The SDN-Governed Ad Hoc Swarm for Mobile Surveillance of Meteorological Facilities

Xi Chen[1,3], Tao Wu[2(✉)], and Ying Tan[1]

[1] The Key Laboratory for Computer Systems of State Ethnic Affairs Commission, Southwest Minzu University, Chengdu, China

[2] School of Computer Science, Chengdu University of Information Technology, Chengdu, China
wut@cuit.edu.cn

[3] School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China

Abstract. Accurate meteorological observation relies heavily on the proper and precise working of meteorological facilities. Nevertheless, a big portion of meteorological facilities are deployed in outdoor environments hardly within the reach of convenient monitoring and reliable networking infrastructure, hence the surveillance challenge. Given such an infrastructure-less/-poor environment (deserts, oceans, etc.) for meteorological facilities, Ad Hoc nodes are possible candidates for mobile surveillance. However, pure Ad Hoc networking without a logical centric node can barely provide consistent collaboration between mobile nodes during monitoring. This paper proposes a tunnelled overlay structure that bridges the Ad Hoc protocol stack and the SDN (Software-Defined Networking) protocol stack based on network virtualization techniques, so that robust distributed Ad Hoc mobile nodes are grouped in the form of an SDN-governed swarm to conduct the mobile surveillance task with joint efforts under the consistent control of the centric SDN controller. In addition, mobile nodes are equipped with TensorFlow-based image recognition feature, capable of transmitting the recognized results of harmful creatures that might cause facility damages, to suggest proper protection measures, control/avoid facility loss, etc. Experiments on the prototype SDN-governed Ad Hoc swarm are carried out in a real-world university campus meteorological station to demonstrate its feasibility and functionalities.

Keywords: SDN (Software-Defined Networking) · Ad Hoc · Image recognition · Mobile surveillance

Supported by National Key Research and Development Program of China (2018YFC1507005), China Postdoctoral Science Foundation (2018M643448), Sichuan Science and Technology Program (2019YFG0110, 2020YFG0189), Fundamental Research Funds for the Central Universities, Southwest Minzu University (2020NQN18), and SEAC Southeast Asia Research Center at Southwest Minzu University (SE2019Y07).

X. Wu et al. (Eds.): QShine 2020, LNICST 381, pp. 57–75, 2021.
https://doi.org/10.1007/978-3-030-77569-8_5

1 Introduction

Meteorological facilities play a key role in various economic aspects, such as weather forecast, agricultural production, tourism, etc. The accuracy of meteorological observation relies heavily on the proper and precise working of meteorological facilities, thus requiring close surveillance of these devices and equipments. However, a big portion of these facilities are usually deployed in outdoor environments hardly within the reach of convenient surveillance and reliable network infrastructure, some of which even have to face with severe environmental conditions. For instance, for those remotely located meteorological facilities in, e.g., deserts, steep ocean cliffs, etc., they might encounter extreme heat, erosion, wind damage, etc. Given such a complex and infrastructure-less/-poor environment and diverse facilities to monitor, simply introducing fixed sensors and cameras might not fully meet the surveillance requirements of outdoor meteorological facilities. For example in a meteorological station as shown in Fig. 1, meteorological and auxiliary facilities might not be perfectly monitored due to that their positions are out of the range of fixed cameras, or obscured by unexpected objects such as growing trees, falling rocks, (see Q1 in Fig. 1, i.e., an obscured humidity sensor) etc. Facilities might also be damaged by wild animals (Q2), e.g., birds, or severe environmental conditions, e.g., strong wind. Besides, meteorological (e.g., temperature sensors, humidity sensors, solar sensors, wind monitors, rain monitors) and auxiliary facilities (e.g., cameras) also need to be periodically or regularly checked/monitored in order to make sure they work properly. For such scenarios, a group of mobile nodes (such as unmanned aerial vehicles or wheeled vehicles in Fig. 1) can be commanded to take a patrol of meteorological facilities and accordingly adapt on-site formations to avoid obstacles, to track harmful creatures, etc., and timely transfer surveillance or damage report to administrators so that facility loss can be controlled or even avoided asap.

Mobile nodes organized and networked through Ad Hoc networking [7] in a flexible and on-demand fashion are possible candidates for mobile surveillance of meteorological facilities deployed in fields. The mobility, flexibility, and robustness in volatile networked environment are desired benefits. However, several issues must be addressed.

- Consistent control over the whole mobile swarm. Mobile nodes in an Ad Hoc swarm work in a distributed and self-determined fashion. Their moving or sensing behaviors might not comply with the requirements of the global surveillance task if no consistent control is given.
- Collaborative on-site formation adaptation. Obstacles might block the surveillance angle. Mobile nodes under the control of a common "head" are able to adjust positions accordingly and collaboratively to form various formations such as triangles, rectangles, etc., for better monitoring views.
- Image recognition capability. Surveillance by only images would require excessive human intervention to identify harmful creatures or damage status if any. If pictures or videos taken by mobile nodes can be timely recognized on-site

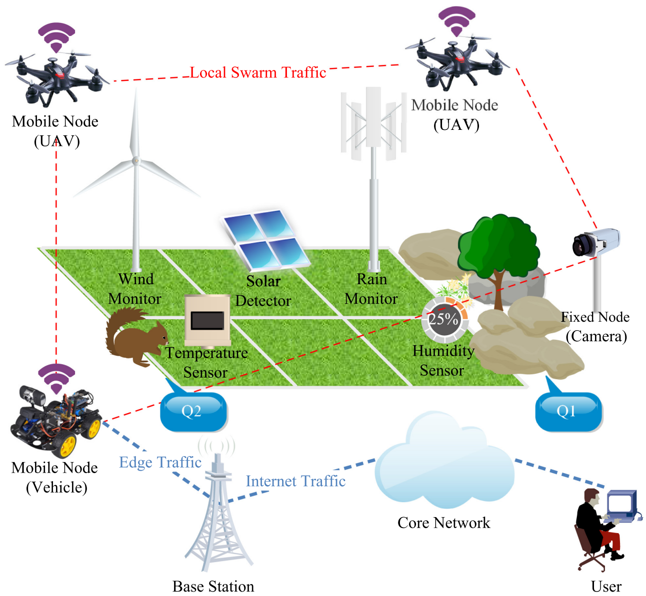


Fig. 1. An outdoor meteorological station under mobile surveillance

and the results of which can be meaningfully transmitted to administrators, appropriate measures are able to be taken as soon as possible.

To summary, pure Ad Hoc networking, which is essentially a distributed structure without central control, can barely afford consistent collaboration between mobile nodes during monitoring, hence the challenge in consistent and streamlined collaboration during mobile surveillance. One possible solution is to apply the centralized SDN (Software-Defined Networking) [15] on top of the distributed Ad Hoc underlay if its mobility, flexibility, and robustness are to be sustained. SDN is widely appliced in DCN (Data center Networks) [21,26,30], WAN (Wide Area Networks) [24,28], network virtualization [37,39], network resources optimization [5,27], QoS provisioning [3,4,6], service function chaining [25,29,32,34], etc. Works are also seen in the literature that integrate SDN into distributed network underlays, to improve security [12,14], to conduct access control and flow scheduling [33], to refine data aggregation [11], to facilitate machine learning integration [22], etc. Nevertheless, how to combine the benefits from the distributed Ad Hoc networking and the centralized SDN, setup the direct control from a uniform controlling interface, and apply the overlaid and hybrid network structure in real-world mobile surveillances of valuable assets are still seldom studied. In this paper, we propose the SDN-governed Ad Hoc swarm, an approach to construct a centralized SDN overlay on top of the distributed Ad Hoc underlay for mobile surveillance of meteorological facilities. The contribution of this paper includes as follows:

– A tunnelled overlay structure based on VxLAN (Virtual eXtensible Local Area Network) [18] that bridges the Ad Hoc protocol stack and SDN protocol stack to implement uniform control over distributed underlay.
– The extended OpenFlow [19] to accommodate self-defined physical actions so that mobility and sensibility of the Ad Hoc swarm can be consistently controlled without involving hardware details.
– Combined with TensorFlow-based image recognition, a prototype of low-cost mobile surveillance Ad Hoc swarm governed by SDN is implemented on Raspberry Pi platform. Extensive experiments of this prototype swarm were conducted in a real-world meteorological station.

The rest of this paper is organized as follows: Sect. 2 introduces the architecture of mobile nodes in the SDN-governed Ad Hoc swarm. Section 3 specifies how the SDN overlay is built on top of the Ad Hoc underlay, and the centralized control by means of the extended OpenFlow. Section 4 specifies the TensorFlow-based image recognition deployed on the swarm. Section 5 conducts various experiments to test the SDN-governed Ad Hoc swarm prototype in a real meteorological station. Related works are summarized in Sect. 6. Finally, this paper is concluded in Sect. 7.

2 Mobile Node Architecture

The architecture of the mobile node is shown in Fig. 2, which is roughly divided into 3 layers. The HW (hardware) layer is based on the Raspberry Pi platform, mainly providing wheeled mobility, camera monitoring and various sensors such as temperature sensors, infrared detectors, etc. Note that other mobility platform such as unmanned aerial vehicle (UAV) can also be considered if affordable. OS (Operating System) layer provides software functionalities like networking, computing, storage, etc. It is able to provide on the operating system several wireless communication protocols, such as WiFi, ZigBee, LoRa, NB-IoT, etc., as well as Ad Hoc routing protocols, such as AODV, DSDV, DSR, OLSR, etc., to enable peer-to-peer distributed control, packed with operating systems or manually installed as needed. Since high data rate transmission is needed in our application scenario to support camera surveillance and photo sharing for image recognition, we adopt the higher-rate WiFi as the fundamental wireless communication mechanism between mobile nodes. Low-power and lower data-rate communications, such as ZigBee for shorter range and LoRa/NB-IoT for longer range can be considered if corresponding chips are available.

The key extension for the centralized control over the distributed Ad Hoc underlay is the SDN layer. Every node is equipped with the SDN data plane, e.g., Open vSwitch (OVS) [23]. Meanwhile, swarm head might be re-assigned to other mobile nodes due to damage, high workload, low battery, etc., hence every node must be eligible to be elected as the swarm head to control and manage other mobile nodes so that the centralized SDN overlay can be continuously maintained. Thus, the SDN control plane must be deployed on every node as well, such as Floodlight, ONOS [1], OpenDaylight [20], etc. In this paper, we adopt OVS as the SDN switch and lightweight Floodlight as the SDN controller. SDN controller (i.e., Floodlight) is activated once a mobile node is elected/designated as swarm head while SDN switch (i.e., OVS) is always standing by on every node to accept control by swarm head through OpenFlow.

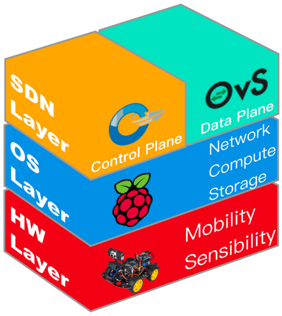


Fig. 2. Mobile node architecture

3 Ad Hoc Swarm Networking

3.1 Ad Hoc Underlay Networking

Every mobile node needs to participate in a two-stage networking to enable the centralized SDN control over the Ad Hoc swarm, i.e., the Ad Hoc underlay networking and SDN overlay networking. And, the Ad Hoc underlay does not have fixed networking infrastructure. Therefore, the wireless NIC (network interface card) of the mobile node should be configured to work in the ad hoc mode. This can be done by editing /etc./network/interfaces. For example, the following configuration (Listing 1.1) sets the IP address of the physical wireless NIC wlan0 as 10.0.0.1. It works in the ad hoc mode as specified in the configuration, so that infrastructure such as access points (AP) are not needed for mutual communication between mobile nodes. Meanwhile, all participating mobile nodes in the same swarm must be configured with exactly the same ESSID (Extended Service Set ID), for example "my-swarm" in Listing 1.1. Network mask "netmask" can be adapted according to the scale of the swarm to be constructed. Take the 10.0.0.0 IP subnet as an example. The IP addresses in such a subnet are by default class A addresses, whose network masks default to 255.0.0.0 (with 8 bits for network IDs and 24 bits for host IDs). It provides an address space capable of hosting almost 2^{24} nodes, which is more than needed in our scenario where an Ad Hoc swarm usually consists of several or tens of collaborative mobile nodes. Therefore, CIDR (classless inter-domain routing) can be used to partition such big address spaces by providing different network masks. For example, 10.0.0.0

subnet with an altered network mask 255.255.255.0 (also written as 10.0.0.0/24) gives a 2^8 space whereas 10.0.0.0/28 gives an even smaller space with 2^4 mobile nodes. Other mobile nodes have similar settings except for different IP addresses. Mobile nodes can ping each other using physical IP addresses (i.e., 10.0.0.x) and receive ICMP replies normally at this point of time, even though no APs are present. The following configuration demonstrates how wireless NICs should be configured when WiFi is adopted as the wireless communication for a distributed Ad Hoc swarm. The above configuration can be scripted and executed to enable automatic Ad Hoc underlay networking.

```
auto wlan0
allow-hotplug wlan0
iface wlan0 inet static
address 10.0.0.1
netmask 255.255.255.0
network 10.0.0.0
broadcast 10.0.0.255
wireless-essid my-swarm
wireless-mode ad-hoc
wireless-channel 3
```

Listing 1.1. The Ad Hoc Configuration

3.2 SDN Overlay Networking

Tunnelled Overlay Structure. Ad Hoc protocol stack and SDN protocol stack are two independent stacks. To combine flexibility from Ad Hoc and manageability from SDN, the two stacks must be somehow bridged. The key to deploying the SDN overlay above the Ad Hoc underlay to offer consistent and streamlined control is the deployment of OVS, the software-ized SDN switch. OVS consists of several modules working in both user space and kernel space of a Linux system. Core modules include ovsdb-server, ovs-vswitchd, ovs kernel module in kernel space, etc. ovsdb-server keep records of configurations. ovs-vswitchd communicates with SDN controllers through OpenFlow. ovs-vswitchd also connects with ovs kernel module through netlink and supports multiple datapaths, i.e., the virtual bridges that forward data. When ovs kernel module receives traffic, the actual forwarding is done by datapaths. Datapaths can be created on demand by OVS, and equipped with multiple vports (virtual ports on a datapath) for traffic ingress and egress. When directing packets to another vport upon traffic arrival, a vport matches packet fields with OpenFlow-installed flow table entries, and forwards matched packets or inquire the SDN controller for miss-matched ones. In other words, traffic forwarding by OVS is essentially instructed by (the flow table entries installed by) OpenFlow. The primary workflow of OVS can be found in Fig. 3.

According to the workflow of OVS mentioned above, to control the traffic flows running through the Ad Hoc underlay, wireless NICs of mobile nodes

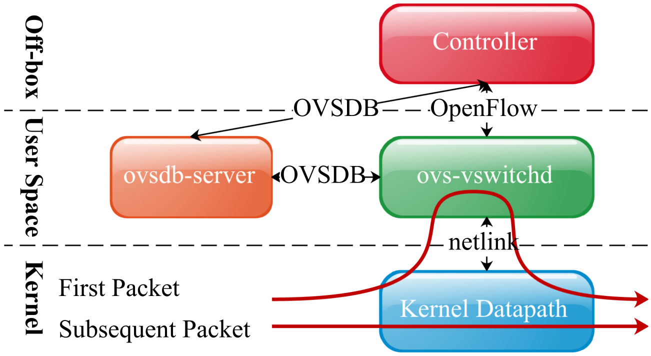


Fig. 3. OVS workflow

must be bound to datapaths created by OVS, so that traffic flows run into OVS modules, leading to an direct control by OpenFlow. Nevertheless the physical wireless NICs have already participated in the Ad Hoc underlay networking in the previous phase, thus simply bounding wireless NICs to datapaths fails the Ad Hoc underlay communication. Alternatively, we can create extra datapaths to encapsulate physical wireless NICs, so that the inner physical wireless NICs inside datapaths are controlled indirectly by OpenFlow. In this way, the SDN overlay becomes possible. In every mobile node, we create an OVS datapath named br0. At this point of time, every node has two forwarding devices, the physical wireless NIC wlan0, participating in the Ad Hoc underlay networking, and the virtual datapath br0, to participate in the SDN overlay controlling. To encapsulate wlan0 without compromising the Ad Hoc underlay communication, it requires end-to-end tunneling over the physical communication channels by means of datapath and vport. To achieve this, VxLAN is adopted for pairwise tunneling. The ovs-vsctl utitity provided by OVS offers VxLAN support. For every other mobile node, br0 adds a vport to tunnel all the way to the wlan0 of that node. Besides, br0 is assigned an independent IP address other than that of its own wlan0. Pseudo code for mobile node with wlan0 assigned 10.0.0.1 (n1 for short) tunnelling another node with wlan0 assigned 10.0.0.2 (n2) looks like Listing 1.2. In the above settings, it not only tunnels n1 and n2 using VxLAN, it also instructs the OVS datapath br0 to be controlled by the SDN controller process residing on 10.0.0.1:6653 through OpenFlow. Note 6653 is the official transport number for OpenFlow. The SDN controller IP address 10.0.0.1 indicates both SDN data plane and control plane are activated in n1 and n1 controls itself. Similar settings must be configured at the other side of the tunnel, i.e., n2, with symmetric changes on remote IP address as 10.0.0.1 and 20.0.0.2/24 for n2 br0. In this way, n1 controls both n1 and n2, forming the SDN overlay.

```
ovs-vsctl del-br br0
ovs-vsctl add-br br0
ovs-vsctl add-port br0 vport1
ovs-vsctl set interface vport1
          type=vxlan
          options:remote_ip=10.0.0.2
ifconfig br0 20.0.0.1/24 up
ovs-vsctl set-controller br0
          tcp:10.0.0.1:6653
```

Listing 1.2. VxLAN Tunneling Configuration

Imagine there are m nodes in an Ad Hoc swarm. It requires $m(m-1) \sim O(m^2)$ tunnels for full collaboration in the Ad Hoc underlay. Therefore, the control and collaboration at runtime is quite complex given larger m, demanding sophisticated algorithms. Nevertheless, if an SDN overlay is constructed, it only imposes $m-1$ OpenFlow channels for full control and centralized collaboration. The administrator controls only the SDN controller through northbound interface, and it gives instructions on behalf of the administrator through OpenFlow, leading to much simpler runtime management. Figure 4 demonstrates the architecture of the SDN-controlled Ad Hoc swarm ($m = 4$) where the upmost node with purple shadow is the SDN controller. The above configuration can be scripted and executed to enable automatic SDN overlay networking. Meanwhile, the selection of the SDN controller (i.e., the swarm head) can be done manually for a small scale swarm. Alternatively, derivatives of cluster head election algorithm such as LEACH [31] widely used in ZigBee can be tuned and used for such a purpose.

Centralized Controlling. We can see from the mobile node architecture that the control and management of the Ad Hoc swarm can be simply achieved by invoking the SDN controller northbound interface residing on the swarm head. In this way, the control and management of the swarm is simplified as the interaction with the swarm head that sends unified OpenFlow directives to other mobile nodes. Therefore, OpenFlow can now be regarded as the middleware to unify heterogenous hardware platforms, offering centralized control and management over peer-to-peer distributed Ad Hoc underlay. This "divide and conquer" approach scales when there are a large amount of mobile nodes under control. In this paper, we go one step further to extend OpenFlow from pure data forwarding actions to physical actions including moving, monitoring, and sensing.

OpenFlow is originally designed to instruct data forwarding on switches. Switches match the incoming packets against flow table, and execute corresponding OpenFlow actions for matched packets. For those miss-matched packets, switches send OpenFlow packet-in PDUs to the SDN controller to inquire how these packets should be processed. The SDN controller replies with packet-out to instruct forwarding actions, and switches also install corresponding flow table entries so that similar packets are to be locally processed the same way without further inquiries. OpenFlow supports several forwarding-related actions (see Listing 1.3, line 3–9) in version 1.3 which is a widely deployed version.

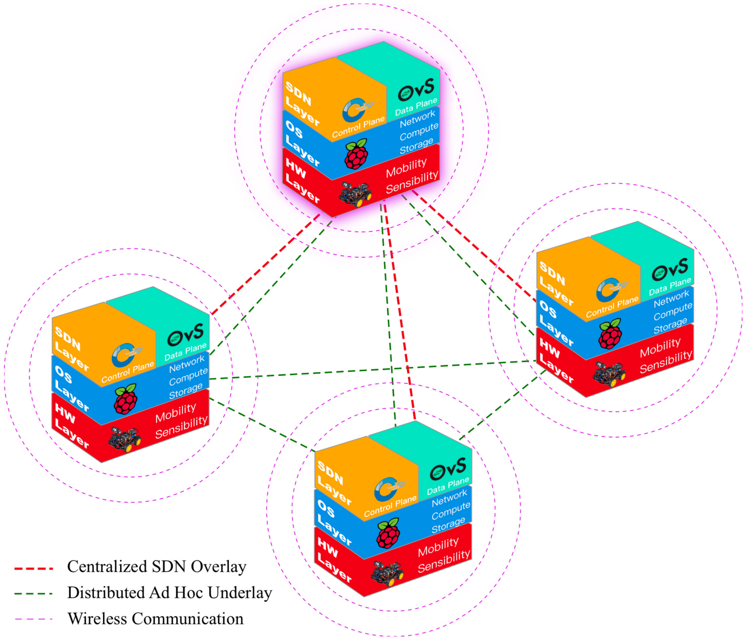


Fig. 4. SDN-governed Ad Hoc Swarm

Since mobile nodes involves moving, sensing, monitoring, etc., to fully implement the control and management of mobile node using OpenFlow, OpenFlow must be extended with corresponding actions. For example in Listing 1.3, action {OFPACT_MOVE_FORWARD, 101} has been added to instruct the movement of the mobile node to move forward; action {OFACT_CAMERA_ON, 201} has been added to instruct the mobile node to turn on the camera and conduct live surveillance; action {OFACT_IMAGE_RECOGNIZE, 201} for image recognition (see Sect. 4). These actions are to be parsed locally on mobile nodes upon arrival. And subsequently, hardware driver APIs are to be invoked internally, transparent to upper layer applications or administrators, to drive corresponding equipments on board of these mobile nodes.

```
static const struct ofpact_map of12[]=
{
  {OFPACT_OUTPUT, 0},
  {OFPACT_SET_MPLS_TTL, 15},
  {OFPACT_DEC_MPLS_TTL, 16},
  {OFPACT_PUSH_VLAN, 17},
  {OFPACT_STRIP_VLAN, 18},
  {OFPACT_PUSH_MPLS, 19},
  {OFPACT_POP_MPLS, 20},
  ...
```

```
    {OFPACT_MOVE_FORWARD, 101},
    {OFPACT_MOVE_BACKWARD, 102},
    {OFPACT_MOVE_LEFT, 103},
    {OFPACT_MOVE_RIGHT, 104},
    ...
    {OFPACT_CAMERA_ON, 201},
    {OFPACT_CAMERA_OFF, 202},
    ...
    {OFPACT_IMAGE_RECOGNIZE, 301},
}
```

Listing 1.3. OpenFlow 1.3 Actions

4 Image Recognition

Image recognition plays a key role in mobile surveillance of meteorological facilities. We developed the image recognition feature deployed on the Raspberry Pi platform based on TensorFlow and OpenCV framework. The purpose of the TensorFlow-based image recognition is to identify several harmful creatures seen around the target meteorological station, i.e., rats, sparrows, dogs, eagles, etc., and send not only the taken images but also the recognized results once damage occurs to the administrator in a timely fashion, to alarm possible creature-caused damages to meteorological facilities (e.g., rat-bites) at early stage and/or to suggest protection measures.

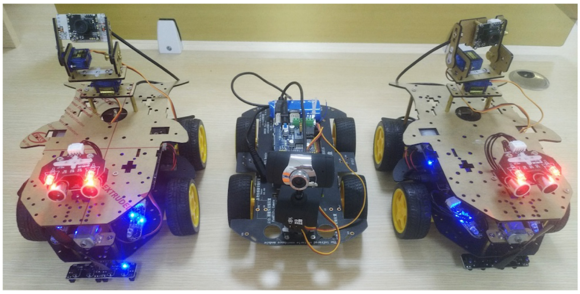
We selected the ssd_mobilenet_v1 network and the pre-trained ssd_mobilenet_v1_coco model to conduct the transfer learning, to accelerate the training. Mobilenet is a lightweight neural network model proposed by Google in 2017, which is suitable for embedded mobile nodes. There are 28 layers in total, each of which has a BatchNorm layer and a ReLU layer. Depth separable convolution is used in mobilenet to divide the standard convolution kernels into depth convolution kernels and 1×1 point-wise convolution kernels, so that the computation load can be reduced for mobile nodes. SSD (single shot multibox detector) is one of the most popular classification frameworks. Its main task is to conduct classification after features extraction from neural networks. The main idea of SSD is to carry out intensive sampling uniformly at different positions in the figure, and adopt different scales and aspect-ratios during sampling to carry out classification and regression. SSD uses convolution to extract the detection from the multi-scale feature map, so that objects of different scales can be simultaneously detected, that is, bigger scales for smaller objects detection and vise versa. ssd_mobilenet adds 8 convolution layers after the last layer of mobilenet, and extracts 6 of them for detection and the rest for convolution kernels for coordinate regression. As of model training, the original ssd_mobilenet_v1_coco model has a 90-dimensional output layer that we substituted with a 4-dimensional (representing rats, sparrows, dogs, and eagles) softmax layer to generate the probabilities indicating which creature the identified object belongs to. The above process applies in the static image recognition. If image recognition must be applied in video streaming such as camera surveillance, OpenCV framework needs to be deployed along with TensorFlow to extract frames from video streaming and conduct the serial image recognition for each frame.

At this point of time, we have constructed an Ad Hoc swarm under the centralized control of the SDN controller activated on the swarm head. Since every mobile node is equipped with OVS, it can be uniformly and compliantly instructed to move, adapt on-site formation, monitor, recognize objects on the taken photos, etc., by means of our extended OpenFlow messages that contain self-defined physical actions.

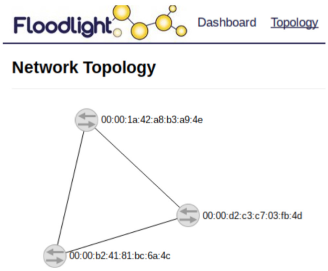
5 Evaluation

5.1 Swarm Networking and Centralized Controlling

The prototype of the SDN-controlled Ad Hoc swarm is implemented based on the Raspberry Pi platform. Raspbian operating system was installed on very Raspberry Pi. Floodlight was deployed as the SDN controller, and OVS is also installed as the SDN data plane, to comprehensively build the SDN layer over every mobile node. For the administrator, northbound REST API provided by Floodlight can be invoked to remotely control the swarm head (i.e., the SDN controller), and the head delivers OpenFlow messages that contain actions to instruct how the Ad Hoc swarm should function and collaborate. For indoor experiments in this section, three such mobile nodes (P1–P3, as shown in Fig. 5a) were participating in the networking, among which, one node was working as the swarm head, which was also an ordinary Ad Hoc swarm node under control itself. The logical SDN overlay topology was discovered by the Floodlight controller and shown in Fig. 5b.



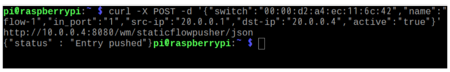
(a) Underlay Ad Hoc Swarm



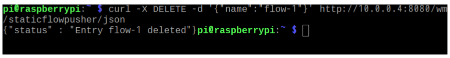
(b) Overlay SDN Topology

Fig. 5. SDN-controlled Ad Hoc swarm

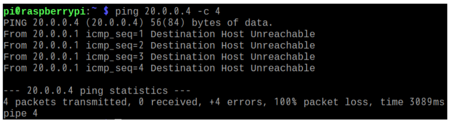
The SDN overlay worked in the 20.0.0.0 IP segment while the Ad Hoc underlay worked in the 10.0.0.0 IP segment. Administrators can push flow entries through Floodlight's northbound API (relative URL is /wm/staticflowpusher/json) to instruct data forwarding, demonstrated by Fig. 6. A flow entry that drops all data from source IP 20.0.0.1 (i.e., P1) to destination IP 20.0.0.4 (i.e., P3) in this experiment was pushed by the administrator, as shown in Fig. 6a. We can see from Fig. 6c that subsequent pingings complained about the destination host unreachable. Dynamic controlling over data forwarding of the Ad Hoc underlay was also implemented, as shown in Fig. 6b where a flow entry that deletes the previous traffic-dropping instruction was pushed by the administrator. We can see from Fig. 6d, the pinging was re-activated. Notice that the pinging operation directly uses the SDN overlay IP addresses (20.0.0.x), instead of those of the Ad Hoc underlay (10.0.0.x). This indicates that the forwarding of the Ad Hoc underlay is fully controlled by the SDN overlay, without any underlying hardware/configuration/communication details involved. Therefore, the administrator interacts with only the Ad Hoc swarm head to implement full control over the whole Ad Hoc swarm through the SDN layer deployed on every node.



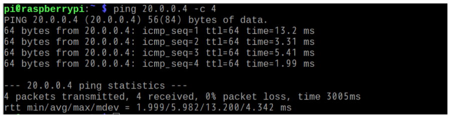
(a) The flow entry that drops traffic between P1 and P3



(b) Deleting the traffic-dropping flow entry



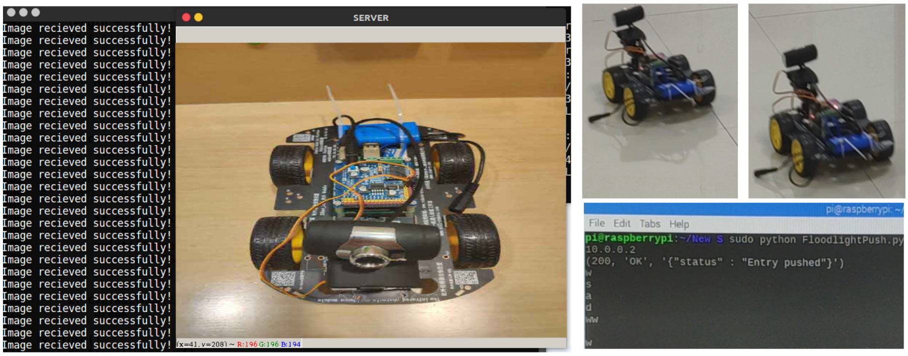
(c) Traffic disabled



(d) Traffic enabled

Fig. 6. Flow entries pushing

Self-defined physical actions of the extended OpenFlow, as we have introduced in Sect. 3.2, can also be pushed from the SDN controller located on the swarm head using Packet-out PDUs, like ordinary OpenFlow actions for data forwarding. A flow entry containing action {OFACT_CAMERA_ON, 201} was pushed, and the Ad Hoc swarm node that received it turned on its camera to conduct live surveillance, as shown in Fig. 7a. Another flow entry containing {OFPACT_MOVE_FORWARD, 101} action was pushed, and the Ad Hoc swarm node that received it moved in different directions for keyboard strokes w (forward), s (backward), a (left) and d (right), as shown in Fig. 7b. These physical actions are particularly useful for collaborative mobile surveillance over valuable assets or facilities, including meteorological facilities remotely deployed, in case of being unexpectedly obscured by obstacles such growing tree branches, damaged by wild animals, etc., as will be shown in outdoor experiments in Sect. 5.2.



(a) SDN-controlled Camera Surveillance (b) SDN-controlled Moving

Fig. 7. Self-defined physical actions of the extended OpenFlow

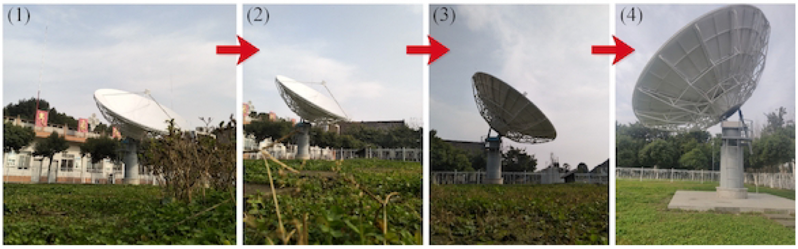
5.2 Mobile Surveillance and Image Recognition

The prototype of the SDN-governed Ad Hoc swarm, augmented with wheels, cameras, sensors, etc., was also deployed in the meteorological station of Chengdu University of Information Technology (CUIT) for outdoor mobile surveillance of meteorological facilities. CUIT features its meteorology, and has a professional meteorological station located in its campus, as shown in Fig. 8a. The size of the main station is about 88 m × 41 m and that of the secondary radar station is about 20 m × 21 m. Some facilities, especially the radar, are obscured by nearby trees, hard to be effectively monitored by the fixed camera, as we can see from Fig. 8b. An SDN-governed Ad Hoc swarm consisting of three wheeled mobile nodes was deployed for evaluation. A flow entry containing action Camera_On was pushed by the SDN controller (i.e., the swarm head), and the node that received it turned on its camera to conduct live surveillance and image recognition as needed. To bypass obstacles and get better monitoring angles, the swarm head also instructed mobile nodes to gradually move around the radar and other meteorological facilities while conducting live surveillance, by sending OpenFlow messages that contained movement actions in different directions. Figure 8c (1)–(3) shows this mobile monitoring process that gradually avoided bushes and grass, and revealed a clear view of the meteorological radar. Besides, mobile surveillance makes it more feasible to conduct multi-angle monitoring compared with fixed camera shooting. For example, Fig. 8c (4) shows a reverse view of the monitored radar, taken by anther mobile node in the triangular on-site formation commanded by the swarm head. In addition, the node-to-node wireless coverage in the field is about 100 m, given the current mobile node design, which is sufficient for the mutual communication between nodes, and the surveillance of the meteorological station.

Image recognition based on TensorFlow was also tested in this experiment due to that small-sized harmful animals are seen around the meteorological facilities, which might cause damage. Figure 8d–8e demonstrate the recognition and marking of birds in a picture. The marking function, which draws a highlighted frame around the recognized object and indicates the recognition confidence, requires TensorFlow's object detection API as well as a more complex offline training process. It says in these pictures the recognition confidence was about 89% in Fig. 8d and 99% in Fig. 8e, respectively. The average recognition precision was above 90% for larger objects in a picture. The delay for image recognition and marking was 0.7 s on average even though the computing platform is the comparatively low-performance mobile Raspberry Pi.

(a) The Meteoreology Station

(b) The Swarm and the Blocked Radar



(c) Mobile Surveillance around the Radar

(d) Bird Recognition

(e) Bird Recognition

Fig. 8. Mobile surveillance of a meteorological radar

6 Related Works

Efforts have been made to extend SDN from wired networking towards the wireless networking. OpenRoads [35] merged SDN with WiFi/WiMAX technologies to deploy in university campus networks to facilitate wireless network innovation. It provides wireless extensions to the original OpenFlow. OpenRoads consists of three layers, namely the flow table layer, which coordinates OpenFlow and WiFi/WiMAX; the slicing layer, for FlowVisor-based network slicing; and the controller layer implemented by NOX [10]. In particular, OpenRoads demonstrated how wireless handover can be effectively conducted by means of SDN in wireless environments. In reference [8], OpenFlow is applied to WMN (Wireless Mesh Network), and an SDN solution based on virtualization technology and cross-layer flow table rules for WMN is proposed to enable unified control. Similarly, applications of SDN in wireless networks are also reported in references [9,36]. These works focus on SDN-equipped wireless networks, lacking the support for ad hoc networking and uniform control, hence difficulty in coordinating Ad Hoc nodes to conduct the consistent and coordinated mobile surveillance.

Recent years have also seen the efforts in integrating SDN into Ad Hoc networking. Reference [16] proposed and envisioned the software-defined VANET architecture and various services. Architecturally, it includes a remote centralized SDN controller located in a telecommunications datacenter, base stations, fixed RSUs (Road Side Unit), and mobile vehicles. Vehicles and RSUs constitute the data plane, whose data transmission is controlled by the SDN controller. Base stations and the remote controller conduct remote communication via wireless communications, such as LTE or WiMAX. This work gave some preliminary envisions on the software-defined VANET. Reference [38] proposed to control the OVS entities deployed on Ad Hoc nodes by the SDN controller, as we do in this paper. It maintains the interoperability with other SDN by applying best-practice SDN methodologies in implementation. However, the main weakness of this work is its lack of consideration of mobility management commonly seen in Ad Hoc networking. Reference [17] proposed to bridge SDN domains with west-east interfaces in a peer-to-peer manner, which offered new possibilities for interconnectivity and collaboration between software-defined MANETs.

SDN has also been introduced into the monitoring of valuable assets in recent years. Reference [2] extended the SDN southbound interface that is dedicated to devices discovery in a way that messages for adaptive device status monitoring are piggybacked in the LLDP (Link Layer Discovery Protocol), and proposed to apply this adaptive monitoring mechanism in energy Internet. Reference [14] proposed to monitor and improve the use of water resources by means of secure SDN techniques. Reference [13] adopted a similar approach to our work in that hardware-specifics are encapsulated using software and exposed as unified interfaces for easy invocation and composition, i.e., the concept of Software-Defined Device. Our work defers in that OpenFlow is used as the middleware to unify data forwarding and physical actions, thus the enhancement of interoperability and compatibility with wired SDN networks.

7 Conclusions

Ad Hoc networking provides good survivability and flexibility in non-static network topologies, thus a good choice for on-demand mobile surveillance for meteorological facilities deployed in fields where communication and monitoring infrastructures might be in shortage. However, the distributed structure prevents it from being consistently controlled hence the difficulty in coordination in mobile surveillance. This paper proposes to ingrate SDN methodologies into Ad Hoc networking to construct a collaborative and controlled swarm, so that moving, sensing, image recognition, etc., are uniformly controlled by the centralized SDN controller by means of extended OpenFlow. This overlaid and hybrid structure simplifies the global control over an Ad Hoc swarm in that only the Ad Hoc swarm head that activates the SDN controller is involved for the manipulation of the whole swarm. Extensive experiments in a real-world meteorological station demonstrate the feasibility. In our future work, topology dynamics such as swarm fusion and fission due to the distance variability will be studied to offer finer-grained control over the distributed underlay.

References

1. Berde, P., et al.: ONOS: towards an open, distributed SDN OS. In: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, HotSDN 2014, pp. 1–6. ACM, New York (2014). https://doi.org/10.1145/2620728.2620744
2. Chen, X., Chen, Y., Sangaiah, A.K., Luo, S., Yu, H.: MonLink: piggybackStatus monitoring over LLDP in software-defined energy internet. Energies **12**(6), 1147 (2019). https://doi.org/10.3390/en12061147, https://www.mdpi.com/1996-1073/12/6/1147
3. Chen, X., et al.: Reinforcement learning based QoS/QoE-aware service function chaining in software-driven 5g slices. Trans. Emerg. Telecommun. Technol **29**(11), e3477 (2018).https://doi.org/10.1002/ett.3477, https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3477
4. Chen, X., Wu, J., Wu, T.: The Top-K QoS-aware paths discovery for source routing in SDN. KSII Trans. Internet Inf. Syst. **12**(6), 2534–2553 (2018). https://doi.org/10.3837/tiis.2018.06.006
5. Chen, X., Wu, T.: Towards the semantic web based northbound interface for SDN resource management. In: IEEE 11th International Conference on Semantic Computing (ICSC), San Diego, CA, United states, pp. 40–47 (2017). https://doi.org/10.1109/ICSC.2017.27
6. Chen, X., Wu, T., Xie, L.: The declarative and reusable path composition for semantic web-driven SDN. IEICE Trans. Commun. **E101.B**(3), 816–824 (2018). https://doi.org/10.1587/transcom.2017EBP3233
7. Corson, D.S.M., Macker, J.P.: Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. No. 2501 in Request for Comments, RFC Editor (1999). https://doi.org/10.17487/RFC2501, https://rfc-editor.org/rfc/rfc2501.txt, published: RFC 2501
8. Dely, P., Kassler, A., Bayer, N.: OpenFlow for wireless mesh networks. In: 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN), pp. 1–6, July 2011. https://doi.org/10.1109/ICCCN.2011.6006100

9. Detti, A., Pisa, C., Salsano, S., Blefari-Melazzi, N.: Wireless mesh software defined networks (wmSDN). In: 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 89–95, October 2013. https://doi.org/10.1109/WiMOB.2013.6673345
10. Gude, N., et al.: NOX: towards an operating system for networks. SIGCOMM Comput. Commun. Rev. **38**(3), 105–110 (2008). https://doi.org/10.1145/1384609.1384625
11. Hakiri, A., Berthou, P., Gokhale, A., Abdellatif, S.: Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications. IEEE Commun. Mag. **53**(9), 48–54 (2015). https://doi.org/10.1109/MCOM.2015.7263372
12. Han, Z., Li, X., Huang, K., Feng, Z.: A software defined network-based security assessment framework for CloudIoT. IEEE Internet Things J. **5**(3), 1424–1434 (2018). https://doi.org/10.1109/JIOT.2018.2801944
13. Hu, P., Ning, H., Chen, L., Daneshmand, M.: An open Internet of Things system architecture based on software-defined device. IEEE Internet Things J. 2583–2592 (2018). https://doi.org/10.1109/JIOT.2018.2872028
14. Kamienski, C., et al.: SWAMP: smart water management platform overview and security challenges. In: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), pp. 49–50, June 2018. https://doi.org/10.1109/DSN-W.2018.00024
15. Kreutz, D., Ramos, F.M.V., Esteves Verissimo, P., Esteve Rothenberg, C., Azodolmolky, S., Uhlig, S.: Software-defined networking: a comprehensive survey. In: Proceedings of the IEEE, vol. 103, pp. 14–76 (2015). https://doi.org/10.1109/JPROC.2014.2371999
16. Ku, I., Lu, Y., Gerla, M., Gomes, R.L., Ongaro, F., Cerqueira, E.: Towards software-defined VANET: architecture and services. In: 2014 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET), pp. 103–110, June 2014. https://doi.org/10.1109/MedHocNet.2014.6849111
17. Lin, P., Bi, J., Chen, Z., Wang, Y., Hu, H., Xu, A.: WE-bridge: west-east bridge for SDN inter-domain network peering. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 111–112, April 2014. https://doi.org/10.1109/INFCOMW.2014.6849180
18. Mahalingam, M., et al.: Virtual eXtensible Local Area Network (VXLAN): a Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks (2014). https://doi.org/10.17487/RFC7348, https://www.rfc-editor.org/info/rfc7348
19. Mckeown, N., et al.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008). https://doi.org/10.1145/1355734.1355746
20. Medved, J., Varga, R., Tkacik, A., Gray, K.: OpenDaylight: towards a model-driven SDN controller architecture. In: Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, pp. 1–6 (2014). https://doi.org/10.1109/WoWMoM.2014.6918985
21. Miao, W., et al.: SDN-enabled OPS with QoS guarantee for reconfigurable virtual data center networks. IEEE/OSA J. Opt. Commun. Netw. **7**(7), 634–643 (2015). https://doi.org/10.1364/JOCN.7.000634
22. Mohammadi, M., Al-Fuqaha, A., Guizani, M., Oh, J.S.: Semisupervised deep reinforcement learning in support of IoT and smart city services. IEEE Internet Things J. **5**(2), 624–635 (2018). https://doi.org/10.1109/JIOT.2017.2712560

23. Pfaff, B., et al.: The design and implementation of open vSwitch. In: 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI), vol. 40, pp. 117–130. USENIX Association, Oakland, CA (2015)
24. Soliman, M., Nandy, B., Lambadaris, I., Ashwood-Smith, P.: Exploring source routed forwarding in SDN-based WANs. In: IEEE International Conference on Communications (ICC), pp. 3070–3075, June 2014. https://doi.org/10.1109/ICC.2014.6883792
25. Sun, G., Li, Y., Liao, D., Chang, V.: Service function chain orchestration across multiple domains: a full mesh aggregation approach. IEEE Trans. Netw. Serv. Manage. **15**(3), 1175–1191 (2018). https://doi.org/10.1109/TNSM.2018.2861717
26. Sun, G., Liao, D., Zhao, D., Xu, Z., Yu, H.: Live migration for multiple correlated virtual machines in cloud-based data centers. IEEE Trans. Serv. Comput. **11**(2), 279–291 (2018). https://doi.org/10.1109/TSC.2015.2477825
27. Sun, G., Zhang, F., Liao, D., Yu, H., Du, X., Guizani, M.: Optimal energy trading for plug-in hybrid electric vehicles based on fog computing. IEEE Internet Things J. **6**(2), 2309–2324 (2019). https://doi.org/10.1109/JIOT.2019.2906186
28. Sun, G., Chang, V., Yang, G., Liao, D.: The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion. Inf. Sci. **432**, 495–515 (2018). https://doi.org/10.1016/j.ins.2017.08.021
29. Sun, G., Li, Y., Yu, H., Vasilakos, A.V., Du, X., Guizani, M.: Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. Fut. Gener. Comput. Syst. **91**, 347–360 (2019).https://doi.org/10.1016/j.future.2018.09.037,http://www.sciencedirect.com/science/article/pii/S0167739X1831848X
30. Sun, G., Liao, D., Zhao, D., Sun, Z., Chang, V.: Towards provisioning hybrid virtual networks in federated cloud data centers. Fut. Gener. Comput. Syst. **87**, 457–469 (2018). https://doi.org/10.1016/j.future.2017.09.065
31. Tyagi, S., Kumar, N.: A systematic review on clustering and routing techniques based upon LEACH protocol for wireless sensor networks. J. Netw. Comput. Appl. **36**(2), 623–645 (2013).https://doi.org/10.1016/j.jnca.2012.12.001,http://www.sciencedirect.com/science/article/pii/S1084804512002482
32. Wen, T., Yu, H., Du, X.: Performance guarantee aware orchestration for service function chains with elastic demands. In: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 1–4, November 2017. https://doi.org/10.1109/NFV-SDN.2017.8169854
33. Wu, D., Arkhipov, D.I., Asmare, E., Qin, Z., McCann, J.A.: UbiFlow: mobility management in urban-scale software defined IoT. In: 2015 IEEE Conference on Computer Communications (INFOCOM). pp. 208–216, April 2015. https://doi.org/10.1109/INFOCOM.2015.7218384
34. Yang, K., Zhang, H., Hong, P.: Energy-aware service function placement for service function chaining in data centers. In: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (Dec 2016). https://doi.org/10.1109/GLOCOM.2016.7841805
35. Yap, K.K., et al.: OpenRoads: empowering research in mobile networks. SIGCOMM Comput. Commun. Rev. **40**(1), 125–126 (2010).https://doi.org/10.1145/1672308.1672331
36. Yap, K.K., et al.: Blueprint for introducing innovation into wireless mobile networks. In: Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures, VISA 2010. pp. 25–32. ACM, New York (2010). https://doi.org/10.1145/1851399.1851404

37. Yu, H., Qiao, C., Wang, J., Li, L., Anand, V., Wu, B.: Regional failure-resilient virtual infrastructure mapping in a federated computing and networking system. IEEE/OSA J. Opt. Commun. Netw. **6**(11), 997–1007 (2014). https://doi.org/10.1364/JOCN.6.000997
38. Yu, H.C., Quer, G., Rao, R.R.: Wireless SDN mobile ad hoc network: from theory to practice. In: 2017 IEEE International Conference on Communications (ICC), pp. 1–7, May 2017. https://doi.org/10.1109/ICC.2017.7996340
39. Yu, H., Wen, T., Di, H., Anand, V., Li, L.: Cost efficient virtual network mapping across multiple domains with joint intra-domain and inter-domain mapping. Opt. Switch. Netw. **14**(PART 3), 233–240 (2014). https://doi.org/10.1016/j.osn.2014.05.020