System Architecture for Real-time Surface Inspection Using Multiple UAVs

V. T. Hoang, M. D. Phung, T. H. Dinh, and Q. P. Ha

Abstract—This paper presents a real-time control system for surface inspection using multiple unmanned aerial vehicles (UAVs). The UAVs are coordinated in a specific formation to collect data of the inspecting objects. The communication platform for data transmission is based on the Internet of Things (IoT). In the proposed architecture, the UAV formation is established via using the angle-encoded particle swarm optimisation to generate an inspecting path and redistribute it to each UAV where communication links are embedded with an IoT board for network and data processing capabilities. Data collected are transmitted in real time through the network to remote computational units. To detect potential damage or defects, an online image processing technique is proposed and implemented based on histograms. Extensive simulation, 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, particle swarm optimisation.

I. INTRODUCTION

Continuous monitoring of health conditions of built infrastructure is essential for their safe operation and long-term serviceability. Since structural deterioration is quite often revealed through visual 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 common, the use of flying vehicles like UAVs is receiving increasing interest thanks to their flexibility in operation, versatility in task allocation and capability of conducting non-intrusive inspection [1]. For instance, UAVs have been used for periodical inspection of critical infrastructures such as oil-gas pipelines, wind turbine blades and power lines [2], [3]. They also have been employed to collect data for defect/damage recognition such as cracks, rust or structural misalignment [4], [5].

Systems for robotic inspection using UAVs are often designed in a layer-based architecture. At the top layer, the UAV is programmed to fly close to the objects to collect inspection data. They are then post-processed to detect potential damage/defects via techniques such as photogrammetry [6] or Haar-like features extraction [2]. The middle layer involves the generation of inspection trajectories whereby path

V. T. Hoang, T. H. Dinh, and Q. P. Ha are with School of Electrical and Data Engineering, University of Technology Sydney (UTS), 81 Broadway, Ultimo NSW 2007, Australia. {VanTruong.Hoang, TranHiep.Dinh, Quang.Ha}@uts.edu.au.

M. D. Phung is with University of Engineering and Technology, Vietnam National University (VNU), 144 Xuan Thuy, Cau Giay, Hanoi, Vietnam. duongpm@vnu.edu.vn.

planning algorithms such as A-star, Dijkstra, rapidly-exploring random tree and probabilistic roadmap can be employed [7]. However, algorithms specifically designed for UAVs using particle swarm optimization [8] or iterative viewpoint resampling [9] are typically more efficient by considering also the coverage among waypoints. At the lowest layer, various techniques have been proposed to track the generated paths, using backstepping, model predictive control, PID control or sliding mode control [10] with sufficient performance.

While the use of single UAVs has proved its applicability in inspection, recent works start to study the coordination of UAVs in a group to achieve higher time efficiency, performance and resilience [11], [12]. A multi-platform UAV system for power line inspection tasks [11] shows that the time to inspect 70 km of the power line can be reduced to 3 hours compared to a week as with the traditional inspection method. Therefore, different aspects of multiple UAV coordination such as formation control, collision avoidance, and data acquisition have been addressed [13], [14].

In a control architecture for inspection using multiple UAVs, the synchronization in control, path planning and data processing is typically required not only between layers within single UAVs but also among them. For real-time inspection, communication channels are therefore 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 [15]. Some studies suggested deploying additional UAVs as communication relay stations to extend the network [16], [17]. This approach is suitable for tasks covering a local range but not large areas like bridges, power lines or wind turbines which require hundreds of kilometres in communication range. Several studies proposed to use satellite communication to overcome this problem [11], [18]. It is however expensive, complex and not always feasible.

In this study, the Internet of Things (IoT) is used as the communication platform to interconnect UAVs and devices in order to efficiently perform inspection tasks. Based on this platform, a robust low-level 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

the 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 streamed to RCU where they are processed to detect potential defects. A detection algorithm based on histogram 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 system.

II. RELATED WORKS

A number of studies have been devoted to the problem of surface inspection using UAVs. Early works typically targeted simple and linear structures such as highways, roads, and canals [19], [20]. They focused on low-level control laws using the data acquired from the incorporated visual systems for quasi-stationary flights above planar surfaces. Later, the inspection of hardly accessible structures in GPS-denied environments using UAVs has been proposed using on-board Simultaneous Localisation and Mapping (SLAM) [21]. While the positioning and self-recovering problems have been addressed, the system was intended for semi-autonomous operations that would require an operator to instruct the UAV. To improve the level of autonomy, images obtained by UAVs can be used to detect inspecting objects and create paths for autonomous flights [22], [3].

For automatic inspection, sophisticated systems featured path planning and path-following control algorithms have been proposed. In [23], an aerial robotic system for the contact-based surface inspection has been introduced using not only optimal trajectory tracking but also accurate force control techniques. In [9], an iterative viewpoint re-sampling path planning algorithm was proposed for the inspection of complex 3D structures. The inspection of outdoor structures under windy conditions was addressed in [24] by using the viewpoint selection and optimal time route algorithms. Besides, an UAV-based inspection system for wind turbines was developed in [25] with the capability of creating smooth and collision-free flight paths based on the data recorded from LIDAR sensors. These studies however focus more on data collection rather than defect detection.

In terms of surface inspection, several studies have been conducted for defect detection tasks. In [26], a fast and effective defect detection method has been proposed using the size-based estimation with data obtained from both color and infrared cameras. In [2], Haar-like features and a cascading classifier were applied to UAV-taken images to identify cracks on wind turbine blade surfaces. Self-organizing map optimization was introduced in [5], based on image recognition and processing model for crack detection to reduce human involvement. On another note, [4] assessed the quality and feasibility of images taken by UAVs for defect detection and discussed methods to improve the image quality.

The use of multi-UAV cooperation for infrastructure inspection has been recently investigated. A multi-platform UAV system for power line inspection was experimented in [11]. Similarly, a multi-UAV positioning and routing for power network damage assessment was introduced in [12] whereas some other studies developed inspection systems for buildings or bridges [27], [10]. However, those systems are rather limited in communication and coordination capability. In fact, it can be seen from the literature that UAV-based inspection is still in its early development stage. There exists a crucial need for an architecture that not only enables the integration of various functional modules like path planning, high-level and low-level control but also allows for the real-time coordination of multiple UAVs within a homogeneous communication platform for efficient surface inspection.

III. SYSTEM OVERVIEW AND DATA COMMUNICATION

Figure 1 presents an overview of the proposed inspection system. Its core includes three UAVs equipped with cameras to collect data of the inspected structure. The UAVs, via IoT boards, form a private network called SkyNet for data exchanges during their cooperative flight. At the same time, they also share data with GCS via a wireless gateway router that relays the data via the Internet to RCU to be processed further. The UAVs localization is conducted via GPS modules combined with inertial measurement units (IMUs) to monitor internal states of the drones. Depending on the inspection task, real-time kinematic (RTK) GPS modules will be used to improve positioning accuracy. The operation of each UAV is monitored via its accompanied remote controller which will take over the control in emergency situations.

A. IoT devices

As illustrated in Fig.1, the inspection system involves real-time cooperation of many components within the control architecture. To equip them with networking and data processing capabilities, IoT boards are installed, each includes a processor and a microcontroller as shown in Fig. 8. With the processor, the IoT-based devices can connect to the wifi network as well as process the receiving data via the Linux operating system installed. The microcontroller enables the devices to work with other low hardware interfaces such as voltage/current interface, AD/DA converter, PWM, etc. The boards thus turn a normal device into a smart one that can be integrated into the Internet protocol (IP) based networks. Nevertheless, the built-in antennas of IoT boards designed for indoor applications are insufficient for outdoor communication which requires high gain external antennas. Therefore, a 6 dBi detachable antenna is placed on the drone to extend the communication range and provide stable signals, especially for moving objects like UAVs. In addition, mobile broadband networks are made available using 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 surface without the need for relay stations nor satellite communication. In remote areas where the mobile network is not available, this approach would be feasible but requires extra equipment to connect to the nearest available access point.
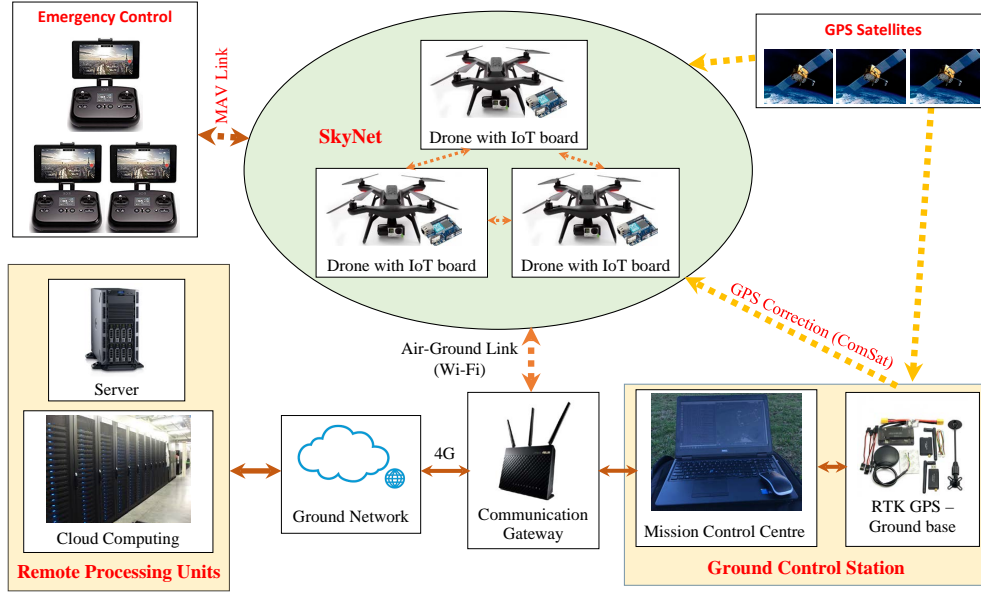
Fig. 1: Data communication structure

B. Communication protocols

Along with the hardware, transport protocols play an important role in ensuring the security and efficiency of the data exchanged. The most popular transport protocols for IoT include the Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Real-Time Transport Protocol (RTP). While TCP was originally designed for the reliable transmission of static data over low-bandwidth, high-error-rate networks, UDP could send datagrams from a device to another as fast as possible but only under good network conditions without considering the state of the network. Developed for delivering real-time multimedia data, RTP can facilitate jitter compensation and detection of out-of-sequence arrival in data. According to those protocol features, RTP is chosen in our systems architecture for photo/video streaming and UDP for sensing data transmitting whereas TCP for delivering administrative data and control commands.

C. Data processing

One of the main benefits of using IoT is the capability of conducting computation and data processing at various layers of the system, i.e., the device layer by IoT boards, the control layer by built-in computers of UAVs and the application layer by networked server computers or cloud computing services, depending on the amount of data to be processed and real-time requirements. In our system, IoT boards are used for processing communication data among UAVs such as position, velocity and other state information to minimize the computational latency. The control algorithm on the other hand is handled by the built-in computers of UAVs to enhance reliability. Other information and imaging data are processed by server computers to cope with the high demand for computation and energy consumption.

IV. DATA ACQUISITION AND CONTROL

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

A. Task assessment

Given the surface to be inspected, at the task assessment level, a set of camera configurations is computed, covering all areas of interest. Each configuration c_i corresponds to a position and an orientation of the camera, and subject to constraints on image quality as per the following requirements:

 (i) images are taken when the camera directed perpendicularly to the inspected surface;
 (ii) their resolution is sufficiently high to distinguish the required smallest feature, s_f; and
 (iii) image overlapping is specified by the sticking algorithm to a given percentage, o_p.

The first requirement confines the camera orientation to the normal of the inspected surface. The resolution requirement suggests the computation of the field of view of the camera as:

$$a_{fov} = \frac{1}{2}r_c s_f, \qquad (1)$$

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

$$d_{f,i} = \frac{a_{fov}f}{s_s}, \qquad (2)$$

where f and s_s are respectively the focal length and sensor size of the camera. Let G be a finite set of geometric primitives g_i for the whole inspected surface, with each g_i corresponding to a surface patch covered by a camera shot. Taken the
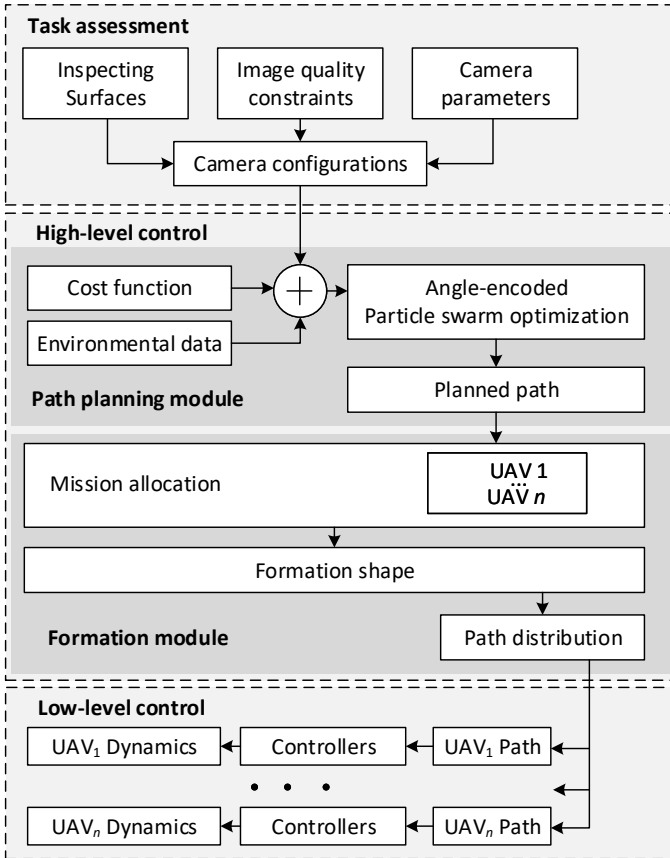
Fig. 2: System architecture for multi-UAV surface inspection

overlapping percentage into account, the geometric primitive g_i is computed as:

$$g_i = (1 - o_p)a_{fov}. \qquad (3)$$

By using (2) and (3), configurations c_i can be sufficiently determined to cover the set of primitives G [8], which is then fed to the high-level control layer for path planning.

B. High-level control

The high-level control consists of two modules, path planning and formation control the generation respectively of the reference path computed from the task assessment, and of trajectories for each UAV based on the reference path and the desired formation shape.

1) Path Planning: When producing a path for the desired motion of multiple UAVs in a group, a number of constraints are required for formation maintenance, 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 reference 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) [28].

In θ-PSO, a set of particles is generated, each seeks for an optimal solution by propagating to compromise between its own experience and the social experience. Initially, each particle is assigned a random position, x_i. This position is represented by a phase angle θ_i of the UAV. The particle motion is then updated by the following equations:

$$\begin{cases} \Delta\theta_{ij}^{k+1} = w\Delta\theta_{ij}^k + c_1 r_{1i}^k(\lambda_i^k - \theta_{ij}^k) + c_2 r_{2i}^k(\lambda_{gj}^k - \theta_{ij}^k) \\ \theta_{ij}^{k+1} = \theta_{ij}^k + \Delta\theta_{ij}^{k+1}, (i = 1, 2, ..., N; j = 1, 2, ..., S) \\ x_{ij}^k = \frac{1}{2}\left[(x_{max} - x_{min})sin\left(\theta_{ij}^k\right) + x_{max} + x_{min}\right], \end{cases}$$
$$(4)$$

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 at iteration j in the searching space; $\lambda_g = [\lambda_{g1}, \lambda_{g2}, ..., \lambda_{gS}]$ and $\lambda_i = [\lambda_{i1}, \lambda_{i2}, ..., \lambda_{iS}]$ are respectively the global and personal best positions; N is the swarm size; S is the searching space dimension; w is the inertial weight; r_{1i} and r_{2i} are 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. (4), it can be seen that a solution of a particle covers 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 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:

$$J_F(\mathcal{T}_F) = \sum_{m=1}^{3} \beta_m J_m(\mathcal{T}_F), \qquad (5)$$

where $\mathcal{T}_F$ is the formation path; β_m is the weighting factor indicating the corresponding threat intensity; and $J_m(\mathcal{T}_F)$, $m = 1, 2, 3$, are the costs associated with the path length, collision violation and flying altitude, respectively. The cost function (5) is formulated from evaluating 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 inter-UAV collisions and to avoid collisions of UAVs with static obstacles as identified by the Mission Planner incorporating a satellite map. It is noted that only static and known position obstacles are considered in this work. External dynamic obstacles can be avoided by using additional sensors such as ultrasonic sensors or Lidars with an extended path planning module. This topic is however beyond the scope of our paper. In operation, the UAVs need to maintain certain distances to the surface as described in (2). Thus, the constraint in flying attitude is also added to the multi-objective cost function (5).

2) UAV Formation: From the reference path generated by the proposed θ-PSO algorithm for the formation centroid, it is necessary to produce a specific path for each UAV to maintain the shape of the formation during the flight. Those paths can be computed from the formation centroid 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$_n$, $n = 1, 2, 3$, in the inertial
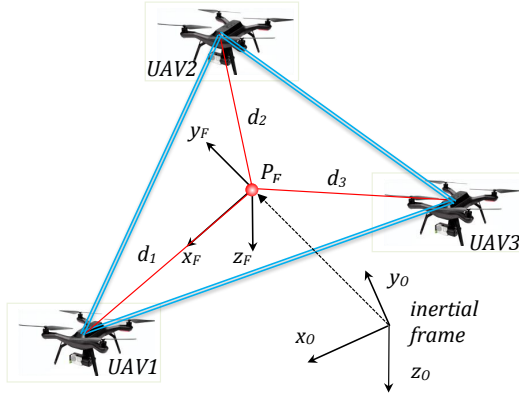
Fig. 3: Inertial and formation frames in UAV formation

frame are denoted as $P_n = \{x_n, y_n, z_n\}$. 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 from the fixed inertial frame as:

$$P_F = \frac{1}{3} \sum_{i=n}^{3} P_n. \tag{6}$$

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

$$R_{OF} = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}, \tag{7}$$

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

Let $P_n = [x_n, y_n, z_n]^T$ and $P_{n,d} = [x_{n,d}, y_{n,d}, z_{n,d}]^T$ be respectively the actual and desired position for each UAV during the flight, in which P_n can be obtained from GPS data of the UAV. The relative position errors of the nth UAV during the flight in the inertial frame are then defined as:

$$\begin{bmatrix} e_{n,x} \\ e_{n,y} \\ e_{n,z} \end{bmatrix} = \begin{bmatrix} x_{n,d} - x_n \\ y_{n,d} - y_n \\ z_{n,d} - z_n \end{bmatrix}. \tag{8}$$

By using the rotation matrix in (7), where $R_{FO}(t) = R_{OF}^{-1}(t)$, the errors in (8) can be converted from the inertial into formation frame as:

$$\begin{bmatrix} e_{n,x}^F \\ e_{n,y}^F \\ e_{n,z}^F \end{bmatrix} = R_{FO}(t) \begin{bmatrix} e_{n,x} \\ e_{n,y} \\ e_{n,z} \end{bmatrix}. \tag{9}$$

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

$$\mathcal{T}_{Fn} = \mathcal{T}_F + \Delta \mathcal{T}_n, \tag{10}$$

where $\Delta \mathcal{T}_n$ is the required trajectory change for the n-th UAV to deviate from the formation centroid. This difference is calculated based on the desired relative distances among the UAVs and the relative position errors in (9), $\Delta \mathcal{T}_n = [e_{n,x}^F, e_{n,y}^F, e_{n,z}^F]^T$. The output $\mathcal{T}_{Fn}$ will be fed to low-level controllers of UAVs for trajectory tracking.

C. Low-level control

The aim of low-level control is to derive the control laws that apply to each UAV's actuators to reach the desired position and attitude. They are derived from 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 $\Theta = (\phi, \theta, \psi)^T$ with the corresponding roll, pitch, and yaw 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.,

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \dot{\Theta}. \tag{11}$$

The transformation from the body to earth frames is then determined by the same rotation matrix as in (7).

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. Therefore, only torque components for orientation of the UAV are considered, with the control input being $\tau = u = [u_\phi, u_\theta, u_\psi]^T$, where u_ϕ, u_θ and u_ψ respectively represent the roll, pitch and yaw components of the thrust torque τ.

The quadcopter dynamics can then be described as follows:

$$\begin{cases} \dot{X}_1 = X_2 \\ \dot{X}_2 = I^{-1} \left[f(X) + u + d \right], \end{cases} \tag{12}$$

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 vector function of the quadcopter moments of inertia I_{xx}, I_{yy}, I_{zz} of the intertia matrix I, and angular rates in the inertial frame [29]:

$$f(X) = \begin{pmatrix} (I_{yy} - I_{zz})qr \\ (I_{zz} - I_{xx})pr \\ (I_{xx} - I_{yy})pq \end{pmatrix}. \tag{13}$$

Of interest at the low level for UAV attitude control are techniques for improving robustness against nonlinearities, uncertainties, and external disturbances. By following the sliding mode control methodology, known to possess highly robust performance, the proposed controller has the form:

$$u(t) = u_{eq}(t) + u_T(t), \tag{14}$$

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 as per desired Euler angle references $X_{1d} = \{\phi_d, \theta_d, \psi_d\}^T$. The sliding function is chosen as:

$$\sigma = \dot{\mathbf{e}} + \Lambda \mathbf{e}, \tag{15}$$

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

The equivalent control u_{eq} is obtained at no disturbance by driving $\dot{\sigma}$ to zero, as follows:

$$u_{eq} = I \left(\ddot{X}_{1d} - \Lambda \dot{e} \right) - f(X). \tag{16}$$
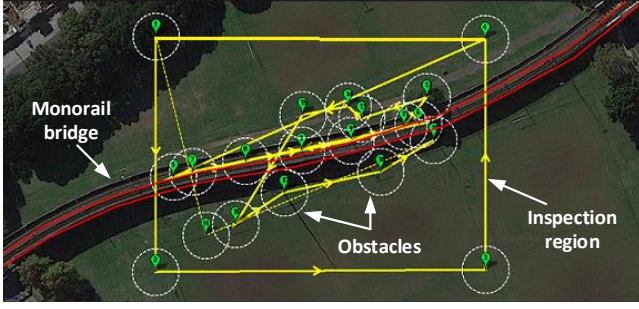
Fig. 4: Mission Planner incorporating Google Satellite Map to create initial information and an inspection plan

Motivated by [30], the discontinuous control here is the twisting controller, $u_{T,i}$, $i = 1, 2, 3$, of the form:

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

where $\mu_i < 1$ is a fixed positive number and $\alpha_i > 0$ is the control gain. By considering the one-stage accelerated twisting algorithm [31] and the adaptive sliding mode control techniques [32] for improving the control transient and tracking performance, the gain α_i in (17) is further adjusted by the adaptation law:

$$\dot{\alpha}_i = \begin{cases} \bar{\omega}_i |\sigma_i(\omega, t)| \text{sign}(|\sigma_i(\omega, t)|^{\rho_i} - \epsilon_i) & \text{if } \alpha_i > \alpha_{m,i} \\ \eta_i & \text{if } \alpha_i \leq \alpha_{m,i}, \end{cases} \quad (18)$$

where $\bar{\omega}_i$, ϵ_i and η_i are positive constants and $\alpha_{m,i}$ is a sufficiently large threshold for adaptation. By assuming the external disturbance d to be bounded, the convergence of the system under the proposed adaptive twisting sliding mode (ATSM) control is obtained in [33].

D. Path planning and formation control implementation

First, the operation space of UAVs is selected according to the infrastructure surface to be inspected. This can be done by using a navigation map with satellite images and a ground control software called "Mission Planner". The software is used to collect initial information about the surface of interest and its surrounding environment based on Google satellite maps (GST). For illustration, Fig. 4 shows the map of a monorail bridge as a testbed subject to inspection loaded by Mission Planner. Correspondingly, the inspection areas, waypoints together with obstacles are identified providing parameters for the cost function and constraints to be defined as in (5). Then, the θ-PSO algorithm, with the pseudo-code presented in Fig.5, is run to obtain the desired path. The path is then uploaded to the UAVs by using Mission Planner for the autonomous flight.

Given the reference 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. After that, the low-level control is applied to UAVs

```
/* Preparation:                                    */
1   Determine the inspection surface;
2   Identify the upper and lower boundaries of the working space, start
    and target positions of the swarm;
3   Identify obstacles in the working space, check and adjust
    obstacles' parameters if needed ;
4   Group all the above data and save in a common file (init file);
/* Initialization:                                 */
5   Initialize the working environment by loading the init file to global
    memory;
6   Initialize 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;
7   Set the range of constraints for each particle's phase angle and
    angular increment in [-π/2, π/2];
/* θ-PSO:                                          */
8   foreach i < (swarm_iteration) do
9       foreach j < (swarm_population) do
10          Calculate new phase angle increment value in the range of
            limitation;      /* using 1st equation in (4) */
11          Calculate new phase angle value in the range of limitation;
                             /* using 2nd equation in (4) */
12          Calculate new position;  /* using 3rd equation in
            (4) */
13          Check Violation cost;
14          Evaluate each path based on the Best_Costs and
            Violation cost;
15          Update each particle_personal_best and the global_best
            positions;
16      end
17      Update global_best and Violation costs;
18  end
19  Save global_best and Violation cost;
/* Path generation:                                */
20  Final path is chosen as the maximum number of iterations is
    reached;
21  Generate individual paths for UAVs using (IV-B2);
```

Fig. 5: Pseudo-code for path generation process

based on (10) to maintain the shape. During the flight, on-board computers calculate the inverse kinematics, obtain position errors with respect to their neighbours and the formation centroid, and then drive these errors to zero with the tracking control.

V. SURFACE INSPECTION

For defect detection, images of the inspected surface taken by UAVs are sent to RCU. Due to a large amount of data to be processed in real time, a fast yet effective image processing algorithm is required. Choosing a particular color space in color image segmentation is largely application dependent [34]. Here, for the sake of camera-based inspection, a surface patch quite often has typically similar or repeated background patterns. Therefore, processing color images may not be as effective as processing grey scale images for the purpose of detecting potential defects, which are defined as abnormal changes in the gray level of pixels. Histogram-based techniques for conversion from $3\times2D$ of RGB images into 1D thus can be used to speed up the calculation. Since existing methods such as the global thresholding (Otsu) [35], valley emphasis (VE) [36], adaptive thresholding (Sauvola) [37], iterative analysis (ITTH) [38] and slope difference distribution (SDD) [39] do not yield satisfactory results if required a higher sensitivity to identify a relatively small number of pixels indicating defects, we seek a new algorithm with an optimal threshold to separate defects from the background.

The proposed algorithm, called iterative thresholding (IT) is illustrated in Fig.6. It begins with the determination of a
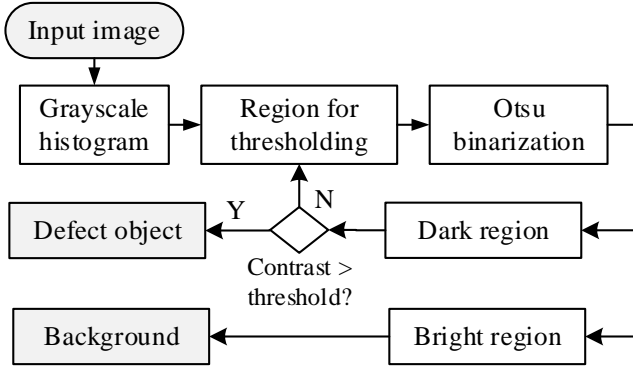
Fig. 6: Flowchart of the defect detection algorithm

threshold value, T^*, similar to the Otsu threshold that can maximize the variance between two classes in the histogram. Given a threshold T, the variance $\sigma^2(T)$ is calculated as:

$$\sigma^2(T) = \omega_0(T)\omega_1(T)(\epsilon_0(T) - \epsilon_1(T))^2, \quad (19)$$

where $\omega_0(T)$ and $\omega_1(T)$ are the weight of foreground and background pixels in the whole image, and $\epsilon_0(T)$ and $\epsilon_1(T)$ are the expectations of the foreground and background intensity:

$$\omega_0(T) = \frac{\sum_{x=0}^{T} y(x)}{\sum_{x=0}^{255} y(x)}, \quad (20)$$

$$\omega_1(T) = \frac{\sum_{x=T+1}^{255} y(x)}{\sum_{x=0}^{255} y(x)}, \quad (21)$$

$$\epsilon_0(T) = \sum_{x=0}^{T} xy(x), \quad (22)$$

$$\epsilon_1(T) = \sum_{x=T+1}^{255} xy(x). \quad (23)$$

The threshold T^* is then given by:

$$T^* = \underset{T \in (0,255)}{\mathrm{argmax}}\, \sigma^2(T). \quad (24)$$

T^* will evenly segment the input image into the dark and bright regions. Without loss of generality, assume that the defect lies in the dark region, our algorithm then continues with an iterative process applying on the dark region.

At iteration k, the threshold T_k^* in (24) will segment the dark region R_{k-1}^D into two new regions R_k^D and R_k^B so that:

$$R_{k-1}^D = R_k^D \cup R_k^B, \quad (25)$$

where R_k^B is the bright region to be treated as the background. To detect defects through the iteration, we use the concept of interclass contrast, a measure for evaluating the segmentation quality, assuming an average intensity for all pixels inside a class [40]. For region R_{k-1}^D, the interclass contrast C_k^D is calculated as:

$$C_k^D = \frac{|\mu_k^D - \mu_k^B|}{\mu_k^D + \mu_k^B}, \quad (26)$$

where μ_k^D and μ_k^B are the intensity means of R_k^D and R_k^B, respectively. As the number of pixels in the thresholding region

decreases after each iteration, the denominator in (26) will decrease leading to the increase of C_k^D. A large value of C_k^D thus indicates a sharp difference in the intensity between the segmenting classes, which implies 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. 7.
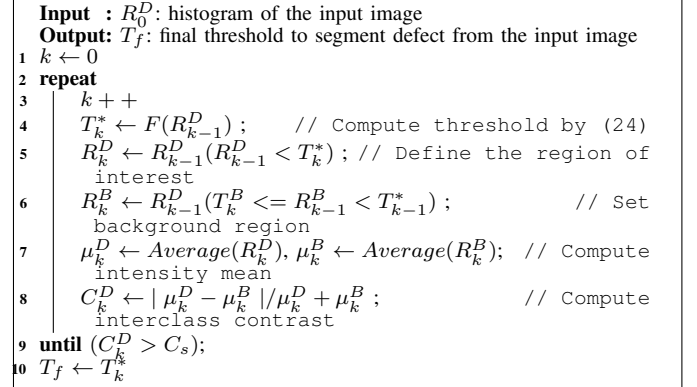
```
Input  : R_0^D: histogram of the input image
Output: T_f: final threshold to segment defect from the input image
1  k ← 0
2  repeat
3      k + +
4      T_k^* ← F(R_{k-1}^D) ;      // Compute threshold by (24)
5      R_k^D ← R_{k-1}^D(R_{k-1}^D < T_k^*) ; // Define the region of
           interest
6      R_k^B ← R_{k-1}^D(T_k^B <= R_{k-1}^B < T_{k-1}^*) ;        // Set
           background region
7      μ_k^D ← Average(R_k^D), μ_k^B ← Average(R_k^B) ; // Compute
           intensity mean
8      C_k^D ← | μ_k^D − μ_k^B |/μ_k^D + μ_k^B ;        // Compute
           interclass contrast
9  until (C_k^D > C_s);
10 T_f ← T_k^*
```

Fig. 7: Pseudo code for defect detection

VI. RESULTS

The performance of our proposed system has been evaluated in a number of surface inspection tasks. This section describes the system testbed and experimental results.

A. Experimental setup

The UAVs used in this study is the 3DR Solo drone shown in Fig. 8. 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 an 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
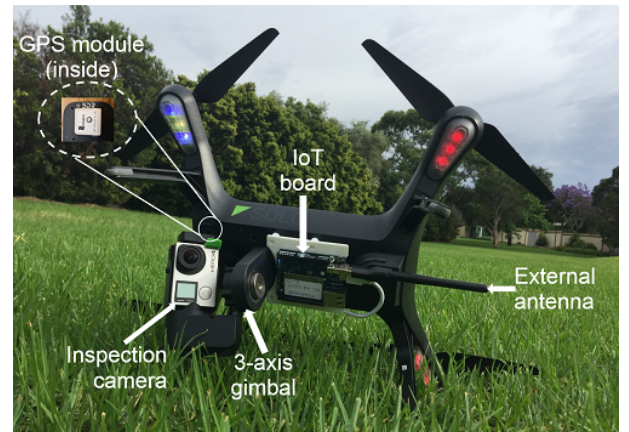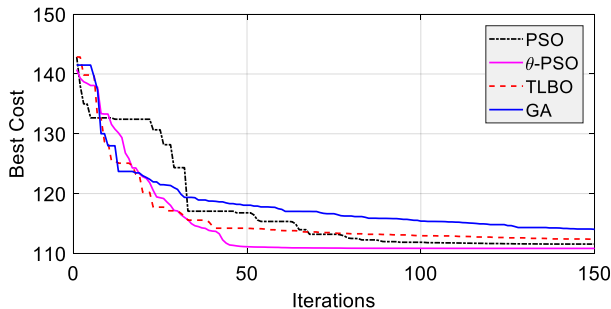

Fig. 8: The 3DR Solo testbed

Fig. 9: Convergence comparison between conventional PSO and θ-PSO

capability. It is attached to a three-axis gimbal with one degree of freedom for controlling its elevation (pitch) angle. The photos taken by this camera will be streamed to RCU. The IoT board named Arduino-yun is attached to the accessory port of the drone and interfaced with the embedded Linux operating system via USB protocol. A 6 dBi Ralink RT5370 detachable antenna is attached to the IoT board to extend the wireless communication range. The gateway router is a TP-Link 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 correction data from the base GPS station to improve positioning precision.

The task assigned in experiments is to inspect a monorail bridge using three UAVs. The operation space is of dimension 141 m $\times$ 101 m $\times$ 40 m equivalent to GST coordinates $\{-33.87601, 151.191182, 0\}$ and $\{-33.875086, 151.192676, 40\}$, as illustrated in Fig. 4. The initial and final positions of the formation centroid (6) are $P_i = \{40.0, 8.0, 30\}$ 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 are $\Delta \mathcal{T}_1 = (0, 0, 2)$ m, $\Delta \mathcal{T}_2 = (3, 0, -1)$ m and $\Delta \mathcal{T}_3 = (-3, 0, -1)$ m.

TABLE I: Performance comparison between GA, TLBO, PSO and θ-PSO

Algorithm	Initial cost	Min cost	Iterations
GA	142.91	114.17	147
TLBO	142.84	113.80	84
PSO	143.00	112.43	102
θ-PSO	142.84	111.02	68

B. High-level control results

The path planning and formation results are presented in this subsection to illustrate the generation of 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 and two other bio-inspired algorithms, Genetic Algorithm (GA) [41] and Teaching-Learning-Based Optimization
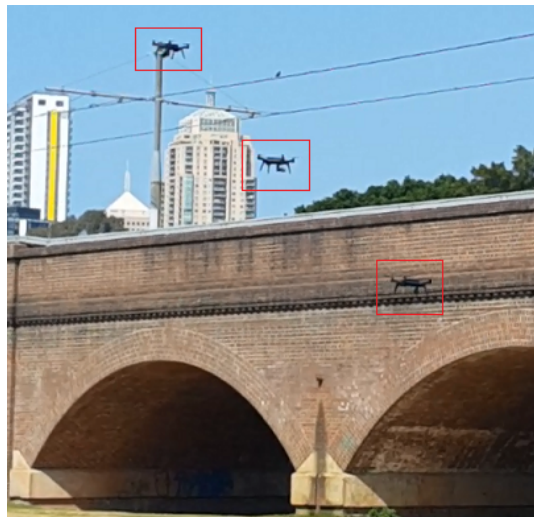
(a) Triangular UAV formation
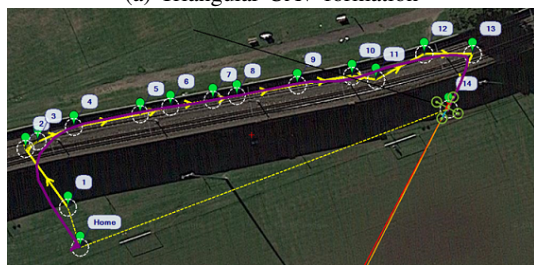
(b) Planned path (yellow) and flown path (violet)

Fig. 10: Bridge inspection with UAV formation

(TLBO) [42]. Figure 9 shows the cost values over iterations, wherein the θ-PSO algorithm exhibits a faster and more stable conversion. The results are confirmed as recorded in Table I showing the cost values and convergence iterations.

Field tests were conducted with the triangular formation automatically navigating along the inspected surface, as depicted in Fig. 10. The 3D trajectories generated from the formation centroid path are shown in Fig. 11, illustrating the three UAVs in taking off, reaching to their individual altitude set-point, descending and finally arriving their target position at almost the same interval while also maintaining the desired triangular shape. This result can be further verified via the altitude time responses of the three UAVs as recorded in Fig. 12. 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. 13 shows the error between the planned and flying paths to indicate the feasibility and reasonable smoothness of the generated path for the deployment of UAV formation.

C. Low-level control results

Control performance of the controller can be 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, it some sudden and significant changes in reference values are applied as $\phi = -10°$, $\theta = 10°$ and $\psi =$
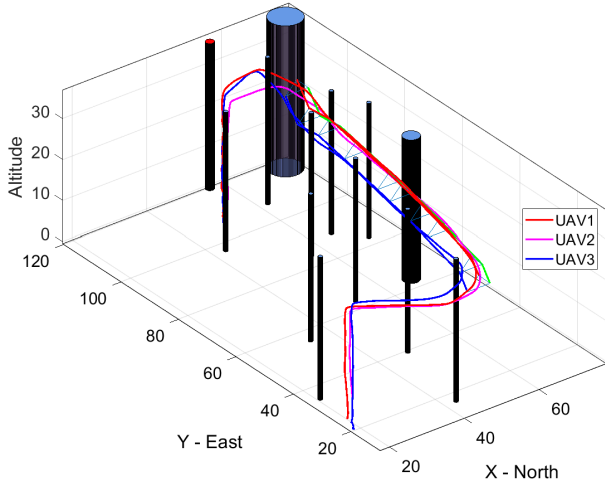
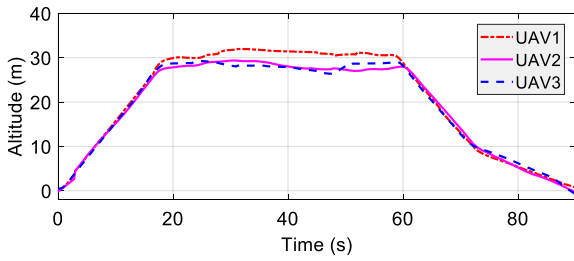Fig. 11: Trajectories of three UAVs tracking the planned paths

Fig. 12: Altitudes of the three UAVs in the formation test

$45°$ at time 0.5 s, 1 s and 2 s, respectively. Simulation results in Fig. 14 illustrate that ATSM effectively rejects disturbances by driving the three angles to their reference values within 2 seconds. Further comparison and evaluation can be found in [33].

D. Surface inspection results

The effectiveness of our proposed defect detection algorithm has been tested on images taken by the camera mounted on UAVs. The results are compared with other methods including Otsu, VE, ITTH, Sauvola and SDD as shown in Figs. 15 and 16 for various surface defects. The input images were taken at different light conditions with various levels of background complexity, i.e., uniform background (Image 1 and 2), patterned background (Image 3), noisy to very noisy background (Image 4 - 10). It can be seen that our algorithm outperforms other methods in all test cases regardless of background complexity or image contrast. Specifically, Otsu
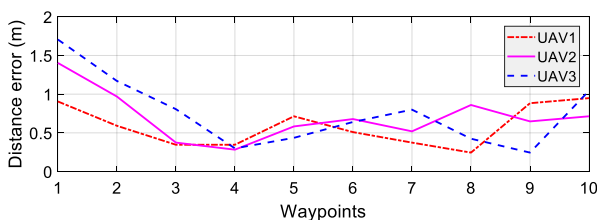

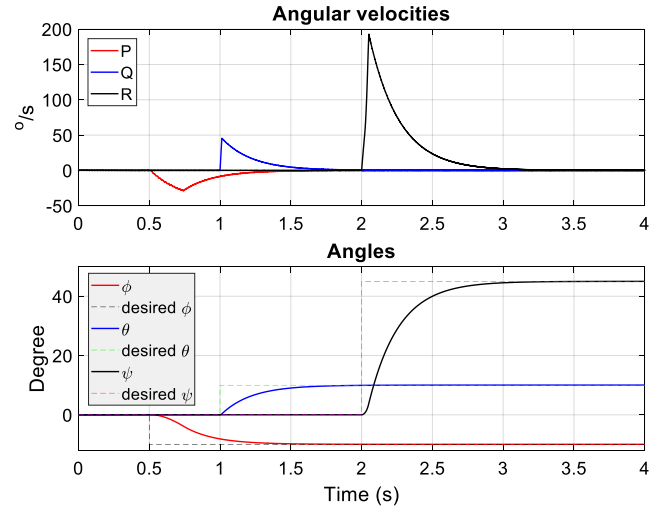Fig. 13: Errors between the planned and flown paths

Fig. 14: Angular velocity and angle responses in the presence of disturbances

and ITTH return a high level of noise on all test images except 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 rather large, but fail to extract cracks on other images. Similarly, SDD could not detect defects in Images 5-8 as it treats bigger classes more favorably.

The comparison results can be quantitatively evaluated via the F-measure, a compromise between recall and precision [43]. Let t_p and t_n be the correctly reported positive and negative results, whereas f_p and f_n be the falsely reported positive and negative results. The F-measure for binary classification is calculated as:

$$F = \frac{2pr}{p + r}, \tag{27}$$

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

$$p = \frac{t_p}{t_p + f_p}, \tag{28}$$

$$r = \frac{t_p}{t_p + f_n}. \tag{29}$$

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.

VII. CONCLUSION

In this paper, we have presented a system architecture for surface inspection in real time using multiple UAVs. The system features a new communication platform based on the Internet of Things which can exploit processing capabilities of RCU and reduce communication distance burdens. Here, a multi-layer paradigm has been introduced to integrate various modules of a complicated system to fulfil a common inspection
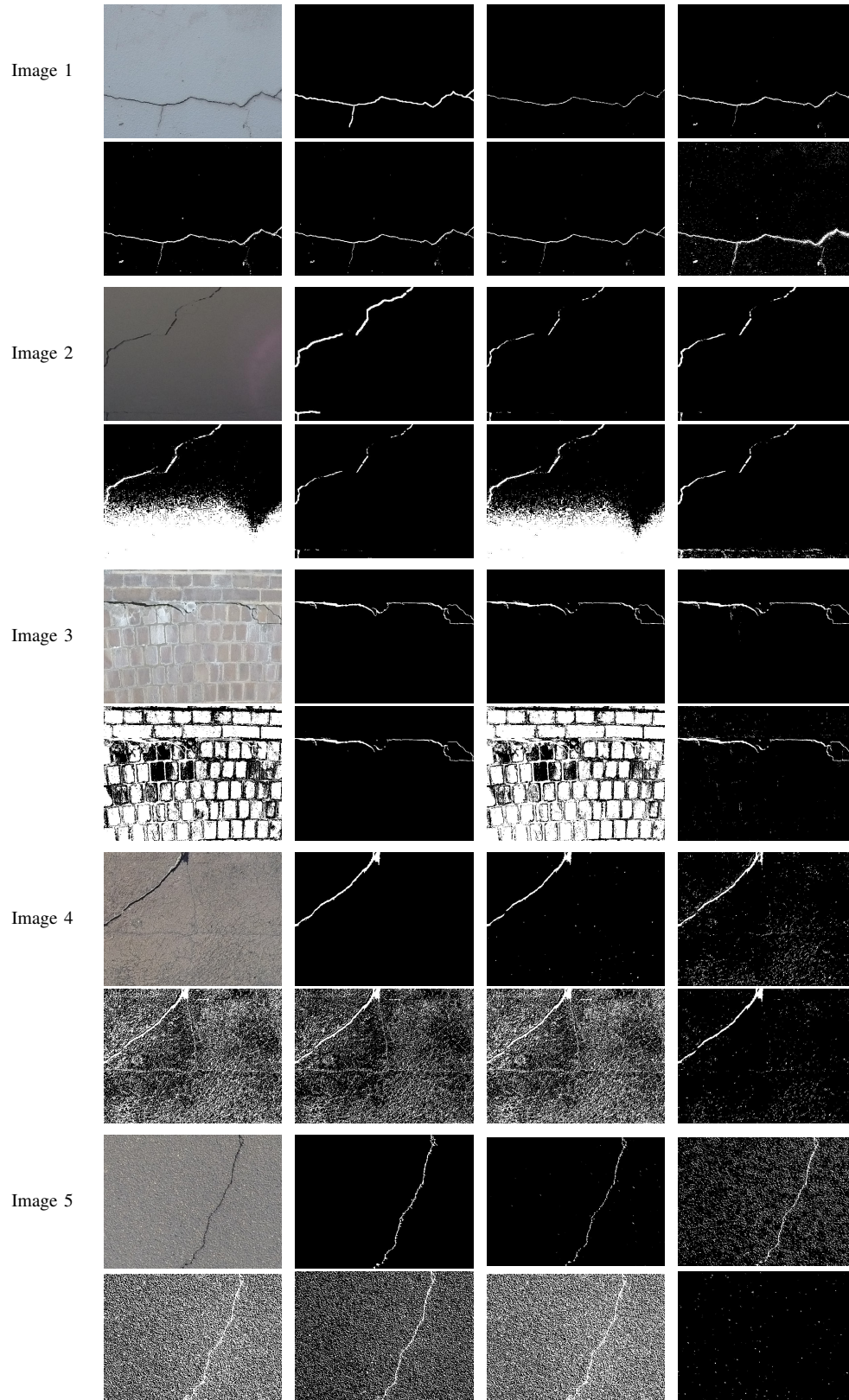
Fig. 15: Defect detection results. First row: image name, original image, ground truth, our result, Sauvola; second row: detection respectively by Otsu, VE, ITTH and SDD.
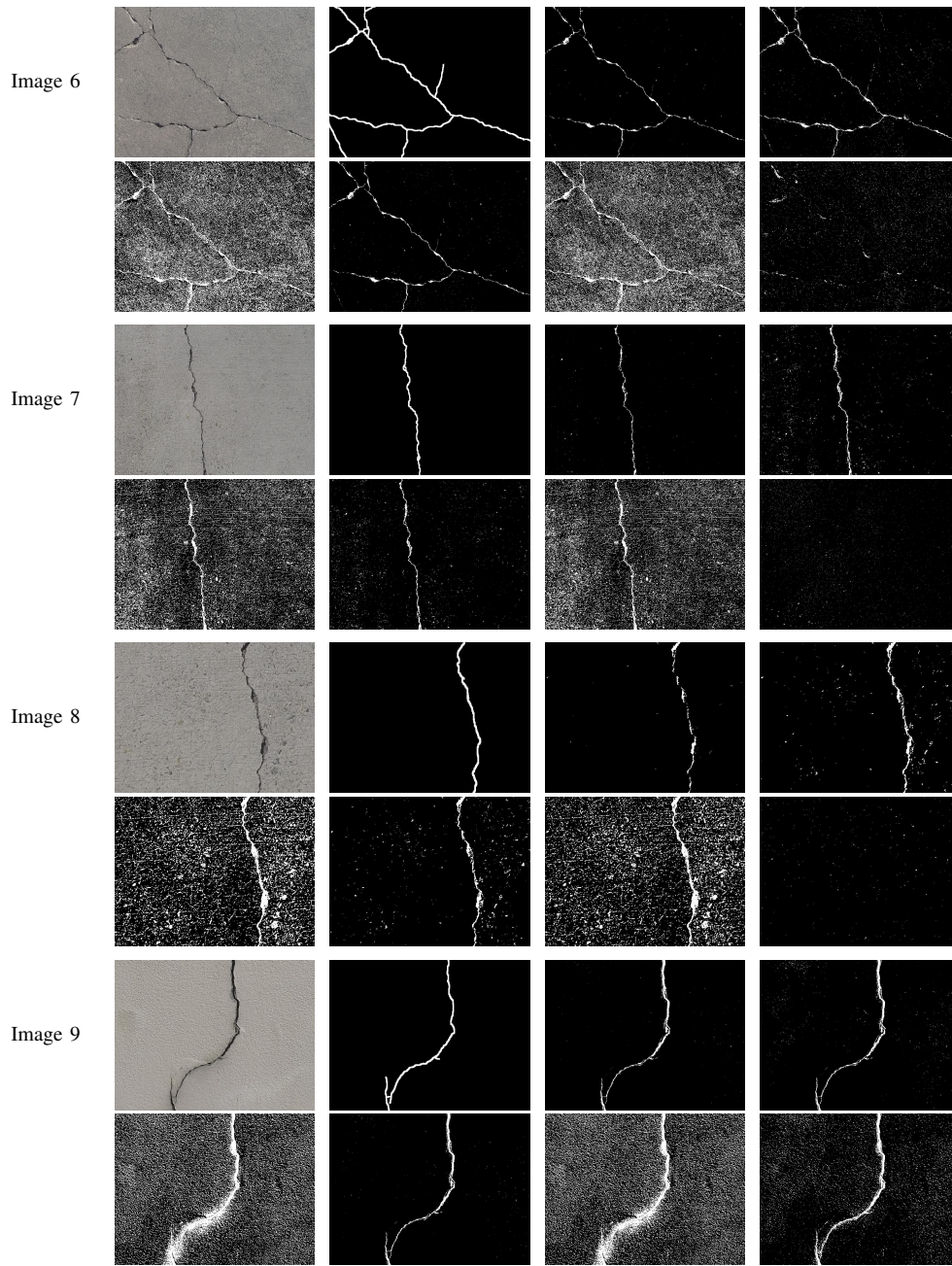
Fig. 16: Defect detection results. First row: image name, original image, ground truth, our result, Sauvola; second row: detection respectively by Otsu, VE, ITTH and SDD.

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

task with UAVs following a predefined geometrical shape. The coordination among UAVs is then carried out via a novel formation algorithm based on the angle-encoded PSO that can create optimal flying paths subject to constraints of the surface to be inspected and collision avoidance. Finally, a histogram-based segmentation algorithm has been developed for online detection of potential defects with high accuracy. A number of experiments have been conducted with real-world defects detected from the data acquired by a triangular formation of UAVs. Comparisons and discussions have been also presented to evaluate the performance of the proposed system.

REFERENCES

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

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous uav guidance," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1, p. 65, Nov 2009.

[8] 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.

[9] 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.

[10] K. Máthé and L. Buşoniu, "Vision and control for uavs: A survey of general methods and of inexpensive platforms for infrastructure inspection," *Sensors*, vol. 15, no. 7, pp. 14 887–14 916, 2015.

[11] 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. Communications*, vol. 9, no. 9, pp. 687–692, 2014.

[12] 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.

[13] J. Lwowski, A. Majumdar, P. Benavidez, J. J. Prevost, and M. Jamshidi, "Bird flocking inspired formation control for unmanned aerial vehicles using stereo camera," *IEEE Systems Journal*, pp. 1–10, 2018.

[14] 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.

[15] E. Stingu and F. L. Lewis, *A Hardware Platform for Research in Helicopter UAV Control*. Dordrecht: Springer Netherlands, 2009, pp. 387–406.

[16] 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.

[17] 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.

[18] 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.

[19] S. Rathinam, Z. Kim, and R. Sengupta, "Vision-based monitoring of locally linear structures using an unmanned aerial vehicle," *Journal of Infrastructure Systems*, vol. 14, no. 1, pp. 52–63, 2008.

[20] 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.

[21] S. Winkvist, E. Rushforth, and K. Young, "Towards an autonomous indoor aerial inspection vehicle," *Industrial Robot: the international journal of robotics research and application*, vol. 40, no. 3, pp. 196–207, 2013.

[22] L. Mejias, J. F. Correa, I. Mondragon, and P. Campoy, "Colibri: A vision-guided uav for surveillance and visual inspection," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, April 2007, pp. 2760–2761.

[23] 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.

[24] J. A. Guerrero and Y. Bestaoui, "Uav path planning for structure inspection in windy environments," *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1-4, pp. 297–311, 2013.

[25] B. E. Schäfer, D. Picchi, T. Engelhardt, and D. Abel, "Multicopter unmanned aerial vehicle for automated inspection of wind turbines," in *2016 24th Mediterranean Conference on Control and Automation (MED)*, June 2016, pp. 244–249.

[26] A. Ellenberg, A. Kontsos, F. Moon, and I. Bartoli, "Bridge deck delamination identification from unmanned aerial vehicle infrared imagery," *Automation in Construction*, vol. 72, pp. 155–165, 2016.

[27] 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.

[28] 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.

[29] 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.

[30] A. Levant, "Sliding order and sliding accuracy in sliding mode control," *International journal of control*, vol. 58, no. 6, pp. 1247–1263, 1993.

[31] Y. Dvir and A. Levant, "Accelerated twisting algorithm," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2803–2807, 2015.

[32] 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.

[33] 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.

[34] N. M. Kwok, Q. P. Ha, and G. Fang, "Effect of color space on color image segmentation," in *2009 2nd International Congress on Image and Signal Processing*, Oct 2009, pp. 1–5.

[35] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.*, vol. 11, no. 285-296, pp. 23–27, 1975.

[36] H.-F. Ng, "Automatic thresholding for defect detection," *Pattern Recognition Letters*, vol. 27, no. 14, pp. 1644 – 1649, 2006.

[37] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern Recognition*, vol. 33, no. 2, pp. 225 – 236, 2000.

[38] 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, March 2014.

[39] 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.

[40] 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.

[41] Y. V. Pehlivanoglu, "A new vibrational genetic algorithm enhanced with a voronoi diagram for path planning of autonomous uav," *Aerospace Science and Technology*, vol. 16, no. 1, pp. 47–55, 2012.

[42] R. Rao, V. Savsani, and D. Vakharia, "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303 – 315, 2011.

[43] I. Pillai, G. Fumera, and F. Roli, "Designing multi-label classifiers that maximize f measures: State of the art," *Pattern Recognition*, vol. 61, pp. 394 – 404, 2017.