Computers and Electrical Engineering 101 (2022) 107992

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Cooperative UAVs framework for Mobile Target Search and tracking[☆]

Aicha Idriss Hentati [a,*], Lamia Chaari Fourati [a], Jihene Rezgui [b]

[a] Digital Research Center of SFAX (CRNS), Laboratory of Signals, systeMs, aRtificial Intelligence and neTworkS (SM@RTS), Tunisia
[b] Laboratoire Recherche Informatique Maisonneuve (LRIMa), Montreal, Canada

ARTICLE INFO

Keywords:
Unmanned Aerial Vehicle
Realistic simulation
Mobile target search
Mobile target tracking
Cooperative UAVs-based systems

ABSTRACT

Recently, cooperative Unmanned Aerial Vehicles (UAVs) have been used in several complex military and civilian applications. Mobile Target Search (MTS) and Mobile Target Tracking (MTT) are among the UAV-based applications that require the involvement of cooperative UAVs. Accordingly, this paper proposes a cooperative UAVs framework for MTS and MTT named (CF-UAVs-MTST). CF-UAVs-MTST is based on the GzUAV co-simulator. It provides a MTS mechanism to generate the aerial waypoints for the UAVs. The MTS algorithm considers the flight speed and altitude, and the resolution of the on-board camera. The MTS algorithm ensures an efficient coverage rate which is in the order of 96.2%. When performing the MTS task, an algorithm runs to detect the target based on a cascade classifier. Further, we provide an MTT mechanism to estimate the target motion and design the optimal tracking path. Simulation results show that CF-UAVs-MTST allows a fast and high precision tracking.

1. Introduction

1.1. Context

Recently, Unmanned Aerial Vehicles (UAVs) [1] have enabled numerous applications thanks to their flexibility, ability to perform complex missions, and low maintenance costs. UAVs have demonstrated their utility and efficiency for Mobile Target Tracking (MTT) [2,3], convoy protection [4], etc.

Mono-UAV-based systems or multi-UAV-based systems can be used for such applications. For mono-UAV-based systems, only UAV-to-Ground (U2G) and Ground-to-UAV (G2U) communications are considered. Thus, the communication between UAVs can only be done through the ground node. Therefore, the performance of a mono-UAV-based system is limited compared to a multi-UAV-based system and especially cooperative UAVs. Cooperative UAVs and swarm UAVs can cover larger areas and perform well in hazardous and harsh environments where conventional equipment or humans cannot stay. Compared to mono-UAVs systems, cooperative and swarm UAVs have a higher success rate compared to mono-UAVs and can quickly accomplish the required task and mission, especially when 3D swarm mobility is exploited. In this work, we use cooperative UAVs for both Mobile Target Search (MTS) and MTT. However, such multi-UAV-based systems require additional algorithms to ensure coordination and self-organization and avoid collisions. Therefore, flight planning is a primary task. The flight plans should be followed by the UAVs without a controlling authority to achieve their common goal. The flight plans are highly dependent on the nature of the task and policies.

☆ This paper is for special section VSI-suav. Reviews were processed by Guest Editor Dr. Bhisham Sharma and recommended for publication.
* Corresponding author.
E-mail address: aicha.idriss@gmail.com (A.I. Hentati).

https://doi.org/10.1016/j.compeleceng.2022.107992
Received 9 November 2021; Received in revised form 25 March 2022; Accepted 6 April 2022
Available online 13 May 2022

1.2. Problem statement

In general, algorithms are designed to activate physical systems. Setting their parameters incorrectly can have disastrous consequences, such as a crash or injury to people. Therefore, before running these algorithms on a real system, they must be properly verified. For example, in the MTS and MTT applications, the biggest challenge is to operate three mechanisms simultaneously (UAVs, flight control, and inter-UAV communication). The UAV simulators such as Gazebo [5] can realistically represent the physical dynamics of UAVs, but the communication between UAVs, which is absolutely necessary for cooperative target search and tracking, may not work well. Network simulators such as the Network Simulator NS-3 [6] can address the problems of packet loss, collisions, or latency. However, this type of simulator cannot support the physical dynamics of UAVs, which is very important for defining paths for MTS and MTT. The above simulator categories do not work together in an integrated way. The research community is working diligently to develop a realistic co-simulation environment that integrates more than one type of simulator.

Although many target detection methods (e.g. deep learning, context-aware classifiers, cascade classifiers) tend to perform MTS, there are still some challenges due to the instability of the image, small size of the target, and fast platform motion. In addition, UAV trajectory planning is an important module for MTS. A flight path must be found that allows the UAV camera to fully cover Region of Interest (RoI). For large RoI coverage, we can use more than one UAV, divide the RoI into some sub-RoIs and assign a UAV to each sub-RoI. Energy is the major constraint for the small UAVs, so the RoI size should match this constraint to fully cover the RoI. Then the selected pattern is applied to each sub-RoI to obtain sub-paths.

Reliability and accuracy of MTT based on cooperative UAVs are challenging tasks because multiple modules are needed to be integrated, including target motion estimation, UAV tracking path generation, coordination and communication between UAVs, etc. Energy is the major constraint for small UAVs, so it is necessary to generate an appropriate tracking path for UAVs. Moreover, systems with multiple UAVs require additional algorithms to ensure cooperative MTT. Coordination can occur in centralized, decentralized, or distributed architectures.

1.3. Contributions & paper organization

We study a variety of co-simulators and evaluate their strategy of interconnection and synchronization of integrated simulators. After reviewing the existing solutions, we select the most appropriate co-simulator that provides a realistic environment for simulating MTS and MTT missions. We propose a cooperative UAVs framework for MTS and MTT named (CF-UAVs-MTST) which uses the co-simulator GzUAV [7]. CF-UAVs-MTST provides a realistic simulation for cooperative UAVs-based systems. It guarantees real-time availability and reliability by integrating three categories of simulators and ensures synchronization between them. Thus, the contributions of this paper are diverse and can be summarized in the following points:

- We propose a cooperative UAVs framework for simulating MTS and MTT algorithms and approaches. CF-UAVs-MTST simulates the UAV software and hardware structure, including the 3D visualization engine Gazebo, the flight controller ArduCopter [8], and the NS-3 [6].
- We provide a MTS mechanism for cooperative UAVs that consists of two mechanisms: (i) a path generation process to generate the ideal ground path and the aerial waypoints for the UAVs and (ii) a mobile target detection process based on a cascade classifier.
- We provide an MTT mechanism for cooperative UAVs that consists of three mechanisms: (i) a target motion estimation process based on the Kalman Filter (KF), (ii) a path planning process to track the target based on its motion estimate, and (iii) a cooperative MTT process that can be invoked by the UAV that has already detected a target.
- We validate the functionalities and analyze the developed framework.

The remainder of this article is organized as follows. Section 2 identifies related work. Section 3 addresses the main concepts of CF-UAVs-MTST. Section 4 provides a cooperative UAVs for MTS mechanism. Section 5 presents a cooperative UAVs for MTT mechanism. Section 6 presents a theoretical analysis of the proposed framework. In Section 7, we explain the simulation environment used to validate the performance of CF-UAVs-MTST and present the evaluation results of the proposed framework. Finally, we draw insightful conclusions in Section 8.

1.4. Abbreviations & Acronyms

All acronyms used in this document can be found at Table 1.

2. Related work

2.1. Co-simulators for UAVs-based systems

UAV-based system simulations can be implemented using a wide range of tools that can be approximate or highly accurate to simulate UAVs running the same real-world program, receiving the same input, and producing the same output. A realistic simulation environment for cooperative UAVs requires the integration of several simulators into a global simulation, called co-simulation. In the following, we examine co-simulators that have been proposed for the simulation of multi-UAV-based systems. We classified them into two categories based on the consideration of the communication aspects of UAVs. Based on a comparative study of these tools, we

Table 1
Acronyms.

AoG	Angle of Gimbal	AoV	Angle of View
API	Application Programming Interfaces	FANET	Flying Ad hoc NETwork
FCU	Flight Control Unit	GCS	Ground Control Station
GDX	Ground Distance in the plan X	GDY	Ground Distance in the plan Y
G2G	Ground-to-Ground	G2U	Ground-to-UAV
KF	Kalman Filter	MTS	Mobile Target Search
MTT	Mobile Target Tracking	PC	Personal Computer
RoI	Region of Interest	ROS	Robot Operating System
SITL	Software-In-The-Loop	UAV	Unmanned Aerial Vehicle
U2G	UAV-to-Ground	U2U	UAV-to-UAV
VNF	Virtual Network Function	ZMQ	Zero Message Queue

select the best tool that provides realistic simulation for cooperative UAVs-based systems. Researchers at [9] have integrated XPlane and OMNeT++ into a co-simulator called AVENS. Communication between the two simulators is done using XML documents. For this reason, bidirectional transmission of telemetry, control data, and sensors is complicated. In addition, AVENS does not consider time synchronization.

The authors of [10] have proposed an integrated simulator based on ArduCopter and NS-3 called FlyNetSim. It includes a middleware layer to connect the two simulators, enabling synchronization to create end-to-end data paths based on the publish–subscribe Zero Message Queue (ZMQ) protocol. The main drawback of FlyNetSim is its lack of accuracy when many events occur in a short period of time.

In [11], the authors have proposed the CUSCUS co-simulator that integrates FL-AIR and NS-3. The communication between the two simulators is done through the tap bridges and the Linux containers. The data exchange based on Linux containers increases the computational costs. In [12], the researchers have proposed an integrated simulator called VENUE. It is based on Linux containers, lightweight Virtual Network Functions (VNFs) and the NS-3 simulator. It enables VNFs to respond with UAV equipment. It covers the simulation phase to the integration phase of real equipment. However, VENUE does not consider mobility model developments.

In [7] the researchers integrated three tools, namely Gazebo, ArduCopter and NS-3 into a realistic co-simulator called GzUAV. They proposed a software module called GzUavChannel to allow the simulators to work with a common time concept. They also used the Application Programming Interfaces (APIs) to exchange data and support distributed simulation.

The researchers of [13], like FlyNetSim [10], used the ZMQ protocol to configure one-to-one correspondence between nodes in NS-3 and UAVs in Gazebo. The proposed co-simulator, named CORNET, creates a continuous data path between UAVs and between UAVs and the Ground Control Station (GCS), with position and time synchronization at both ends. CORNET suffers from the same drawback as FlyNetSim, namely the lack of accuracy for many events in a short time. Based on a deep study on multi-UAV co-simulators, we choose GzUAV for the development of our CF-UAVs-MTST because of its realistic simulation. Moreover, GzUAV guarantees real-time availability and reliability and ensures synchronization between the integrated tools thanks to the use of APIs. GzUAV is extended to support robust algorithms for MTS and MTT.

2.2. Related work of MTS algorithms

The cooperative MTS algorithm consists of finding drone flight paths that cover each point of RoI. The MTS performance depends on the RoI coverage rate. MTS approaches are divided into two categories, including exact cellular decomposition and approximate cellular decomposition [14]. The size of the cells is proportional to the footprint of the camera in the UAV. Several characteristics should be considered when choosing a flight pattern, including route length, number of turns, turn angles, and optimal speeds to minimize UAV energy consumption. Three categories of flight patterns are distinguished in the literature: Random, Spiral, and Lawnmower flight patterns.

- **The random flight pattern:** The authors of [15] chose the exploration strategy random walk. Despite the use of a mechanism to avoid revisiting areas, the full coverage of RoI is not guaranteed.
- **The spiral flight pattern:** It consists of straight lines and 90-degree turns to the left or right. This flight pattern starts from the center of RoI and extends to the borders. The spiral flight pattern is exploited in [16]. In this exploration strategy, a larger number of turning maneuvers are performed, which requires more time and higher energy consumption of the drone. In addition, the 90-degree turn angle adopted by the spiral flight pattern is not suitable for UAV path planning because UAVs are constrained by their turning radius.
- **The Lawnmower flight pattern:** It consists of repeated outbound and return flight lines. This flight pattern is a simple geometric flight pattern and is adopted by the most common flight control software (e.g. Mission Planner). This flight pattern has recently been used in [17]. Compared to the spiral flight pattern, the Lawnmower flight pattern reduces energy consumption by minimizing the number of turns, thus reducing the time of exploration. We adopt this flight pattern in the proposed CF-UAVs-MTST framework.

2.3. Related work of MTT algorithms

In this section, we briefly present previous work on MTT algorithms based on computer vision. The authors of [18] proposed a UAV-based system to track an unmanned ground vehicle. The target detection algorithm uses an improved context-aware correlation filter to reduce the influence of occlusion, target deformation, and light changes. The proposed algorithm used only the gradient histogram feature. Adding other features such as depth can improve the performance of the tracking task. In addition, the researchers of [19] have proposed a UAV-based system for tracking multiple targets and collecting data. UAVs must collect data from critical areas and transmit it to a central location. An intelligent matching algorithm between UAVs and targets is performed. Optimal data collection is determined in a distributed manner using a game theoretic approach. The authors of [20] have proposed a Deep Learning-based method for detecting and tracking a high-speed vehicle. The target detection module relies on a stereo image processing algorithm to detect the target and calculate its location. The target position is transmitted to the UAV controller to track the vehicle. Although the use of a Robot Operating System (ROS) to transmit the captured image to a separate computer for processing is required, the framework requires higher computing power and specialized hardware. In addition, the authors of [21] proposed a hierarchical deep learning task distribution framework. The outsourcing optimization algorithm was performed to minimize the overall weighted costs such as energy consumption and tracking delay, taking into account the quality of the video images. Although the proposed algorithm provided efficient offloading, it did not guarantee high inference accuracy.

3. CF-UAVs-MTST design & architecture

3.1. Integrated tools

The proposed CF-UAVs-MTST framework is based on the integration of three tools: Gazebo, NS3, and ArduCopter. UAVs are simulated using Gazebo; the network and communication are simulated using the network simulator NS-3 and the flight stack is executed using the ArduCopter tool. Below is a brief description of each tool.

- **Gazebo:** It provides realistic simulations for the UAVs and their sensors. It can simulate a large number of UAVs (their models can be defined via an XML file that specifies the geometry and dynamics of the UAV modules, the UAVs' handling characteristics, and the sensors to be used). Gazebo has a modular software architecture that can be extended by plugins. Plugins can interact with Gazebo's APIs as well as with external processes.
- **ArduCopter:** It provides control mechanisms for UAVs. It runs on the FCU, supports connection to a companion computer, has real-time UAV status, and sends velocity and position setpoints. The connection between the companion computer and the FCU can be established using the MAVLink protocol. MAVLink can be used to remotely control the UAV from a GCS via wireless telemetry radios and to send emergency commands and monitor status.
- **NS-3:** It is capable of simulating different types of network protocols and infrastructures. It provides models for different technologies and protocols. The simulations are programmed using simulation scripts.
- **Discussion:** The tools presented above form the basic components for creating a simulation for cooperative UAVs. However, they cannot work in an integrated manner. CF-UAVs-MTST has to solve two main problems. The first problem is related to data exchange. The previously presented tools have different interfaces, so a suitable "translator" must be created. The second problem is related to the synchronization of the clock. The tools presented above should provide a realistic simulation, but each of them has a different notion of time. The integrated simulation environment CF-UAVs-MTST must have a common notion of time. Therefore, another "translator" must interact with each tool to make them work with a synchronized way.

3.2. CF-UAVs-MTST: System design

The architecture of CF-UAVs-MTST consists of several modules related to a specific UAV component in the real scenario. The software modules are the Gazebo visual model, the Gazebo physical model, the CF-plugin, the ArduCopter process, the UAV node, and the mission layer. The Gazebo visual model and the Gazebo physical model define the frame and inertia of the UAV. These models are encoded via the same XML file used by Gazebo to simulate the physics of the UAV and represent it in the 3D scenario. The frame must be connected to the UAV flight stack. ArduCopter needs to get the information about the position of the frame (Euler angle and Euler angle rates, geographic coordinates, etc.) and send the correct commands to the motors. The CF-Plugin is derived from the ArduCopter plugin available in Gazebo. It has been modified to include synchronization of activities with the other components of the integrated simulation environment.

The ArduCopter process is compiled to run on Personal Computer (PC) platforms (not a real FCU) and supports SITL mode. It can monitor the flight of the simulated UAV by interacting with the CF-Plugin.

The UAV node represents the wireless interface of the companion computer. Its instance runs in the NS-3 process to simulate the wireless communication channel. CF-UAVs-MTST supports both IEEE 802.15.4 and IEEE 802.11 wireless standards, which can be selected when the simulation is started.

The mission layer is the module that runs on top of the user software and implements the cooperative UAVs for both MTS and MTT. It is a Python script that uses the services of two Python APIs: DroneKit and NS3interface. It contains the behavior of a single drone at the mission level, therefore each UAV has its own Python process.

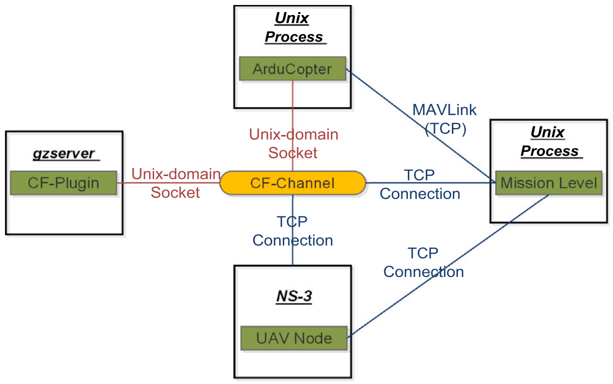

Fig. 1. Relationships between the CF-Channel and the components [7].

Fig. 1 presents the relationship between the software components of the CF-UAVs-MTST and CF-Channel. Communication between ArduCopter and the mission level instances is done via a TCP connection used through DroneKit to encapsulate the MAVLink protocol. The transmission of data packets across the UAV networks is also done using a TCP connection. The use of TCP connections allows the distribution of the different components to different computers.

3.3. Time synchronization

The dynamical systems handle time based on the specific nature of the system. The physical systems simulated by Gazebo are modeled based on differential equations to control dynamics and kinematics. At each iteration, the sampling time corresponds to a time interval used to update the state information. Such simulation policy is called time-controlled. Moreover, the ArduCopter flight controller implements the control algorithms based on differential equations. Thus, it is a time-driven process.

The simulated network systems do not support continuous timing, but they are characterized by sporadic events, e.g. the UAV is silent until it has packets to send. Such simulation policy is referred to as event-driven. Events are retrieved and processed by an appropriate scheduler. In addition, events can have a timestamp indicating when the event needs to be processed. As we can see, CF-UAVs-MTST has to consider tools with different simulation strategies: event-driven for NS-3 and time-driven for Gazebo and ArduCopter.

In the CF-UAVs-MTST simulation environment, the sampling period is generated by Gazebo. When an event occurs, all plugins of Gazebo are notified, including CF-Plugins. When CF-Plugin is notified, it retrieves the appropriate data from the associated UAV sensors and sends it to CF-Channel. This message informs CF-Channel about the beginning of a new simulation period. The simulation period is divided into two different phases. In the first phase, the UAV related activities are executed and in the second phase, NS-3 performs the network simulation.

The division of the simulation period into two different phases allows the synchronization of the transmission activities. Namely, in the first phase the mission algorithm can be executed (like MTS, MTT algorithms). This phase must include (i) the calculation of the next speed or position of the UAVs and their transmission to ArduCopter through the MAVLink protocol (ii) the transmission of messages to be simulated by NS-3. In the second phase, NS-3 starts simulating the data transmission. When it is necessary to transmit a data packet to a target UAV, NS-3 contacts the corresponding UAV node.

4. Cooperative UAVs for MTS

Compared to MTS based on ground vehicles, MTS based on air vehicles, presents several challenges. For example, a ground vehicle can make 90-degree turns to rotate in place, while UAVs are constrained by their turning radius. In addition, energy efficiency is an important challenge for aerial MTS. The size of the RoI depends on various factors, such as the UAV's flight duration, maximum flight time, flight speed, acquisition width, energy required, fuel consumed to reach the starting point for sweeping an RoI, etc. Therefore, the path generation process should take into account the capabilities of the UAVs, their characteristics such as fuel consumption, and flight speed, etc.

In this paper, we consider a team of n UAVs, $UAVs = \{UAV_1, UAV_2, .., UAV_n\}$. It is assumed that there is only one target in a wide RoI. RoI is decomposed into n sub-RoI, $RoI = \{RoI_1, RoI_2, .., RoI_n\}$. Let $\mathrm{RoI_i}$ be associated with $\mathrm{U_i}$. It is assumed that the sub-RoIs are equals.

Each UAV is equipped with a sensor camera and must provide full coverage of its sub-RoI with low power consumption. Therefore, the projection of the UAV camera on the ground, called the camera-ground footprint distance, is important to generate the ideal ground path and thus the aerial waypoints for the UAVs.

Area decomposition and path generation are performed by the GCS in a centralized manner to reduce computational overhead. The GCS knows the capabilities and characteristics of the drone such as fuel consumption, maximum flight speed, camera resolution, etc. Based on this information, the GCS calculates the appropriate ground path and converts it into aerial waypoints.

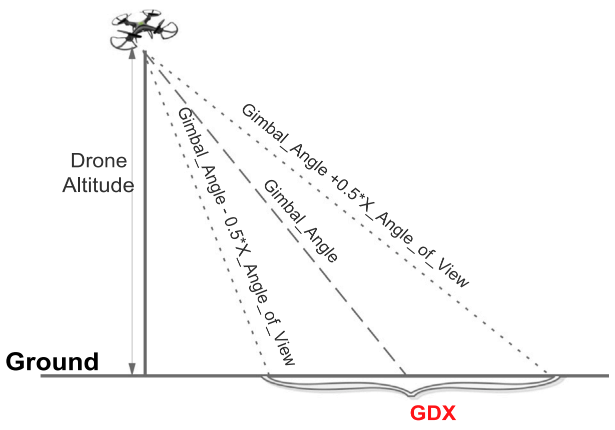
Drone Altitude
Gimbal_Angle +0.5*X_Angle_of_View
Gimbal_Angle
Gimbal_Angle - 0.5*X_Angle_of_View
Ground
GDX

Fig. 2. Ground distance in the plan X.

4.1. Camera ground footprint distance

The ground footprint distance of the UAV camera depends on several factors, including flight altitude alt, focal length $focallen$, camera resolution $(width * height)$, and the Angle of Gimbal (AoG).

The Ground Distance in plane X (GDX) and the Ground Distance in plane Y (GDY) are calculated with the formulas (1) and (2), respectively.

$$\begin{aligned} GDX = alt * (tan(AoG + 0.5 * X\_AoV))- \\ alt * (tan(AoG - 0.5 * X\_AoV)) \end{aligned} \tag{1}$$

$$\begin{aligned} GDY = alt * (tan(AoG + 0.5 * Y\_AoV))- \\ alt * (tan(AoG - 0.5 * Y\_AoV)) \end{aligned} \tag{2}$$

where X_AoV and Y_AoV are the Angles of View (AoV) representing the visible part of the space through the camera. The AoV depends on the camera characteristics, namely the width (mm), the height (mm), and the $focallen$(mm).

The X_AoV and Y_AoV are calculated with the formulas (3) and (4), respectively.

$$X\_AoV = 2 * atan(width/(2 * focallen)) \tag{3}$$

$$Y\_AoV = 2 * atan(height/(2 * focallen)) \tag{4}$$

Fig. 2 illustrates the method to calculate GDX.

4.2. Path generation process

The path generation process consists of generating the ideal ground path and the aerial waypoints for the UAVs. It is a primary mechanism that depends on the distance of the camera from the ground. Any location on the ground should be captured with one pass of the UAV camera. Therefore, it is necessary to establish the ideal ground path of the camera's center of gravity. Then, the resulting ground path is converted into aerial waypoints that the UAVs will follow.

- Generating the Ground Path
  The ground path is the path of the camera footprint in the RoI. We chose to use the lawnmower pattern, which is a repetitive back-and-forth configuration based on the camera's ground distance (GDX and GDY). The ground path allows to capture any location on the ground in one pass. Fig. 3 presents a typical lawnmower pattern based on the perimeter pattern of RoI. In Fig. 3, the red path provides full coverage of the area, but it lets UAV fly beyond the perimeter of the RoI, so the green path is the correct pattern. Note that a path with many turns requires more time and more energy than a path with few turns. Fig. 4 shows the two possible paths for an RoI. It is necessary to generate a path that minimizes the number of turns of the UAV to accomplish the task MTS.

- Waypoints Transformation
  After the ground pattern is created for each UAV, the waypoint transformation is performed. The request to reach a specific waypoint is made through DroneKit. The UAV must follow the generated waypoints to the desired ground point while maintaining the camera footprint. Due to minimum turning radius specifications, the flight path may be outside the sector assigned to the UAV. A final check should ensure that all waypoints are within the allowed flight envelope.

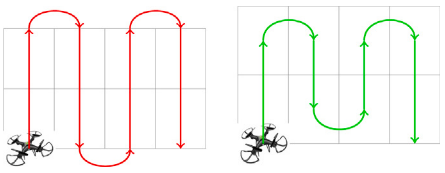
Fig. 3. Two possible lawnmower pattern paths.

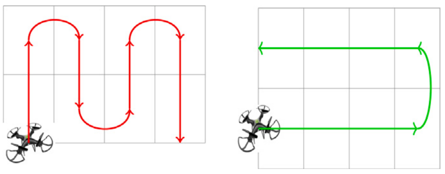
Fig. 4. Two possible paths for an RoI.

Based on the list of generated waypoints, each UAV performs the MTS task in its RoI. While performing this task, an algorithm runs to detect the target.

4.3. Mobile target detection

For target detection, we adopted the famous approach of [22], which combines AdaBoost with a cascade process. This allows extremely fast recognition rates to be achieved. To compute the Haar features quickly, integral image representation for images is introduced. The features are used as weak learners, and AdaBoost is used to weight these learners. The cascade classifier is used to discard multiple regions of the image background and focus the computations on the target regions. The cascade function uses a variety of .xml files with different feature sets. In our case, we use the haarcascade_fullbody.xml and haarcascade_pedestrian.xml files to identify the features of the pedestrian's body with multiscale detection.

Implementing the target detection algorithm in the context of CF-UAVs-MTST framework requires performing several steps. The first step is the insertion the target into the Gazebo 3D visualization interface. Gazebo allows the inclusion of a variety of other objects in the simulated environment. In Gazebo, an animated model is referred to as an actor that does not interact with the rest of the simulation. We added a skeleton animation that moves randomly at different speeds. Assuming that the target is intelligent, when it notices UAV pursuit, it performs evasive maneuvers, such as increasing its speed to avoid tracking.

The second step is to include the target detection module at the mission level. We used the CascadeClassifier function from the OpenCV library to detect the pedestrian's body on the captured image.

4.4. Multi-UAVs for MTS

The proposed CF-UAVs-MTST framework uses the IEEE 802.11 wireless standard for the created FANET that is simulated by NS-3. The NS-3 interface contains functions for sending and receiving packets. For the encapsulation/decapsulation processes, which consist of packing/unpacking data packets during network communication, we used Python's *struct* library. It can handle the binary data of the network. The FANET consists of a GCS and a number of UAVs. Initially, each FANET node has a MAVLink identifier. This identifier is used when MAVLink messages are transmitted. Among the first packets sent from the GCS to each UAV $\mathrm{U_i}$ are the packets containing the coordinates of the starting point, air waypoints, altitude, airspeed, and AoG. The received waypoints are converted into commands to be executed by the UAV. It should be noted that the UAV firstly flies in "GUIDED" mode (e.g. to perform the turn) and then switches to "AUTO" mode to receive the commands.

5. Cooperative UAVs for MTT

Once the target is detected, it is necessary to predict the next target position (after a short time). This information is important to change the trajectory of the UAV to track this target.

5.1. Target motion estimation

The KF [23] is used to estimate the next target position. KF is a recursive algorithm for a linear filtering problem with discrete data. KF has been used extensively in research, especially in the field of assisted or autonomous navigation. KF is essentially a set of mathematical formulas for a predictor–corrector estimator that minimizes the estimated error covariance. KF consists of two steps, namely the prediction step and the update step. In the prediction step, the next state of the system is predicted based on the previous measurements. In the update step, the current state of the system is estimated based on the measurements in that time step.

5.2. Path planning

Based on the target's motion prediction, the UAV should change its path while maintaining a certain distance from the target. The target should be in the center of the camera image. To enable automatic sequencing and timing, there are soft transitions between waypoints based on the estimation of the target speed. The list of waypoints to be reached in a sequence is generated by the mission module based on the estimated target speed. The request to reach a specific waypoint is executed via DroneKit.

5.3. Cooperative UAVs for MTT

To improve the MTT success rate, the MTT task is performed cooperatively. Once the target is detected, the UAV should send this information to the GCS while continuing to send its current waypoint to neighboring UAVs (which are within its communication range). The decision to request tracking assistance can be made in two critical cases.

- If the UAV cannot track the target for a period of P, it sends a request for help to all neighbor UAVs.
- If the UAV's fuel is insufficient to complete the MTT task, the UAV can select another UAV from the list of nearest UAVs to complete the MTT task, or the GCS will activate another recovery UAV.

6. Theoretical analysis & computational complexity

To explore the characteristics, benefits, and limitations of the proposed framework, a theoretical analysis is presented to provide further insight into building effective cooperative UAVs networks.

6.1. Theoretical analysis

Limited energy and flight time are the main challenges for wide-area coverage by cooperative UAVs. Therefore, we need to adjust some parameters to overcome these challenges. We need to find an optimal distance to the ground while ensuring accurate coverage of camera movements. Suppose the area size is $1000 * 1000$ m^2 and 9 UAVs must be deployed to cover the entire area. As mentioned earlier, the area coverage depends on the distance of the UAV camera from the ground. Based on the Eqs. (1), (2), (3) and (4), the ground coverage depends on several factors, including the flight altitude, focal length, camera resolution, and AoG. The proposed CF-UAVs-MTS framework uses a Iris quadcopter equipped with a CGO gimbal camera that has the following specifications: the focal length $focallen$ is equal to 91.28 mm and the camera resolution is $(640 * 480)$ pixels. The gimbal angle is fixed at 30 degrees.

The flight time T is calculated with the formula (5).

$$T = \frac{L}{V} \tag{5}$$

where V is the flight speed and L is the flight path length. The trajectory length L is calculated using the formula (6).

$$L = m * SL + (m-1) * (XTA + YTA) \tag{6}$$

where

- m (m is an integer ≥ 1) and SL are the number and length of straight lines, respectively.
- XTA and YTA are the lengths of the turning arcs on the planes X and Y, respectively.

The formula (7) defines the number m of straight lines.

$$m = \frac{min(RoI\_width, RoI\_height)}{GDX} \tag{7}$$

where RoI_width and RoI_height are the width and height of the RoI, respectively.

If $min(RoI\_width,\ RoI\_height)\ mod\ GDX \neq 0$ then m is increased by 1.

The formula (8) defines the length of straight lines.

$$SL = max(RoI\_width, RoI\_height) - 2 * GDY \tag{8}$$

Formulas (9) and (10) define the length of turning arcs XTA and YTA, respectively.

$$XTA = GDX * \pi/2 \tag{9}$$

$$YTA = GDY * \pi/2 \tag{10}$$

The CF-UAVs-MTST framework uses the Iris quadcopter. This copter is constrained by the average flight time, which should be between 10 and 15 min. The acceleration and the time needed to go from the starting point to sub-RoI and to return from sub-RoI to the starting point must be taken into account when calculating the maximum flight time. For this reason, we have assumed a maximum flight time of 10 min. Using a higher speed can solve the flight time problem, but the flight speed will affect the battery life of the drone. In the CF-UAVs-MTST framework, we used a flight speed of 8 m/s, based on the study of the authors of [24], which shows that the power consumption remains below 150 W up to 8 m/s, above which the power consumption increases rapidly.

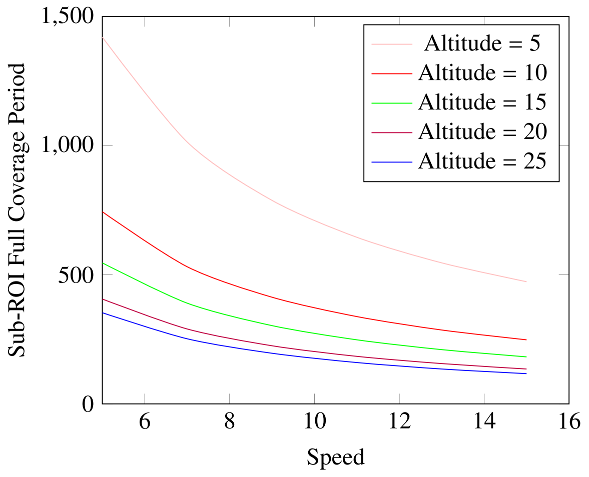

Fig. 5. Coverage period per the UAV speed.

Since high flight altitudes provide basic information for mapping, we used low flight altitudes to obtain more accurate information and improve detection accuracy. However, as the flight altitude decreases, the ground footprint distance decreases. The aforementioned challenges are studied in detail to select the optimal flight speed and altitude for accurate MTS and MTT. Fig. 5 shows the full coverage period per flight speed at different flight altitudes. As can be shown, by selecting the period of covering the sub-RoI less than the maximum flight time (10 min) and the optimal flying speed 8 m/s, the optimal flying altitude is 10 m. Another prominent metric to be considered is the average coverage rate, which is often used to evaluate MTS algorithms. It describes the percentage of RoI covered by the cooperative UAVs. According to the formulas (1) and (2), GDX and GDY are proportional to the UAV altitude alt. Moreover, according to the formulas (9) and (10), GDX and GDY are proportional to the turning arcs XTA and YTA, respectively. Therefore, XTA and YTA are proportional to the UAV altitude alt. However, larger turning arcs let the edge of the RoI uncovered and may reduce the coverage rate, as shown in Fig. 6, which plots the coverage rate of RoI as a function of UAV altitude. As can be shown, an increase in altitude leads to a decrease in coverage rate. Based on the previous theoretical analysis, the altitude of flight is set at 10 m. At this altitude, the average coverage rate is 96.2% which presents an efficient coverage rate.

6.2. Computational complexity

To reduce the computational complexity of the MTS algorithm, the computation of the path of the UAVs is performed centrally by the GCS. Thus, this complexity is independent of the number of UAVs. It is related to the rules executed by each UAV. In the worst-case, the target detection is performed at the end of the MTS. In this case, the complexity of the MTS algorithm is $\mathcal{O}(m)$, where m is the number of straight lines of the lawnmower flight pattern. The limited camera FOV leads to greater complexity. This complexity can be reduced by using high-resolution cameras. The Haar feature-based cascade classifier is used for target detection. For each image, an integral image is computed with a few operations. The computation of Haar features for the integral image enables accurate target detection in constant time. The total complexity of the cascade classifier is $\mathcal{O}(width * height)$, where $width$ and $height$ are the image width and height, respectively. Therefore, the calculation of the integral image can drastically reduce the calculation time. The entire complexity of the detection algorithm is $\mathcal{O}(I\_width * I\_height)$ where I_width and I_height are the integral image width and height, respectively.

KF is used for the estimation of the target state. Its complexity is $\mathcal{O}(b^3)$, where b is the number of system states. This complexity is low compared to the complexity of a Particle Filter which uses a set of particles to estimate the distribution of a process. The complexity of the Particle Filter is $\mathcal{O}(B * b^2)$, where b is the number of particles. This number should be large enough for the particle filter to perform well.

The total complexity of the proposed framework is the sum of the complexities presented earlier, it is $\mathcal{O}(m + I\_width * I\_height + b^3)$.

7. Framework performance analysis

7.1. Simulation parameters

Table 2 presents the simulation parameters.

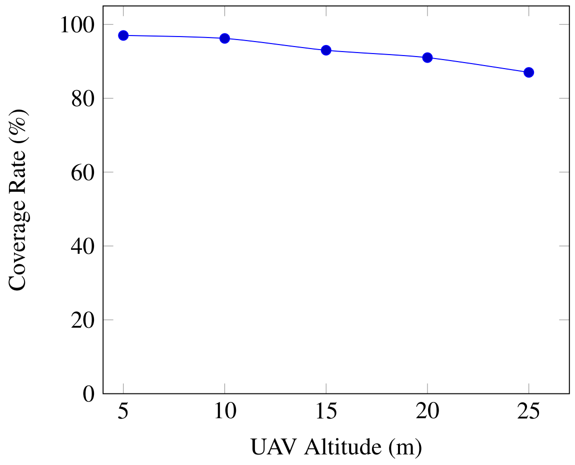

Fig. 6. Coverage rate per the UAV altitude.

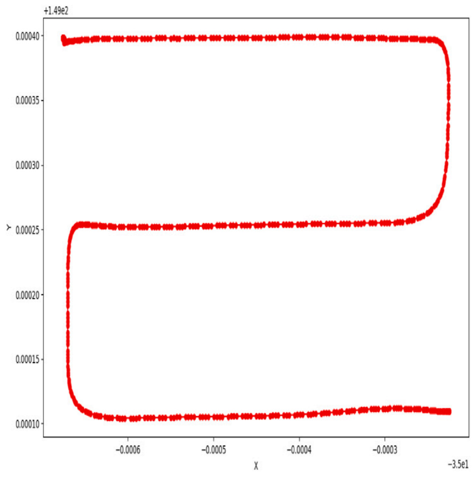

Fig. 7. Typical path for a UAV Performing MTS.

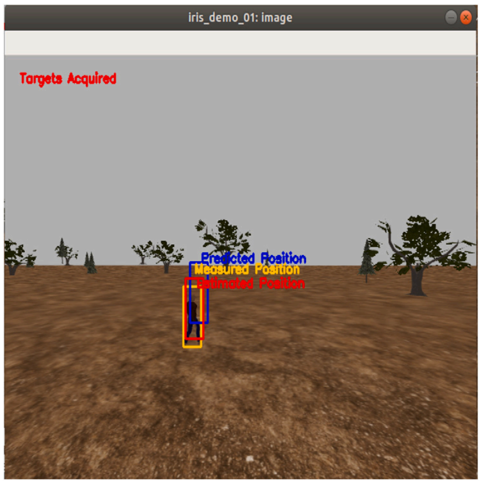

Fig. 8. UAV camera capture when detecting the target.

Table 2
Simulation parameters.

Parameters	Value
UAV type	Quad-rotor, Iris
UAVs number	9
Camera type	CGO with Gimbal
Frame rate	30 f/s
Image format	BGR
Image resolution	640 ∗ 480 pixels
UAV altitude	10 m
Flight speed	8 m/s
Area size	1000 ∗ 1000 m
Wireless standard	IEEE 802.11 n
Carrier frequency	2.4 GHz
Maximum transmission range	250 m
Packet size	512 Bytes
Data rate	11 Mb/s

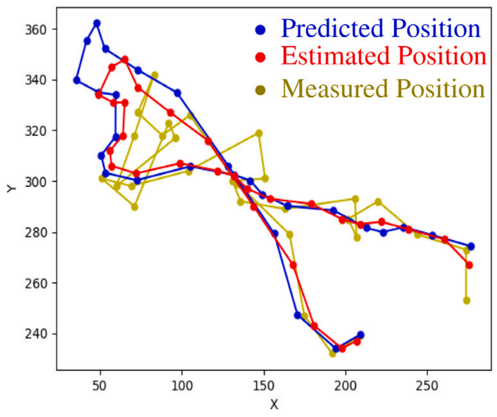

Fig. 9. Estimated, predicted, and measured paths of the target.

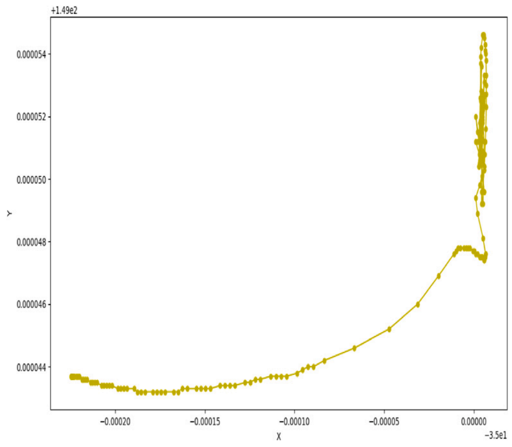

Fig. 10. Path of the UAV detecting the target.

7.2. Simulation results

During MTS, the path generation algorithm worked well for the simulations. It generated the ideal aerial waypoints for the UAVs. Fig. 7 shows a typical path for a UAV performing MTS. UAVs aim to detect targets that roughly resemble the human skeleton. Fig. 8 shows the image of the UAV camera detecting the target. The image contains three bounding boxes for the target: the red rectangle shows the estimated position, the blue rectangle shows the predicted position, and the yellow rectangle shows the measured position.

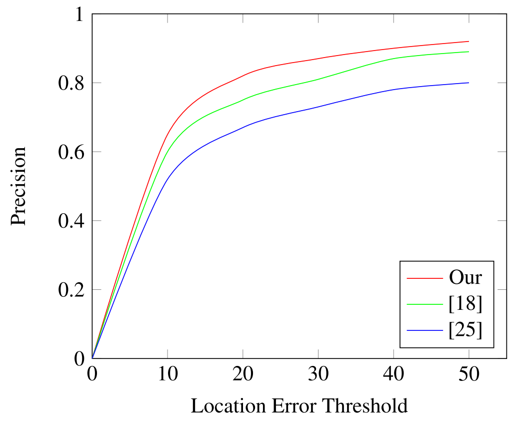

Fig. 11. Precision plot.

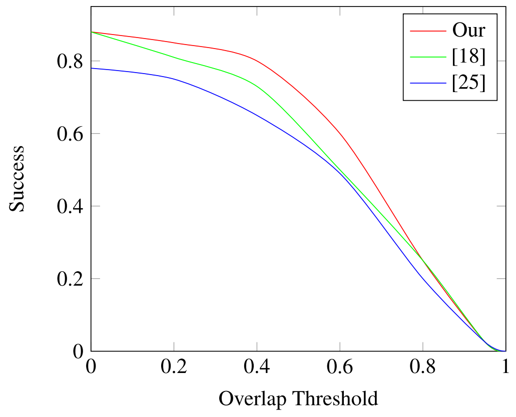

Fig. 12. Success plot.

Fig. 9 returns the estimated, predicted, and measured positions of the target during the simulation. As can be shown, KF worked well to estimate the movement of the target. Fig. 10 presents the path of the UAV that detects the target.

Precision and success plots are the most common metrics used to evaluate MTT algorithms. Precision is the average of the Euclidean distance between the center of the estimated bounding boxes and the center of the measured bounding boxes. The precision plot shows the rate of frames whose precision is below the position error (pixels) threshold. Fig. 11 shows the precision plot. As can be seen, the proposed MTT algorithm performs well and has a high precision rate compared to [18,25]. During the simulations, the framework shows almost continuous tracking of the target even when the scale of the target changes and the target is occluded.

The success is the intersection of the pixels of the estimated bounding boxes and the measured bounding boxes. The success plot presents the rate of frames whose success is greater than the overlap threshold. Fig. 12 reveals the success plot. As can be seen, the proposed MTT algorithm performs well and presents a high success rate compared to [18,25]. Despite its robustness and accuracy, the CF-UAVs-MTST framework suffers from some limitations. For instance, it does not consider the case where the target speed is considerably higher than the UAV flight speed. Moreover, it does not support 3D target motion estimation.

8. Conclusion

The proposed CF-UAVs-MTST framework presents a realistic simulation environment for simulating MTS and MTT tasks based on cooperative UAVs-based systems using the integration of three tools, namely Gazebo, NS-3 and ArduCopter. The MTS algorithm takes into account the onboard camera resolution, flight time and speed, and limited energy constraints to generate a search path that should guarantee nearly complete coverage of RoI. Each UAV is assigned to a specific RoI and given the aerial waypoints. When performing the MTS process, the target detection algorithm based on a cascade classifier is running, which allows identifying the features of the pedestrian's body. The UAV that detects the target must estimate its trajectory based on the KF to track it. When the UAV detects a target, it should inform the GCS and its neighbor UAVs to let them know its current location and the location of the target. Sharing this information among UAVs is important to perform the MTT task cooperatively and improve tracking accuracy. The proposed framework is evaluated through extensive simulations. The simulation results prove the robustness of the developed framework CF-UAVs-MTST. Our current work focuses on proposing a routing protocol for FANETs that takes into account the characteristics of UAVs such as high mobility, limited energy and network lifetime, etc.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Hentati AI, Fourati LC. Comprehensive survey of uavs communication networks. Comput Stand Interfaces 2020;103451.
[2] Hentati AI, Fourati LC. Framework for uav mobile object tracking based on ue4sim. In: 2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM). IEEE; 2019, p. 1–6.
[3] Hentati AI, Fourati LC. Mobile target tracking mechanisms using unmanned aerial vehicle: Investigations and future directions. IEEE Syst J 2019.
[4] Hentati AI, Fourati LC. A convoy of ground mobile vehicles protection using cooperative uavs-based system. In: 2021 International Symposium on Networks, Computers and Communications (ISNCC). IEEE; p. 1–6.
[5] Hentati AI, Krichen L, Fourati M, Fourati LC. Simulation tools environments and frameworks for uav systems performance analysis. In: 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC). IEEE; 2018, p. 1495–500.
[6] Carneiro G. Ns-3: Network simulator 3. In: UTM Lab Meeting April, Vol. 20. 2010, p. 4–5.
[7] D'Urso F, Santoro C, Santoro FF. An integrated framework for the realistic simulation of multi-uav applications. Comput Electr Eng 2019;74:196–209.
[8] Meier L, Honegger D, Pollefeys M. Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE; 2015, p. 6235–40.
[9] Marconato EA, Rodrigues M, Pires RdM, Pigatto DF, Pinto AR, Branco KR. Avens-a novel flying ad hoc network simulator with automatic code generation for unmanned aircraft system. In: Proceedings of the 50th Hawaii International Conference on System Sciences. 2017.
[10] Baidya S, Shaikh Z, Levorato M. Flynetsim: An open source synchronized uav network simulator based on ns-3 and ardupilot. In: Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. 2018, p. 37–45.
[11] Zema NR, Trotta A, Natalizio E, Di Felice M, Bononi L. The cuscus simulator for distributed networked control systems: Architecture and use-cases. Ad Hoc Netw 2018;68:33–47.
[12] Sanchez-Aguero V, Valera F, Nogales B, Gonzalez LF, Vidal I. Venue: Virtualized environment for multi-uav network emulation. IEEE Access 2019;7:154659–71.
[13] Acharya S, Bharadwaj A, Simmhan Y, Gopalan A, Parag P, Tyagi H. Cornet: A co-simulation middleware for robot networks. In: 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS). IEEE; 2020, p. 245–51.
[14] Cabreira TM, Brisolara LB, Ferreira Jr PR. Survey on coverage path planning with unmanned aerial vehicles. Drones 2019;3(1):4.
[15] Albani D, Nardi D, Trianni V. Field coverage and weed mapping by uav swarms. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2017, p. 4319–25.
[16] Song Z, Zhang H, Zhang X, Zhang F. Unmanned aerial vehicle coverage path planning algorithm based on cellular automata. In: 2019 15th International Conference on Computational Intelligence and Security (CIS). IEEE; 2019, p. 123–6.
[17] Guo Y, Liu C, Coombes M. Spraying coverage path planning for agriculture unmanned aerial vehicles. In: 2021 26th International Conference on Automation and Computing (ICAC). IEEE; 2021, p. 1–6.
[18] Liang X, Shen M, Du G, Chen G. Real-time moving target tracking algorithm of uav/ugv heterogeneous collaborative system in complex background. Politehn Univ Bucharest Sci Bull Ser C 2019;81(1):119–36.
[19] Patrizi N, Fragkos G, Ortiz K, Oishi M, Tsiropoulou EE. A uav-enabled dynamic multi-target tracking and sensing framework. In: GLOBECOM 2020-2020 IEEE Global Communications Conference. IEEE; 2020, p. 1–6.
[20] Shastry AK, Sinha H, Kothari M. Autonomous detection and tracking of a high-speed ground vehicle using a quadrotor uav. In: AIAA Scitech 2019 Forum. 2019, p. 1188.
[21] Yang B, Cao X, Yuen C, Qian L. Offloading optimization in edge computing for deep-learning-enabled target tracking by internet of uavs. IEEE Internet Things J 2020;8(12):9878–93.
[22] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Vol. 1. CVPR 2001, Ieee; 2001, p. I.
[23] Laaraiedh M. Implementation of kalman filter with python language. 2012, arXiv preprint arXiv:1204.0375.
[24] Fotouhi A, Ding M, Hassan M. Understanding autonomous drone maneuverability for internet of things applications. In: 2017 IEEE 18th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). IEEE; 2017, p. 1–6.
[25] Li F, Fu C, Lin F, Li Y, Lu P. Training-set distillation for real-time uav object tracking. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE; 2020, p. 9715–21.

Aicha Idriss Hentati is a Ph.D. student at National Engineering School of Sfax (ENIS), Tunisia. Her scope of research is communications related to UAV-based systems and collaborative networking with complex operations and high levels of independence. The main objective of her Ph.D. research project is the design and validation of a collaborative inter-UAVs system for autonomous tracking of intruders.

Lamia Chaari Fourati is a professor at Computer Science and Multimedia Higher Institute of Sfax (ISIMS), Tunisia. Her research activities are related to digital telecommunication networks, in particular wireless sensor networks, vehicular networks, software defined networks, information centric networks, wireless body area networks, and UAV networks. She is the laureate for the Kwame Nkrumah Regional Awards for women 2016.