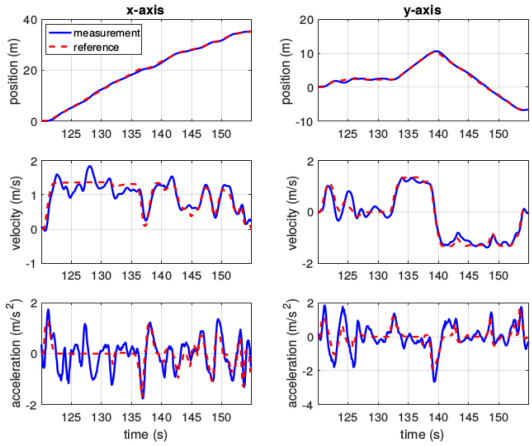


Fig. 8. Position and velocity response of corridor flight testing.

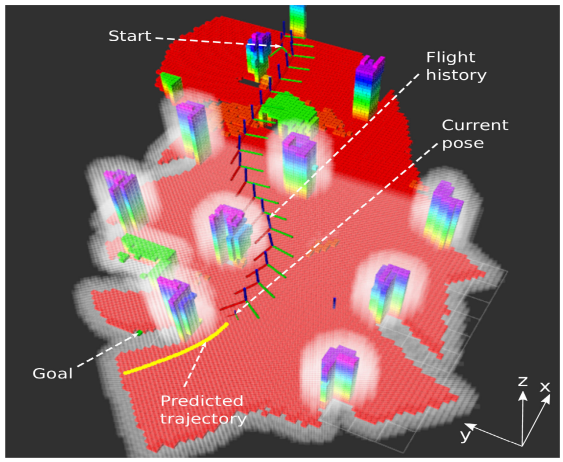


Fig. 9. Map from the simulated environment. In the colored one is the 3-D OG map, and in white is 3-D EDT map. The MAV is able to plan a trajectory that can successfully avoid all the obstacles on its way to the final target (shown as a green dot).

system in many ways: First, the relatively long distance, ever-changing brightness caused by the semiopen space, narrow space of the corridor, as well as the interference from the environment to the magnetometer requiring our state estimation algorithm to be accurate enough and have low-drift during long-range flight; second, our mapping has to be accurate enough to avoid

different types of obstacles such as pillars and people in the hall without any prior information; third, the motion planning has to work efficiently to choose the optimal local target and generate obstacle-free trajectory with the given setpoints. In the corridor flight testing, we set the target in the corridor behind pillars and a wall, and the proposed method was able to avoid all the obstacles on the way autonomously with the onboard sensors and computational power described in Section III. The performance of corridor flight testing is shown in Fig. 8. The map from the simulated environment is shown in Fig. 9.

VII. Conclusion

In this article, we presented an MAV system that can navigate autonomously in GPS-denied and obstacle-cluttered environments while avoiding obstacles on the way. Various advanced technologies were designed, including a stereo visual–inertial state estimation that can handle scenarios without external localization resource, a GPU-based EDF mapping that significantly improves the real-time performance without sacrificing accuracy, as well as a model predictive local motion planning with BSCPs solved by PSO, providing increased performance for tasks that require precise and timely maneuver. The results of simulation and real flight testing in both indoor and semiopen scenarios with various tasks proved the effectiveness of the proposed system for challenging practical applications. We believe this system is suitable enough for various tasks. However, there are still many improvements that can be achieved to improve the whole system further. Concerning state estimation, we need to reduce the drift in high speed flight. The prediction of dynamic obstacles will also be included in the future extensions of this work.

References

[1] C. C. Olsen, K. Kalyanam, W. P. Baker, and D. L. Kunz, "Maximal distance discounted and weighted revisit period: A utility approach to persistent unmanned surveillance," *Unmanned Syst.*, vol. 7, no. 04, pp. 215–232, 2019.

[2] J. Hu, J. Xu, and L. Xie, "Cooperative search and exploration in robotic networks," *Unmanned Syst.*, vol. 01, no. 01, pp. 121–142, 2013. [Online]. Available: https://doi.org/10.1142/S2301385013500064

[3] S. Zhao *et al.*, "A robust real-time vision system for autonomous cargo transfer by an unmanned helicopter," *IEEE Trans. Ind. Electron.*, vol. 62, no. 2, pp. 1210–1219, Feb. 2015.

[4] S. Park and S. Hashimoto, "Autonomous mobile robot navigation using passive RFID in indoor environment," *IEEE Trans. Ind. Electron.*, vol. 56, no. 7, pp. 2366–2373, Jul. 2009.

[5] J. Li, Y. Bi, K. Li, K. Wang, F. Lin, and B. M. Chen, "Accurate 3D localization for MAV swarms by UWB and IMU fusion," in *Proc. IEEE 14th Int. Conf. Control Autom.*, 2018, pp. 100–105.

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

[7] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proc. SIGGRAPH*, 1996, pp. 303–312.

[8] M. Lan, S. Lai, and B. M. Chen, "Towards the realtime sampling-based kinodynamic planning for quadcopters," in *Proc. 11th Asian Control Conf.*, Dec. 2017, pp. 772–777.

[9] T. Qin, S. Cao, J. Pan, and S. Shen, "A general optimization-based framework for global pose estimation with multiple sensors," 2019, *arXiv:1901.03642*.

[10] K. Sun *et al.*, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 965–972, Apr. 2018.

[11] K. Mohta *et al.*, "Fast, autonomous flight in GPS-denied and cluttered environments," *J. Field Robot.*, vol. 35, no. 1, pp. 101–120, 2018.

[12] L. Campos-Macías, R. Aldana-López, R. de la Guardia, J. I. Parra-Vilchis, and D. Gómez-Gutiérrez, "Autonomous navigation of MAVs in unknown cluttered environments," 2019, *arXiv:1906.08839*.

[13] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

[14] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *Int. J. Robot. Res.*, vol. 36, no. 10, pp. 1053–1072, 2017.

[15] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 15–22.

[16] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3923–3929.

[17] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots*, 2013. [Online]. Available: http://octomap.github.com

[18] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d Euclidean signed distance fields for on-board MAV planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 1366–1373.

[19] R. A. Newcombe *et al.*, "Kinectfusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality*, Oct. 2011, pp. 127–136.

[20] L. Han, F. Gao, B. Zhou, and S. Shen, "Fiesta: Fast incremental Euclidean distance fields for online motion planning of aerial robots," 2019, *arXiv:1903.02144*.

[21] W. Feng and S. Xiao, "To GPU synchronize or not GPU synchronize?" in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2010, pp. 3801–3804.

[22] J. Chen, K. Su, and S. Shen, "Real-time safe trajectory generation for quadrotor flight in cluttered environments," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Dec. 2015, pp. 1678–1685.

[23] S. Liu *et al.*, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1688–1695, Jul. 2017.

[24] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 5332–5339.

[25] J. Tordesillas, B. T. Lopez, J. Carter, J. Ware, and J. P. How, "Real-time planning with multi-fidelity models for agile flights in unknown environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2019, pp. 725–731.

[26] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenÃ©t frame," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 987–993.

[27] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1294–1310, Dec. 2015.

[28] B. T. Lopez and J. P. How, "Aggressive 3-D collision avoidance for high-speed navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2017, pp. 5759–5765.

[29] M. Hehn and R. DâAndrea, "Real-time trajectory generation for quadrocopters," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 877–892, Aug. 2015.

[30] B. T. Lopez and J. P. How, "Aggressive collision avoidance with limited field-of-view sensing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 1358–1365.

[31] S. Lai, M. Lan, and B. M. Chen, "Model predictive local motion planning with boundary state constrained primitives," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3577–3584, Oct. 2019.

[32] B. M. Chen, T. H. Lee, and V. Venkataramanan, *Hard Disk Drive Servo Systems*. Berlin, Germany: Springer, 2002.

[33] S. M. Weiss, "Vision based navigation for micro helicopters," Ph.D. dissertation, ETH Zurich, Zurich, Switzerland, 2012.

[34] R. A. Newcombe *et al.*, "Kinectfusion: Real-time dense surface mapping and tracking," *ISMAR*, vol. 11, no. 2011, 2011, pp. 127–136.

[35] C. R. Maurer, R. Qi, and V. Raghavan, "A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 2, pp. 265–270, Feb. 2003.

[36] S. Lai, M. Lan, and B. M. Chen, "Model predictive local motion planning with boundary state constrained primitives," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3577–3584, Oct. 2019.

Yu Zhou received the B.Eng. degree in automation and the M.Eng. degree in control engineering from Northeastern University, Shenyang, China, in 2014 and 2017, respectively.

She is currently working as an Associate Scientist with Temasek Laboratories, National University of Singapore, Singapore. Her research interests include visual SLAM, motion planning, optimization, and reasoning under uncertainty.

Shupeng Lai received the B.Eng. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, in 2012, and the Ph.D. degree in Integrative Sciences and Engineering from the National University of Singapore, Singapore, in 2016.

His research interests include motion planning, nonlinear model predictive control, multiagent systems, and aerial systems.

Huimin Cheng received the B.Eng. degree in electrical engineering from the National University of Singapore, Singapore, in 2019. He is currently working toward the M.Comp. degree at NUS School of Computing, Singapore.

He is currently working as an Associate Scientist with Temasek Laboratories, Singapore focusing on visual SLAM algorithm and multicamera perception system design.

Abdul Hamid Mohamed Redhwan received the B.Sc. degree in aircraft engineering from Kingston University, Kingston, London, in 2014.

He is a UAV pilot and engineer who works on hardware and aircraft design.

Zhi Gao received the B.Eng. and Ph.D. degrees from the Photogrammetry Department, Wuhan University, Wuhan, China, in 2002 and 2007, respectively.

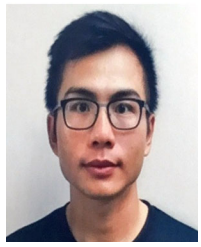
Since 2008, he joined the Interactive and Digital Media Institute, National University of Singapore (NUS), Singapore, as a Research Fellow (A) and Project Manager. In 2014, he joined Temasek Laboratories, NUS (TL@NUS) as a Research Scientist (A) and Principal Investigator. He is currently working as a Full Professor with the School of Remote Sensing and Information Engineering, Wuhan University. Since 2019, he has been supported by the distinguished professor program of Hubei Province, China. He has published more than 70 research papers on top journals and conferences, such as *International Journal of Computer Vision*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTION ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, *IEEE Geoscience and Remote Sensing Letters*, *IEEE Signal Processing Letters*, *Journal of Machine Vision and Applications*, *Journal of Real-Time Image Processing*, ACM journals, *European Conference on Computer Vision*, *Asian Conference on Computer Vision*, *British Machine Vision Conference*, etc. His research interests include computer vision, machine learning, remote sensing and their applications. In particular, he has strong interests in UAV-based surveillance research and applications.

Pengfei Wang received the B.Eng. degree in mechanical science and engineering, from the Huazhong University of Science and Technology, Wuhan, China, in 2012, and the Ph.D. degree in mechanical engineering from the Department of Mechanical Engineering, National University of Singapore, Singapore, in 2016.

He is a Research Scientist with Temasek Laboratories, National University of Singapore, Singapore. His research interest focuses on sense and avoid of UAVs, deep learning based detection, and depth estimation on UAV platform.

Junji Zhu received the B.E. degree in mechanical engineering and the M.Eng. degree in electrical engineering from the National University of Singapore, Singapore, in 2016 and 2019, respectively.

He is currently an Associate Scientist with Centre for Flight Sciences, Temasek Laboratories, National University of Singapore. His research interests include UAV control and automation, 3-D modeling and manufacturing, and 3-D simulation.



Zhengtian Ma received the B.E. degree in electrical engineering from the National University of Singapore (NUS), Singapore, in 2019.

He is currently working as an Associate Scientist with Centre for Flight Sciences, Temasek Laboratories, NUS. His research interests include multirotor control design and implementation as well as control design and implementation with data-driven methods.

Yingcai Bi received the B.Eng. degree in automation engineering from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, the M.Eng. degree in control science and engineering from the Beihang University (BUAA), Beijing, China, and the Ph.D. degree in integrative sciences and engineering from the National University of Singapore (NUS), Singapore, in 2018.

His research interests lie on localization, mapping and autonomous navigation for unmanned systems.

Feng Lin received the B.Eng. degree in computer science and control and the M.Eng. degree in system engineering from the Beihang University, Beijing, China, in 2000 and 2003, respectively. He received the Ph.D. degree in computer and electrical engineering from the National University of Singapore (NUS), Singapore, in 2011.

He is currently working as a Senior Research Scientist with the Temasek Laboratories, NUS, and a Research Assistant Professor with Department of Electrical & Computer Engineering, National University of Singapore. His main research interests are unmanned aerial vehicles, vision-aided control and navigation, target tracking, robot vision, as well as embedded vision systems.

Dr. Lin has served on the Editorial Board for Unmanned Systems. He was the recipient of the Best Application Paper Award, 8th World Congress on Intelligent Control and Automation, Jinan, China, in 2010.

Ben M. Chen (Fellow, IEEE) is currently a Professor with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong, and a Professor with the Department of Electrical and Computer Engineering, National University of Singapore (NUS), Singapore. He was a Provost's Chair Professor with NUS. His current research interests are in unmanned systems, robust control, and control applications. He has published more than 400 journal and conference articles and authored/coauthored ten research monographs in systems and control theory, industrial applications, unmanned systems, and financial market modeling.

Dr. Chen had served on the editorial boards for several international journals including IEEE TRANSACTIONS ON AUTOMATIC CONTROL and *Automatica*. He currently serves as an Editor-in-Chief for Unmanned Systems. He was the recipient of a number of research awards nationally and internationally. His research team has actively participated in international unmanned aerial vehicles competitions and won many championships in the contests.