Drones for Cooperative Search and Rescue in Post-Disaster Situation

Jin Q. Cui[1], Swee King Phang[2], Kevin Z.Y. Ang[1], Fei Wang[1], Xiangxu Dong[1], Yijie Ke[2]
Shupeng Lai[3], Kun Li[2], Xiang Li[2], Feng Lin[1], Jing Lin[2], Peidong Liu[2]
Tao Pang[1], Biao Wang[4], Kangli Wang[2], Zhaolin Yang[1], Ben M. Chen[2]

***Abstract*— In this work, we report our solutions to the problems given in the 2014 International Micro Aerial Vehicle Competition, held in Delft, the Netherlands, August 2014, which involves using micro air vehicles in urban post-disaster search and rescue missions. Solutions to all key mission elements of the competition, including real-time map stitching, indoor navigation and roof-top perching, are documented and highlighted in this manuscript. The proposed solutions are successfully demonstrated in the competition and help us win the championship.**

I. Introduction

While the term drone is gaining more popularity in this modern world, many researchers around the globe start to discover more advanced applications for drones, or in other words, the micro air vehicles (MAVs). Due to its small size 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 major roles in many military and civilian applications, especially in aerial reconnaissance, search and rescue and post-disaster area exploration [2]. 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 apply various linear and non-linear 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 [3]. The design of MAV system resembles the design of other systems consisting of both mechanics and electronics modules [4].

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 mission using a swamp of MAVs. The competition has attracted many research teams from various countries across the world to submit their proposals to the mission requirements. Fourteen short listed finalists are then required to demonstrate the capabilities of their MAVs on site in Delft, the Netherlands.

Fig. 1. IMAV 2014 task assignment

The main objective of the competition is to simulate a search and rescue mission using MAVs in a post-disaster village. The competition mission is divided into four different elements (see Fig. 1). The first element of the competition involves a drone to inspect the targeted area by performing aerial photography and map stitching. The contestants will need to identify several possible routes without obstacles for the rescuers to enter the village from the stitched map. The second element of the competition requires a drone to search and identify each house along the main street of the village. The contestants will need to identify the number of survival and the corresponding house number via the feedback from the attached onboard camera. The third element requires the drone to navigate in a two-story building, and to identify objects, such as chairs and photo frames, in the building. The last mission element involves roof-top perching of a drone at a specific building, and then observes visually a display panel placed on the neighbor's house.

Our 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 [5]. This manuscript describes the key solutions proposed by our team to overcome the mission elements of this competition. These sophisticated solutions help us win the championship of the competition. The manuscript is divided into the following sections. Brief introduction is given in this section.

[1] J.Q. Cui, Z.Y.K. Ang, F. Wang, X.X. Dong, F. Lin, T. Pang and Z. Yang are with National University of Singapore (NUS), Singapore, cuijinqiang@nus.edu.sg

[2] S.K. Phang, Y. Ke, K. Li, X. Li, J. Lin, P. Liu, K. Wang and B.M. Chen are with Department of Electrical & Computer Engineering, National University of Singapore, Singapore. bmchen@nus.edu.sg

[3] S. Lai is with Graduate School for Integrative Sciences & Engineering, National University of Singapore. shupenglai@nus.edu.sg

[4] B. Wang is with Nanjing University of Aeronautics and Astronautics, Nanjing, China. wangbiao@nuaa.edu.cn

978-1-4673-7338-8/15/$31.00

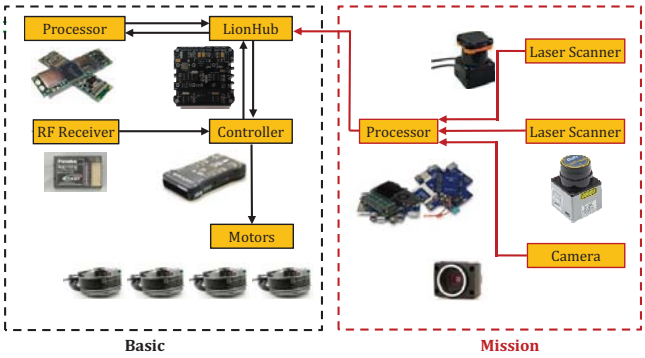

Fig. 2. The avionics system configuration for IMAV 2014 task C

Fig. 3. The assembled platform for task C

Section II presents the overall hardware and software configuration of the MAVs. Section III presents the key solutions developed for this competition, including real-time image stitching, number detection, indoor navigation and vision-based pose estimation. Concluding remarks are given in the last section.

II. System Configuration

The competition consists of four different tasks for the same mission. To accomplish the tasks in the allocated time, we have designed our MAV platforms based on a quadrotor with different configurations of the avionics system. The first incentive is to share as many resources as possible while meeting the different requirements. Both the hardware and software design follow this principle.

The platform is chosen to be of a quadrotor type because of its simple mechanical structure and stable control performance. The avionics system is designed to stabilize the attitude and translation dynamics at the same time. All of the platforms share the same attitude controller 'Pixhawk' and use a Gumstix Overo Fire to implement algorithms like automatic control, sensor fusion, servo driving, and so on. To meet the different mission elements' requirements, another powerful computer Mastermind is assembled to interface with various sensors and to implement those computationally intensive algorithms such as simultaneous localization and mapping (SLAM), path planning, and vision processing algorithms. Fig. 2 depicts the structure of avionics system for task C and Fig. 3 is the fully assembled platform. Different avionic modules are used to meet various mission elements, which are summarized by Table I. An noteworthy point is that we use a fifth MAV to carry a WiFi router to provide wireless relay for the other four MAVs.

TABLE I
Platform Configuration List

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

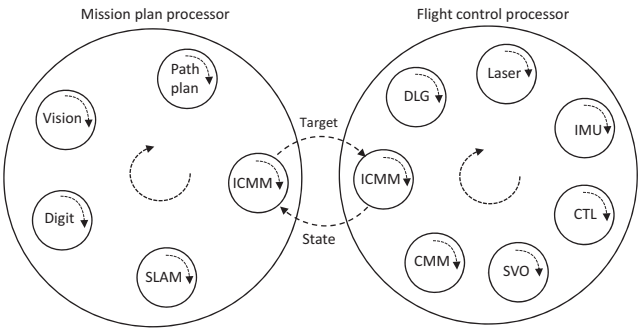

Fig. 4. Software structure of MAV navigation system

According to this hardware configuration, the software system is implemented in different threads allocated in two computers: the gumstix Overo Fire and the Mastermind. They are labeled as *Flight control processor* and *Mission plan processor* in Fig. 4 respectively. Since the Mastermind processor 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. 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 System (GCS) for user monitoring and 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.

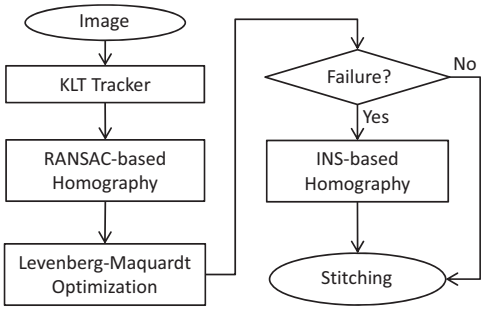

Fig. 5. Image stitching flow chart

III. 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 software structure, different algorithms are developed, including real-time image stiching, indoor navigation, digit-detection and vision-based pose estimation.

A. Fast Onboard Image Stitching

In order to provide fast evaluation of the surveyed area, we need a high resolution stitched image the instant we have the 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 use by disaster response teams.

The whole working flow of image stitching is shown in Fig. 5. The basic idea is to transform the current image using the homography extracted between each two consecutive images and transform the current image to a reference canvas using the accumulated homography. The homography is extracted by pairs of corresponding feature points between the two images.

We first evaluate the performance of different feature detectors, descriptors and matchers with respect to the computational time. 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 that seem homogeneous to human eyes such as that of grass patches and also foliage areas. During our flight over the area of interest, we have taken over 1000 images and performed our stitching algorithm based on this number of images. The total time taken for stitching our map is 153 seconds, achieving an update rate of 5 Hz.

Panoramic stitching relies on the projective transformation between two sets of matched points from two images, which represents the camera motion between the two images. The camera motion consists of rotation as well as translation and could be represented by the Homography [6]. 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,

$$x_i' = H x_i \,, \tag{1}$$

where H is the Homography matrix of size 3×3.

In an image set, it usually consists of many features 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 such, we implement random sample consensus (RANSAC [7]) strategy with the large number of feature points. The criteria of determining whether two points are inliers is the re-projection error defined in homogeneous coordinates. The re-projection error is defined as,

$$\text{Reprojection Error} = \left\| x_i' - H \times x_i \right\| . \tag{2}$$

Finally, the computed homography is refined further with the Levenberg-Marquardt method [8] to further reduce the re-projection error.

The RANSAC-based homography is still prone to errors due to the noisy image or too large motion for the KLT tracker to manage. To make a robust image stitching algorithm, we have developed a failsafe mechanism, 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 shrinking that should occur due to slight height differences while our MAV system 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 IMU states which

TABLE II
Comparison of detectors & 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

procedure IMAGE SIZE CHECK
  $k = 1$ or $0 \leftarrow$ Check results
  Vector(points) $\leftarrow$ Projective Transform from H
  Area ratio, A $\leftarrow$ Area between *Vector(points)*
  if $|1 - A| \geq 0.2$ **then**
    $k \leftarrow 0$
  else
    $k \leftarrow 1$
procedure IMAGE TRANSLATION CHECK
  Centroid Diff, D $\leftarrow$ Dist of centroid of *Vector(points)*
  if $|D| \geq 0.5 \times diag(img)$ **then**
    $k \leftarrow 0$
  else
    $k \leftarrow 1$
procedure FAILURE CHECK RECTIFICATION
  $H_{INS} \leftarrow$ INS states input
  if $k = 1$ **then**
    continue;
  else
    $H_{INS} \leftarrow$ INS-based Homography Calculation

encompasses the euler angles as well as the GPS coordinates. Otherwise, the image stitching continues with the RANSAC-based homography.

During the IMAV 2014 competition, we are required to fly over the military village of Oostdorp, the Netherlands. This was quite an undertaking as we are required to fly over an area where there are buildings and trees as high as 15 meters in height. Our algorithm, with its robust RANSAC-based homography and the dedicated fail-safe check, was able to reject those features that were detected on the buildings and trees and produce a stitched map shown in Fig. 6. We were able to produce the stitched map immediately after our MAV landed and thus enabled our team to obtain really high scores.

B. 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 famous problem of SLAM. Many theoretical works and practical implementations of SLAM on ground robots [9], and on MAV platforms [10] have been published in literature. 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 algorithm. In consequence, some of them only work in controlled lab environments with short 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 light-weight 2D Lidar sensors in the MAV platform. The size of the MAV is sufficiently small to fly through windows and doorways with a width of 0.9 meter (Fig. 7). In addition,

Fig. 6. Stitched image (left) compared with Google map view (right)

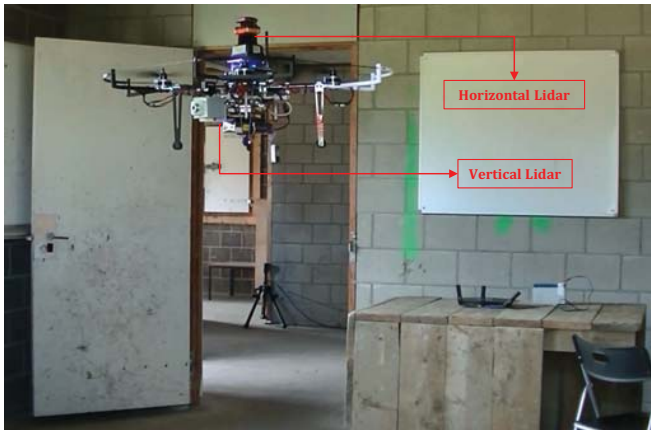

Fig. 7. MAV platform operating in IMAV Task C mission

in order to realize a robust and real-time SLAM algorithm, two innovative yet reasonable assumptions about the indoor environment are made:

1) The line features are orthogonal to each other or off-set by multiples of a constant angle displacement, such as 30° or 45°.
2) The environment can be described by sparse features, which include corners and straight line segments.

The above two assumptions are fulfilled for most of 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. We estimate the planar pose the first horizontal Lidar and estimate the height with the second vertical Lidar.

The planar localization algorithm via the first Lidar con-

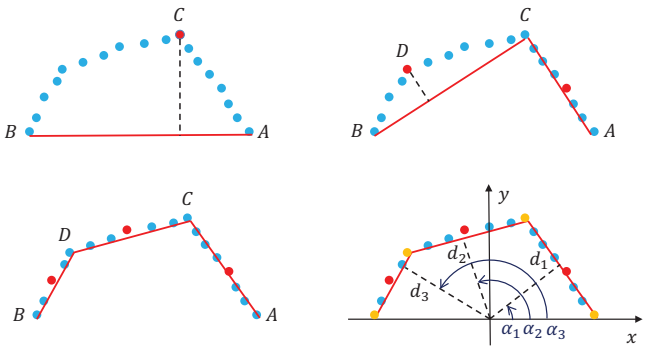

Fig. 8. The *split-and-merge* and line extraction algorithm (line features in red; point features in orange).

tains 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 rotational motion can be done by comparing the difference between line gradients instead of relying on point feature matching, thus making the estimation of rotational motion decoupled from translational motion. This decoupling feature is very beneficial because rotational motion usually results in inconsistent point matching results, especially when the feature points are far away from the sensor source. The planar localization algorithm include five steps, namely feature extraction, rotation tracking, point feature association, line feature association and position tracking.

The feature extraction process seeks to find line and point features in the laser scans. Each scan is passed into a segmentation algorithm called *split-and-merge* [11] to generate a series of line segments. Fig. 8 gives a graphical illustration of *split-and-merge*. After obtaining the clusters of points, we use 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. 8 for reference).

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

$$\begin{aligned} \beta_m &= \psi_t + \beta_l \\ &= \psi_{t-1} + \Delta\psi_t + \beta_l \\ &= n_i\alpha, \end{aligned} \tag{3}$$

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

$$\Delta\psi_t = -[(\psi_{t-1} + \beta_l)\,\%\,\alpha], \tag{4}$$

where the operator % is defined as:

$$a\%b = \begin{cases} (a \mod b) & , \text{ if } (a \mod b) \le b/2 \\ (a \mod b) - b & , \text{ otherwise.} \end{cases} \tag{5}$$

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

$$\begin{aligned} \psi_t &= \psi_{t-1} + \Delta\psi_t \\ &= \psi_{t-1} - [(\psi_{t-1} + \beta_l)\,\%\,\alpha]. \end{aligned} \tag{6}$$

According to (6), 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 (6) 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 rotational 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 trackable. 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 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 represent the translational motion of the MAV in either x-axis or 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 $[\Delta x_t, \Delta y_t]$:

$$[x_t, y_t] = [x_{t-1}, y_{t-1}] + [\Delta x_t, \Delta y_t], \tag{7}$$

where

$$\begin{bmatrix} \Delta x_t \\ \Delta y_t \end{bmatrix} = \frac{\sum w_{\mathrm{p}}(\mathbf{p}_t - \mathbf{p}_{t-1})}{\sum w_{\mathrm{p}}} + \begin{bmatrix} \sum w_{\mathrm{l},x}(d_{x,t} - d_{x,t-1}) / \sum w_{\mathrm{l},x} \\ \sum w_{\mathrm{l},y}(d_{y,t} - d_{y,t-1}) / \sum w_{\mathrm{l},y} \end{bmatrix}, \tag{8}$$

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 (8) can be seen as an weighted average of all the associated features' displacement. In practice, the points which are further away and the

Fig. 9. Result of map reconstruction in the IMAV competition fly-off

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 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 (CG) 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. Fig. 9 shows the reconstructed map which is generated by projecting the laser scans on the poses estimated with the presented method.

C. Pose Estimation with Monocular Camera

It is not practical to land on the rooftop using only GPS measurement, thus 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 [12]. The 3D points are the corners of the defined marker as shown in Fig. 10 and the 2D points are the corresponding image points of these corners. The marker is designed to consist of two square contours, one inner cross contour and one triangle shape. The two square contours are

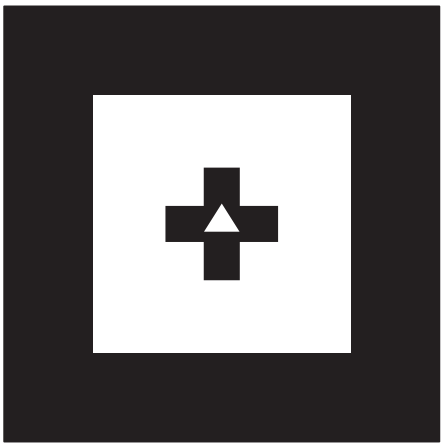

Fig. 10. Planar marker used for pose estimation

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. 11. 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. 12. 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. 13 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.

D. 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

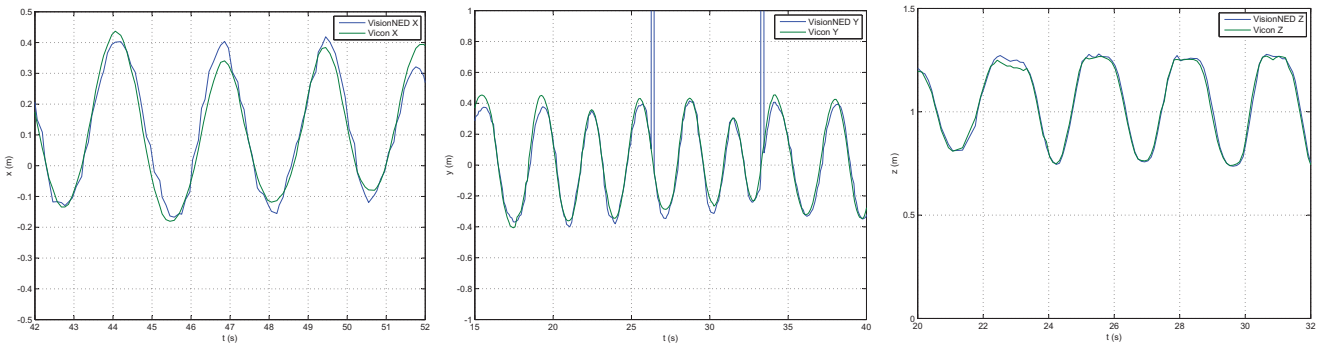

Fig. 13. Comparison of measurements between vision algorithm and VICON

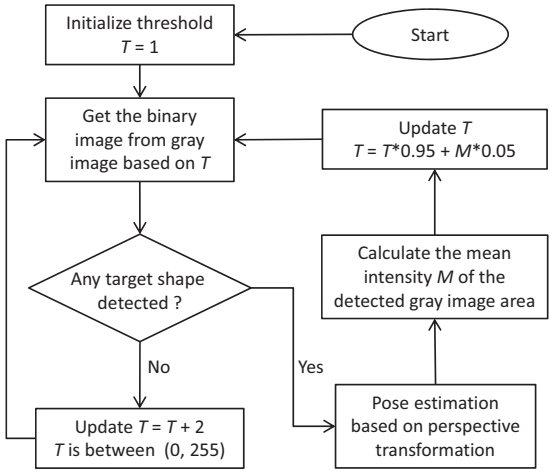

Fig. 11. Vision processing pipeline for pose estimation

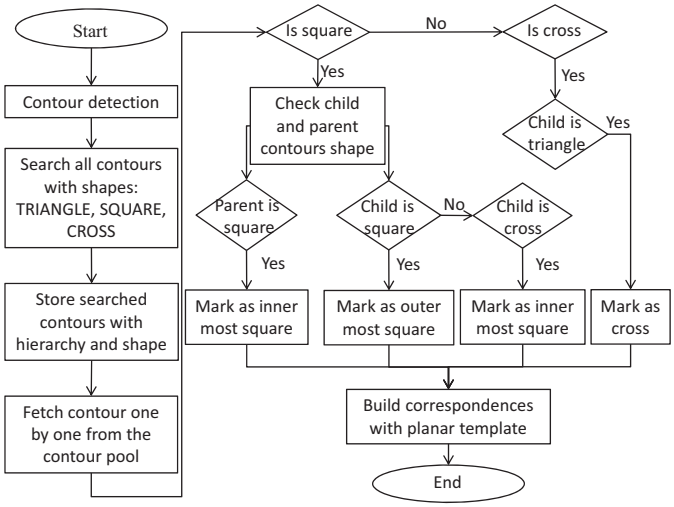

Fig. 12. Detailed vision algorithm for target detection

pan-tilt mechanism to expand the searching zone of the ROI. We also implement a strategy shown as in Fig. 14 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

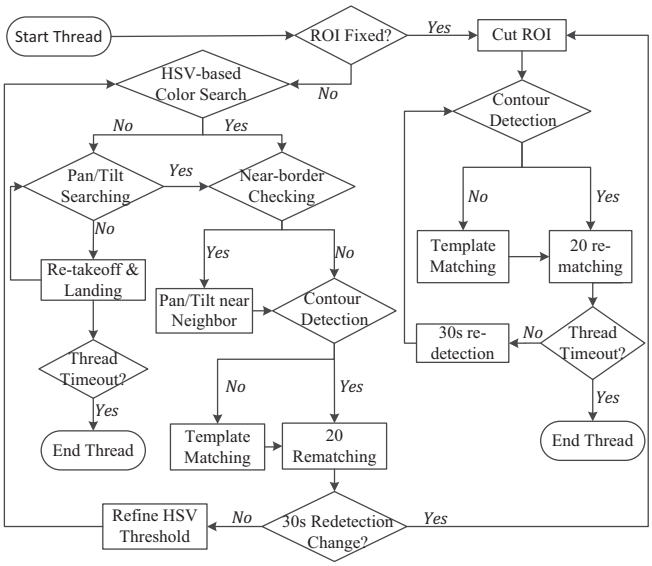

Fig. 14. Digit detection flowchart

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 there are enough corresponding orange pixels in the image, it 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 seconds, if the digit changes (the digit's 7 segments are controlled by 7 servos to produce a new digit every 30 seconds), the ROI is determined. Otherwise, the ROI is considered to be falsely detected and has to be re-search with another threshold.

In practice, the image collected onboard in the competition site is always prone to noises considering the complex illumination conditions. The HSV segmentation will generate a binary image which consists not only the contours of the

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

Fig. 15. Matching templates

TABLE III
MATCHING RESULTS WITH 4 TEMPLATES METHOD

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

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

$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y')I'(x+x',y+y'))}{\sqrt{\sum_{x',y'}T'(x',y')^2\sum_{x',y'}I'(x+x',y+y')^2}}, \qquad (9)$$

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. 15. Each image needs to be tested with the 4 templates and obtain 4 similarity values. Table. III shows the outputs of these combinations. This method is tested to be more 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. Fig. 16(a) shows one patch of the onboard image recorded on the actual competition day. Fig. 16(b) is the detected number with clear contours.

(a) Onboard Image (b) Detected Number

Fig. 16. Digit detection on the actual competition day

IV. CONCLUSION

In this manuscript we have presented our solution of using multiple MAVs for cooperative search and rescue in post-disaster situations. We have first 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 and help the team won the championship of the competition. A video footage describing the missions is available at `http://youtu.be/wNV0IqGKW3U`.

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*, R. Alami, R. Chatila, and H. Asama, Eds. 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," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, May 2014, pp. 274–283.

[3] F. Wang, J. Cui, S. K. Phang, B. Chen, and T. Lee, "A monocamera and scanning laser range finder based uav indoor navigation system," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, May 2013, pp. 694–701.

[4] C. K. Pang, T. S. Ng, F. Lewis, and T. H. Lee, "Managing Complex Mechatronics R&D: A Systems Design Approach," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 42, no. 1, pp. 57–67, Jan. 2012.

[5] L. Liu, "Robust cooperative output regulation problem for non-linear multi-agent systems," *Control Theory Applications, IET*, vol. 6, no. 13, pp. 2142–2148, Sept. 2012.

[6] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[7] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[8] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis*. Springer, 1978, pp. 105–116.

[9] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun, "6D SLAM with an application in autonomous mine mapping," in *IEEE International Conference on Robotics and Automation*, 2004.

[10] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 2878–2883.

[11] G. Borges and M. J. Aldon, "A split-and-merge segmentation algorithm for line extraction in 2D range images," in *15th International Conference on Pattern Recognition*, 2000.

[12] G. Schweighofer and A. Pinz, "Robust pose estimation from a planar target," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2024–2030, Dec 2006.