Multiple UAV Coordination based on the Internet of Things for Real-time Surface Inspection

Van Truong Hoang, *Student Member, IEEE,* Manh Duong Phung, *Member, IEEE,*
Tran Hiep Dinh, *Student Member, IEEE,* and Quang Ha, *Senior Member, IEEE*

***Abstract*—This paper presents a real-time system for surface inspection which is essential in structural health monitoring. The system uses multiple unmanned aerial vehicles (UAVs) coordinated in a specific formation to collect data of the inspecting objects and the Internet of Things (IoT) as the communication platform to transmit data. The UAV formation is obtained via using the angle-encoded particle swarm optimisation to create the inspecting path and redistribute it to each UAV. The communication is implemented by installing an IoT board to each component of the system to equip them with network and data processing capabilities. Through the network, collecting data is transmitting in real time to remote computational units where an enhanced thresholding image processing technique is implemented to detect defects/damages in real time. Various simulations, experiments and comparisons have been conducted to verify the validity and performance of the proposed system.**

***Index Terms*—UAV formation, Internet of things, surface inspection, structural health monitoring, particle swarm optimisation.**

I. Introduction

Continuous monitoring health conditions of built infrastructure is essential for their safe operation and long-term serviceability. Due to the nature deterioration of structures which reveals through the surface appearance, the foremost task to be conducted in monitoring is typically surface inspection. This task can be carried out by three main approaches including using foot patrol, ground vehicles, and flying-assisted robots. While the first two approaches are more common, the use of flying vehicles like UAVs is receiving increasing interest thanks to their flexibility in operating environment, versatility in task allocation and capability of conducting non-instructive inspection [1], [2]. For instance, UAVs have been used for periodical inspection of critical infrastructures such as oil-gas pipelines, wind turbine blades and power lines [3], [4]. They also have been employed to collect data for defect/damage recognition such as crack, rust or structural misalignment [5], [6], [7], [8].

To fulfil the surface inspection tasks, developments at various layers of the UAV system are often required. At the top layer, the UAV is programmed to fly closely to the inspected objects to collect data. The data is then post-processed to detect damages/defects via techniques such as photogrammetry [9], [10], Haar-like features extraction [3], self organizing map optimization [8] and dense structure-from-motion [11]. The middle layer relates to the generation of trajectories used to collect data. Here, classic path planning algorithms such as A-star, Dijkstra, rapidly-exploring random tree and probabilistic roadmap can be employed [12], [13], [14], [15]. However, algorithms specifically designed for UAVs such as the quasi-autonomous programming based algorithm [16] or iterative viewpoint resampling [17] are typically more efficient as they optimise the inspecting path considering the coverage among waypoints. At the lowest layer, low-level control algorithms are used to track the generated paths. Various techniques have been proposed such as backstepping techniques, PID control or sliding mode control [18], [19] with sufficient performance.

As using single UAVs has proved its applicability in inspection, recent works started to study the coordination of a group of UAVs to achieve higher time efficiency, superior performance and durable resilience [20], [21], [22]. Deng et al. developed a multi-platform UAV system for power line inspection tasks [20]. The experiments show that the time to inspect 70 km of power line can be reduced to 3 hours compared to a week as with the traditional inspection method. In [21], a power network damage assessment system using multiple UAVs was presented allowing the minimization of operating time and cost. Other studies addressed different aspects of multiple UAV coordination such as formation control, collision avoidance, and data acquisition [23], [24], [25].

Through the literature, it is recognisable that UAV-based inspection systems require the synchronization in control, path planning and data processing not only between layers within single UAVs but also among them. For real-time inspection, extra communication channels are needed among UAVs, ground control stations (GCS), and other remote computational units (RCU) such as server computers or cloud computing systems to tackle the high demand of data fusion and processing. Those requirements pose the need for a homogeneous communication platform that allows all components of an inspection system to be integrated. The current use of radio transceiver modules with a pulse mode modulation 2.4GHz uplink and 2.4/5.8GHz downlink is insufficient as they can only form a private network among on-site devices [26]. Some studies suggested to deploy additional UAVs as communication relay stations to extend the network [27], [28]. Although this approach is suitable for tasks that cover a certain structure or area, it is ill-use for structures located in remote areas like bridges, power line or wind turbines which require hundreds-of-kilometre communication range. Several studies proposed

The authors are with School of Electrical and Data Engineering, Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney (UTS), 81 Broadway, Ultimo NSW 2007, Australia. {VanTruong.Hoang, ManhDuong.Phung, TranHiep.Dinh, Quang.Ha}@uts.edu.au.

This work was supported by a UTS FEIT 2017 Data Arena research grant.
Manuscript received April 19, 2005; revised August 26, 2015.

to use satellite communication to overcome this problem [20], [29]. It is however expensive, complex and not always feasible.

Our approach in this study is using the Internet of Things (IoT) as the communication platform to interconnect UAVs and devices in order to efficiently perform inspection tasks. Based on this platform, a robust low-lever controller is introduced for each UAV to track the inspection path under harsh working conditions. The high-level control is then developed to coordinate multiple UAVs via formation and path-planning modules. The formation configuration is decided based on the results of mission assessment and defect detection requirements. A multi-objective particle swarm optimisation algorithm using angle-encoded PSO (θ-PSO) is exploited to generate an optimal path for the centroid of the group. This path is finally translated into individual trajectories for UAVs to follow. The images captured during the flight are then streamed to remote RCU where they are processed to detect defects. A detection algorithm based on thresholding analysis is proposed for the image processing tasks. Extensive simulations and experiments have been conducted in real and practical scenarios to evaluate the validity and feasibility of the proposed approach.

The remaining structure of the paper is organised as follows. Section II presents the overview and data communication structure of the system. Section III introduces the control and data acquisition processes. The image processing algorithm for defect detection is illustrated in section IV. Section V presents the experimental results. The paper ends with discussion and conclusion presented in sections VI and VII.

II. System Overview and Data Communication

Figure 1 presents an overview of the proposed inspection system. Its core includes three UAVs equipped with vision sensors to collect data of the inspecting structure. The UAVs fly cooperatively and share data in real-time with GCS via a wireless communication gateway. The gateway relays the data via the Internet to RCU where the data will be processed to detect defects. The localisation for UAVs is conducted via GPS modules combined with inertial measurement units (IMUs) that monitor internal states of UAVs. Depending on the inspection task, real-time kinematic (RTK) GPS modules will be used to improve the positioning accuracy. The operation of each UAV is monitored via its accompanied remote controller which will take over the control in emergency situations.

In the system, the communication among UAVs, GCS and RCU are carried out via the IoT. The idea is to equip each device of the inspection system with processing and network capabilities so that it can connect to the Internet to communicate with others via common protocols such as the transmission control protocol (TCP) and user datagram protocol (UDP). In our system, we used the IoT boards based on the the ATmega32u4 and Atheros AR9331 processors as shown in Fig. 1. These boards provide devices capabilities to not only connect to the wifi network but also process the receiving data via Atheros AR9331 processor with the Linux operating system installed. The ATmega32u4 enables the device to work with other low hardware interfaces such as voltage/current interface, AD/DA converter, PWM, etc. The boards thus turn normal devices into smart ones that can be integrated into the Internet protocol (IP) based networks. To obtain the Internet access, we utilised the mobile broadband networks through wireless gateway routers that have SIM card slots. As mobile broadband networks present almost everywhere, this approach allows the inspection system to be deployed for any structure without the need for relay stations nor satellite communication.

III. Control and Data Acquisition

Given the data communication framework, we propose an architecture for control and data acquisition consisting three layers: task assessment, high-level control and low-level control as shown in Fig. 2. Details are described as follows.

A. Task assessment

Given the task to be conducted, the task assessment layer determines requirements for data collection. For vision-based surface inspection, requirements typically include the area to be covered, the resolution of collected images and the overlapping between images sufficient for stitching and detecting defects. Let s_f be the resolution required to distinguish the smallest features. The field of view of the camera is computed as:

$$a_{fov} = \frac{1}{2} r_c s_f, \tag{1}$$

where r_c is the camera resolution. Thus, the distance $d_{f,i}$ from the ith UAV to the inspecting surface can be found as:

$$d_{f,i} = \frac{a_{fov} f}{s_s}, \tag{2}$$

where f and s_s are respectively the focal length and sensor size of the camera. That information is fed to the high-level control layer to determine the formation configuration for UAVs.

B. High level control

The high-level control layer consists of two modules, path planning and formation control. The path planning generates the inspection path that fulfils requirements computed from the task assessment. The formation control bases on the generated path and target shape to compute trajectories for each UAV.

1) Path Planning: When producing a path for the desired motion of multiple UAVs in a group, a number of constraints are required to maintain the formation, UAV maneuverability, operating space, and obstacle avoidance. In this work, all constraints will be incorporated into a multi-objective function. The path planning problem then can be simplified to creating a feasible path for the formation centroid. Since our goal is to construct optimal paths for all UAVs in the group, it is essential to speed up the convergence of the optimization process for the whole formation. Therefore, we propose to use the angle-coded PSO (θ-PSO) [30].

In θ-PSO, a set of particles is generated, each seeks for the optimum solution by moving in a way that compromises between its own experience and the social experience. Initially, each particle is assigned a random position, x_i. This position

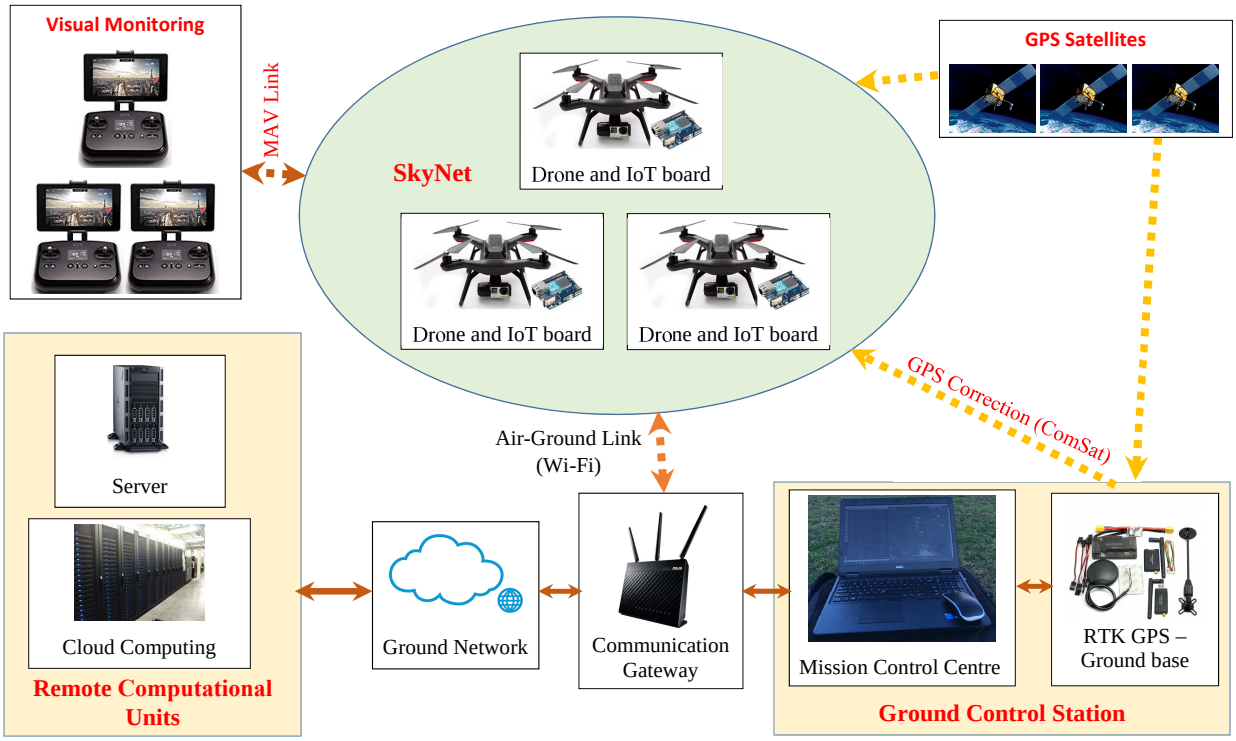


Fig. 1: Data communication structure

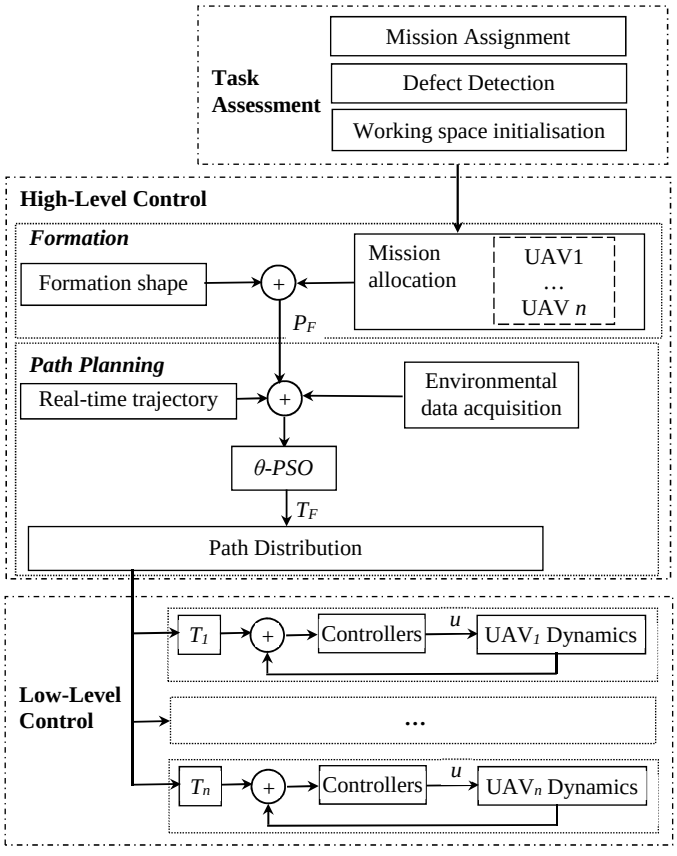


Fig. 2: System architecture

is represented by a phase angle θ_i of the UAV. The particle motion is then updated by the following equations:

$$\begin{cases} \Delta\theta_{ij}^{k+1} = w\Delta\theta_{ij}^{k} + c_1 r_{1i}^{k}(\lambda_{ij}^{k} - \theta_{ij}^{k}) + c_2 r_{1i}^{k}(\lambda_{gj}^{k} - \theta_{ij}^{k}) \\ \theta_{ij}^{k+1} = \theta_{ij}^{k} + \Delta\theta_{ij}^{k+1}, (i = 1, 2, ..., N; j = 1, 2, ..., S) \\ x_{ij}^{k} = \dfrac{1}{2}\left[(x_{max} - x_{min})sin\left(\theta_{ij}^{k}\right) + x_{max} + x_{min}\right], \end{cases} \tag{3}$$

where $\theta_{ij} \in [-\pi/2, \pi/2]$ and $\Delta\theta_{ij} \in [-\pi/2, \pi/2]$ are respectively the phase angle and phase angle increment of the ith particle in dimension j; $\lambda_g = [\lambda_{g1}, \lambda_{g2}, ..., \lambda_{gS}]$ and $\lambda_i = [\lambda_{i1}, \lambda_{i2}, ..., \lambda_{iS}]$ are respectively the global and personal best positions; N is the swarm size; S is dimension of the searching space; w is the inertial weight; r_1 and r_2 are two pseudorandom scalars; c_1 and c_2 are the gain coefficients; x_{max} and x_{min} are the upper and lower restrictions of the search space; and subscript k is the iteration index.

In Eq. (3), it can be seen that a solution of a particle is the three alternative options: to track its private trajectory $\Delta\theta_i$, to follow its best prior position λ_i, or to move toward the global best position λ_g. The correlation among them depends on three coefficients w, c_1 and c_2. The values of λ_i and λ_g are evaluated based on the cost function in the following form:

$$J_F(T_{Fi}) = \sum_{n=1}^{3} \beta_n J_n(T_{Fi}), \tag{4}$$

where T_{Fi} is the formation path; β_n is the weighting factor indicating the corresponding threat intensity; and $J_n(T_{Fi})$, $n = 1, 2, 3$, are the costs associated with the path length, collision violation and flying altitude, respectively. The cost

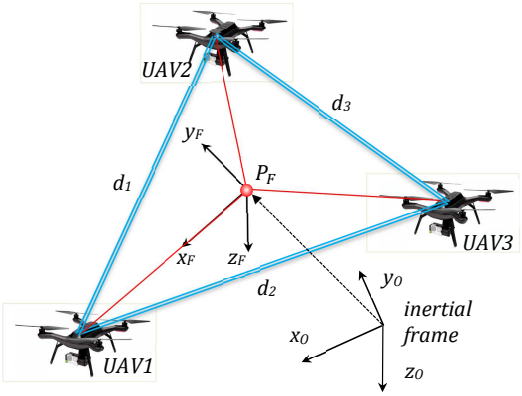


Fig. 3: Inertial and formation frames in UAV formation

Fig. 4: Mission Planner incorporating Google Satellite Map to create initial information and an inspection plan

function (4) forms by two major evaluations, the length and violation cost of the path. The former helps to minimize the total travelling distance of the path whereas the latter is to avoid collisions of UAVs with each other and with obstacles. In inspection, the UAVs also need to maintain certain distances to the surface as described in (2). Thus, the constraint in flying attitude is added to form the multi-objective cost function (4).

2) UAV Formation: Given the path generated by the θ-PSO for the formation centroid, it is necessary to produce a specific path for each UAV so that the shape of the formation during the flight can be maintained. Those paths can be computed based on the PSO's generated path and the desired relative distances among the UAVs.

Figure 3 shows the inertial and formation frames that represent a triangular UAV formation used in this study. All measurements are referred to the inertial frame O with axes x_O, y_O and z_O. Positions of UAV$_i$, $i = 1, 2, 3$, in the inertial frame are denoted as $P_i = \{x_i, y_i, z_i\}$. The formation frame, $\{x_F, y_F, z_F\}$, is defined such that the origin P_F is coincident with the centroid of the triangle. This allows to determine the centroid of the formation respected to the fixed inertial frame as:

$$P_F = \frac{1}{3}\sum_{i=1}^{3} P_i. \tag{5}$$

The rotation matrix which represents the relation between the formation and inertial frames is given by:

$$R_{OF} = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}, \tag{6}$$

where $s_x = sin(x)$, $c_x = cos(x)$, and ϕ, θ and ψ are Euler angles of the shape.

Let $P_{i,d} = [x_{i,d}, y_{i,d}, z_{i,d}]^T$ be the desired position for each UAV during the flight and $P_i = [x_i, y_i, z_i]^T$ be the actual position, P_i can be obtained from the GPS data of the UAV. We then define the relative position errors of the ith UAV during the flight in the inertial frame as:

$$\begin{bmatrix} e_{ix} \\ e_{iy} \\ e_{iz} \end{bmatrix} = \begin{bmatrix} x_{i,d} - x_i \\ y_{i,d} - y_i \\ z_{i,d} - z_i \end{bmatrix} \tag{7}$$

Using the rotation matrix in (6), with $R_{FO}(t) = R_{OF}^{-1}(t)$, the errors in (7) can be converted into the errors in the formation frame as:

$$\begin{bmatrix} e_{ixF} \\ e_{iyF} \\ e_{izF} \end{bmatrix} = R_{FO}(t) \begin{bmatrix} e_{ix} \\ e_{iy} \\ e_{iz} \end{bmatrix} \tag{8}$$

The customized path for each UAV then can be represented in term of trajectory control command as:

$$T_{iF} = T_r + \Delta T_i, \tag{9}$$

where T_r is the trajectory command of the centroid and ΔT_i is the amount added to direct the UAV away from the centroid. ΔT_i is calculated based on the desired relative distances among the UAVs and the relative position errors in (8), $\Delta T_i = [e_{ixF}, e_{iyF}, e_{izF}]^T$. The output T_{Fi} will be fed to the low-level controller of UAVs for trajectory tracking.

3) Path planning and formation implementation: The implementation starts with choosing the operation space of UAVs and the infrastructure to be inspected. This can be done by using a navigation map with satellite images. For example, here a monorail bridge as a testbed subject to inspection can be loaded on the Mission Planner, as shown in Figure 4. The obstacles are also identified based on this map. Furthermore, range sensors such as lidars can be used to form a 3D map of the environment as in our previous work [31]. Based on those inputs, the cost function together with constraints can be defined as described in the previous section. The θ-PSO algorithm will then be run to obtain the desired path. Figure 5 present the pseudo code for path generation process.

Given the desired path, the formation flight starts with an initialisation phase in which each UAV needs to reach its desired initial position from an arbitrary location without collision. While UAVs typically can take off automatically based on the built-in software, their movements to the desired initial positions require a number of checks as follows:

- Crossing check: verify whether or not the path of a UAV$_i$ crosses the paths of other UAVs taken into account marginal clearance ranges;
- Conflict check: verify whether or not a UAV$_i$ enters the conflict region of other UAVs at a same time instant;
- Deadlock check: verify whether or not a UAV$_i$ enters the assigned location of another UAV in the formation.

```
/* Preparation:                                                    */
   Determine the inspection surface;
   Identify the upper and lower boundaries of the working space, start
   and target positions of the swarm;
   Identify obstacles in the working space, check and adjust
   obstacles' parameters if needed ;
   Group all the above data and save in a common file (init file);
/* Initialisation:                                                 */
   Initialise the working environment by loading the init file to global
   memory;
   Initialise the θ-PSO parameters, i.e., w, swarm_population, and
   swarm_iteration and generate a random path to connect the start
   and the target points;
   Set the range of constraints for each particle's phase angle and
   angular increment in [π/2, π/2];
/* θ-PSO:                                                          */
foreach i < (swarm_iteration) do
    foreach j < (swarm_population) do
        Calculate new phase angle increment value in the range of
          limitation;   /* using 1st equation in (3) */
        Calculate new phase angle value in the range of limitation;
          /* using 2nd equation in (3) */
        Calculate new position;  /* using 3rd equation in
          (3) */
        Check Violation cost;
        Evaluate each path based on the Best_Costs and
          Violation cost;
        Update each particle_personal_best and the global_best
          positions;
    end
    Update global_best and Violation costs;
end
   Save global_best and Violation cost;
/* Path generation:                                                */
   Final path is chosen as the maximum number of iterations is
   reached.;
   Generate individual paths for UAVs using (III-B2);
```

Fig. 5: Pseudo code for path generation process.

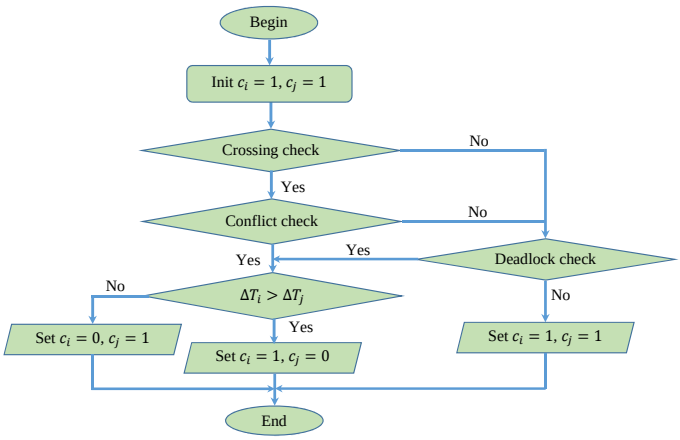


Fig. 6: Initialisation process for UAV formation.

Those checks can be implemented by introducing a binary action variable c_i to each UAV. The value of $c_i = 0$ implies the UAV$_i$ stop to avoid the collision whereas the value of $c_i = 1$ enables it to continue to fly. Denoting ΔT_i as the remaining flight time to the desired initial position of UAV$_i$, the initialisation process represented via the value of c_i that avoids the collision between every two UAVs is then presented as in Fig. 6.

After the initialisation, the low-level control is applied to UAVs based on (9) to maintain the shape. During the flight, on-board computers calculate the inverse kinematics (the formation variables based on the positions of UAVs), and then compare it with their neighbours and the formation centroid to obtain position errors. Those errors are then eliminated by the tracking control action generated.

C. Low-level control

The aim of low-level control is to derive the control laws that apply to each UAV to reach the desired position and attitude. The derivation is based on the mathematical model of the quadcopter with two main frames, i.e., the inertial frame (x_E, y_E, z_E) and the body frame (x_B, y_B, z_B). The translational motion of the quadcopter in the inertial frame is determined by its position, $\xi = (x, y, z)^T$, and velocity, $\dot{\xi} = (\dot{x}, \dot{y}, \dot{z})^T$. The UAV attitude is described by Euler angles roll, pitch, and yaw, $\Theta = (\phi, \theta, \psi)^T$ with the corresponding angular rates $\dot{\Theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$. Let $\omega = [p, q, r]^T$ be the angular rate of the quadcopter in the inertial frame, i.e.,:

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \dot{\Theta}, \tag{10}$$

The transformation from the body to earth frames is then determined by the same rotation matrix as in (6).

In this study, the position of the UAVs is controlled by the built-in PID controller of the flight controller whereas the attitude is governed by the adaptive twisting sliding mode control. In this case, only torque components for orientation of the UAV are considered. They include the torque caused by thrust forces τ, by body gyroscopic effects τ_b, by propeller gyroscopic effects τ_p, and by aerodynamic friction τ_a. In our system, the gyroscopic and aerodynamic torques are considered as external disturbances. Thus, the control inputs mainly depend on the thrust torque $\tau = [\tau_\phi \ \tau_\theta \ \tau_\psi]^T$. Let us denote the control inputs $u = [u_\phi, u_\theta, u_\psi]^T$, where u_ϕ, u_θ and u_ψ respectively represent the roll, pitch and yaw torques.

The quadcopter dynamics then can be represented as follows:

$$\begin{cases} \dot{X}_1 = X_2 \\ \dot{X}_2 = I^{-1}\left[f(X) + u + d\right], \end{cases} \tag{11}$$

where $X_1 = \Theta$, $X_2 = \dot{\Theta}$, $X = [X_1, X_2]^T$ is the state vector, $d = [d_\phi, d_\theta, d_\psi]^T$ is the disturbance vector, and $f(X)$ is the matrix represented as

$$f(X) = \begin{pmatrix} (I_{yy} - I_{zz})qr \\ (I_{zz} - I_{xx})pr \\ (I_{xx} - I_{yy})pq \end{pmatrix}. \tag{12}$$

Based on this dynamic model, we use the adaptive twisting sliding mode control to derive the control law of the form:

$$u(t) = u_{eq}(t) + u_T(t), \tag{13}$$

where $u_{eq}(t) = (u_{eq,i})^T$ and $u_T(t) = (u_{T,i})^T$, $i = 1, 2, 3$, are respectively the equivalent control and the discontinuous part containing switching elements. Given the desired angle reference $X_{1d} = \{\phi_d, \theta_d, \psi_d\}^T$, the sliding surface equation is chosen as:

$$\sigma = \dot{\mathbf{e}} + \Lambda \mathbf{e}, \tag{14}$$

where $\Lambda = \text{diag}(\lambda_\phi, \lambda_\theta, \lambda_\psi)$ is a positive definite matrix being designed, and $\mathbf{e}$ is the control error, $\mathbf{e} = X_1 - X_{1d}$.

The time derivative of σ is then found as:

$$\dot{\sigma} = -\ddot{X}_{1d} + I^{-1}\left[f(X) + u\right] + \Lambda\dot{e}. \tag{15}$$

When the sliding mode has been induced, u can be considered as the equivalent control u_{eq}. The equivalent control rule can be obtained by driving $\dot{\sigma}$ to zero, as follows:

$$u_{eq} = I\left(\ddot{X}_{1d} - \Lambda\dot{e}\right) - f(X). \tag{16}$$

The discontinuous control in our work is the twisting controller [32], $u_{T,i},\ i = 1, 2, 3$ is adopted here as:

$$u_{T,i} = \begin{cases} -\mu_i\alpha_i \text{sign}(\sigma_i) & \text{if } \sigma_i\dot{\sigma}_i \leq 0 \\ -\alpha_i \text{sign}(\sigma_i) & \text{if } \sigma_i\dot{\sigma}_i > 0, \end{cases} \tag{17}$$

where $\mu_i < 1$ is a fixed positive number and $\alpha_i > 0$ is the control gain. To improve the control transient and tracking performance, the gain α_i in (17) could be selected to satisfy the following condition for the one-stage accelerated twisting algorithm [33]:

$$\alpha_i = \max\{\alpha_{*,i}, \gamma_i|\sigma_i|^{\rho_i}\}, \tag{18}$$

where $\alpha_{*,i}$, γ_i and ρ_i are positive constants. Given that fixed time stability is required over a large operational region of the UAV, and motivated by the simplicity of the one-stage accelerated twisting algorithm mentioned above, we propose to adjust the gain α_i in (17) adaptively as in [34], [35], to be constructed based on the following equation:

$$\dot{\alpha}_i = \begin{cases} \bar{\omega_i}\left|\sigma_i(\omega, t)\right| \text{sign}(|\sigma_i(\omega, t)|^{\rho_i} - \epsilon_i) & \text{if } \alpha_i > \alpha_{m,i} \\ \eta_i & \text{if } \alpha_i \leq \alpha_{m,i}, \end{cases} \tag{19}$$

where $\bar{\omega_i}$, ϵ_i and η_i are positive constants and $\alpha_{m,i}$ is an adaptation threshold, chosen to be greater than $\alpha_{*,i}$. By assuming that the disturbance d is bounded, the convergence of the system is found [36].

IV. Surface inspection

Through the data acquisition, photos of the inspected surface are taken and sent to RCU for defect detection. Due to the large amount of data to be processed, a fast yet effective image processing algorithm is required. In this work, we propose a thresholding-based approach that can accurately detect defects in real time.

According to our observation, defects normally appear to be darker than the surrounding area on a gray scale image and hence, can be extracted by selecting a proper threshold. The threshold can be determined automatically by a binarization algorithm, such as the global (Otsu) [37], valley emphasis (VE) [38], adaptive (Sauvola) [39], iterative (ITTH) [40] and slope difference distribution (SDD) [41] thresholding technique. Sauvola's algorithm is widely used in binarization applications due to its adaptability to the content of the images. The threshold in Otsu is calculated by minimizing the within-class variance between the foreground and background objects and might lead to segmentation error when one of the classes has a large variance. VE is an improved version of Otsu's method that focuses on a threshold with a small probability of occurrence and its effectiveness on various defect detection applications has been demonstrated. ITTH are claimed to overcome the limitation of Otsu when segmenting weak objects or fine structures by iteratively applying Otsu's method on the third class between the foreground and background until a preset criterion is met. SDD has been recently proposed based on the slope difference distribution of the histogram peaks to deal with the thresholding selection of different type of vision-based applications. In the case of defect detection, when the ratio between the defect and the background is small, the aforementioned techniques are unable to deliver a good segmentation. In this study, we propose an enhanced thresholding technique that continuously focuses on the lower intensity range of the image histogram taken into account the contribution of defect pixels.

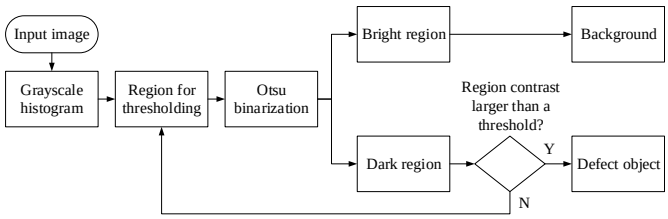


Fig. 7: Flowchart of the proposed thresholding method.

The flowchart of the proposed algorithm is presented in Fig. 7. Specifically, the threshold derived by Otsu method is first applied to the histogram of the input image to segment it into the dark and bright regions. The contrast between those regions is then calculated and compared with a pre-defined threshold to determine whether the dark region can be considered as a defect object. If the stop condition has not been met, the thresholding and segmentation process on the dark region will be iterated whereas all pixels in the bright region will be assigned as the background. At iteration k, the Otsu algorithm, denoted as an operator F applying on the dark region R_i^{k-1}, generates a threshold T_i^k such that:

$$T_i^k = F(R_i^{k-1}). \tag{20}$$

This threshold segments R_i^{k-1} into R_i^k and R_b^k so that:

$$R_i^{k-1} = R_i^k \cup R_b^k, \tag{21}$$

where R_b^i is the bright region to be treated as the background. To detect defects through the iteration, we use the concept of interclass contrast which is defined as a measurement to evaluate the quality of segmentations assuming that the pixels inside one class have the same intensity as the average intensity of that class [42]. For region R_i^{k-1}, the interclass contrast C_i^k is calculated as:

$$C_i^k = \frac{|\ \mu_i^k - \mu_b^k\ |}{\mu_i^k + \mu_b^k}, \tag{22}$$

where μ_i^k and μ_b^k are the intensity means of R_i^k and R_b^k, respectively. As the number of pixels in the thresholding region decreases after each iteration, the denominator in (22) will

decrease leading to the increase of C_i^k. A large value of C_i^k thus indicates a sharp difference in the intensity between the segmenting classes or in another word, the existence of defect-like objects. Let C_s be the threshold for that occurrence, the pseudo code of our algorithm is then presented as in Fig. 8.

Input : R_i^0: histogram of the input image
Output: T_u: ultimate threshold to segment defect from the image background

$k \leftarrow 0$
repeat
  $k++$
  $T_i^k \leftarrow F(R_i^{k-1})$; // Compute new threshold
  $R_i^k \leftarrow R_i^{k-1}(R_i^{k-1} < T_i^k)$; // Define the region of interest
  $R_b^k \leftarrow R_i^{k-1}(T_i^k <= R_i^{k-1} < T_i^{k-1})$; // Set background region
  $\mu_i^k \leftarrow Average(R_i^k)$, $\mu_b^k \leftarrow Average(R_b^k)$; // Compute intensity mean
  $C_i^k \leftarrow |\mu_i^k - \mu_b^k| / \mu_i^k + \mu_b^k$; // Compute interclass contrast
until $(C_i^k > C_s)$;
$T_u \leftarrow T_i^k$

Fig. 8: Pseudo code for defect detection.

V. Results

We have conducted a number of surface inspection tasks to evaluate the performance our proposed approach. Details are as follows.

A. Experimental setup

The UAVs used in this study is the 3DR Solo drone shown in Fig. 9. It has three processors, two are Cortex M4 168 MHz running Pixhawk firmware for low-level control and the other is an ARM Cortex A9 running Linux operating system for data processing and high-level control. The UAV is retrofitted with a RTK compatible GPS receiver, an IoT board, a detachable antenna, a camera, a 3D gimbal and other sensors for data acquisition.

The camera used is a Hero 4 Black with the focal length of 34.4 mm, resolution of 12 megapixels, and wireless network capability. It is attached to a three-axis gimbal with one degree

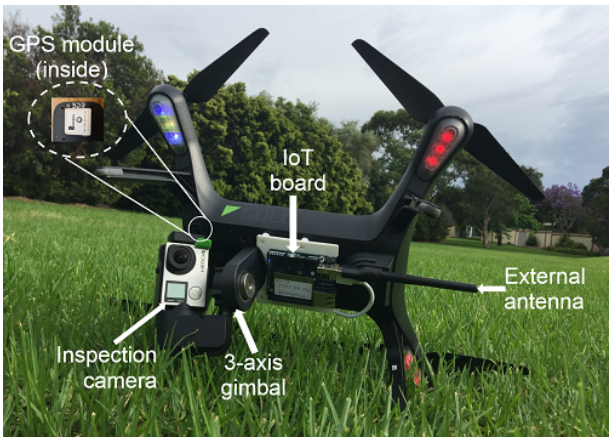


Fig. 9: The 3DR Solo Drone with retrofitted components

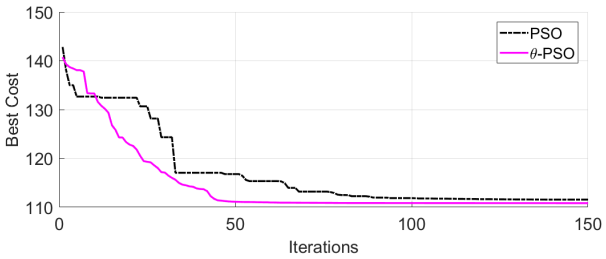


Fig. 10: Convergence comparison between conventional PSO and θ-PSO

of freedom for controlling its elevation (pitch) angle. The photos taken by using this camera will be streamed to RCU where the thresholding algorithm is run to detect defects.

The IoT board is attached to the accessory port of the drone and interfaced with the embedded Linux operating system via USB protocol. A 6 dBi detachable antenna is attached to the IoT board to extend the wireless communication range. The gateway router is a TPLink wireless N 4G LTE router that has a SIM card slot to use mobile broadband networks. The RTK compatible GPS receiver is based on the u-blox NEO-M8P-2-10 module. It periodically updates the correction data from the base GPS station to improve the positioning precision.

The task assigned in experiments is to inspect surfaces of monorail bridges using three UAVs. The Mission Planner ground control software is used to collect initial information about the structure and its surrounding environment based on Google satellite maps (GST). The operation space is chosen with the dimensions of 141 m $\times$ 101 m $\times$ 40 m equivalent to the GST coordinates of $\{-33.87601, 151.191182, 0\}$ and $\{-33.875086, 151.192676, 40\}$, as illustrated in Fig. 4. The initial and final positions are chosen as $P_i = \{40.0, 8.0, 30\}$ and $P_f = \{64, 108, 34\}$, respectively. Therein, ten obstacles are identified, each with a different radius.

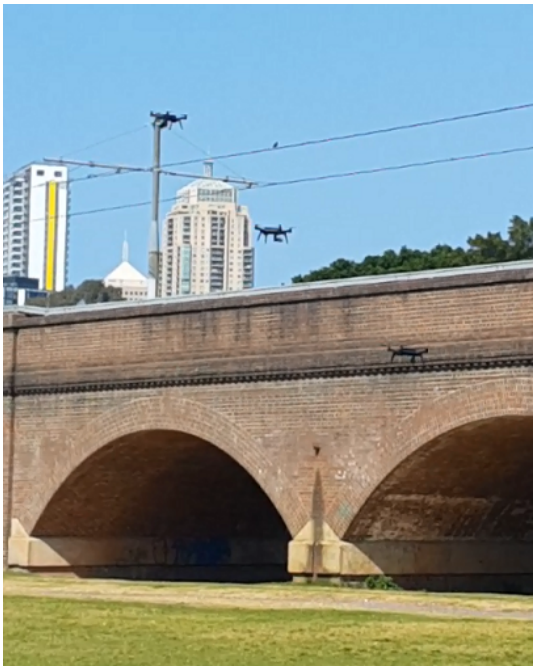
In our path planning algorithm, the number of particles, waypoints, and iterations are respectively selected as 150, 10, and 300. Parameters of the three quadcopters with respect to the centroid of the formation are $\Delta T_1 = (0, 0, 2)$ m, $\Delta T_2 = (3, 0, -1)$ m and $\Delta T_3 = (-3, 0, -1)$ m.

B. High-level control results

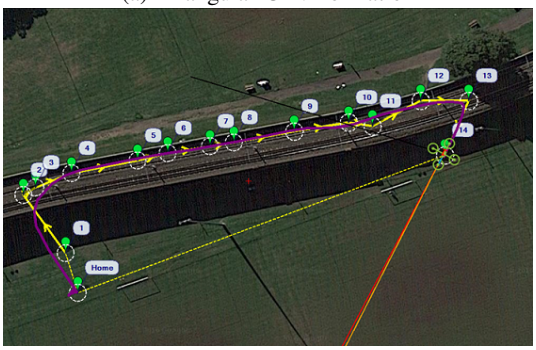
The path planning and formation results are presented in this subsection, whereby it is expected that the designed method can generate collision-free paths for the three UAVs with sufficiently fast convergence using the proposed algorithm. For this, let us first compare the performance of the proposed θ-PSO with a conventional PSO algorithm. Figure 10 shows the cost values over iterations. It can be seen that although both algorithms are convergent, the θ-PSO introduces a faster and more stable conversion. The results are confirmed as recorded in Table I which shows the average cost value and convergence iterations.

TABLE I: PSO and θ-PSO performance comparison

Algorithm	Min cost	Max cost	Iterations
PSO	112.43	143.0	102
θ-PSO	111.02	142.84	68



(a) Triangular UAV formation



(b) Planned path (yellow) and flown path (violet)

Fig. 11: Bridge inspection with UAV formation

Given the generated path, we have conducted field-test experiments in which the triangular formation automatically navigated along the inspected surface, as depicted in Fig. 11. The 3D trajectories generated from that of the formation centroid path are shown in Fig. 12, where it can be seen that the three UAVs can take off, reach to their individual altitude set-point, descend and finally arrive their target position at almost the same interval, while maintaining the desired triangular shape. This result can be further verified via the altitude time responses of the three UAVs as recorded in Fig. 13. It is clear that the UAVs are capable of avoiding obstacles and preserving the desired formation configuration during the inspection task. For further evaluation, Fig. 14 shows the error between the planned and flown paths computed by selecting the closest coordinates of the flight trajectories to the reference points. Those small errors imply the feasibility and reasonable smoothness of the generated path for the deployment of UAV formation.

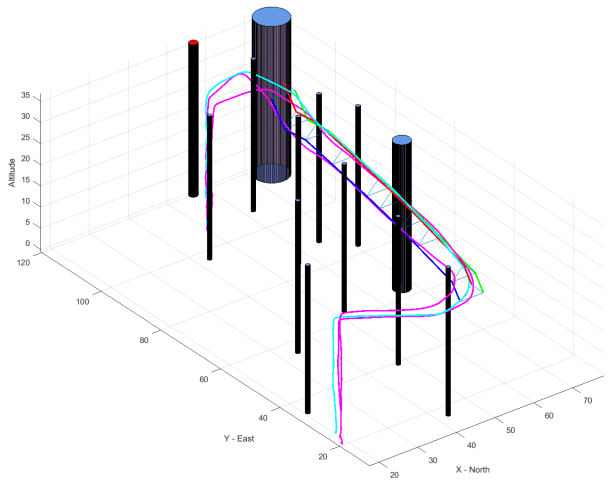


Fig. 12: Trajectories of three UAVs tracking the planned paths

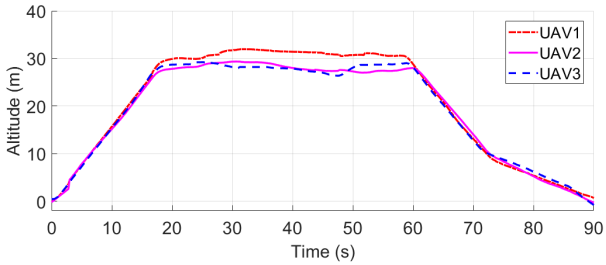


Fig. 13: Altitudes of the three UAVs in the formation test

C. Low-level control results

Performance of the controller is judged under disturbances and parametric variations. In our experiments, the UAV was initially staying at the steady state in the air, where all angles and angular velocities are zeros. To test with the real inspection situation, some sudden and significant change in reference values are added as $\phi = -10°$, $\theta = 10°$ and $\psi = 45°$ at time 0.5 s, 1 s and 2 s, respectively. Results in Fig. 15 illustrate that ATSM effectively rejects disturbances by driving the three angles to their reference values within 2 seconds. Other comparison and evaluation results can be found in our previous work [36].

D. Surface inspection results

The effectiveness of our proposed defect detection algorithm has been tested on images taken by the camera mounted on

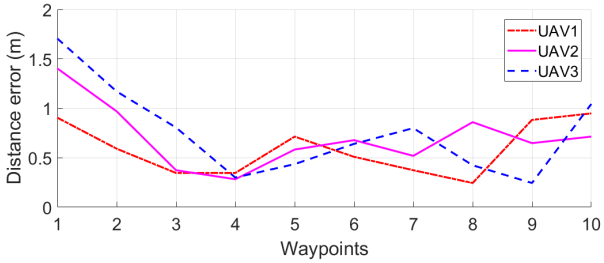


Fig. 14: Errors between the planned and flown paths

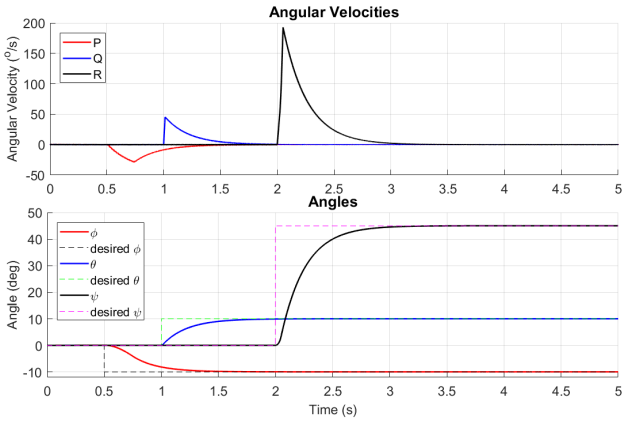


Fig. 15: Angular velocity and angle responses in the presence of disturbances.

UAVs. The results have been compared with other methods including Otsu, ITTH and Sauvola. Figs. 16 and 17 present some examples from our experiment where the input images are taken at different light conditions with various levels of background complexity, e.g. uniform background (Image 1 and 2), patterned background (Image 3), noisy to very noisy background (Image 4 - 10). Image 3 was taken at the monorail bridge mentioned in the experimental setup, whereas the remaining ones were obtained at a suburban bridge in New South Wales. Notably, Otsu and ITTH return a high level of noise on all test images except on Image 1 where the contrast is high and the background is uniform. Sauvola and VE return a proper segmentation on Images 1-3 where the contrast between the object and the background is still obvious but fails to distinguish the crack and the background on other images when the contrast decreases. As SDD treats bigger classes more favorably, the defect features might be mis-detected on noisy images (e.g Images 5-8). Thus, our algorithm outperforms those methods in all test cases of different background complexity and image contrast. Those results can be quantitatively evaluated via F-measure, a compromise between recall and precision. Let tp, tn be the correctly reported positive and negative results, fp and fn the falsely reported positive and negative results, the F-measure for binary classification is calculated as:

$$F = \frac{2 \times p \times r}{p + r}, \tag{23}$$

where p and r denote the precision and recall measure defined respectively as:

$$p = \frac{tp}{tp + fp}, \tag{24}$$

$$r = \frac{tp}{tp + fn}. \tag{25}$$

Table II reports the F values obtained for each method which clearly shows the superiority of our proposed algorithm.

We also evaluated the processing time executed by using MATLAB R2018b on an Intel(R) Core(TM) i5-5300U CPU @2.30 GHz with 64 bit Windows 7. The average processing time is 50 ms per image or 20 frames per second which is sufficient for real-time inspection.

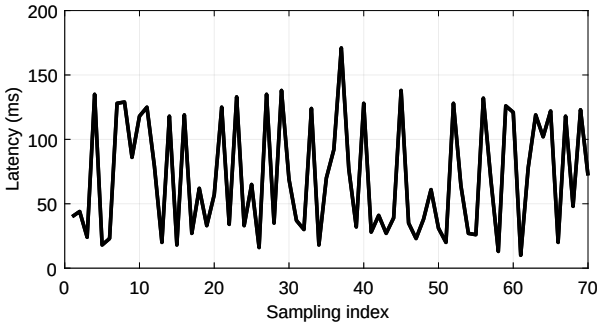


Fig. 18: Network delay during the inspection

VI. Discussion

While the inspection system has been implemented successfully, it is worth to discuss on problems during the development for further investigation. The first issue relates to the communication between UAVs and the gateway router. As most current IoT boards are designed for indoor applications, their built-in antenna is insufficient for outdoor communication in inspection tasks. A high gain external antenna is therefore needed to extend the communication range as well as provide stable signals, especially for moving objects like UAVs. Nevertheless, the communication range between UAVs and the gateway router is still limited to approximately 100 m due to the transmitting power of the IoT board. While this range is sufficient for inspecting structures like buildings or wind turbines, it may be ill-use for long structures like bridges. A possible solution is to use more gateway routers as repeaters.

Another issue relates to the network latency. It is interesting that the average transmission delay in our experiments is 71 ms, which is even larger than the processing time to detect defects. Moreover, the network jitter is also quite large with the standard deviation of 45 ms as shown in Fig. 18. In the worst case, the delay together with the processing time limit the real-time capability of the inspection system to 4 frames per second. The rationale is the use of mobile broadband networks which require wireless data transmission via multiple intermediate relay stations. Further investigation is thus necessary to improve the real-time response.

The final issue relates to the formation of UAVs. In rigid-body structure, the shape is preserved during the inspection process and thus is relevant for structures with clear flying paths. For more complicated structures, for example with corners and varied/narrow passages, a flexible formation structure is often required [43]. In this case, the UAVs need have capabilities to do online path planning as well as communicate with other UAVs to determine its role in the new formation. Another approach is to pass all the sensing information to the RCU for calculating new paths and re-upload them to the UAVs. In either approach, extra complication is added to the system which may pose further issues.

VII. Conclusion

In this paper, we have introduced a system for inspecting structural surfaces in real time using the IoT and UAV formation. Our contributions include: (i) a new communication

TABLE II: Comparison in defect detection between the proposed method and Sauvola, Otsu, VE, ITTH and SDD methods

Image	Our method	Sauvola	Otsu	VE	ITTH	SDD
Image 1	**0.9889**	0.9163	0.8800	0.9206	0.9416	0.6881
Image 2	**0.9774**	0.9743	0.5101	0.9736	0.5112	0.7626
Image 3	**0.9818**	0.9547	0.5083	**0.9818**	0.5076	0.8534
Image 4	**0.9542**	0.6251	0.5141	0.5203	0.5147	0.6974
Image 5	**0.9451**	0.5239	0.5071	0.5119	0.5069	0.4965
Image 6	**0.9140**	0.8256	0.5183	0.8774	0.5208	0.5198
Image 7	**0.8693**	0.6776	0.5117	0.6371	0.5125	0.5017
Image 8	**0.8685**	0.7333	0.5214	0.6809	0.5244	0.5029
Image 9	**0.8474**	0.7876	0.5197	0.8367	0.5208	0.5730
Image 10	**0.7537**	0.6893	0.5179	0.7362	0.5222	0.5706

platform based on the Internet of Things which can exploit vast processing capabilities of RCU and remove the burdens in communication distance; (ii) a multi-layer architecture allowing various modules of a complicated system to be integrated to fulfil a common inspection task; (iii) a new formation algorithm for UAVs based on the angle-encoded PSO that can create optimal flying paths constrained to the requirements in data acquisition; and (iv) an enhanced thresholding algorithm to detect defects in real time at high accuracy. A number of experiments have been conducted with real defects detected from the acquired data. Comparisons and discussions have been also presented to evaluate the performance of the proposed system and address issues for future investigation.

REFERENCES

[1] G. Cai, J. Dias, and L. Seneviratne, "A survey of small-scale unmanned aerial vehicles: Recent advances and future development trends," *Unmanned Systems*, vol. 2, no. 02, pp. 175–199, 2014.

[2] M. C. Tatum and J. Liu, "Unmanned aircraft system applications in construction," *Procedia Engineering*, vol. 196, pp. 167–175, 2017.

[3] L. Wang and Z. Zhang, "Automatic detection of wind turbine blade surface cracks based on uav-taken images," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 9, pp. 7293–7303, 2017.

[4] Y. Zhang, X. Yuan, t. Li, and S. Chen, "Automatic power line inspection using uav images," *Remote Sensing*, vol. 9, no. 8, p. 824, 2017.

[5] G. Morgenthal and N. Hallermann, "Quality assessment of unmanned aerial vehicle (uav) based visual inspection of structures," *Advances in Structural Engineering*, vol. 17, no. 3, pp. 289–302, 2014.

[6] K. Peng, L. Feng, Y. Hsieh, T. Yang, S. Hsiung, Y. Tsai, and C. Kuo, "Unmanned aerial vehicle for infrastructure inspection with image processing for quantification of measurement and formation of facade map," in *Applied System Innovation (ICASI), 2017 International Conference on*. IEEE, 2017, pp. 1969–1972.

[7] D. Reagan, A. Sabato, and C. Niezrecki, "Feasibility of using digital image correlation for unmanned aerial vehicle structural health monitoring of bridges," *Structural Health Monitoring*, p. 1475921717735326, 2017.

[8] J.-H. Chen, M.-C. Su, R. Cao, S.-C. Hsu, and J.-C. Lu, "A self organizing map optimization based image recognition and processing model for bridge crack inspection," *Automation in Construction*, vol. 73, pp. 58–66, 2017.

[9] L. J. Sánchez-Aparicio, B. Riveiro, D. Gonzalez-Aguilera, and L. F. Ramos, "The combination of geomatic approaches and operational modal analysis to improve calibration of finite element models: A case of study in saint torcato church (guimarães, portugal)," *Construction and Building Materials*, vol. 70, pp. 118–129, 2014.

[10] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart, "A uav system for inspection of industrial facilities," in *Aerospace Conference, 2013 IEEE*. IEEE, 2013, pp. 1–8.

[11] A. Khaloo, D. Lattanzi, K. Cunningham, R. DellAndrea, and M. Riley, "Unmanned aerial vehicle inspection of the placer river trail bridge through image-based 3d modelling," *Structure and Infrastructure Engineering*, vol. 14, no. 1, pp. 124–136, 2018.

[12] F. Ducho, A. Babinec, M. Kajan, P. Beo, M. Florek, T. Fico, and L. Juriica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59 – 69, 2014, modelling of Mechanical and Mechatronic Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S187770581403149X

[13] A. Ammar, H. Bennaceur, I. Châari, A. Koubâa, and M. Alajlan, "Relaxed dijkstra and a* with linear complexity for robot path planning problems in large-scale grid environments," *Soft Computing*, vol. 20, no. 10, pp. 4149–4171, Oct 2016. [Online]. Available: https://doi.org/10.1007/s00500-015-1750-1

[14] A. Bircher, K. Alexis, U. Schwesinger, S. Omari, M. Burri, and R. Siegwart, "An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees," *Robotica*, vol. 35, no. 6, p. 13271340, 2017.

[15] F. Yan, Y.-S. Liu, and J.-Z. Xiao, "Path planning in complex 3d environments using a probabilistic roadmap method," *International Journal of Automation and Computing*, vol. 10, no. 6, pp. 525–533, Dec 2013. [Online]. Available: https://doi.org/10.1007/s11633-013-0750-9

[16] S. Siebert and J. Teizer, "Mobile 3d mapping for surveying earthwork projects using an unmanned aerial vehicle (uav) system," *Automation in Construction*, vol. 41, pp. 1–14, 2014.

[17] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 6423–6430.

[18] N. Metni and T. Hamel, "A uav for bridge inspection: Visual servoing control law with orientation limits," *Automation in construction*, vol. 17, no. 1, pp. 3–10, 2007.

[19] K. Alexis, G. Darivianakis, M. Burri, and R. Siegwart, "Aerial robotic contact-based inspection: planning and control," *Autonomous Robots*, vol. 40, no. 4, pp. 631–655, 2016.

[20] C. Deng, S. Wang, Z. Huang, Z. Tan, and J. Liu, "Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications," *J. Commun*, vol. 9, no. 9, pp. 687–692, 2014.

[21] G. J. Lim, S. Kim, J. Cho, Y. Gong, and A. Khodaei, "Multi-uav pre-positioning and routing for power network damage assessment," *IEEE Transactions on Smart Grid*, 2016.

[22] A. Banaszek, S. Banaszek, and A. Cellmer, "Possibilities of use of uavs for technical inspection of buildings and constructions," in *IOP Conference Series: Earth and Environmental Science*, vol. 95, no. 3. IOP Publishing, 2017, p. 032001.

[23] C. Rosales, P. Leica, M. Sarcinelli-Filho, G. Scaglia, and R. Carelli, "3d formation control of autonomous vehicles based on null-space," *Journal of Intelligent & Robotic Systems*, vol. 84, no. 1-4, pp. 453–467, 2016.

[24] H. G. Tanner, G. J. Pappas, and V. Kumar, "Leader-to-formation stability," *IEEE Transactions on robotics and automation*, vol. 20, no. 3, pp. 443–455, 2004.

[25] C. A. Kitts and I. Mas, "Cluster space specification and control of mobile multirobot systems," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 2, pp. 207–218, 2009.

[26] E. Stingu and F. L. Lewis, *A Hardware Platform for Research in Helicopter UAV Control*. Dordrecht: Springer Netherlands, 2009, pp. 387–406. [Online]. Available: https://doi.org/10.1007/978-1-4020-9137-7_21

[27] S. Kim, H. Oh, J. Suk, and A. Tsourdos, "Coordinated trajectory planning for efficient communication relay using multiple uavs," *Control Engineering Practice*, vol. 29, pp. 42 – 49, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066114001282

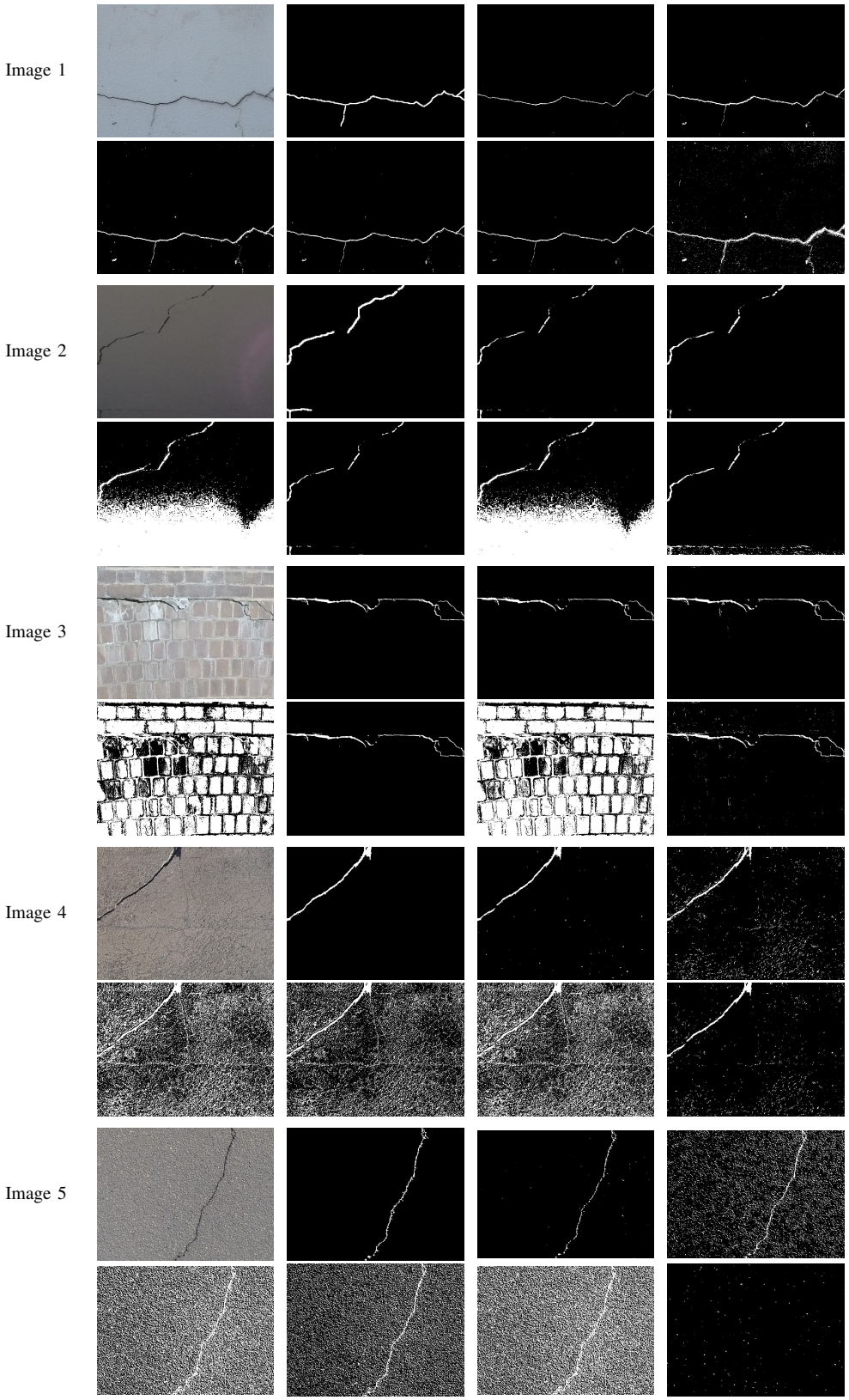


Fig. 16: Image segmentation results. First row: image name, original image, ground truth, our result, Sauvola; second row: segmentation respectively by Otsu, VE, ITTH and SDD.

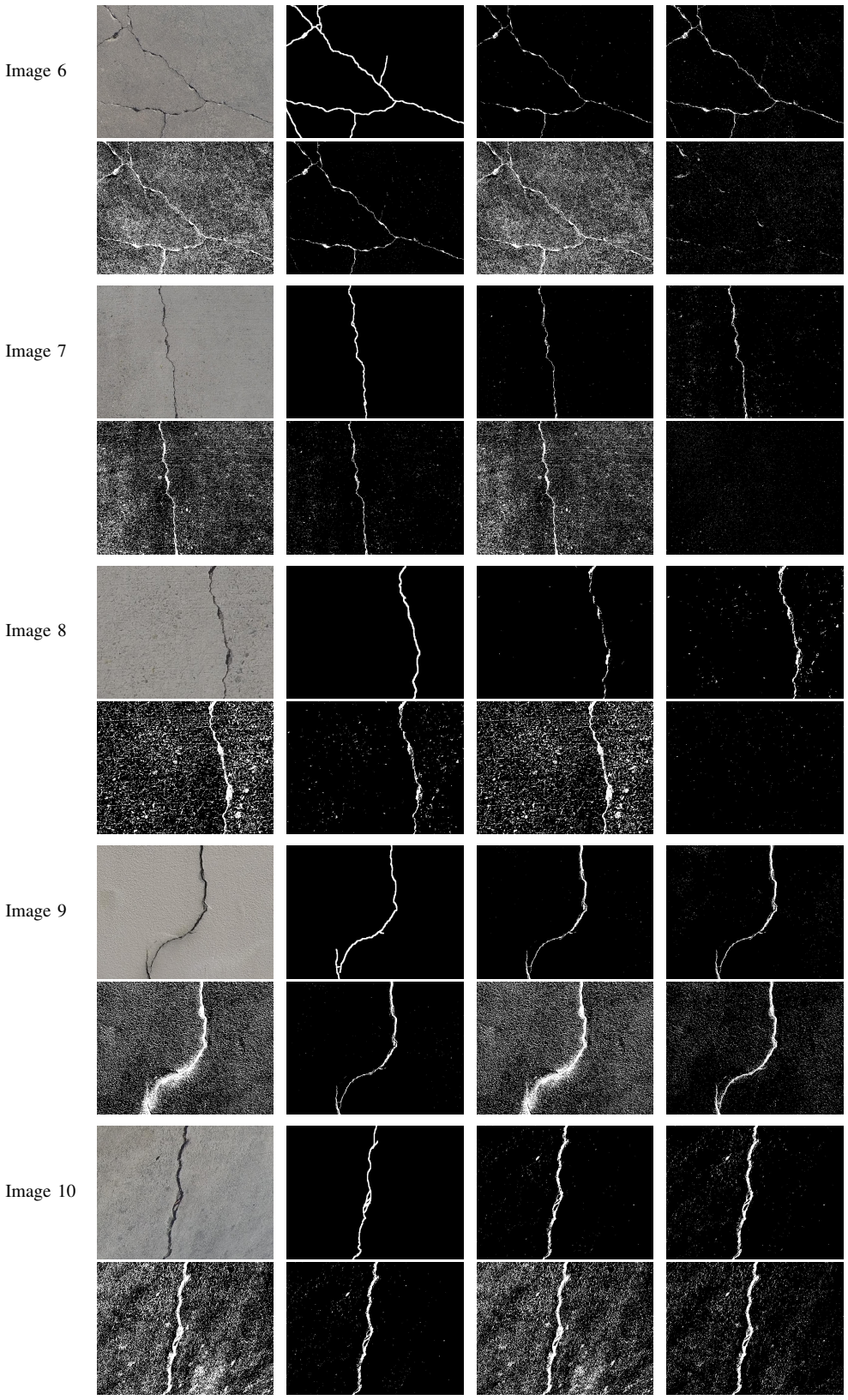


Fig. 17: Image segmentation results. First row: image name, original image, ground truth, our result, Sauvola; second row: segmentation respectively by Otsu, VE, ITTH and SDD.

[28] O. Cetin and I. Zagli, "Continuous airborne communication relay approach using unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1, pp. 549–562, Jan 2012. [Online]. Available: https://doi.org/10.1007/s10846-011-9556-6
[29] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, May 2016.
[30] V. T. Hoang, M. D. Phung, T. H. Dinh, and Q. P. Ha, "Angle-encoded swarm optimization for uav formation path planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 5239–5244.
[31] M. D. Phung, C. H. Quach, T. H. Dinh, and Q. Ha, "Enhanced discrete particle swarm optimization path planning for uav vision-based surface inspection," *Automation in Construction*, vol. 81, pp. 25–33, 2017.
[32] A. Levant, "Sliding order and sliding accuracy in sliding mode control," *International journal of control*, vol. 58, no. 6, pp. 1247–1263, 1993.
[33] Y. Dvir and A. Levant, "Accelerated twisting algorithm," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2803–2807, 2015.
[34] F. Plestan, Y. Shtessel, V. Bregeault, and A. Poznyak, "New methodologies for adaptive sliding mode control," *International journal of control*, vol. 83, no. 9, pp. 1907–1919, 2010.
[35] V. T. Hoang, A. M. Singh, M. D. Phung, and Q. P. Ha, "Adaptive second-order sliding mode control of uavs for civil applications," in *Automation and Robotics in Construction (ISARC), 2017 International Symposium on*, 2017, pp. 816–822.
[36] V. T. Hoang, M. D. Phung, and Q. P. Ha, "Adaptive twisting sliding mode control for quadrotor unmanned aerial vehicles," in *2017 11th Asian Control Conference (ASCC)*, Dec 2017, pp. 671–676.
[37] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.*, vol. 11, no. 285-296, pp. 23–27, 1975.
[38] H.-F. Ng, "Automatic thresholding for defect detection," *Pattern Recognition Letters*, vol. 27, no. 14, pp. 1644 – 1649, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016786550600119X
[39] J. Sauvola and M. Pietikinen, "Adaptive document image binarization," *Pattern Recognition*, vol. 33, no. 2, pp. 225 – 236, 2000.
[40] H. Cai, Z. Yang, X. Cao, W. Xia, and X. Xu, "A new iterative triclass thresholding technique in image segmentation," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1038–1046, 2014.
[41] Z. Wang, J. Xiong, Y. Yang, and H. Li, "A flexible and robust threshold selection method," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2220–2232, Sep. 2018.
[42] M. D. Levine and A. M. Nazif, "Dynamic measurement of computer generated image segmentations," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 2, pp. 155–164, 1985.
[43] H. Rezaee and F. Abdollahi, "A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 347–354, Jan 2014.