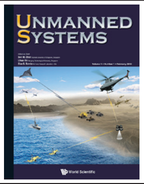
Unmanned Systems, Vol. 4, No. 1 (2016) 83–96
© World Scientific Publishing Company
DOI: 10.1142/S2301385016400094

UNMANNED SYSTEMS
http://www.worldscientific.com/worldscinet/us



Search and Rescue Using Multiple Drones in Post-Disaster Situation

Jin Q. Cui[*,¶], Swee King Phang[§,‖], Kevin Z. Y. Ang[*,**], Fei Wang[†,††], Xiangxu Dong[*,‡‡], Yijie Ke[§,§§], Shupeng Lai[‡,¶¶], Kun Li[§,‖‖], Xiang Li[†,***], Jing Lin[*], Peidong Liu[‡], Tao Pang[*,†††], Kangli Wang[§,‡‡‡], Zhaolin Yang[*,§§§], Feng Lin[*,¶¶¶], Ben M. Chen[§,‖‖‖]

[*]*Temasek Laboratories, National University of Singapore, Singapore*

[†]*AeroLion Technologies, The Galen, 61 Science Park Rd, #06-01 Science Park II, Singapore 117525*

[‡]*Graduate School for Integrative Sciences & Engineering, National University of Singapore, Singapore*

[§]*Department of Electrical & Computer Engineering, National University of Singapore, Singapore*

We present the development and application of multiple autonomous aerial vehicles in urban search and rescue missions. The missions are designed by the 2014 International Micro Aerial Vehicle Competition, held in Delft, the Netherlands, August 2014. Different mission tasks are identified for search and rescue missions, such as aerial photography, low altitude flight in urban environment, indoor navigation and rooftop landing. These tasks are all of paramount importance for rescuers in a disaster-hit place. We have designed a team of micro aerial vehicles with specific configurations to meet the mission requirements. A range of key technologies have been developed, including robust controller design, real-time map stitching, indoor navigation and roof-top perching. The proposed solutions are successfully demonstrated in the competition.

Keywords: Multiple aerial vehicles; aerial photography; indoor navigation; rooftop landing; digit detection.

1. Introduction

While the term ‘drone’ is gaining more popularity in this modern world, many researchers around the globe start to investigate more advanced applications for drones, or in other words, the micro air vehicles (MAVs). Due to its small size factor and ease of use, many applications which were previously hard to achieve are now realizable with a swamp of MAVs, in a collaboration manner either by doing the same task, or by working on different sub-tasks of a main mission [1].

In recent years, MAVs play increasingly important roles in many civilian applications, such as in aerial reconnaissance, search and rescue and post-disaster area exploration [2]. Using a team of MAVs to build up the communication system has been investigated in [3], in which the authors have discussed about the formation strategy and the design of an end-to-end communication system. For aerial surveillance, a single camera onboard an MAV could collect images along a predefined trajectory so that a 3D terrain map can be extracted [4]. For fast forest fire evaluation, a team of MAVs carrying infrared cameras are also investigated to assess the propagation of large forest fires [5]. Time-varying formation control of multiple MAVs has been investigated in [6] with experiments carried out in quad-rotor platforms.

While the sensors and processors are getting more intelligent and smaller, MAVs can now be realized in smaller packages. This results in the shift of research direction from outdoor navigation to GPS-denied indoor navigation of MAVs. While extensive research has been conducted to

Received 27 October 2015; Revised 25 December 2015; Accepted 26 December 2015; Published 5 February 2016. This paper was recommended for publication in its revised form by Special Issue Guest Editor, Guoqiang Hu.
Email Addresses: [¶]*cuijinqiang@nus.edu.sg*, [‖]*king@nus.edu.sg*, [**]*tslangzy@nus.edu.sg*, [††]*wangfei@aerolion.com*, [‡‡]*tsldngx@nus.edu.sg*, [§§]*A0109975@nus.edu.sg*, [¶¶]*shupenglai@nus.edu.sg*, [‖‖]*kunli89@nus.edu.sg*, [***]*lixiang@aerolion.com*, [†††]*tslpt@nus.edu.sg*, [‡‡‡]*a0055862@nus.edu.sg*, [§§§]*tslyangzl@nus.edu.sg*, [¶¶¶]*linfeng@nus.edu.sg*, [‖‖‖]*bmchen@nus.edu.sg*

apply various linear and nonlinear control laws for the MAVs, many researchers are also focusing on MAV localization and mapping methods using smart sensors such as laser range finders, cameras and ultrasonic sensors [7]. The design of MAV system resembles the design of other systems consisting of both mechanics and electronics modules [8].

In August 2014, an annual international micro air vehicle (IMAV) competition was held in Delft, the Netherlands. This competition was organized by the MAVLab from TU Delft, with the aim of crowdsourcing technical solutions to help in search and rescue missions using a swamp of MAVs. The competition has attracted many research teams from various countries to submit their proposals to meet the mission requirements. Fourteen short-listed finalists are then required to demonstrate the capabilities of their MAVs on site in Delft, the Netherlands.

The main objective of the competition is to simulate a search and rescue mission using MAVs in a post-disaster village. In order to achieve a fast and accurate evaluation of the disaster, the whole mission is divided into four complementary mission elements. Each mission element has a specific task and forms an indispensable part for search and rescue missions. As shown in Fig. 1, the four tasks are categorized as follows:

- Task A: **Photomapping a village**, the mission element requires a drone to inspect the targeted area defined by the rectangle 'A' in Fig. 1. A high resolution overview image has to be provided in 30 min after the drone lands. It is preferable to perform real-time aerial photography and map stitching using the onboard computer in order to provide the stitched map in time. The map has to be of high resolution so that several possible routes without obstacles can be identified for rescuers to enter the village.
- Task B: **Fast house inspection**, this mission element requires a drone to search each house along the main street of the village, which is labeled by the polygon 'B' in Fig. 1. The drone has to fly at a height below the rooftop to look through the windows to check if there are victims trapped in the houses. The number of each house has to be recognized autonomously.
- Task C: **Indoor inspection**, the mission element requires a drone to navigate in a two-story building (labeled as 'C' in Fig. 1), and to determine the number of victims or identify objects, such as chairs and photo frames, in each room.
- Task D: **Rooftop observation**, the mission element demands landing an MAV on a rooftop and observing the situation in another building across the road. To simplify the observation scenario, a seven-segment digit is placed on the wall of the neighboring house. The digit keeps changing every 30 s. Precision landing on the rooftop and digit recognition are required to accomplish the observation task.

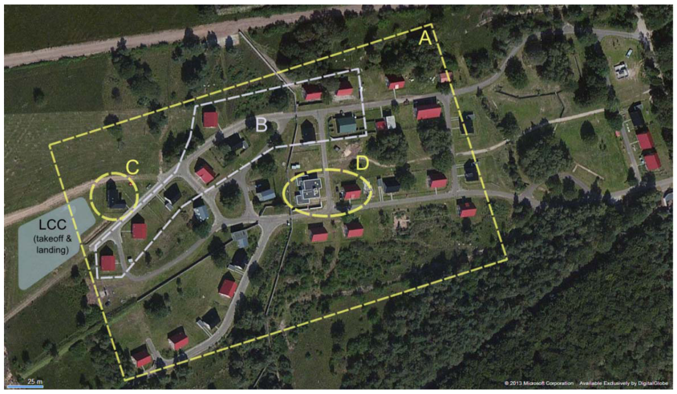

A
B
C
D
LCC (takeoff & landing)
25 m
© 2013 Microsoft Corporation Available Exclusively by DigitalGlobe

Fig. 1. Overview of task assignment in IMAV 2014.

The UAV team from the National University of Singapore has taken part in this competition, and demonstrated successfully our solutions to all the four tasks using multiple MAVs [9]. This manuscript describes the key technologies proposed by our team to meet the requirements of the mission elements in this competition. These sophisticated solutions help us win the championship of the competition. The remaining part of the manuscript is organized as follows: Section 2 analyzes the requirements of each tasks and presents the hardware and software configuration of the MAVs for each mission. Section 3 presents the key technologies developed for this competition, including robust controller design, real-time image stitching, number detection, indoor navigation and vision-based pose estimation. Section 4 concludes the manuscript.

2. System Configuration

The four tasks in the competition focus on different aspects of a search and rescue mission. Every aspect requires a specific configuration of the platforms, both in the system hardware and the software aspects. On the other hand, it is highly desirable that these platforms could share as many resources as possible to minimize the development cost and time. This section presents the configuration of MAVs for the four tasks, illustrating the hardware configurations and the software framework.

The hardware configuration of an MAV consists of the platform selection and the avionics system design. In a wide range of prospective platforms, we have selected quadrotor as the MAV platform due to its simple mechanical structure and widely available autopilots. The simple structure makes it easy to maintain and to assemble additional mission

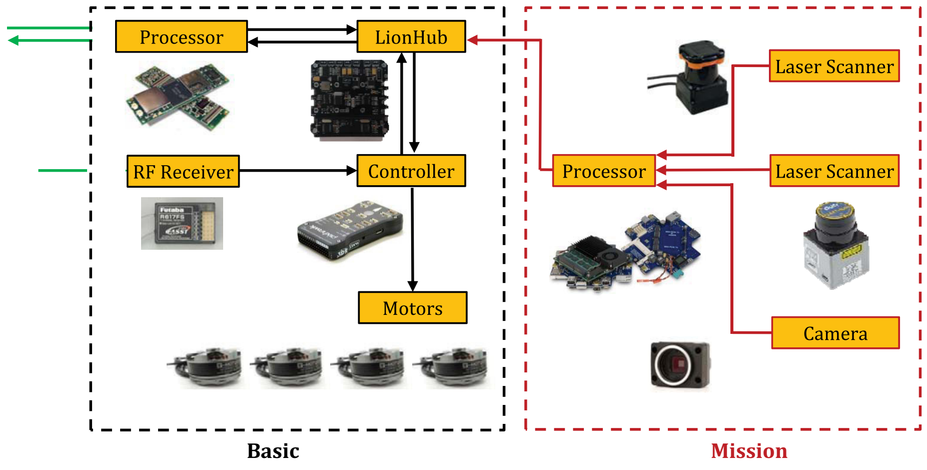


Fig. 2. The avionics system configuration for IMAV 2014 task C.

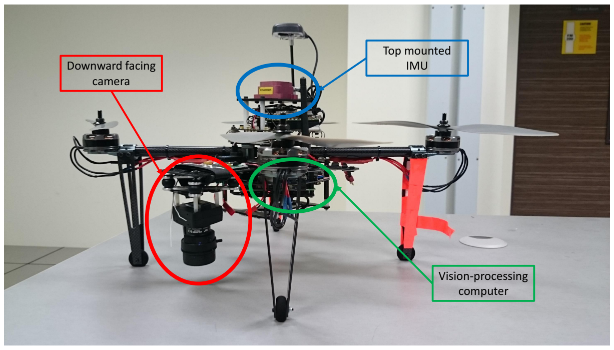
components. It is also due to the simple structure of quadrotors that the mathematical model is easy to be derived, facilitating the design and implementation of autonomous controllers to stabilize its angular dynamics and to achieve tracking external position references.

The design of the avionic system needs to provide all the hardware support for the MAV navigation and the mission requirements. The navigation of MAVs requires basic hardware modules, such as inertial measurement unit (IMU), processors, servo driving circuits. Configuration of different avionics modules has to cater to the requirements of different mission tasks. As a result, the avionics system is designed to have a common basic part and a specific mission-dependent part. The basic part is the same for the four tasks, including the attitude controller 'Pixhawk' and the position controller 'Gumstix Overo Fire'. It also contains a small-size computer called 'Mastermind' which serves as the mission processor. The mission computer interfaces with various sensors required by the mission to perform those computationally intensive algorithms, such as simultaneous localization and mapping (SLAM), path planning, and vision processing algorithms. For example, Task C (as shown in Fig. 2) requires navigating in indoor environments and detecting objects in rooms. We use two Hokuyo laser range finders and one camera to meet the mission requirements. The Hokuyo 'UTM-30LX' is assembled horizontally to scan the environment at 40 Hz to provide information for indoor localization and mapping. Another Hokuyo 'URG-04LX' is assembled vertically to scan the ground plan to get relative height in complex terrain conditions. The camera is used for survey the surrounding environment for object detection.

Other MAV avionics system share a similar structure but with different mission modules, which are summarized by Table 1. For task A, a high accuracy GPS-based navigation is required for the MAV to navigate above the defined area and capture sharp images. We use 'IG-500N' as the navigation module and assemble a downward facing camera for the image collection. For task B, navigation of MAV is supposed to take place below rooftops, making it impossible to use GPS as navigation sensors. We come out with a solution that uses optical flow sensor to estimate the relative velocity of the MAV for onboard state estimation. At the same time, we still use the GPS signal as the guidance reference once GPS is ready. A laser range finder (UTM-30LX) is also used to avoid obstacles, such as walls or cars parking in the streets. For task D, precision landing of the MAV on the corner of a rooftop is required so that the MAV get the best observation angle. Thus we use a downward facing camera

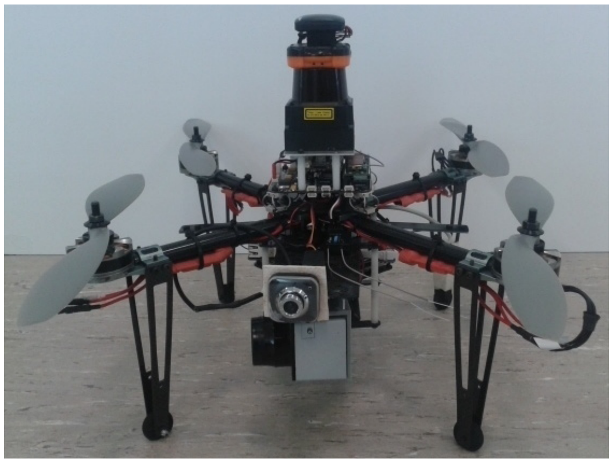
Table 1. Platform configuration list.

Platform	Modules	Mission elements
A	IG-500N Downward facing camera	GPS navigation Real-time image stitching
B	UTM-30LX PX4Flow Forward facing camera	Path planning Urban navigation House number recognition
C	UTM-30LX URG-04LX Forward looking camera	Indoor navigation Height measurement Object recognition
D	IG-500N Downward facing camera Forward facing camera	GPS waypoint navigation Vision-guided rooftop landing Digit recognition
E	WiFi router	WiFi relay

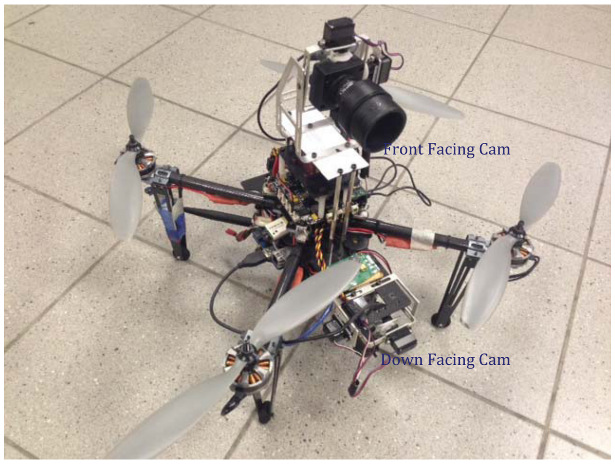


(a) UAV for image stitching

(b) UAV for urban navigation



(c) UAV for indoor navigation



(d) UAV for rooftop landing and surveillance

Fig. 3. Overview of platforms for different missions at IMAV2014.

to implement vision-guided rooftop landing and use a forward looking camera to observe the changing digit in the neighboring house. Last but not least, we use a fifth MAV to carry a WiFi router to provide wireless communication between the four MAVs and their ground control stations. Figure 3 shows close view of each platform developed for the competition.

According to the above hardware configuration, the software system is implemented in different threads allocated in the two computers: the 'Gumstix Overo Fire' and the 'Mastermind'. As shown in Fig. 4, they are labeled as *Flight control processor* and *Mission plan processor*, respectively. Since the 'Mastermind' possesses powerful processing capabilities, high level tasks such as *SLAM*, *Vision* and *Path planning* are scheduled. For the flight control subsystem, different tasks are realized in the threads running on the 'Gumstix Overo Fire'. The sensor fusion is in *IMU* and the control task in *CTL*. Motor driving signals are sent to the MAV motors from the *SVO* task to achieve the 6 degree of freedom (DOF) movement. Other auxiliary tasks are also implemented: the communication task *CMM* is to send status data back to ground control station (GCS) for user monitoring and to receive user commands; the data logging task *DLG* is used to record flight status data for post flight analysis. Finally, to pass high level navigation data to *Flight control processor* and share MAV status with *Mission plan processor*, the inter-processor communication task *ICMM* is implemented on both processors.

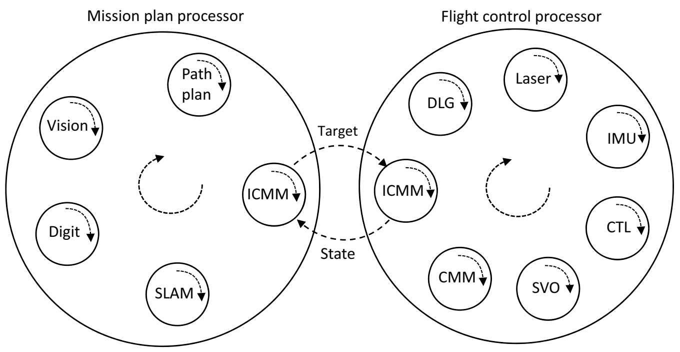


Fig. 4. Software structure of MAV navigation system.

3. Key Technologies Development

The collaborative operations of multiple MAVs require good performance of each individual platform with specific mission capabilities. Based on the quadrotor platform and the software structure, different algorithms are developed, including the design of a robust controller for position tracking, real-time image stitching, indoor navigation, vision-based pose estimation and digit-detection.

3.1. *Robust controller design*

The controller of an MAV is usually separated into two loops: an inner-loop running at a higher rate to stabilize the attitude dynamics of the aircraft, and an outer-loop running at a lower rate to control the position or linear velocity of the aircraft [10]. As mentioned in Sec. 2, the attitude dynamics of the platform is stabilized by the commercial ‘Pixhawk’ autopilot. The ‘Pixhawk’ autopilot comes with open source attitude controller, which is available freely online [11]. Thus in this manuscript, only the design of the outer-loop controller for position tracking is covered.

In order to achieve precise and robust position tracking, a robust perfect tracking (RPT) controller [12] is applied. Theoretically, a system controlled by this method is able to track any given reference with arbitrarily fast settling time subjected to disturbances and initial conditions. The basic idea can be generalized as follows. Given a linear time-invariant system,

$$\Sigma = \begin{cases} \dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + E\mathbf{w}, \\ \mathbf{y} = C_1\mathbf{x} + D_1\mathbf{w}, \\ \mathbf{h} = C_2\mathbf{x} + D_2\mathbf{u} + D_{22}\mathbf{w} \end{cases} \tag{1}$$

with $\mathbf{x}, \mathbf{u}, \mathbf{w}, \mathbf{y}, \mathbf{h}$ being the state, control input, disturbance, measurement and controlled output, respectively, the task of the RPT controller is to formulate a dynamic measurement control law of the form of

$$\begin{aligned} \dot{\mathbf{v}} &= A_c(\varepsilon)\mathbf{v} + B_c(\varepsilon)\mathbf{y} + G_0(\varepsilon)r + \cdots + G_{\kappa-1}(\varepsilon)r^{\kappa-1}, \\ \mathbf{u} &= C_c(\varepsilon)\mathbf{v} + D_c(\varepsilon)\mathbf{y} + H_0(\varepsilon)r + \cdots + H_{\kappa-1}(\varepsilon)r^{\kappa-1}, \end{aligned}$$

so that when a proper $\varepsilon^* > 0$ is chosen,

(1) The resulted closed-loop system is asymptotically stable subjected to zero reference.
(2) If $e(t,\varepsilon)$ is the tracking error, then for any initial condition $\mathbf{x}_0$, there exists:

$$\|e\|_p = \left(\int_0^\infty |e(t)^p| dt\right)^{1/p} \to 0, \quad \text{as } \varepsilon \to 0. \tag{2}$$

Similar to the case introduced in [13], the outer dynamics of a quadrotor MAV is differentially flat. This means all its state variables and inputs can be expressed in terms of algebraic functions of flat outputs and their derivatives. A proper choice of the flat output could be

$$\boldsymbol{\sigma} = [x, y, z, \psi]^{\mathrm{T}}. \tag{3}$$

It can be observed that the first three outputs, x, y, z, are totally independent. We can consider the MAV as a mass point with constrained velocity, acceleration, jerk and any higher order. Hence, a stand-alone RPT controller based on a multiple-layer integrator model in each axis can be designed to track the corresponding reference in that axis. For the x-axis or the y-axis, the nominal system can be written as

$$\begin{cases} \dot{\mathbf{x}}_n = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\mathbf{x}_n + \begin{bmatrix} 0 \\ 1 \end{bmatrix}u_n, \\ \mathbf{y}_n = \mathbf{x}_n, \end{cases} \tag{4}$$

where $\mathbf{x}_n$ contains the position and velocity state variables and u_n is the desired acceleration.

To achieve better tracking performance, it is common to include an error integral to ensure zero steady-state error subjected to step inputs. This requires an augmented system to be formulated as

$$\begin{cases} \dot{\mathbf{x}}_{xy} = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}\mathbf{x}_{xy} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}u_{xy}, \\ \mathbf{y}_{xy} = \mathbf{x}_{xy}, \\ \mathbf{h}_{xy} = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]\mathbf{x}_{xy}, \end{cases} \tag{5}$$

where $\mathbf{x}_{xy} = [\int(p_e) \ \ p_r \ \ v_r \ \ a_r \ \ pv]^{\mathrm{T}}$ with p_r, v_r, a_r as the position, velocity and acceleration references in the controlled axis, p, v as the actual position and velocity and $p_e = r_p - p$ as the tracking error of position. By following the procedures in [12], a linear feedback control law can be formed as follows:

$$u_{xy} = \mathbf{F}_{xy}\mathbf{x}_{xy}, \tag{6}$$

where

$$\begin{aligned} \mathbf{F}_{xy} = &\left[\frac{k_i\omega_n^2}{\varepsilon^3} \ \ \frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} \ \ \frac{2\zeta\omega_n + k_i}{\varepsilon}\right. \\ &\left. 1 - \frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} - \frac{2\zeta\omega_n + k_i}{\varepsilon}\right]. \end{aligned} \tag{7}$$

Here, ε is a design parameter to adjust the settling time of the closed-loop system. ω_n, ζ, k_i are the parameters that determines the desired pole locations of the infinite zero structure of (5) through

$$p_i(s) = (s + k_i)(s^2 + 2\zeta\omega_n s + \omega_n^2). \tag{8}$$

The z-axis control is similar but in a lower-order form. As the inner-loop is directly looking for velocity reference in this axis, it is straightforward to model the outer loop as a single integrator from velocity to position, leading to the following augmented system,

$$\begin{cases} \dot{\mathbf{x}}_z = \begin{bmatrix} 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_z + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_z, \\ \mathbf{y}_z = \mathbf{x}_z, \\ \mathbf{h}_z = [1 \quad 0 \quad 0 \quad 0]\mathbf{x}_z, \end{cases} \tag{9}$$

where $\mathbf{x}_z = [\int(p_e) \;\; p_r \;\; v_r \;\; p]^{\mathrm{T}}$. This leads to a linear feedback control law of

$$u_z = \mathbf{F}_z \mathbf{x}_z, \tag{10}$$

where

$$\mathbf{F}_z = \left[-\frac{\omega_n^2}{\varepsilon} \quad \frac{2\omega_n\zeta}{\varepsilon^2} \quad 1 \quad -\frac{2\omega_n\zeta}{\varepsilon^2} \right].$$

Theoretically, when the design parameter ε is small enough, the RPT controller can give arbitrarily fast responses. However, due to the constraints of the MAV physical dynamics and its inner-loop bandwidth, it is safer to limit the bandwidth of the outer loop to be much smaller than that of the inner-loop dynamics. For the MAV designed in this paper, the following design parameters are used:

$$x\text{-},\ y\text{-axes}: \begin{cases} \varepsilon = 1, \\ \omega_n = 0.99, \\ \zeta = 0.707, \\ k_i = 0.25, \end{cases} \qquad z\text{-axes}: \begin{cases} \varepsilon = 1, \\ \omega_n = 0.559, \\ \zeta = 2. \end{cases}$$

The designed RPT controller have been implemented in multiple autonomous flight tests under various weather conditions. Figure 5 shows the tracking performance of the controller in both x- and y-directions, including the position tracking and the velocity tracking. The maximum tracking error for the position is 0.4 m and 0.1 m/s for the velocity.

3.2. *Fast onboard image stitching*

In order to provide a fast evaluation of the targeted area, we need a high resolution stitched image the instant we have collected the images. The stitching algorithm has to be robust and reliable enough and runs in real time. Therefore, we eliminate the common appearance enhancements, which aim to beautify the stitched map usually found in panoramic stitching algorithms. Some of the enhancement algorithms are gain compensation, multi-band blending and seam line detection, which require additional computational time. In consequence, our stitched image may not be as beautiful as some other panorama stitching, but it gives us an instant result suitable for disaster response teams.

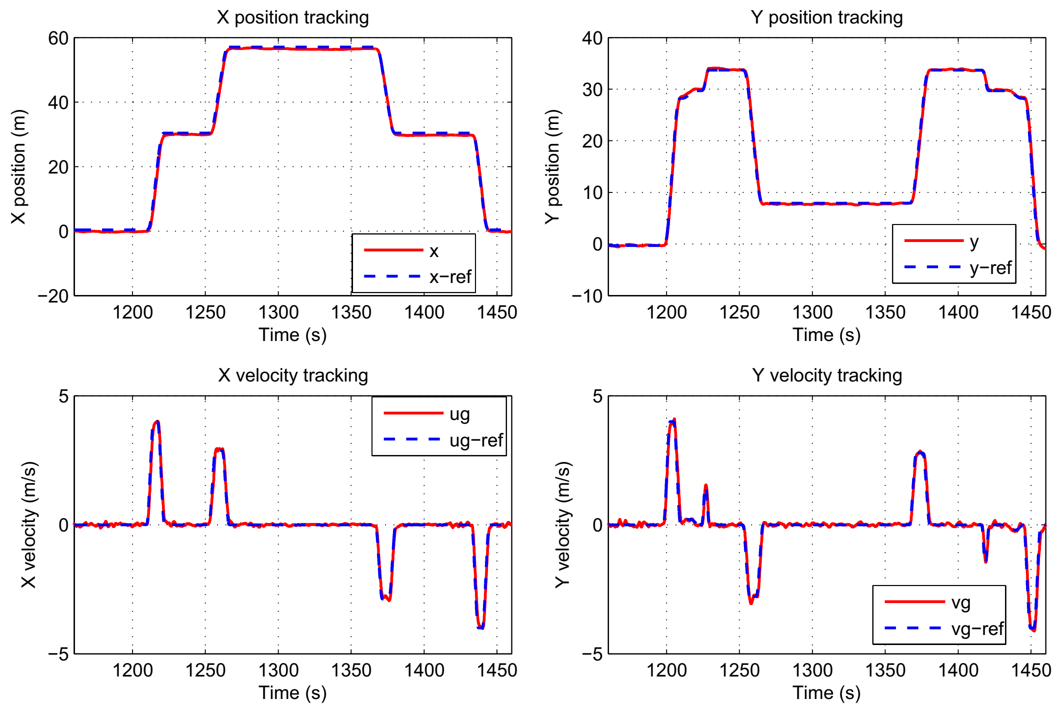


Fig. 5. Performance of the RPT controller for position and velocity tracking.

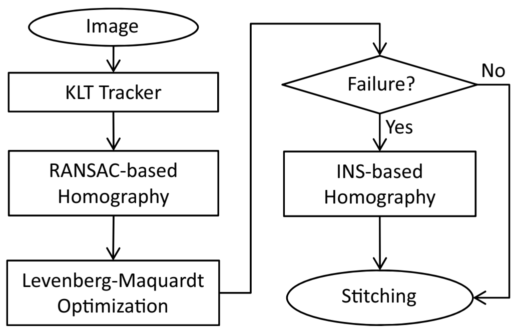


Fig. 6. Image stitching flowchart.

The working flow of the image stitching is shown in Fig. 6. The basic idea is to extract an incremental homography from each two consecutive images from two sets of associated features in the two images. The incremental homography transform is then accumulated to provide an absolute transform with respect to the canvas frame, which is the frame when the first image with sufficient features is initialized. The current image is then transformed to the reference canvas using the accumulated homography to produce an overall stitched image.

We first evaluate the performance of different feature detectors, descriptors and matchers with respect to the computational time. The results are listed in Table 2. The Kanade–Lucas–Tomasi (KLT) feature detection and tracking is chosen due to its acceptable performance with fast computation time. The KLT tracker uses optical flow tracking that is calculated over different Gaussian pyramids of the two images. It is proven to work well even in areas which seem homogeneous to human eyes such as that of grass patches and foliage areas. During our flight over the area of interest, we have taken over 1000 images and performed our stitching algorithm based on the collected images. The total time taken for stitching the map is 153 s, achieving an update rate of 5 Hz.

Panoramic stitching relies on the projective transformation between two sets of matched points from the two images, which represents the camera motion between the two time instants when the images are taken. The camera motion consists of rotation and translation which could be represented by the homography [14]. The homography transformation maps the pixel coordinates from one image onto another in 2D homogeneous coordinates $\mathbf{x}'_i = (x'_i, y'_i, 1)$ and $\mathbf{x}_i = (x_i, y_i, 1)$ such that,

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i, \tag{11}$$

where $\mathbf{H}$ is the homography matrix of size 3×3.

In an image set, it usually consists of many feature points that could be detected and tracked across different images. We have more feature points than needed to calculate the homography matrix but many of these feature points are noisy and could represent bad matches. As a result, we implement random sample consensus (RANSAC [15]) strategy with the large number of feature points. The criteria of determining whether two points are inliers is the reprojection error which is defined as

$$\text{Reprojection Error} = \|\mathbf{x}_i{}' - \mathbf{H} \times \mathbf{x}_i\|. \tag{12}$$

In the RANSAC implementation, random sets of four corresponding points are chosen to estimate the homography matrix using a simple least-squares algorithm. With the homography matrix estimate, we then compute the inlier ratio of the computed homography based on the reprojection error threshold shown in Eq. (12). The eventual best subset is then used to produce the initial estimate of the homography matrix with its set of inliers. Finally, the computed homography is refined further with the Levenberg–Marquardt method [16] to further reduce the re-projection error.

The RANSAC-based homography is still prone to errors due to image noises or too large motion for the KLT tracker to manage. To achieve a robust image stitching algorithm, we have developed a failsafe mechanism by introducing the homography induced from the inertial navigation system (INS) as a complementary option.

The failure check pipeline is illustrated in Algorithm 1. Two parameters are evaluated to define whether the RANSAC-based homography is valid. The first parameter is the difference of image size between the current transformed image and the last one. We allow an image size change of $\pm 20\%$ due to the skewing of the image and also any enlargement or shrinkage that should occur due to slight height differences while the MAV was flying.

Secondly, we performed a check on the overall translation of the image as compared to the previous image. This was done by calculating the centroid of the image that has been projectively transformed by the calculated homography matrix. As we run our algorithm at 5 Hz frequency, we expect the translation of the image to be very small. Therefore, we allow the translation to be less than half the diagonal distance of the original image. If one of the two failure check fails, an interim homography matrix will be calculated and used. This interim homography matrix is calculated from the onboard INS which encompasses the Euler angles as well as the GPS coordinates.

During the IMAV 2014 competition, we are required to fly over the military village of Oostdorp, the Netherlands.

Table 2. Comparison of detectors and matchers.

Detector	Descriptor	Matcher	Time
FAST	BRIEF	Brute force	0.118 s
GFTT	Optical flow	Optical flow	0.153 s
SURF	SURF	FLANN	0.292 s

Algorithm 1 Homography Failure Checks

```
1:  procedure IMAGE SIZE CHECK
2:      $k = 1$ or $0 \leftarrow$ Check results
3:      $Vector(points) \leftarrow$ Projective Transform from H
4:      $Area\ ratio,\ A \leftarrow$ Area between $Vector(points)$
5:      if $|1 - A| \geq 0.2$ then
6:          $k \leftarrow 0$
7:      else
8:          $k \leftarrow 1$
9:  procedure IMAGE TRANSLATION CHECK
10:     $Centroid\ Diff,\ D \leftarrow$ Dist of centroid of $Vector(points)$
11:     if $|D| \geq 0.5 \times diag(img)$ then
12:         $k \leftarrow 0$
13:     else
14:         $k \leftarrow 1$
15: procedure FAILURE CHECK RECTIFICATION
16:     $H_{INS} \leftarrow$ INS states input
17:     if $k = 1$ then
18:         continue;
19:     else
20:         $H_{INS} \leftarrow$ INS-based Homography Calculation
```

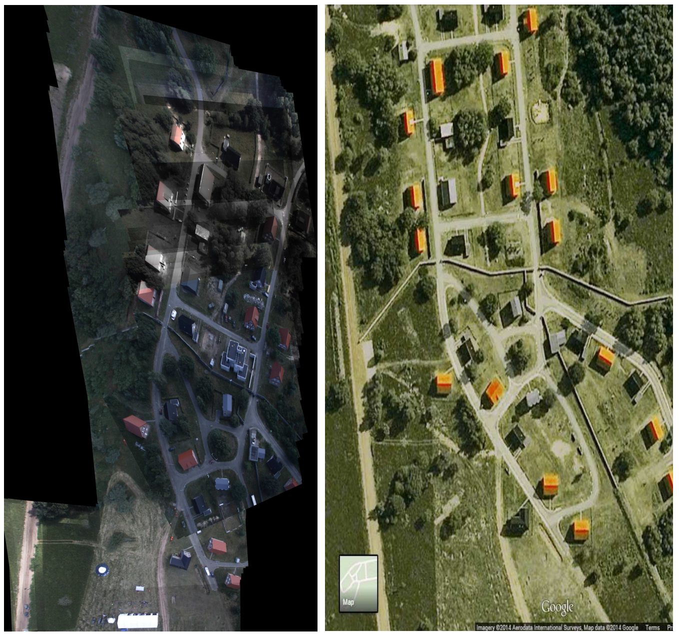

Map
Google
Imagery ©2014 Aerodata International Surveys, Map data ©2014 Google Terms

Fig. 7. Stitched image (left) compared with Google map view (right).

This was quite an undertaking as we are required to fly over an area where there are buildings and trees as high as 15 m. Our algorithm, with its robust RANSAC-based homography and the dedicated fail-safe checking mechanism, was able to reject those features that were detected on the buildings and trees and produced a stitched map shown in Fig. 7. Part of the requirement of the stitched map was to be able to allow the users to identify potential blockages or obstacles that might be a hindrance to rescue workers. Figure 7 shows we were able to detect roadblocks and obstacles in the stitched map very clearly.

3.3. *Indoor navigation*

Since GPS signal is unavailable in an indoor environment and no prior information about the indoor structure is given, this mission boils down to the problem of SLAM. Among the extensive published literature in SLAM, many theoretical works and practical implementations of SLAM are based on ground robots [17]. However, few of them have considered the computation limitation on MAVs, and they usually exploit the unlimited payload on ground robots or rely on high-bandwidth communication to the ground control station (GCS) where a powerful computer is running a computationally intensive algorithms. In consequence, some of these published works are limited in controlled lab environments with short-range and line-of-sight communication.

For real-life complicated scenarios such as this IMAV indoor mission, we have developed a more practical and robust navigation solution which only relies on two lightweight 2D Lidar sensors in the MAV platform. The navigation solution is implemented on an MAV with a tip-to-tip size of 0.76 m, which is sufficiently small to fly through windows and doorways (Fig. 8). The robustness and fast speed of the developed navigation algorithms are based on two innovative yet reasonable assumptions about the indoor environment listed as follows:

(1) In indoor environments, it is possible to extract sparse features from 2D Lidar measurments, such as corners and line segments.
(2) There exists a specific angle between each two non-parallel line features. Two line features can be orthogonal to each other or off-set by multiples of a constant angle displacement, such as 30° or 45°.

The above two assumptions are fulfilled for most modern indoor environments. In practice, even a small number of outliers will not affect too much on the estimation performance to jeopardize the navigation. The MAV pose in the map frame can be represented by its 3D position coordinates x, y, z and the heading angle ψ. We first divide them into two groups: namely the planar pose (x, y, ψ) and the altitude z. As described in Sec. 2, we have two Lidar sensors assembled onboard the MAV. We estimate the planar pose using the first horizontal Lidar and estimate the height with the second vertical Lidar.

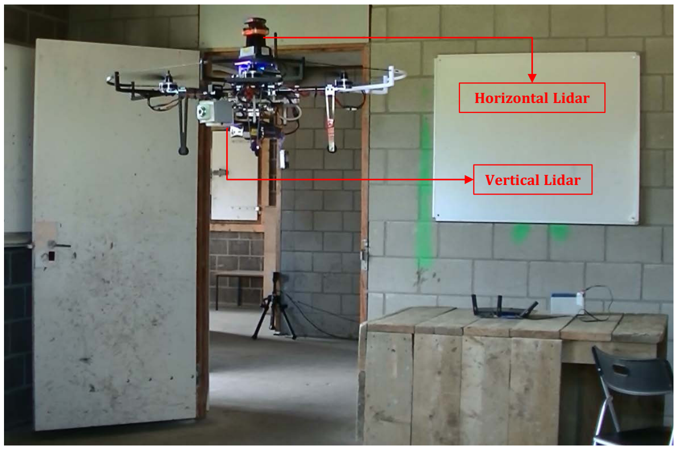


Fig. 8. MAV platform operating in Task C.

The planar localization algorithm via the first Lidar contains the fundamental ideas that make the whole navigation algorithm robust and efficient. With assumption 1, the conventional point cloud matching algorithm can be avoided, reducing the number of point matching pairs from thousands to dozens. With assumption 2, the estimation of rotation can be done by comparing the difference between line gradients instead of relying on point feature matching, making the estimation of the rotation motion decoupled from the translation motion. This decoupling feature is beneficial because the rotation motion usually lead to inconsistent point matching results, especially when the feature points are far away from the sensor center. The planar localization algorithm includes five steps, namely feature extraction, rotation tracking, point feature association, line feature association and position tracking.

The feature extraction process seeks to find the line and point features in the laser scans. Each scan is passed into a segmentation algorithm called *split-and-merge* [18] to generate a series of line segments. Figure 9 gives a graphical illustration of *split-and-merge*. After obtaining many clusters of points, we use the least-mean-square fitting to extract the line feature parameters. At the same time, the end points of the line segments are chosen to be the point features. Each line feature can be represented by two parameters, namely the line's normal direction α_k and its perpendicular distance to the center of laser scanner d_k, and each point feature can be represented by its 2D coordinates (see the bottom-right sub-figure of Fig. 9 for reference).

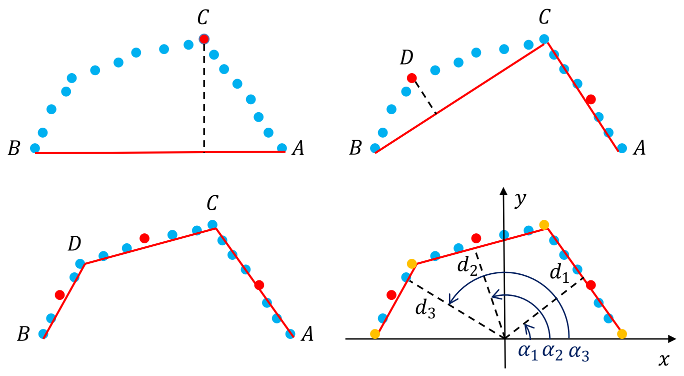


Fig. 9. (Color online) The *split-and-merge* and line extraction algorithm (line features in red; point features in orange).

With the line segments identified, we utilize assumption 2 in an innovative way to keep track of the robot's heading direction ψ. Without loss of generality, let the map frame x-axis align with one of the walls. Then all the walls will have their directions at $n\alpha$, where α is the constant angle displacement and n can be any integers. Choose one of the walls currently observable and let its direction be β_l in the laser scanner frame. Then we have this wall's direction β_m in the map frame as

$$\begin{aligned}\beta_m &= \psi_t + \beta_l \\ &= \psi_{t-1} + \Delta\psi_t + \beta_l \\ &= n_i\alpha,\end{aligned} \tag{13}$$

where ψ_t and ψ_{t-1} are the MAV headings in the current frame and previous frame, respectively, and $\Delta\psi_t$ is the inter-frame heading increment. Obviously, $(\psi_{t-1} + \Delta\psi_t + \beta_l)$ is divisible by α, which leads to

$$\Delta\psi_t = -[(\psi_{t-1} + \beta_l)\%\alpha], \tag{14}$$

where the operator $\%$ is defined as:

$$a\%b = \begin{cases} (a \bmod b), & (a \bmod b) \le b/2, \\ (a \bmod b) - b, & \text{otherwise.} \end{cases} \tag{15}$$

After obtaining $\Delta\psi_t$, the MAV heading can be updated as

$$\begin{aligned}\psi_t &= \psi_{t-1} + \Delta\psi_t \\ &= \psi_{t-1} - [(\psi_{t-1} + \beta_l)\%\alpha].\end{aligned} \tag{16}$$

According to (16), we can see that the MAV heading ψ_t is only related to the previous heading ψ_{t-1} and the line segment heading β_l. If we initialize the MAV heading to be zero at the program start, the heading estimate of the MAV using (16) is thus always absolute heading without drift. In practice, the longest line extracted for the current frame can be used for the heading alignment because it is the most reliable. However, it should be noted that this heading tracking algorithm only works when the MAV inter-frame rotational increment $\Delta\psi_t$ is less than $\alpha/2$. Fortunately, the 2D Lidar scans fast enough (40 Hz) to ensure the condition is met.

Once the MAV rotation motion has been resolved, the translation motion can be obtained using the extracted point and line features. The main idea in this step is to associate locally observed point and line features from one frame to the next so that the incremental planar displacement of the MAV is tracked. The data association is performed in the global heading frame by transforming the point and line features using the estimated MAV heading ψ_t. The point feature association uses the Euclidean distance error metric. For the line features, we partition globally transformed lines into two groups: the horizontal lines and the vertical lines. For each group of lines, they are compared exhaustively with those in the previous scan. If their respective d and α parameters are sufficiently close they are associated, and their difference in d represents the translation motion of the MAV in x-axis and y-axis, respectively.

The current position can be iteratively estimated based on the previous-frame position $[x_{t-1}, y_{t-1}]$ and an incremental change $[\Delta x_t, \Delta y_t]$:

$$[x_t, y_t] = [x_{t-1}, y_{t-1}] + [\Delta x_t, \Delta y_t], \tag{17}$$

where

$$\begin{bmatrix} \Delta x_t \\ \Delta y_t \end{bmatrix} = \frac{\sum w_{\mathrm{p}}(\mathbf{p}_t - \mathbf{p}_{t-1})}{\sum w_{\mathrm{p}}} + \begin{bmatrix} \sum w_{\mathrm{l},x}(d_{x,t} - d_{x,t-1}) \Big/ \sum w_{\mathrm{l},x} \\ \sum w_{\mathrm{l},y}(d_{y,t} - d_{y,t-1}) \Big/ \sum w_{\mathrm{l},y} \end{bmatrix}, \tag{18}$$

where $\mathbf{p}_t$ and $\mathbf{p}_{t-1}$ are the matched point features, w_{p}, $w_{\mathrm{l},x}$ and $w_{\mathrm{l},y}$ are the weights to tune the importance of point features and line features. Equation (18) can be seen as an weighted average of all the associated features' displacement. In practice, the points which are further away and the shorter lines are more prone to noises. Therefore, closer point features and longer line features are given larger weights.

For the MAV height measurement, a second Hokuyo URG-04LX Lidar is mounted vertically. Similar to the line extraction algorithm mentioned above, the same *split-and-merge* method can be applied. After filtering out those line segments with dissimilar gradients to the ground plane, the rest line segments are sorted by their perpendicular distances to the laser scanner center. The furthest line segments are kept, among which the longest one is believed to be the true ground. Finally, the MAV height can be calculated as the perpendicular distance of this line segment to the laser scanner center, compensated by the offset between the laser scanner and the MAV center of gravity as well as the MAV attitude angles.

In the actual competition, this customized SLAM algorithm was implemented onboard of the MAV. With only some waypoint to guide the MAV inside different rooms, the MAV successfully traveled to all the defined rooms using the state estimation presented in this section. Figure 10 shows the reconstructed map which is generated by projecting the laser scans on the poses estimated with the presented method.

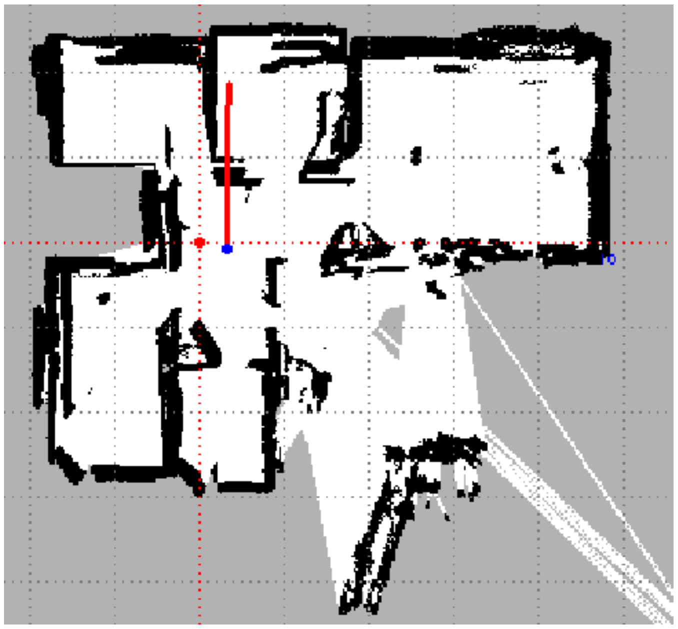


Fig. 10. Result of map reconstruction in the IMAV competition.

3.4. *Pose estimation with monocular camera*

Task D requires the MAV to land on the corner of a rooftop and observe the targeted area. In practice, it is not possible to land on the rooftop precisely using only normal GPS receiver. We developed a vision-based pose estimation algorithm to guide the MAV for precise landing. It is designed to extract the pose of the MAV with respect to a predefined planar marker board on the rooftop. The pose is extracted from a number of 3D-to-2D point correspondences [19]. The 3D points are the corners of the defined marker as shown in Fig. 11 and the 2D points are the corresponding image points of these corners. The marker is designed to

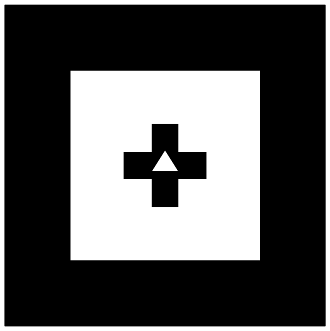


Fig. 11. Planar marker used for pose estimation.

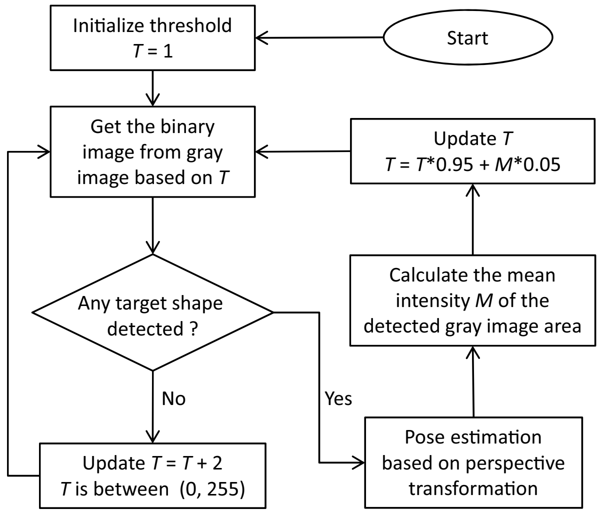

Start
Initialize threshold $T = 1$
Get the binary image from gray image based on T
Update T $T = T*0.95 + M*0.05$
Any target shape detected ?
No
Yes
Calculate the mean intensity M of the detected gray image area
Update $T = T + 2$ T is between (0, 255)
Pose estimation based on perspective transformation

Fig. 12. Vision processing pipeline for pose estimation.

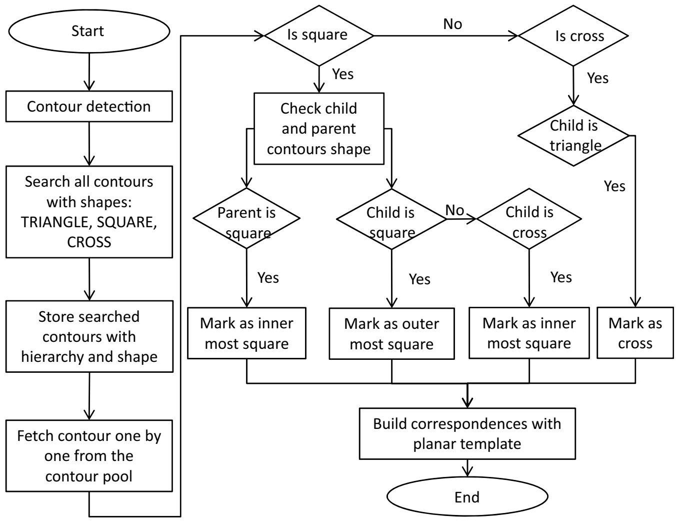

Start
Contour detection
Search all contours with shapes: TRIANGLE, SQUARE, CROSS
Store searched contours with hierarchy and shape
Fetch contour one by one from the contour pool
Is square
No
Is cross
Yes
Check child and parent contours shape
Child is triangle
Parent is square
Child is square
Child is cross
Mark as inner most square
Mark as outer most square
Mark as inner most square
Mark as cross
Build correspondences with planar template
End

Fig. 13. Detailed vision algorithm for target detection.

consist of two square contours, one inner cross contour and one triangle shape. The two square contours are for pose estimation in longer distances and the inner cross shape and the triangle shape are for pose estimation in shorter distances.

The whole image processing pipeline is shown in Fig. 12. The main idea is to binarize the image and produce a series of contours with shape and hierarchy information. To address the challenging illumination conditions in outdoor environments, the segmentation threshold T is searched between 0 and 255 before the marker shape is detected. Once the marker is detected, the threshold is adaptively changed using a low-pass filter combining the current working threshold and the average intensity of the detected marker area.

The detailed algorithm for the target detection is shown in Fig. 13. Contours are detected with hierarchy and shape information. The algorithm then sequentially searches the outer-loop square, inner-loop square and the cross shape. If one of the them can be found, the marker is assumed to be detected. The correspondences between the marker corners with known dimensions and the contour corners from the image can be built. With such correspondences information, the camera pose relative to the marker is extracted using the perspective transform algorithm, which is implemented by a built-in function "solvePnP" in OpenCV.

To verify the position estimation of the vision algorithm, experiments with a motion capture system (VICON) as the ground truth were conducted. The VICON system can provide precise position measurements in millimeter accuracy. It can be shown from Fig. 14 that the position estimation from the developed vision algorithm matched well with the measurement provided by VICON. The spikes shown in the figure are due to the blockage of camera in VICON system.

3.5. *Digit detection*

Digit panel detection and observation is another important mission element. A prerequisite for digit panel observation is to locate the area of the digit panel, i.e., the region of interest (ROI). This requires precise landing of the MAV at the predefined heading angle. However, even though with the vision-guided landing, the requirement is not certain to be met. Therefore, we install the forward-looking camera on a pan-tilt mechanism to expand the searching zone of the ROI. We also implement a strategy shown as in Fig. 15 to search for the ROI, either by panning and tilting the camera or by taking-off and landing again.

The digit panel is a 7-segment digit number in orange color on a black panel, which provides important information for detecting the ROI. The image is first converted to hue-saturation-value (HSV) color space and the algorithm will try to determine whether there are enough number of orange pixels in the image. Once enough orange pixels are confirmed, the current image is regarded as the correct frame. Based on the ROI, we further check if the ROI is near the border. This is indispensable as the digit may be falsely detected if the ROI is at the borders or only partially viewed. If the ROI is at the borders, the pan/tilt mechanism is activated to move the ROI into the center of the image. When the digit is detected within the current frame and keeps constant in the next 20 frames consecutively, the digit number is confirmed. After 30 s, if the digit changes (the digit's seven segments are controlled by seven servos to produce a new digit every 30 s), the ROI is determined. Otherwise, the ROI is considered to be falsely detected and has to be searched again with another threshold.

In practice, the image collected in the competition site is always prone to noises considering the complex

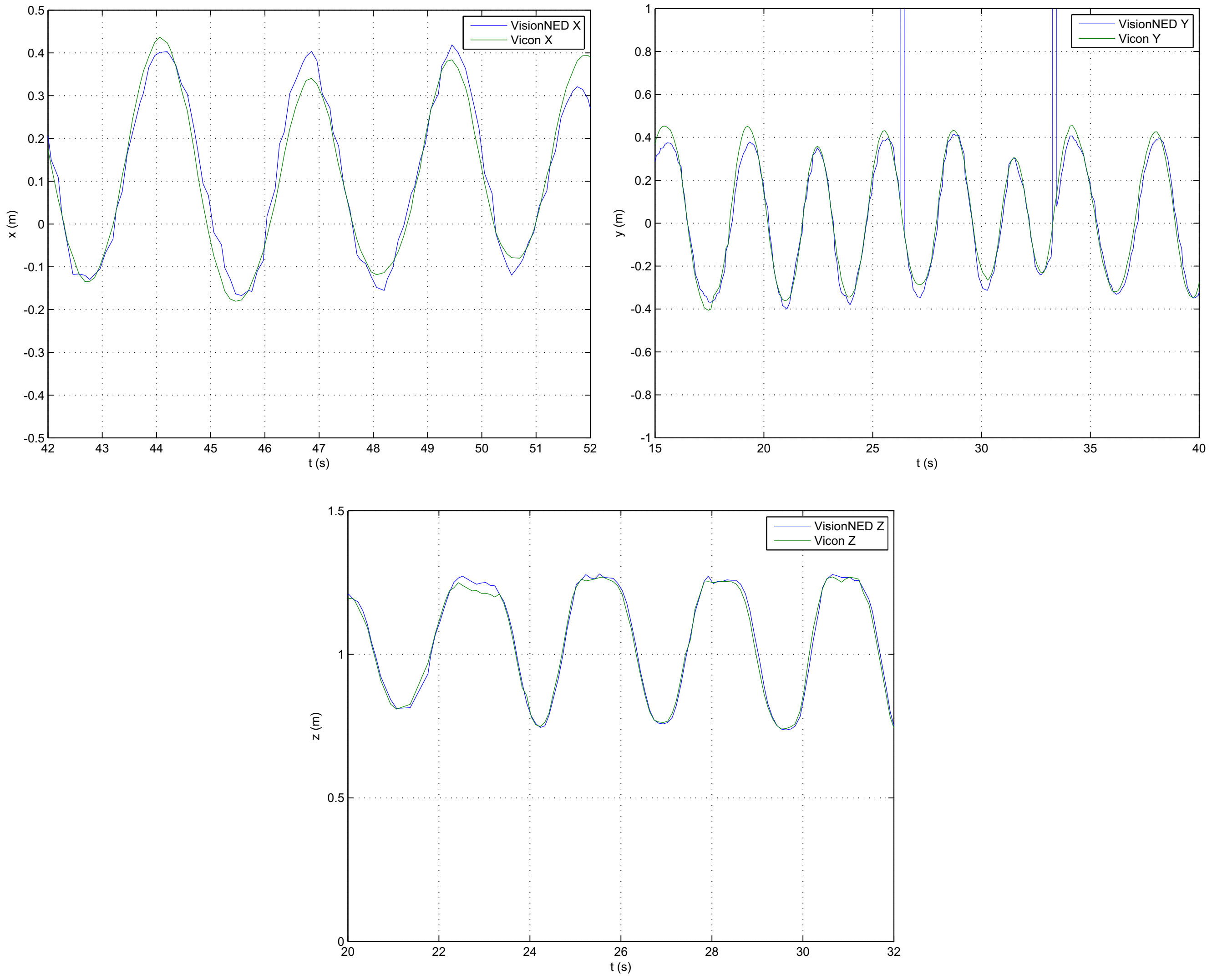


Fig. 14. Comparison of measurements between vision algorithm and VICON.

illumination conditions. The HSV segmentation will generate a binary image which consists not only the contours of the digit segments, but also other objects. With the digit size given, we apply several descriptors to validate the contours in the binary image, such as the area, the length-width ratio, and the relative topological relationship among the contours. Once the candidate contours are identified, we run a template matching on the binary image to recognize the digit.

The basic concept of the template matching is to calculate the similarity of a template patch and a patch in the sample image with the same area and find the patch location with the highest similarity. Several methods of calculation similarity have been provided in OpenCV libraries and the best method tested for this application is based on

$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y')I'(x+x',y+y'))}{\sqrt{\sum_{x',y'}T'(x',y')^2\sum_{x',y'}I'(x+x',y+y')^2}}, \quad (19)$$

where T and I indicate the values in the image pixel channels and (x',y') and (x,y) are the points in the template patch and starting location in the sample image, respectively.

Instead of feeding direct digit as the templates, the template patches are designed as four templates in Fig. 16. Each image needs to be tested with the four templates and obtain four similarity values. Table 3 shows the outputs of these combinations. This method is tested to be more

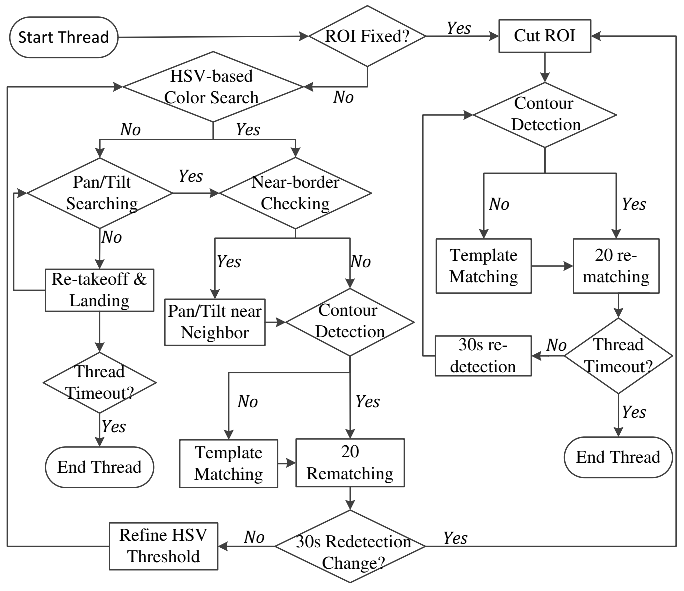
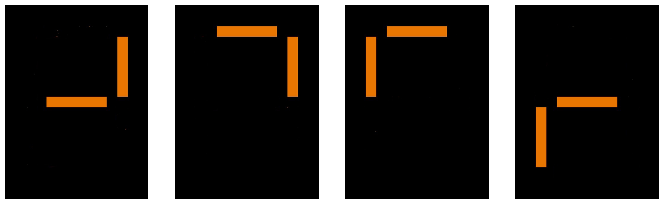


Fig. 15. Digit detection flowchart.



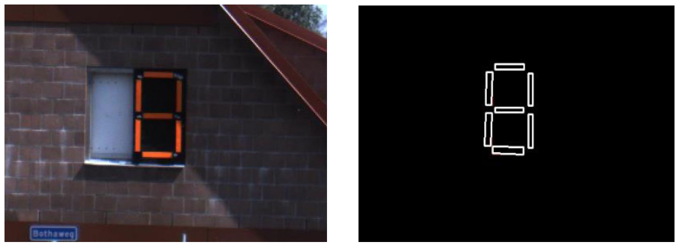
(a) A (b) B (c) C (d) D

Fig. 16. Matching templates.

robust and reliable compared to the direct digit template method, because this method relies on the composition of four template matching result while the direct method depends only on one template. Figure 17(a) shows one

Table 3. Matching results.

Digit value	A	B	C	D
Number 0	Low	High	High	Low
Number 1	Low	Low	Low	Low
Number 2	High	High	Low	High
Number 3	High	High	Low	Low
Number 4	High	Low	Low	Low
Number 5	Low	Low	High	Low
Number 6	Low	Low	High	High
Number 7	Low	High	Low	Low
Number 8	High	High	High	High
Number 9	High	High	High	Low



(a) Onboard image (b) Detected number

Fig. 17. Digit detected on the actual competition day.

patch of the onboard image recorded on the actual competition day. Figure 17(b) is the detected number with clear contours.

4. Conclusion

In this manuscript, we have presented our solution of using multiple MAVs for search and rescue in post-disaster situations. We have presented the system configuration, with the idea of sharing as many hardware and software resources as possible. The key technologies developed for the mission have been discussed, including real-time image stitching, indoor navigation, vision-based pose estimation and digit number recognition. All the presented techniques have been successfully demonstrated in IMAV 2014. A video footage describing the missions is available at `http://youtu.be/wNVOIqGKW3U`.

References

[1] I. Maza and A. Ollero, Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms, in *Distributed Autonomous Robotic Systems 6*, eds. R. Alami, R. Chatila and H. Asama (Springer Japan, 2007), pp. 221–230.

[2] C. Ezequiel, M. Cua and N. T. Libatique, UAV aerial imaging applications for post-disaster assessment, environmental management and infrastructure development, *Int. Conf. Unmanned Aircraft Systems (ICUAS)*, May 2014, pp. 274–283.

[3] G. Tuna, B. Nefzi and G. Conte, Unmanned aerial vehicle-aided communications system for disaster recovery, *J. Netw. Comput. Appl.* **41** (2014) 27–36.

[4] S. Siebert and J. Teizer, Mobile 3D mapping for surveying earthwork projects using an Unmanned Aerial Vehicle (UAV) system, *Autom. Construct.* **41** (2014) 1–14.

[5] D. W. Casbeer, D. B. Kingston, R. W. Beard and T. W. McLain, Cooperative forest fire surveillance using a team of small unmanned air vehicles, *Int. J. Syst. Sci.* **37**(6) (2006) 351–360.

[6] X. Dong, B. Yu, Z. Shi and Y. Zhong, Time-varying formation control for unmanned aerial vehicles: Theories and applications, *IEEE Trans. Control Syst. Technol.* **23** (2015) 340–348.

[7] F. Wang, J. Q. Cui, S. K. Phang, B. M. Chen and T. H. Lee, A monocamera and scanning laser ranger finder based UAV indoor navigation system, *2013 Int. Conf. Unmanned Aircraft Systems*, Atlanta, US (2013), pp. 693–700.

[8] C. K. Pang, T. S. Ng, F. Lewis and T. H. Lee, Managing complex mechatronics R&D: A systems design approach, *IEEE Trans. Syst., Man Cybern. A, Syst. Hum.* **42** (2012) 57–67.

[9] L. Liu, Robust cooperative output regulation problem for non-linear multi-agent systems, *Control Theor. Appl., IET* **6** (2012) 2142–2148.

[10] S. K. Phang, K. Li, K. H. Yu, B. M. Chen and T. H. Lee, Systematic design and implementation of a micro unmanned quadrotor system, *Unmanned Syst.* **2**(2) (2014) 121–141.

[11] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer and M. Pollefeys, Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision, *Auton. Robots* **5**(1–2) (2012) 21–39.

[12] B. M. Chen, *Robust and* H∞ *Control* (Springer, New York, 2000).

[13] D. Mellinger and V. Kumar, Minimum snap trajectory generation and control for quadrotors, *2011 IEEE Int. Conf. Robotics and Automation (ICRA)*, Shanghai, China (2011), pp. 2520–2525.

[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision* (Cambridge University Press, 2003).

[15] M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**(6) (1981) 381–395.

[16] J. J. Moré, The levenberg-marquardt algorithm: Implementation and theory, *Numerical Analysis* (Springer, 1978), pp. 105–116.

[17] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg and S. Thrun, 6D SLAM with an Application in Autonomous Mine Mapping, in *Proceedings IEEE 2004 International Conference Robotics and Automation*, New Orleans, USA (2014), pp. 1998–2003.

[18] G. Borges and M. J. Aldon, A split-and-merge segmentation algorithm for line extraction in 2D range images, *15th International Conference on Pattern Recognition*, Barcelona, Spain (2000), Vol. 1, pp. 441–444.

[19] G. Schweighofer and A. Pinz, Robust pose estimation from a planar target, *IEEE Trans. Pattern Anal. Mach. Intell.* **28** (2006) 2024–2030.