algorithms

Article

An IoT System for Social Distancing and Emergency Management in Smart Cities Using Multi-Sensor Data

Rosario Fedele [1] and Massimo Merenda [1,2,*]

1 Department of Information Engineering, Infrastructure and Sustainable Energy (DIIES), University Mediterranea of Reggio Calabria, 89124 Reggio Calabria, Italy; rosario.fedele@unirc.it
2 HWA srl-Spin Off dell'Università Mediterranea di Reggio Calabria, Via Reggio Campi II tr. 135, 89126 Reggio Calabria, Italy
* Correspondence: massimo.merenda@unirc.it; Tel.: +39-0965-1693-441

Received: 11 August 2020; Accepted: 3 October 2020; Published: 7 October 2020

Abstract: Smart cities need technologies that can be really applied to raise the quality of life and environment. Among all the possible solutions, Internet of Things (IoT)-based Wireless Sensor Networks (WSNs) have the potentialities to satisfy multiple needs, such as offering real-time plans for emergency management (due to accidental events or inadequate asset maintenance) and managing crowds and their spatiotemporal distribution in highly populated areas (e.g., cities or parks) to face biological risks (e.g., from a virus) by using strategies such as social distancing and movement restrictions. Consequently, the objective of this study is to present an IoT system, based on an IoT-WSN and on algorithms (Neural Network, NN, and Shortest Path Finding) that are able to recognize alarms, available exits, assembly points, safest and shortest paths, and overcrowding from real-time data gathered by sensors and cameras exploiting computer vision. Subsequently, this information is sent to mobile devices using a web platform and the Near Field Communication (NFC) technology. The results refer to two different case studies (i.e., emergency and monitoring) and show that the system is able to provide customized strategies and to face different situations, and that this is also applies in the case of a connectivity shutdown.

Keywords: smart city; emergency management; monitoring; social distancing; neural network; internet of things; wireless sensors network; near field communication technology; computer vision

1. Introduction

The application of Information and Communication Technologies (ICTs) in real contexts is a key factor for the development of smart cities, where proper levels of quality of life and environment, safety, and sustainability are the main targets [1].

The benefits related to the use of ICTs affect everyday life in both normal and emergency conditions. The occurrence of accidental or catastrophic natural events (e.g., sudden floods, earthquakes, fires, etc.) and the adoption of inadequate management strategies are the main causes of the occurrence of emergency conditions in structures and infrastructures. These causes can affect the structural health status of the abovementioned assets, which in turn can affect the health and safety of highly populated areas (i.e., urban contexts, buildings, occasional assembly points for crowds, theme parks, etc.). Significant improvements in disaster management are expected if artificial and human intelligence are integrated (e.g., Disaster City Digital Twin paradigm; cf. [2]), affecting the efficiency of real-time monitoring (by social and remote sensing), of the data analysis (to detect and monitor human activities, damages, and relief needs), and of the scenario simulations (for various and fair allocation of resources, or for training and planning purposes). Consequently, the main disaster management stages (e.g., preparedness and risk reduction, disaster response, recovery) can be boosted if proper information sharing, which includes different systems and stakeholders, is carried out [3].

Furthermore, emergency conditions are defined as the presence or the spreading of high concentrations of pollutants (environmental risk), or of diseases and epidemics caused by bacteria or viruses (biological risk). In the first case, thresholds can be determined, and pollutants can be monitored (e.g., using ICT solutions based on fixed and mobile sensor nodes that actively include citizens in the information gathering [4]) to verify that the pollutants' concentrations are under the given thresholds, or to trigger predefined alarms. In the second case, it is crucial to manage the spatiotemporal distribution of the crowds by using technologies that allow for the implementation of specific and well-designed measures, such as physical distancing (or social distancing) and movement restrictions [5], which are well-known strategies to reduce infection and mortality risks [6,7].

1.1. Literature Review on Available Solutions

In order to face the above problems, several approaches have been proposed. Among them, noteworthy examples of Internet of Things Wireless Sensor Networks (IoT -WSNs), monitoring platforms and recommender systems are reported in the following (Table 1), also exploiting the use of machine learning (ML) in IoT systems [8]. Importantly, the applicability of the WSNs mainly depends on the lifetime of the sensor nodes, and, for this reason, it is important to design this type of system while bearing in mind this crucial aspect and selecting the more convenient energy efficient routing protocol [9–11]. Other important design parameters are [10,12,13]: (i) the limited storing and computational resources of each sensing nodes, (ii) the costs (i.e., cheap sensors are prone to failure, while expensive sensors need good housing and cannot be used for dense deployments), (iii) the position of each sensing node, which cannot be predetermined and depends on the accessibility of the point where the node should be placed, (iv) the sensing nodes' deployment (to collect the needed data, to have the required coverage and connectivity, to extend the network lifetime, and to minimize energy consumption), and (v) the minimum number of time slots required to aggregate data along the edges of a data-gathering tree spanning all the nodes in a WSN (a.k.a., minimum aggregation delay), if the gathered data are aggregated before the transmission to the control center. The solution presented in this paper was designed while bearing in mind all the design parameters mentioned above, focusing, in particular, on maximizing the exploitation of the nodes' storing and computational resources, on minimizing the system cost and on optimizing the system deployment.

Table 1. Examples of Wireless Sensor Networks (WSN) and Internet of Things (IoT)-WSN solutions for emergency and disaster management.

Reference	Main Characteristics	Limitations
[14]	WSN based on energy-efficient wireless sensor nodes equipped with an ultrasonic sensor, which were tested in a field experiment (explosion in a building) to confirm functionality and reliability in terms of collision-free data transmission during the emergency.	Buildings only; Ultrasonic sensor only; Explosion emergency only; Does not consider the overcrowding.
[15]	WSN based on the idea of monitoring the earthquake precursors (e.g., unusual movement of animals, ground water pressure, radon emission, etc.), which was designed for early earthquake warnings and disaster management.	Earthquake emergency only; Does not consider the overcrowding.
[16]	WSNs used together with Unmanned Aerial Vehicles (UAV) for monitoring, forecast, early warning, information fusion and sharing, logistics, evacuation, search and rescue mission, damage assessment.	Does not consider the overcrowding.

Table 1. *Cont.*

Reference	Main Characteristics	Limitations
[17]	WSN paradigm for real-time applications in smart cities, which aims at balancing performance and energy consumption, and uses the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) optimization technique to find the shortest data transmission path.	Paradigm; Does not consider the overcrowding.
[18]	WSN based on a method inspired by biological intracellular signaling, which was designed to perform smog pollution sensing, and taking into account the ad hoc demand routing protocol (AODV) and bellman-ford and interzone routing protocol (IERP) for data transmission.	Air pollution only; Does not consider the overcrowding.
[19]	IoT-WSN based on an evidence-based interactive trust management system for disaster management.	Medical emergency only; Communications between autonomous and adaptive nodes.
[20]	WSN that aims at detecting, in a disaster scenario, moving people without 'tracking devices' (i.e., carrying out the so-called Device free Passive Localization, DfPL).	Does not consider the emergency detection.
[21]	IoT-WSN based on smart fire sensors, cameras, and a Convolutional Neural Network (CNN), acting as a surveillance monitoring system for detecting disasters that occur in a remote area (e.g., a forest).	Remote area only; Does not use sensors for structural monitoring.
[22]	IoT-WSN based on machine learning algorithms that run in a cloud server and includes a modular redundancy fault tolerant scheme to obtain an accurate prediction from sensor data (gas and force sensors) managed by the ultra-low power MSP430 board and a Raspberry Pi, which was designed for early warning in an industry environment.	Gas and force sensors only; Industry environment only; Does not consider overcrowding.
[23]	IoT-WSN that uses the Advanced Adaptive Wavelet Sampling Algorithm (AAWSA) for prolonging the lifetime and power consumption of sensor nodes that include several sensors (i.e., moisture sensors, pressure sensor, rain gauge, tilt meters, and strain gauge), which was developed for disaster prediction in an urban region.	Does not consider overcrowding.
[24]	IoT-based architecture that collects real-time data from the city (from existing sensors at home, parking, vehicular networking, surveillance, weather and water monitoring system, etc.), implemented in the Hadoop ecosystem that allows the processing of Big Data, to obtain a "Smart Digital City".	Architecture; Based on existing sensors.

Another group of solutions for the problems mentioned above is represented by software platforms (Table 2).

Table 2. Examples of platforms for emergency and disaster management.

Reference	Main Characteristics	Limitations
[25]	2D and 3D WebGIS-based platform that has a scalable network architecture and uses a three-tier software architecture, which was designed for effective landslide multilevel management, and an emergency response.	Landslide emergency only; Does not consider overcrowding.
[26]	SENS-ME platform that aims at exploiting the functionalities of Commercial Off-The-Shelf (COTS) smartphones to carry out opportunistic networking, mobile sensing, and distributed information processing.	Does not consider emergency detection. Smartphones as sensors.
[27]	Flood disaster management system (FDMS) that carries out environmental model selection and disaster-related data binding.	Flood disaster only.

Table 2. *Cont.*

Reference	Main Characteristics	Limitations
[28]	DECATASTROPHIZE (a.k.a., DECAT) platform that aims at managing disasters or multiple and/or simultaneous natural and man-made hazards by means of a Geospatial Early-warning Decision Support System (GE-DSS) that allows early warning, decision making, rapid mapping, impacts assessment and mitigation, and geospatial data/information dissemination.	Does not consider overcrowding.
[29]	A web platform developed by the Emergency and Security Coordinating Centre to improve the decision making process of the Canary Islands' Authorities, which provides a geographical and temporal incident distribution and which is able to forecast and classify incidents.	Emergency and security incident distribution only.
[30]	A Building Information Modeling (BIM)-based platform that was designed for building fire emergency management in a dynamic way, i.e., using building users' behavior decisions (e.g., escape, wait for rescue, and fire extinguishing) and both fire and users' positions, which plans action routes and provides visual route guidance.	Buildings only; Fire emergency only; Does not consider overcrowding.
[31]	A smartphone-based platform for city-wide crowd management (through a "heat map-like" system for a real-time overview of the spatiotemporal distribution of crowds in given areas and through specific messaging for real-time, smart, adaptive emergency response and evacuation strategies), which aims at having smart crowds in smart cities and which was used in at least three European countries (UK, Netherlands, Switzerland).	Smartphones as sensors; Does not consider emergency detection.
[32]	A cloud-based architecture for emergency management and first-responders localization (landmark-based and landmark-free), which aims at supporting coordinated emergency management in smart cities based on the localization of first responders during crisis events.	Localization of first responders only.
[33]	Smart disaster management system for transportation applications in smart cities, which gathers information from multiple sources and locations (using VAENTS, i.e., Vehicular Ad hoc Networks, such as Vehicle-to-Vehicle, V2V, Vehicle-to-Infrastructure, V2I, or smartphones or other technologies), detects the point of incidence, makes strategies and decisions (using, e.g., high-performance computing, HPC), and propagates the information to vehicles and other nodes in real time.	Traffic incident only.

Other important and innovative examples of ICT solutions for emergency management refer to: (a) smartphone-based information systems [34]; (b) mobile post-disaster management systems based on free and open source technologies [35]; (c) satellite remote sensing for disaster management [36]; (d) exploiting data from social media [37]; (e) using deep learning to identify survivors in debris from images gathered by smart infrastructures [38]; (f) using an IoT ecosystem (wellbeing-wearable and home automation system sensors with varying communication protocols) to empower the elderly population to self-manage their own health and stay active, healthy, and independent as long as possible within a smart and secured living environment [39].

Recommender systems technology aims at reducing the consumer over-choice due to the huge amount of information available on the web [40]. These systems use information (e.g., location, preferences, past interaction, item features, etc.), gathered through few interactions with its user, to create a customized selection of items (e.g., recommendation list) that can interest the user and can facilitate a better user experience [40,41]. Recommender systems could be further improved if Machine Learning (ML; users extract the features from raw data to feed the algorithm) and Deep Learning (DL; the algorithm automatically extracts features from raw data) algorithms were used for information retrieval [40]. Noteworthy applications of recommender systems in smart cities include: 1) a mobile IoT recommender system for users that need to find Park-and-Ride infrastructures to switch from a private to a public transportation mode [42]; 2) an autonomous situation-aware evacuation route recommender architecture, optimized in real time, to obtain smart buildings [43]; 3) a recommendation system based on the concept of Social IoT (SIoT), i.e., use and reuse data generated by various IoT applications, and adapt the services provided by the single IoT system, to improve the user experience [44].

Based on the literature, different strategies can be adopted to derive/recognize safe routes and assembly points. Traditionally, the problems above can be solved using Shortest Path Finding (SPF) algorithms that belong to the mathematical field of Combinatorial Optimization. Noteworthy examples are: (1) Prim's algorithm, which can be used to derive the Minimum Spanning Tree (MST; i.e., a tree diagram without circles) [45]; and (2) the Dijkstra's and the A* algorithms, which allow one to derive the shortest path between two points [46]. In the Matlab environment, it is possible to find different tools (e.g., "shortestpath" and "digraph") that allow one to solve this problem [47].

This study also refers to a bidirectional exchange of information between a platform and its users, which can be satisfied using NFC technology [48].

The efficiency of the solutions presented above is strictly related to the real-time knowledge of the spatiotemporal distribution of the population, which is fundamental for risk analysis and emergency management (i.e., for an estimation of human exposure and vulnerability) [49]. One of the most used techniques to model or track moving crowds is Computer Vision [50]. Convolutional Neural Networks (CNNs) represent the key algorithms in computer vision, and recently [51], enhanced versions of the tree growth and firefly metaheuristics algorithms have been proposed to automatize the hyperparameters' optimization process (i.e., find the right set of hyperparameters that allow one to obtain the best CNN model accuracy), obtaining better performances (in terms of image classification accuracy and the use of computational resources) than the traditional approaches (that require time expertise). Furthermore, faster regions CNN (Faster R-CNN) is considered one of the most important techniques for automatic pedestrian detection from video [52], and, if automatic color enhancement (ACE) is used, good performances (in terms of recognition rate and offset of target selection) can be obtained because of the reduced susceptibility to the diversity of pedestrians' appearances and the light intensity in specific scenarios (e.g., a subway) [52]. Other suitable techniques, such as satellite imagery processing, could be used as an aid for the automatic recognition of vegetation, landslides, and geospatial data [53,54]. Note that an easy detection of a full-body person in perspective can be carried out using other methods, such as Mask R-CNN [55], Pose2Seg [56], and EfficientDet [57]. Finally, Convolutional Neural Networks are methods specialized in a grid-like structure or multiple arrays form [58,59].

1.2. Objectives and Scopes

In light of the above, previous works [60,61] presented an innovative IoT-WSN platform that was designed as a decision support tool for an Italian theme park. The main limit of the proposed platform was to provide solutions (emergency plans) that neglected the people density (number and spatiotemporal distribution). However, the literature review reported above allows one to identify solutions, i.e., the recommender system and the computer vision applications, which can be effectively used to improve the abovementioned tool. Hence, for this reason, in the study described in the following sections, we report the results of the integration of these solutions into the previous platform.

The main novelty of the study refers to the integration in an all-in-one solution, for the first time as per the authors' knowledge, of available and new algorithms and ICT solutions that allow one to solve two important problems related to the theme park, i.e., the overcrowding and the emergency management, which were presented as two case studies in this paper.

Consequently, the remaining part of the paper is organized as follows. The following section (Section 2) describes the IoT system, based on the platform cited above (Hardware in Section 2.1), and how the system uses input data (Software in Section 2.2) to carry out the main objective of this study, i.e., obtain a multipurposes IoT-WSN-based system that can be used by authorities for emergency and crowd management, and by authorized customers as a decision support tool. Subsequently, Section 3 describes two case studies that were used to validate the proposed system, including dataset generation and algorithms application. Section 4 reports the main results related to the case studies mentioned above, which were derived from simulations. Finally, the last two sections contain the main conclusions derived from this study and some anticipatory remarks about future works.

2. IoT System Description

In this section, the main components of the IoT system presented in this paper will be described. The system consists of two components, i.e., a hardware (i.e., an IoT-WSN) and a software (NN-based and SPF algorithms for emergency management and social distancing purposes).

2.1. Hardware: A Multisensor IoT-WSN

The IoT-WSN proposed in this paper consists of wireless sensing nodes, which are fed using a suitable tuned photovoltaic energy harvester [62]. Figure 1 shows the main components of the first prototype of one wireless sensing node of the WSN. In particular, the figure cited above shows (1) the power supply unit of the sensing node (which consists of a 5 W-18 V polycrystalline silicon photovoltaic panel, a 12 V-20 A recharge circuit, and a 12 V-12 Ah battery), and (2) an ultralow-power IoT board [63], which is the core of the sensing node.

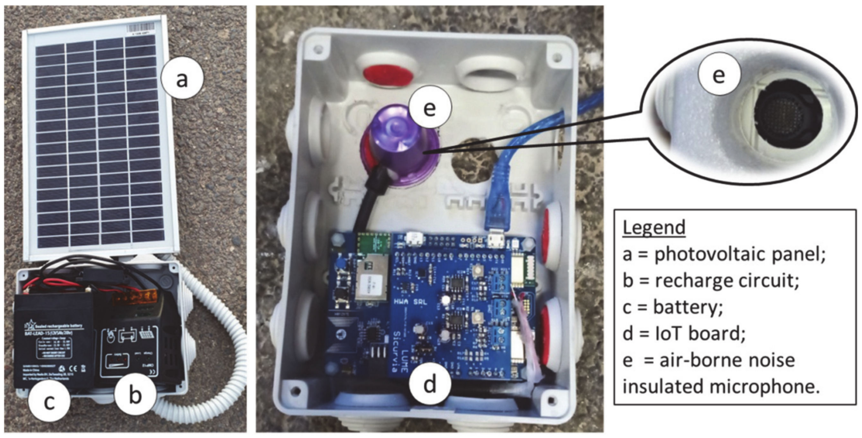


Figure 1. First prototype of the wireless sensing node of the proposed WSN.

In more detail, the IoT board includes several MEMS sensors (i.e., a 3D accelerometer, a microphone, a temperature/humidity sensor, magnetometer, barometer, etc.) and a microcontroller. The IoT board can wirelessly transmit the data gathered by the sensors, using different standards and protocols, i.e., Wi-Fi, Bluetooth, and NFC tags. It is important to underline that the NFC protocol allows for further applications, e.g., the interaction/exchange of information with the theme park visitors during normal and emergency conditions, also in the absence of a remote connection, thus allowing a backup communication channel with the users. Note that, to obtain a more detailed environmental and structural monitoring, the IoT board was also equipped with a smoke sensor (to detect carbon monoxide, liquid petroleum gas, and smoke), a flame sensor, and an additional microphone. The latter was added to receive the vibro-acoustic response [64–66] of the structure on which the units were installed. In more detail, the additional microphone was isolated from the airborne noise through a cover (inside the box) and isolating material (between the box and the structure), and was able to receive the acoustic signals (a.k.a., vibro-acoustic signature of the structure) that travelled into the structure. Proper analyses of these acoustic signals (i.e., feature extraction, multidomain analysis for cracks identification and monitoring, structural health status classification; [64–66]) allow one to carry out the Structural Health Monitoring (SHM) of the structures where the sensing node is attached, which is also applicable in the case of road monitoring, where the use of self-powered and smart sensors is envisioned [67].

The study reported in this paper refers to two case studies that were carried out using, as a reference place, an Italian theme park. Figure 2a shows the map of the park, where lines point out boundaries and main routes (solid and dashed, respectively), numbers (from 1 to 6) point out the positions in which the sensing nodes of the WSN were installed, and the positions of the two exits (called Exit 1 and Exit 2) and the two assembly points (called AP 1 and AP 2) are indicated. Figure 2 also shows three prototypes of wireless sensing nodes installed at different points of the theme park, i.e., a road pavement near the Exit 1 (see Figure 2b, which refers to point 1 in Figure 2a), a masonry structure (historical military fort; see Figure 2c, which refers to point 5 in Figure 2a), a light pole (see Figure 2d, which refers to point 6 in Figure 2a). Importantly, the prototype cited above have been subsequently equipped with Wi-Fi cameras that allow computer vision analyses aimed at defining the spatiotemporal distribution of the visitors of the theme park per given area.

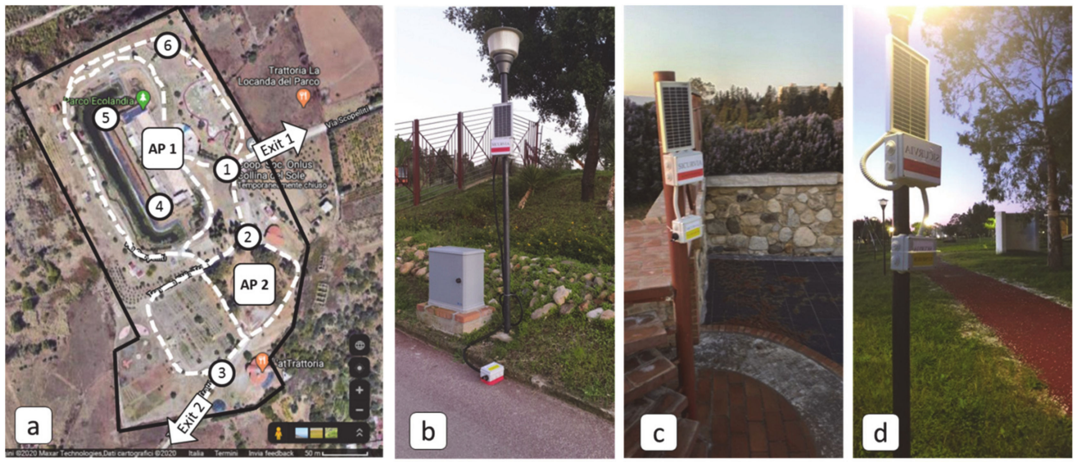


Figure 2. (**a**) Map of the theme park where the WSN was installed and three self-powered wireless sensing nodes installed on (**b**) a road pavement, (**c**) a masonry structure, and (**d**) a light pole in the park.

2.2. *Software: NN-Based and SPF Algorithms for Emergency Management and Social Distancing*

The framework of the proposed IoT system is depicted in Figure 3. In particular, the system includes the WSN (which was described in the previous section and is represented by box 1 in Figure 3) and a procedure, consisting in several steps (represented by boxes 2 to 13 in Figure 3), which allows one to analyze the data that come from the WSN (i.e., sensing nodes and cameras).

The sensing nodes of the WSN (see box 1 in Figure 3) gather data from the environment using their sensors, while the Wi-Fi cameras detect the people's position in a given area. Sensing nodes and cameras are fed by a photovoltaic-based power supply system. The aforementioned data are sent, with a proper timing (e.g., every five minutes), to a local server (see box 2 in Figure 3) for analysis/backup purposes. At the same time, the IoT board converts the collected data to text strings (JSON format) and sends the converted data to a web server (see box 3 in Figure 3) using the MQTT protocol (e.g., the MQTT broker Eclipse Mosquitto™) for backup purposes. Note that, sampling and data transmission frequencies are defined so as to minimize the transmission cost, latency, network bandwidth, and resource requirements, and to increase data privacy and data transfer reliability.

The data stored in the web local server can be accessed by an authorized customer of the system (i.e., subscribing to a specific topic) for further analyses, while the data stored in the local server are analyzed as follows using three different algorithms (see boxes 4–6 in Figure 3).

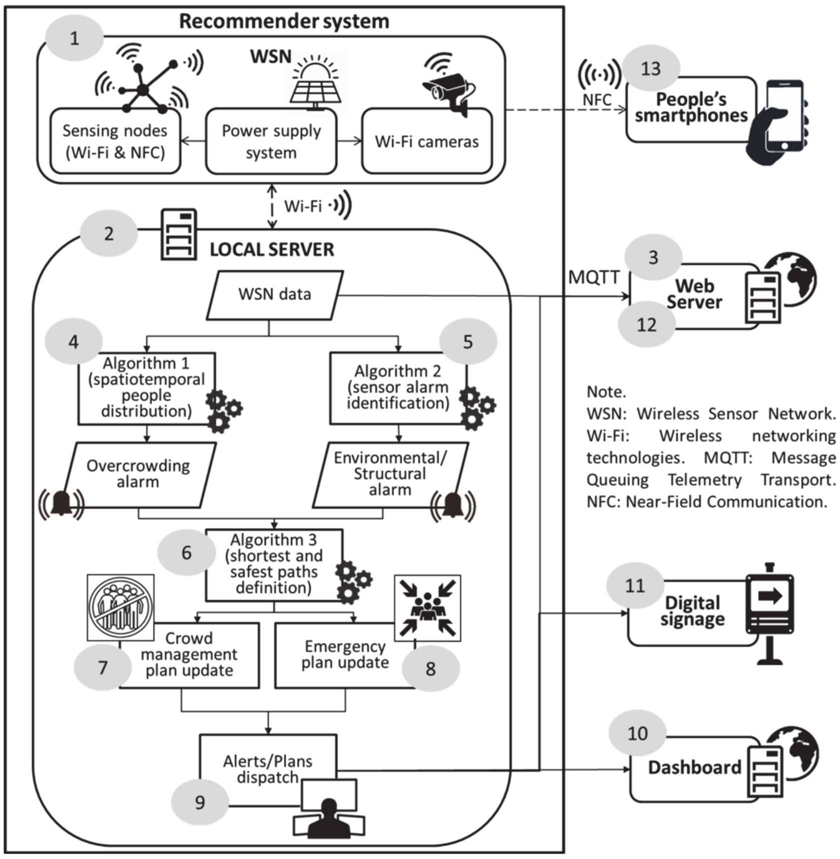

Recommender system
1
WSN
Sensing nodes (Wi-Fi & NFC)
Power supply system
Wi-Fi cameras
Wi-Fi
2
LOCAL SERVER
WSN data
4
Algorithm 1 (spatiotemporal people distribution)
5
Algorithm 2 (sensor alarm identification)
Overcrowding alarm
Environmental/ Structural alarm
6
Algorithm 3 (shortest and safest paths definition)
Crowd management plan update
7
Emergency plan update
8
Alerts/Plans dispatch
9
NFC
13
People's smartphones
MQTT
3
Web Server
12
Note.
WSN: Wireless Sensor Network.
Wi-Fi: Wireless networking technologies.
MQTT: Message Queuing Telemetry Transport.
NFC: Near-Field Communication.
11
Digital signage
10
Dashboard

Figure 3. Framework of the proposed IoT system.

The first algorithm (Algorithm 1, represented in box 4 in Figure 3) is used to process the data that come from the cameras using computer vision to define the spatiotemporal distribution of the people in each area and trigger the overcrowding alarm (if people in a given area exceed a predefined threshold). Hence, Algorithm 1 can be used as an automatic tool when the social distancing measure must be respected or to limit the access to a given area, avoiding overcrowding. In more detail (cf. Table 3), this algorithm uses object detection frameworks to classify and locate objects in a visual field. Facial detection would take an image input of some kind, check for the person or face class of the objects, and locate them in the frame. Additionally, facial recognition would pick out eyes, mouths, and various other features to compare them to a known dataset. The applications detect humans in the visual field via processing blocks pre-trained by crunching a huge number of images with a deep learning artificial intelligence system. The extracted features, as the number of humans and the direction of the people flow, are used in the following to determine the spatiotemporal distribution.

Table 3. Main characteristics of the Algorithm #1.

Scopes	Input	Output	Steps
Count the number of people entering and leaving a given area using data from cameras.	• Video stream from cameras installed at strategic points of the theme park.	• Number of people entering and leaving a given area.	- Receive data from cameras; - Count the number of people entering and leaving a given area using data from cameras.

The second algorithm (Algorithm 2; see box 5 in Figure 3) is used to process the raw data from the sensors (cf. Table 4). In particular, raw data are preprocessed in order to extract meaningful features (i.e., statistical indicators such as maximum, minimum, average, root mean square, etc.), which are used as input in a NN for clustering purposes. This NN is used to identify the occurrence of one or more environmental and/or structural alarms detected by one or more sensing nodes of the WSN. The output of the NN cited above is one of all the possible combinations of alarm. The number of alarm combinations is equal to the permutations of the number of sensing nodes without repetitions and can be calculated using the following expression:

$$A = \sum_{i} \frac{N!}{K_i! \cdot (N - K_i)!}, \tag{1}$$

with i = 1, . . . , 6, and where N is the length of the array {1, 2, 3, 4, 5, 6} that represent the sensing nodes of the WSN, and K is an integer that shows how many nodes detected an alarm at the same time. Note that the WSN used in this study consists of six sensor nodes. Hence, the number of alarm combinations A without repetitions (e.g., the condition "alarm actives at node 1 + alarm actives at node 3" is equal to the condition "alarm actives at node 3 + alarm actives at node 1") is 63.

Table 4. Main characteristics of the Algorithm #2.

Scopes	Input	Output	Steps
Classify environmental and structural conditions and trigger related alarms when predefined thresholds are exceeded.	• Sensor data; • Thresholds.	• Features from sensor data; • Emergency (environmental, and/or structural) alarm(s).	- Receive sensor data; - Extract features from sensor data; - Classify sensor data identifying environmental and structural conditions based on predefined thresholds.

The third algorithm (Algorithm 3, represented by box 6 in Figure 3) is used to define the shortest and the safest paths if the Environmental and/or Structural alarm or the Overcrowding alarm is active in at least one node of the WSN. In particular (cf. Table 5), the Algorithm 3 automatically updates: (1) when the overcrowding alarm is active, a crowd management plan that consists of an image that highlights, with red areas and white signals, the overcrowded areas (see two examples in Figure 6); (2) when the environmental/structural alarm is active, an Emergency plan that consists of an image (see two examples in Figure 8) that shows in which nodes the alarm is active (with red circles and yellow triangles), if an exit is closed (with a no entry sign), and the exit or the assembly point that must be reached (with green and white signs, respectively). In more detail, Algorithm 3 is based on the graphs shown by Figure 4 (i.e., graph in Figure 4a was used in the case study 1, while graph in Figure 4b was used in the case study 2) and was developed using Matlab tools (i.e., using the "shortestpath" and "digraph" functions [47]). It takes as input the connections among the graphs and the alarm conditions, and takes the people in each area as weight; it then returns the sequence of nodes that should be passed through, which corresponds to the safest path for the people who, because of active alarm/s, must reach a predefined point (which could be an exit or an assembly point) from their starting node.

In summary, the fastest (i.e., on paths traveled by few people) and safest (i.e., avoiding nodes where an alarm is active) path is suggested at the same time, considering that the spatiotemporal distribution of the people could affect the speed of the flow to reach the available exits or, alternatively, the assembly points.

Note that Algorithm #3 is based on the assumption that all the edges of the graph have the same endless capacity, and the main parameter used to evaluate the availability of each edge is the number of people that are on the edge when an alarm is activated. Based on this assumption, the objective of the algorithm is not the flow of people that can move on the graph per unit of time but to recognize and provide, as a solution, the less busy path to reach an exit or an assembly point. Importantly, although the Matlab function "shortestpath" was used, Algorithm #3 does not select the "shortest" path but selects the path that includes edges that allow one to reach a final point (exit or assembly point) while avoiding slowdowns, even though the path is the longest one.

Table 5. Main characteristics of the Algorithm #3 (consisting of two parts).

Scopes	Input	Output	Steps
• Close area(s) in case of overcrowding alarm; • Find the shortest and the safest paths in case of an emergency alarm.	For part 1: • Graph of the park; • Number of people in each area of the park; • Number of people entering and leaving each area of the park.	For part 1: • Graph of the park with closed and open areas; • Number of people leaving overcrowded area(s); • Number of people entering and leaving not overcrowded area(s).	For part 1: - Compare number of people (P) inside an area with a threshold (T); - If P>T, trigger the overcrowding alarm; - Close overcrowded area(s); - Define the flows of people leaving the overcrowded area(s). - Update the digital signage and the dashboard.
	For part 2: • Graph of the park; • Active alarm(s) at the sensing nodes.	For part 2: • Graph of the park with nodes with activated alarms; • Shortest and safest paths to reach an available exit(s) or assembly point(s).	For part 2: - Update graph's node(s) N with activated alarm(s); - If N is(are) closed to exit(s), exit(s) is(are) closed. - If all exits are closed, the assembly points are activated; - Shortest and the safest paths are calculated to leave all the nodes and reach an available exit(s) or assembly point(s). - Update the dashboard.

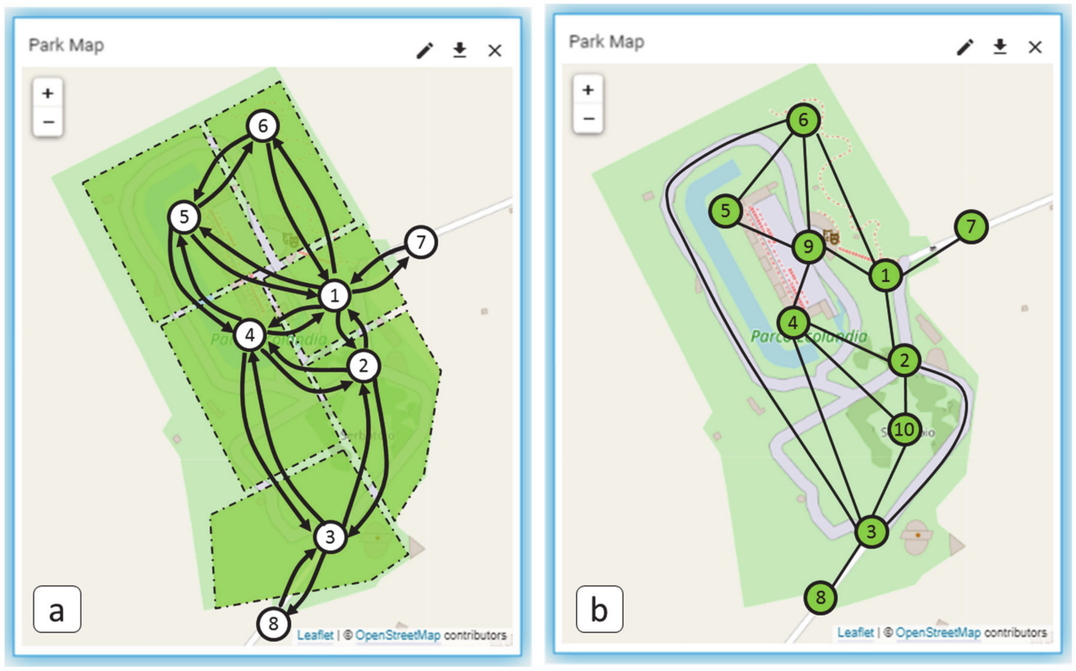


Figure 4. Graphs used by the Algorithm 3 in case of (**a**) Overcrowding alarm (Case study 1: monitoring scenario and social distancing); (**b**) Environmental and/or Structural alarm (Case study 2: emergency scenario and emergency management).

Finally, the local server dispatches the plans defined above using different ways, i.e., (1) using internet via a dashboard of the IoT system (e.g., an open-source server-side platform dashboard for controlling IoT devices and visualizing the sensor data in real time was used in this study (Thingsboard [68]); see box 10 in Figure 3; Figure 5a) that can be accessed by the users of the IoT system (e.g., staff members or visitors of the park); (2) using a digital signage that is carefully spread throughout the park (see box 11 in Figure 3); (3) using internet via the abovementioned web server (see box 12 in Figure 3); (4) via the NFC interface of the sensing nodes (see box 13 in Figures 3 and 5b) in case of connectivity shutdown or if the user is close to the node and has a smartphone.

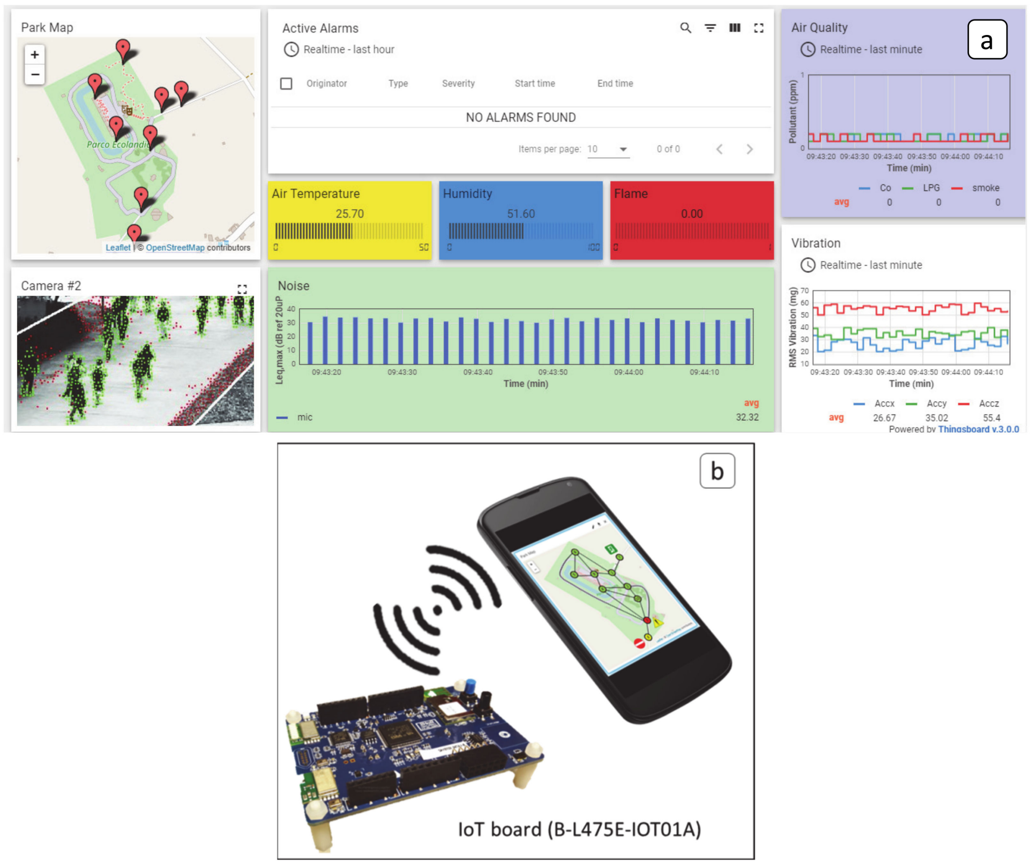


Figure 5. (**a**) IoT system dashboard; (**b**) Emergency plan provided by the NFC module-smartphone connection in the form of a smartphone compatible map.

3. Case Studies

This section of the paper contains the description of two simulated cases studies in which the IoT system was applied to face two different problems, i.e., an emergency scenario and a monitoring scenario, occurring in the abovementioned theme park.

3.1. Case Study 1: Monitoring Scenario (Social Distancing)

The first case study refers to a monitoring scenario. In particular, the IoT system is envisioned as a tool to control the spatiotemporal distribution of the people in a given area and to trigger, automatically, the overcrowding alarm that allows one to respect social distancing. Figure 6 shows two examples of a crowd management plan, where the overcrowded areas are highlighted in red, while white signals indicate which sensing node detected the overcrowding alarm.

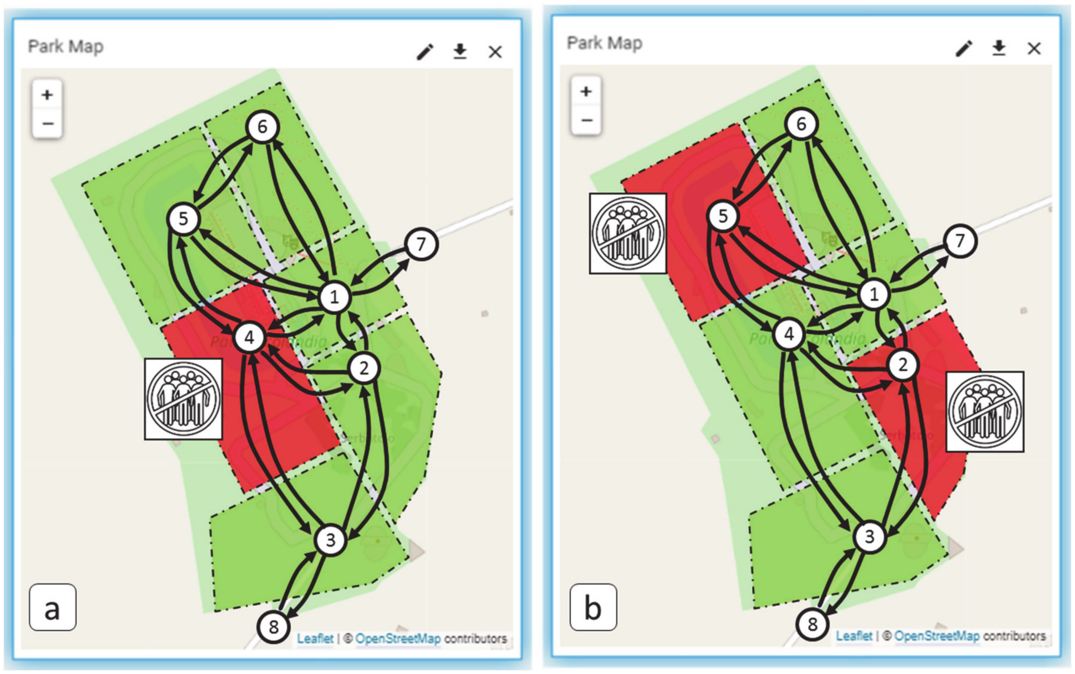


Figure 6. Two examples of a crowd management plan in case of an overcrowding alarm active at (**a**) node 4, and (**b**) nodes 2 and 5.

3.1.1. Case Study 1: Data Generation

As described above (cf. description of Figure 3), the data for this application consists of video from Wi-Fi cameras sent using a Raspberry Pi. The used cameras provide a video with 30 fps and a resolution of 1280x960 pixels. Figure 7 reports an example of the output obtained from the Algorithm 1 (note that, in this case, the algorithm counts the people between the two white lines on the pavement).

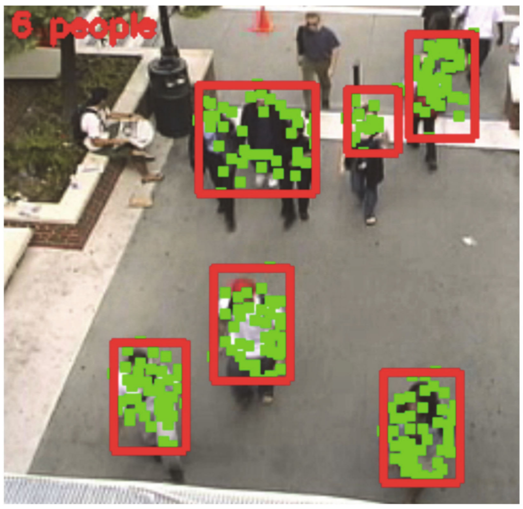


Figure 7. Example of the application result of the Algorithm 1 (computer vision).

3.1.2. Case Study 1: Algorithms

Based on the above, a computer vision algorithm—the Algorithm 1—with the following characteristics can be used to derive the spatiotemporal distribution of the people in each monitored area of the park (see Figure 4a) from the dataset described above (i.e., video from Wi-Fi cameras) and is used to trigger the overcrowding alarm. The software was developed using the programming language Python.

In pursuing the implementation of person detection and tracking, we use a two-step procedure for (a) the detection of the position of the people in the video frame, carried out periodically due to the large computational cost and implemented with the built-in OpenCV algorithm HOG (Histogram of Oriented Gradients) + Linear SVM model [69], and (b) the tracking of the people's movement around the video frames, using the built-in OpenCV algorithm Discriminative Correlation Filter with Channel and Spatial Reliability (DCF-CSR) [70].

For every camera installed in every single node, the output of the software task is the value of people actually in the area, with a refresh rate of 1 sec. The maximum number of people admitted in every single area is a priori known, so an alarm is raised if the number of people in the area is above the threshold. If the overcrowding alarm is detected by Algorithm 1, the Algorithm 3 is triggered to define a crowd management plan, as shown in the previous section. In this case, the "digraph" function [46] is automatically activated to derive the shortest path that the people in the overcrowded area must follow to leave the area.

3.2. *Case Study 2: Emergency Scenario (Emergency Management)*

The second case study refers to an emergency scenario, in which sensor alarms are active and the IoT system is envisioned as a tool to control the environmental and structural conditions of the areas and assets of the park and to automatically recognize several sensor alarm conditions and provide, in real-time, customized emergency plans that show safe paths for the park's visitors, who need to reach an exit or an assembly point from one of the six areas of the park because of the emergency.

Figure 8 shows two examples of emergency plans, which contain: (i) a graph that indicates the connections among the nodes, exits, and assembly points; (ii) sensing nodes that are represented by red and green circles if an alarm is active or not, respectively; (iii) exits that are represented by yellow and green circles if the exit is safe or not, respectively; (iv) yellow triangles with a black exclamation point that indicate the point where the alarm is active; (v) green signals that indicate an available exit and red access-denied signals that point out an unavailable exit (closed); and (vi) white signals that show the assembly points.

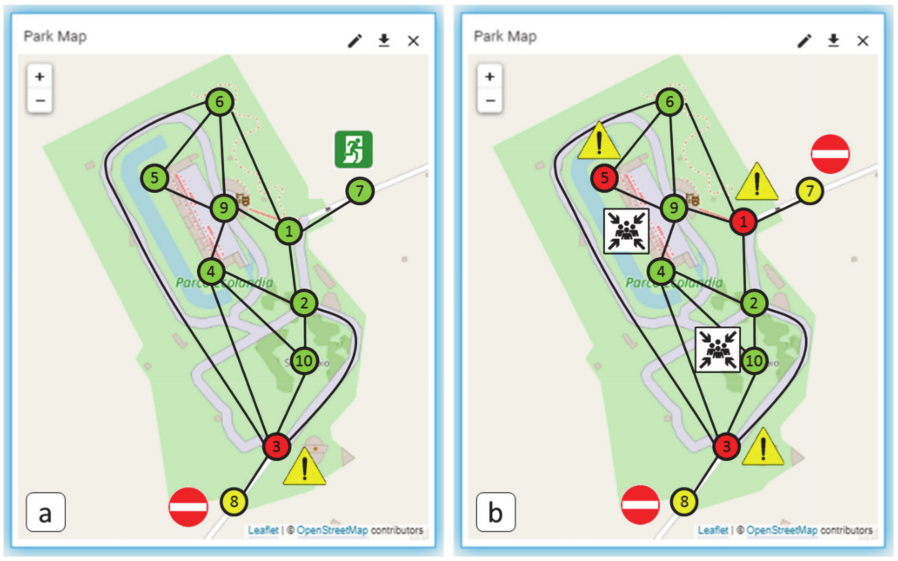


Figure 8. Two examples of emergency plans in case of an environmental/structural alarm active at (**a**) node 3, and (**b**) nodes 1, 3, and 5.

3.2.1. Case Study 2: Dataset Generation

The dataset for this case study consists of one matrix (see Figure 9), which is used to train the Algorithm 2 (described below) and contains sensor data (10 different sensors per each sensing node of the WSN). This matrix simulates the 64 possible park conditions (i.e., the condition 0—no sensor alarm—together with the 63 sensor alarm conditions defined above) and a number from 0 to 63 that represents all the possible park conditions. In more detail, the matrix was created using a MATLAB script that generates, for each sensor, a suitable value in the range of operations using a random function with a Gaussian distribution. Furthermore, some of the data is "faulted" to inject alarm conditions. The matrix contains 32,000 records of data (10 sensors for each of the six sensor nodes) and has been randomized and standardized during the preprocessing task. The last column of the matrix represents, for each record, the labeled alarm condition.

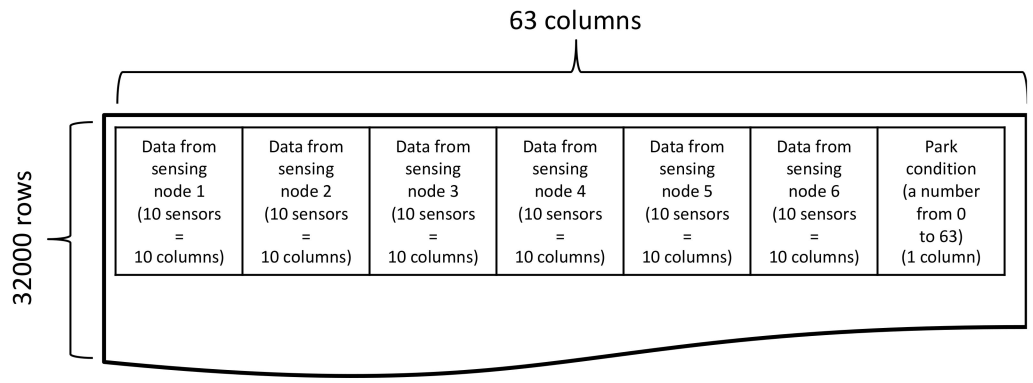

63 columns
32000 rows
Data from sensing node 1 (10 sensors = 10 columns)
Data from sensing node 2 (10 sensors = 10 columns)
Data from sensing node 3 (10 sensors = 10 columns)
Data from sensing node 4 (10 sensors = 10 columns)
Data from sensing node 5 (10 sensors = 10 columns)
Data from sensing node 6 (10 sensors = 10 columns)
Park condition (a number from 0 to 63) (1 column)

Figure 9. Table description of the input matrix used by the algorithm for emergency management.

Note that every row of the above matrix is a complete record of a single sensors' data acquisition and has a 1D grid-like structure (1×63). Importantly, the position of every column is fixed, i.e., containing a specific order of the sensor data (accelerometer, temperature, humidity, etc.). Hence, every row is a different set of sensors' data in a specific column position for a single acquisition. In this way, the NN is able to learn the relationship between data related to each alarm condition, using every single row as a full set of information.

3.2.2. Case Study 2: Algorithms

As described above (cf. description of Figure 3), in an emergency scenario the Algorithm 2; Algorithm 3 are used. In particular, Algorithm 2 is a clustering NN, which was used in this study to detect environmental/structural alarm from the above matrix and has the following characteristics: (i) Input size = 63; (ii) Output size = 64; (iii) Hidden layers = 55; and (iv) Batch size = 50.

Furthermore, Algorithm 2 uses the Stochastic Gradient Descent (SGD) as an optimizer function (i.e., an iterative method for optimizing an objective function with suitable smoothness properties). Meanwhile, the activation function REctified Linear Unit (relu) is used for hidden layers (relu is the most commonly used activation function in neural networks, especially in NNs). The output layer uses the Softmax function, which takes as input a vector of K real numbers, and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. The outputs of the algorithm are the nodes in which the alarms are not active. Using this information, the definition of the graph G is modified accordingly, thus eliminating the unsafe nodes and the possibility of passing through a damaged infrastructure, creating a simplified graph G^*.

Moreover, data from Algorithm 1 are also used to update the weights of the graph connections. If an environmental or structural alarm is detected by Algorithm 2, the Algorithm 3 is used to define an Emergency plan, using the graph G^* as inputs. In this case, the "digraph" function [46] is automatically activated to derive the fastest and safest paths that the people near to each node must follow to reach an exit (if available) or an assembly point (if the exits are closed).

4. Results and Discussions

Algorithm 1 was tested using Anaconda, which is a free and open-source distribution of the Python and R programming languages for scientific computing. The performances were estimated using several prerecorded videos that compared the output to the manual counting of the people, showing an accuracy above 90%.

Algorithm 2 was tested using Keras, a deep-learning library that allows for easy and fast prototyping. Algorithm 2 showed a test accuracy of about 85%.

Algorithm 3 (see Figure 10) was tested using Matlab, taking as inputs the output of Algorithms 1 and 2 and modifying, accordingly to the above results, the G graph. When it is executed with data coming from Algorithm 1, the Algorithm 3 shows the areas that are closed because of the activation of the overcrowding alarm. In the case of data coming from Algorithms 1 and 2, the Algorithm 3's output represents the safest and fastest path to reach an exit or assembly point in case of active environmental or structural alarms.

(a)Algorithm 3 (part 1):
Crowd Management

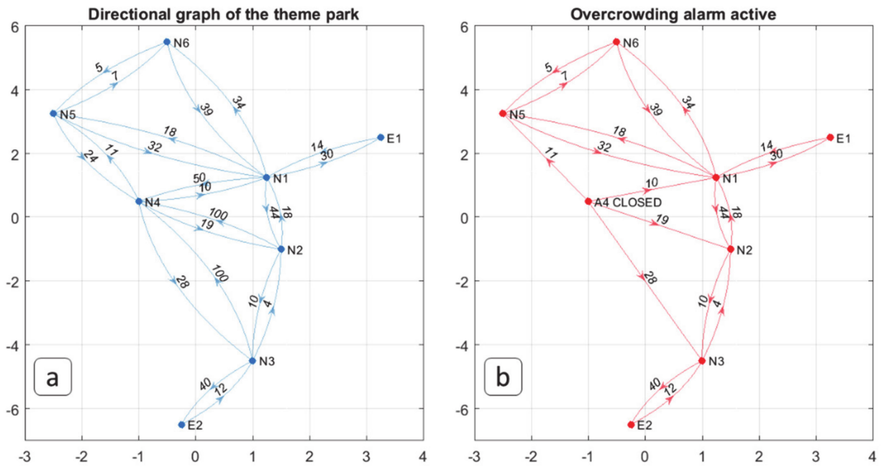
```
1)  Create a weighted graph using the Matlab
    function "digraph";
2)  Define the array "thresholds";
3) For each area:
  3.1) Count the people coming in (P_IN) and
       out (P_OUT) the area based on the
       weights of the graph;
  3.2 Calculate the people inside each area
       (P_IN-P_OUT);
  3.3 If P_IN > P_OUT and P_IN > threshold,
       trigger the overcrowding alarm at the i-
       th area;
  3.4) Close the area removing the edges
       directed to the area;
4)  Create a new graph where active alarm(s)
    and closed area(s) are indicated.
```

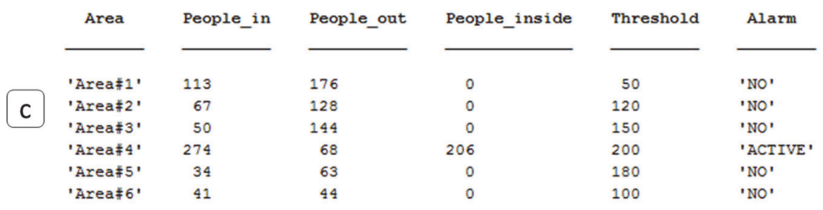
(b) Algorithm 3 (part 2):
Emergency Management

```
1)  Create a weighted graph using the Matlab function "graph";
2)  Import the array "EMERGENCY" from the Algorithm 2;
3)  Find the targets of the algorithm from the exits and the
    assembly points based on the array above;
4)  Find the node(s) Ni (with i=1,…,6 when alarms are detected
    in all the nodes) where the alarm(s) was/were detected;
        For each node in which the alarm is active:
      5.1) Create a new graph Matlab function "graph" indicating
           active alarm(s) and closed exit(s);
      5.2) Find the possible safe paths with the Matlab function
           "shortestpath";
      5.3) Select the fastest path among the paths above using the
           weights;
5)  For each node in which the alarm does not active: repeat the
    steps from 6.1 to 6.3;
6)  Create a new graph where active alarm(s) and closed exit(s)
    are indicated.
```

Figure 10. Pseudo-code of the Algorithm 3 for: (**a**) crowd and (**b**) emergency management.

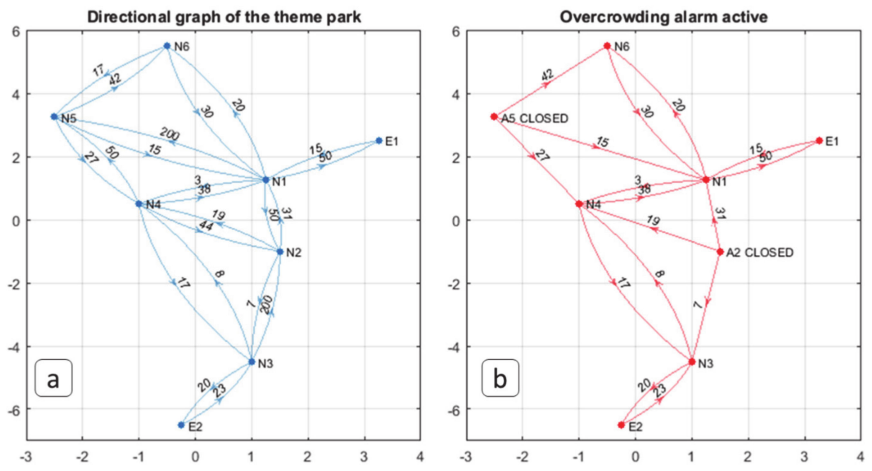
The following figures (Figures 11 and 12) show the results of the Algorithm 1 in two simulated cases of overcrowding alarm according to those shown by Figure 6. Note that Figure 11a is an example of a graph of the theme park without alarm, while Figure 11b is an example of a graph with alarm. The latter was automatically generated because of the activation of the overcrowding alarm at node 4, which led to the closing of the area number 4 (please note that the name of node 4 changed from "N4" to "A4 CLOSED"). Meanwhile, Figure 11c is a table that shows the overall conditions and that consists in the columns: (i) "Area", which reports the six areas into which the theme park was divided; (ii) "People_in", which reports the number of people entering in each area (which will be provided by the Algorithm 1 in the real application of the IoT system); (iii) "People_out", which reports the number of outgoing people from each area (provided by the Algorithm 1 in the real application); (iv) "People_inside", which reports the number of people that are inside each area at the time t_1 (when the crowd management plan was generated), which was calculated by subtracting the entering and outgoing people in relation to each area (note that only the positive values are reported, while the null and the negative ones are represented with zeros); (v) "Threshold", which reports the maximum number of people allowed to stay at the same time in each area; and (vi) "Alarm", which can be "NO" and "ACTIVE" if the overcrowding alarm is inactive and active, respectively. Note that, when an area is closed, it is only possible to leave the area (see the arrows on the graphs), and specific messages, shown by the digital signages installed in the park, prevent the access to the closed area.

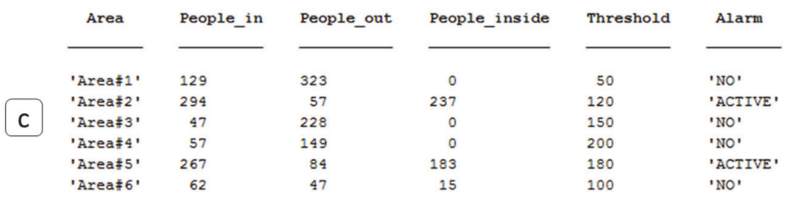

Directional graph of the theme park
Overcrowding alarm active
N1
N2
N3
N4
N5
N6
E1
E2
A4 CLOSED
a
b


c

Area	People_in	People_out	People_inside	Threshold	Alarm
'Area#1'	113	176	0	50	'NO'
'Area#2'	67	128	0	120	'NO'
'Area#3'	50	144	0	150	'NO'
'Area#4'	274	68	206	200	'ACTIVE'
'Area#5'	34	63	0	180	'NO'
'Area#6'	41	44	0	100	'NO'

Figure 11. Example of the results of the Algorithm 3 in case of an overcrowding alarm at node 4, where (**a**) is the graph without an alarm, (**b**) is the graph in case of an alarm, and (**c**) is a table showing the overall condition.


Directional graph of the theme park
Overcrowding alarm active
N1
N2
N3
N4
N5
N6
E1
E2
A5 CLOSED
A2 CLOSED
a
b


c

Area	People_in	People_out	People_inside	Threshold	Alarm
'Area#1'	129	323	0	50	'NO'
'Area#2'	294	57	237	120	'ACTIVE'
'Area#3'	47	228	0	150	'NO'
'Area#4'	57	149	0	200	'NO'
'Area#5'	267	84	183	180	'ACTIVE'
'Area#6'	62	47	15	100	'NO'

Figure 12. Example of the results of the Algorithm 3 in case of an overcrowding alarm at nodes 2 and 5, where (**a**) is the graph without an alarm, (**b**) is the graph in case of an alarm, and (**c**) is a table that shows the overall condition.

As in Figure 11, Figure 12 shows the results of the Algorithm 3 in case of an overcrowding alarm, but, in this case, this alarm was detected by nodes 2 and 5, contemporaneously.

The next figures (Figures 13–18) show two examples of the Algorithm 3's output, which were obtained by carrying out two simulations according to Figure 8 (i.e., environmental/structural alarm active at node 3 (see Figures 13–15) and at nodes 1, 3, and 5 (Figures 16–18)).

In particular, Figure 13a shows the weighted graph (the weights are the people on the path between two nodes) of the theme park without an alarm, while Figure 13b shows the graph in case of an alarm. In more detail, as shown by Figure 13b, when an alarm is detected by the Algorithm 2 at an i-th node (node 3 in this case), the Algorithm 3: (1) changes the name of the i-th node in the graph to point out the node in which the alarm is active (see "N3 ALARM" in Figure 13b); (2) removes the edge between the i-th node and the other node without an alarm; (3) if the i-th nodes are those that are near one exit (i.e., nodes 1 and 3), the name of the exit changes as well (see "E2 CLOSED" in Figure 13b).

Note that, in order to improve the clarity of the figure below, the position of node 6 was defined so as to show the node 6-node 3 connection in a better way, and that, in case of an alarm, the weights were removed from the related graphs.

Figures 14 and 15 show the safest (i.e., that do not include nodes with active alarms) and fastest (i.e., that consider the people on the edges as a resistance to the crowd flow) paths. These paths can be used by the people near each node of the system (from node 1, N1, to node 6, N6) to face the environmental/ structural emergency detected by node 3 (N3). Importantly, the Algorithm 3 also provides a path for people that are near N3 where the alarm was detected. In addition, a table that summarizes the overall conditions is reported in Figure 15c. This table shows: (i) the "Start Node", which is the node from which the paths start; (ii) the "End Node", which is the final destination of the people reaching a safe place (i.e., an assembly point); (iii) the safest and fastest path, consisting of the edges between the i-th node (Ni) and the n-th node (Ni+3); (iv) the node(s) where the alarm(s) was/were detected.

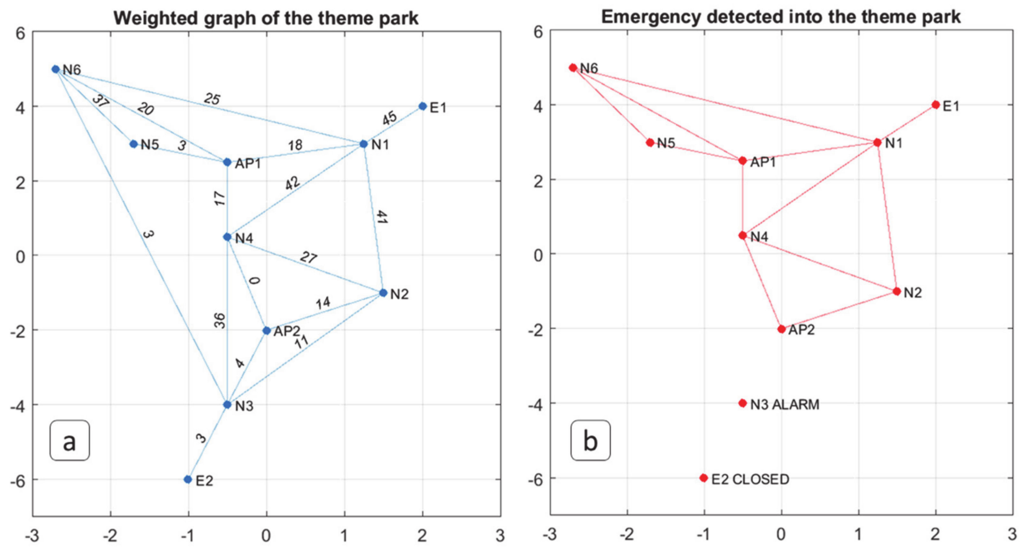

Weighted graph of the theme park
N6
N5
AP1
N1
E1
N4
N2
AP2
N3
E2
37
20
25
45
3
18
42
17
41
3
27
0
14
36
11
4
3
a
Emergency detected into the theme park
N3 ALARM
E2 CLOSED
b

Figure 13. Example of the results of the Algorithm 3 in case of an environmental/structural alarm at node 3 (N3), where (**a**) is the graph before the alarm triggering and (**b**) is the graph after the alarm triggering.

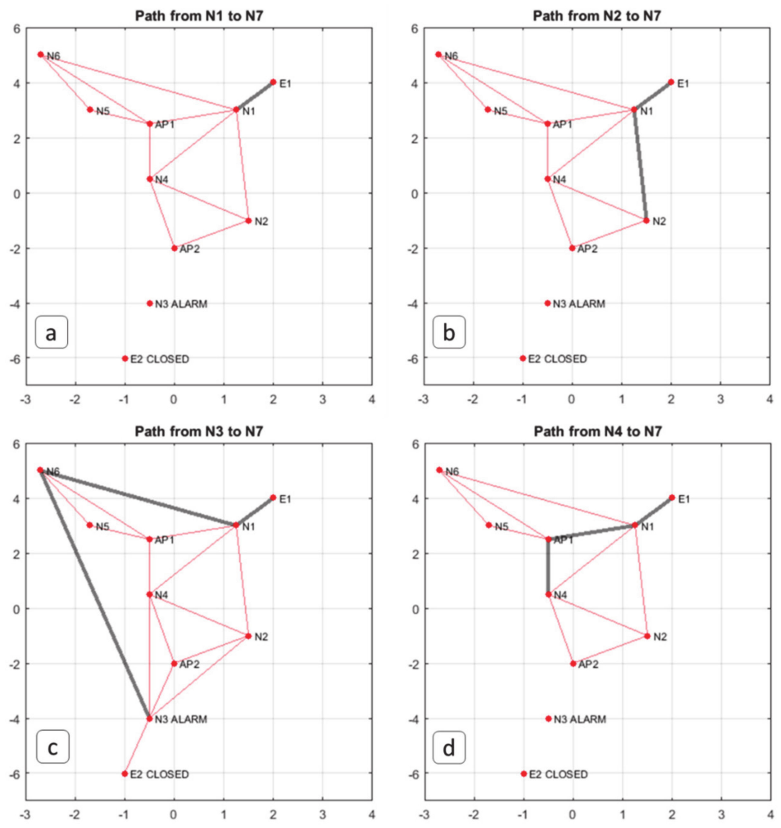
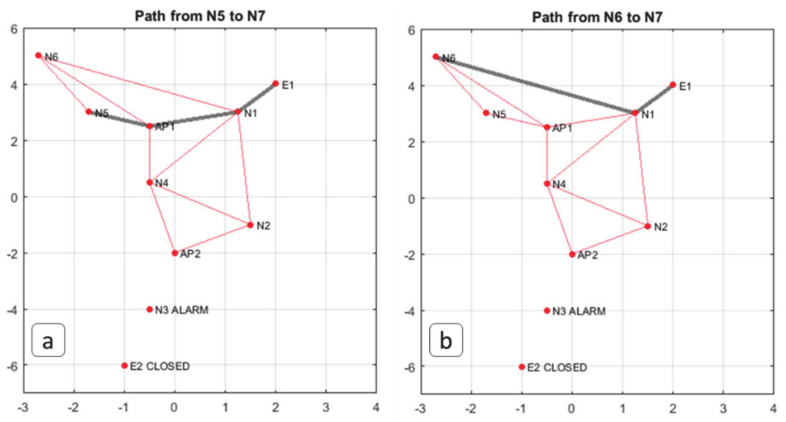
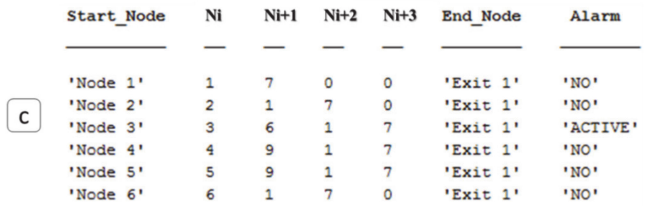


Figure 14. Example of the safest and fastest paths provided by the Algorithm 3 in case of an environmental/structural alarm at node 3 (N3), which can be used by people at (**a**) node 1 (N1), (**b**) node 2 (N2), (**c**) node 3 (N3), and (**d**) node 4 (N4) to reach the exit 1 (E1 = N7).





```
   Start_Node     Ni    Ni+1    Ni+2    Ni+3    End_Node     Alarm
   __________     __    ____    ____    ____    ________    ________

    'Node 1'      1      7       0       0      'Exit 1'    'NO'
    'Node 2'      2      1       7       0      'Exit 1'    'NO'
c   'Node 3'      3      6       1       7      'Exit 1'    'ACTIVE'
    'Node 4'      4      9       1       7      'Exit 1'    'NO'
    'Node 5'      5      9       1       7      'Exit 1'    'NO'
    'Node 6'      6      1       7       0      'Exit 1'    'NO'
```

Figure 15. Example of the safest and fastest paths provided by the Algorithm 3 in case of an environmental/structural alarm at node 3 (N3), for people at (**a**) nodes 5 (N5) and (**b**) 6 (N6), and (**c**) is a table that shows the overall condition.

Finally, Figures 16–18 report a second example of the application of the Algorithm 3, which was derived according to that shown in Figure 8b. In this case, nodes 1, 3, and 5 detected an alarm.

Consequently, the exits 1 and 2 (herein called E1 and E2) were closed, and the target of the Algorithm 3 became the Assembly Points 1 and 2 (that are also indicated as nodes 9 and 10, or N9 and N10, in the figure below).

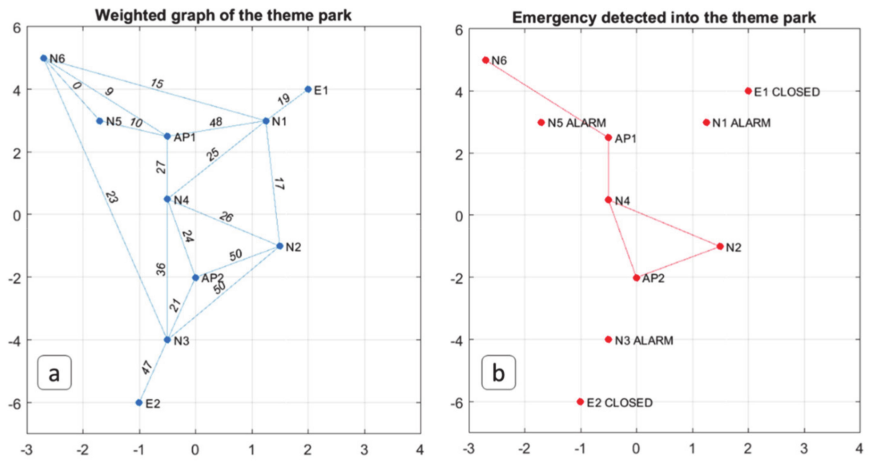


Figure 16. Example of the results of the Algorithm 3 in case of an environmental/structural alarm at nodes 1, 3, and 5 (N1, N3, and N5), where (**a**) is the graph before the alarm triggering and (**b**) is the graph after the alarm triggering.

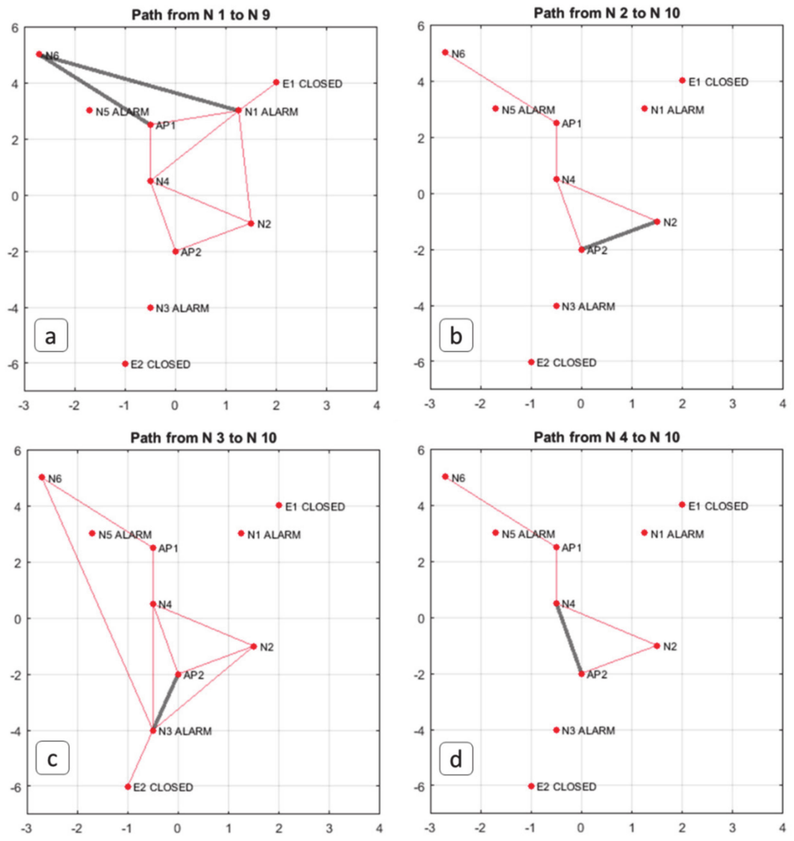
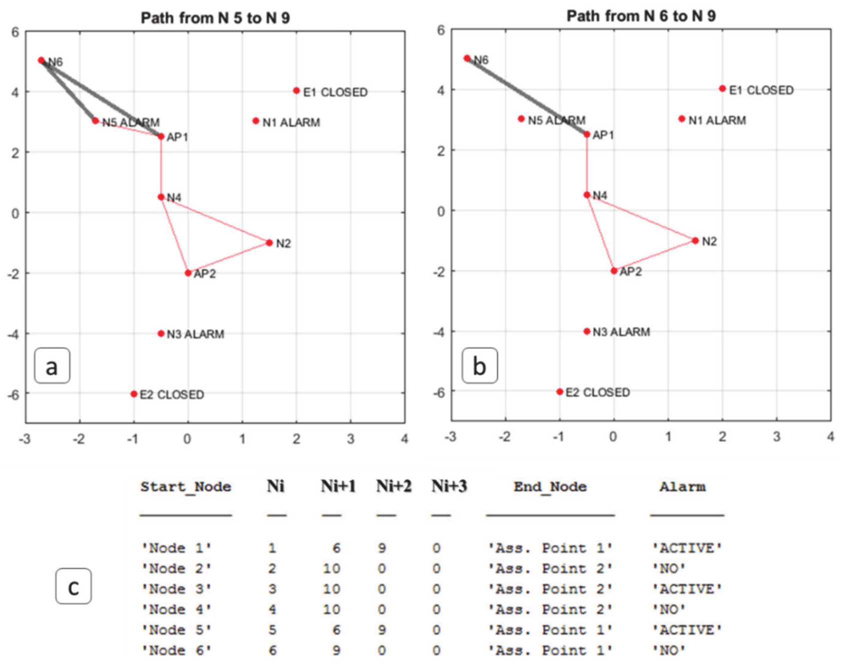


Figure 17. Example of the safest and fastest paths provided by the Algorithm 3 in case of an environmental/structural alarm at nodes 1, 3, and 5 (N1, N3, and N5), which can be used by the people at (**a**) node 1 (N1), (**b**) node 2 (N2), (**c**) node 3 (N3), and (**d**) node 4 (N4) to reach the Assembly Points 1 and 2 (AP1 and AP2, which are also nodes 9, N9, and 10, N10).



Start_Node	Ni	Ni+1	Ni+2	Ni+3	End_Node	Alarm
'Node 1'	1	6	9	0	'Ass. Point 1'	'ACTIVE'
'Node 2'	2	10	0	0	'Ass. Point 2'	'NO'
'Node 3'	3	10	0	0	'Ass. Point 2'	'ACTIVE'
'Node 4'	4	10	0	0	'Ass. Point 2'	'NO'
'Node 5'	5	6	9	0	'Ass. Point 1'	'ACTIVE'
'Node 6'	6	9	0	0	'Ass. Point 1'	'NO'

Figure 18. Example of the safest and fastest paths provided by the Algorithm 3 in case of an environmental/structural alarm at nodes 1, 3, and 5 (N1, N3, and N5), for the people at (**a**) nodes 5 (N5) and (**b**) 6 (N6), and (**c**) is a table that shows the overall condition.

Note that Table 6 and Equation 2 report the results of a scalability analysis carried out considering a system consisting in 6, 60, 240, 480, 960 and 1200 nodes, two exits and two assembly points. Alarms were simulated considering that, in each case, 20% of the nodes (randomly selected and indicated in the following with "A") triggered an emergency alarm. The results refer to one of the worst conditions, i.e., the definition of the safest and shortest path to reach the exit 2 from node 1 (cf. Figure 2a) considering that: (i) node 1 is adjacent to the exit 1; (ii) the exit 1 is closed; and (iii) the people around node 1 have to cross the whole park while avoiding all the emergencies (i.e., the nodes "A") to reach the exit 2. In each of the six abovementioned cases, the system's response times, using a desktop PC equipped with Intel® Core™ i5-7400 CPU @ 3.00GHz, 8.00 GB of RAM, were calculated by averaging the results of 50 simulations for each case. The results of the scalability analysis described above are reported in Table 6.

Table 6. Results of the scalability analysis.

Condition	# of Node That Make Up the System	# Nodes in Alarm (i.e., 20% of All the Nodes)	Average Time of Response of the Algorithm 3 (s)
1	6	1	0.5513
2	60	12	0.9253
3	240	48	1.7147
4	480	96	2.9778
5	960	192	5.9606
6	1200	240	7.3738

The results in Table 6 were used to define the law that better describes how the response time of the algorithm 3 decays if the number of nodes increases (Equation (2) was derived).

$$t = 0.1706 \cdot N^{0.487}, \tag{2}$$

where t is the response time of the algorithm 3 (seconds) as a function of the number of sensing nodes N.

Finally, all the algorithms have been developed and tested in a desktop PC and in a Raspberry PI, showing a computational time compatible with the level of risk associated with the safety application (<3 sec) in the case of actual nodes. The possibility to run a Raspberry PI on a Wireless Local Area Network (WLAN) is also powerful in the case of a network communication interruption, as the algorithms can be implemented in low computational and standalone devices.

Future works include (1) the implementation on low computational devices (e.g., microcontrollers), (2) the comparison with other simple neural network classifiers, such as a K-Means algorithm or a tree-based method (e.g., random forest or decision tree), or Convolutional Neural Networks and (3) the usage of a "flow graph" created by means of a min-cut max flow algorithm in order to maximize the number of people per unit of time who can move out of the park (e.g., using the number of people per unit of time who can use the path as capacities, and the actual number of people who are expected to use it at any given time based on a decision for evacuation as a flow value. In this way, crowds can be redirected from all the nodes simultaneously, and the flow is maximized).

5. Conclusions

In this study, an IoT system consisting of a system and algorithms that allow environmental and structural monitoring and emergency management is presented. Using powerful Machine Learning and Shortest Path Finding algorithms, fed with simulated datasets, useful information is gathered from sensors (environmental and structure-related), cameras, and using NFC technology (number and location of visitors) in order to obtain an alarm detection, safe path suggestion, and social distancing alarm for users and platform managers. Raw and elapsed data are sent to a platform dashboard for the additional online monitoring of environmental and structural conditions. The implementation is envisioned on low computational devices such as microcontrollers, which are also powerful in the case of a network communication interruption because they do not require a remote data exchange.

Author Contributions: Conceptualization, M.M. and R.F.; methodology, M.M.; software, R.F.; validation, M.M. and R.F.; investigation, R.F.; writing—original draft preparation, R.F.; writing—review and editing, M.M.; supervision, M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bibri, S.E. The IoT for smart sustainable cities of the future: An analytical framework for sensor-based big data applications for environmental sustainability. *Sustain. Cities Soc.* **2018**, *38*, 230–253. [CrossRef]
2. Fan, C.; Zhang, C.; Yahja, A.; Mostafavi, A. Disaster City Digital Twin: A vision for integrating artificial and human intelligence for disaster management. *Int. J. Inf. Manag.* **2019**, (in press). [CrossRef]
3. Sakurai, M.; Murayama, Y. Information technologies and disaster management – Benefits and issues. *Prog. Disaster Sci.* **2019**, *2*, 100012. [CrossRef]
4. Bacco, M.; Delmastro, F.; Ferro, E.; Gotta, A. Environmental Monitoring for Smart Cities. *IEEE Sens. J.* **2017**, *17*, 7767–7774. [CrossRef]
5. Government of South Australia COVID-19. Available online: https://www.health.gov.au/news/health-alerts/novel-coronavirus-2019-ncov-health-alert/how-to-protect-yourself-and-others-from-coronavirus-covid-19/limits-on-public-gatherings-for-coronavirus-covid-19 (accessed on 14 April 2020).
6. Caley, P.; Philp, D.J.; McCracken, K. Quantifying social distancing arising from pandemic influenza. *J. R. Soc. Interface* **2008**, *5*, 631–639. [CrossRef]
7. Stein, R.A. COVID-19 and rationally layered social distancing. *Int. J. Clin. Pract.* **2020**, *74*, e13501. [CrossRef]
8. Merenda, M.; Porcaro, C.; Iero, D. Edge machine learning for ai-enabled iot devices: A review. *Sensors* **2020**, *20*, 2533. [CrossRef]
9. Nakas, C.; Kandris, D.; Visvardis, G. Energy efficient routing in wireless sensor networks: A comprehensive survey. *Algorithms* **2020**, *13*, 72. [CrossRef]

10. Jin, Z.; Jian-Ping, Y.; Si-Wang, Z.; Ya-Ping, L.; Guang, L. A survey on position-based routing algorithms in wireless sensor networks. *Algorithms* **2009**, *2*, 158–182. [CrossRef]
11. Hedar, A.R.; Abdulaziz, S.N.; Sewisy, A.A.; El-Sayed, G.A. Adaptive scatter search to solve the minimum connected dominating set problem for efficient management of wireless networks. *Algorithms* **2020**, *13*, 35. [CrossRef]
12. Meghanathan, N. A, Benchmarking algorithm to determine minimum aggregation delay for data gathering trees and an analysis of the diameter-aggregation delay tradeoff. *Algorithms* **2015**, *8*, 435–458. [CrossRef]
13. El Khamlichi, Y.; Tahiri, A.; Abtoy, A.; Medina-Bulo, I.; Palomo-Lozano, F.A. Hybrid algorithm for optimal wireless sensor network deployment with the minimum number of sensor nodes. *Algorithms* **2017**, *10*, 80. [CrossRef]
14. Erd, M.; Schaeffer, F.; Kostic, M.; Reindl, L.M. Event monitoring in emergency scenarios using energy efficient wireless sensor nodes for the disaster information management. *Int. J. Disaster Risk Reduct.* **2016**, *16*, 33–42. [CrossRef]
15. Rahman, M.U.; Rahman, S.; Mansoor, S.; Deep, V.; Aashkaar, M. Implementation of ICT and Wireless Sensor Networks for Earthquake Alert and Disaster Management in Earthquake Prone Areas. Proceedings of International Conference on Computational Modeling and Security, Bangalore, India, 11–13 February 2016; Volume 85, pp. 92–99.
16. Erdelj, M.; Król, M.; Natalizio, E. Wireless Sensor Networks and Multi-UAV systems for natural disaster management. *Comput. Netw.* **2017**, *124*, 72–86. [CrossRef]
17. Jain, B.; Brar, G.; Malhotra, J.; Rani, S. A novel approach for smart cities in convergence to wireless sensor networks. *Sustain. Cities Soc.* **2017**, *35*, 440–448. [CrossRef]
18. Alam, S.; De, D. Bio-inspired smog sensing model for wireless sensor networks based on intracellular signalling. *Inf. Fusion* **2019**, *49*, 100–119. [CrossRef]
19. Uma Priyadarsini, P.S.; Sriramya, P. Disaster management using evidence-based interactive trust management system for wireless sensor networks by Internet of Things. *Comput. Electr. Eng.* **2019**, *75*, 164–174. [CrossRef]
20. Deak, G.; Curran, K.; Condell, J.; Asimakopoulou, E.; Bessis, N. IoTs (Internet of Things) and DfPL (Device-free Passive Localisation) in a disaster management scenario. *Simul. Model. Pract. Theory* **2013**, *35*, 86–96. [CrossRef]
21. Cui, F. Deployment and integration of smart sensors with IoT devices detecting fire disasters in huge forest environment. *Comput. Commun.* **2020**, *150*, 818–827. [CrossRef]
22. Pillai, A.S.; Chandraprasad, G.S.; Khwaja, A.S.; Anpalagan, A. A service oriented IoT architecture for disaster preparedness and forecasting system. *IoT* **2019**, 100076. (in press). [CrossRef]
23. Tao, Z. Advanced Wavelet Sampling Algorithm for IoT based environmental monitoring and management. *Comput. Commun.* **2020**, *150*, 547–555. [CrossRef]
24. Rathore, M.M.; Paul, A.; Hong, W.H.; Seo, H.C.; Awan, I.; Saeed, S. Exploiting IoT and big data analytics: Defining Smart Digital City using real-time urban data. *Sustain. Cities Soc.* **2018**, *40*, 600–610. [CrossRef]
25. Chen, W.; He, B.; Zhang, L.; Nover, D. Developing an integrated 2D and 3D WebGIS-based platform for effective landslide hazard management. *Int. J. Disaster Risk Reduct.* **2016**, *20*, 26–38. [CrossRef]
26. Aloi, G.; Briante, O.; Di Felice, M.; Ruggeri, G.; Savazzi, S. The SENSE-ME platform: Infrastructure-less smartphone connectivity and decentralized sensing for emergency management. *Pervasive Mob. Comput.* **2017**, *42*, 187–208. [CrossRef]
27. Qiu, L.; Du, Z.; Zhu, Q.; Fan, Y. An integrated flood management system based on linking environmental models and disaster-related data. *Environ. Model. Softw.* **2017**, *91*, 111–126. [CrossRef]
28. Damalas, A.; Mettas, C.; Evagorou, E.; Giannecchini, S.; Iasio, C.; Papadopoulos, M.; Konstantinou, A.; Hadjimitsis, D. Development and Implementation of a DECATASTROPHIZE platform and tool for the management of disasters or multiple hazards. *Int. J. Disaster Risk Reduct.* **2018**, *31*, 589–601. [CrossRef]
29. Pérez-González, C.J.; Colebrook, M.; Roda-García, J.L.; Rosa-Remedios, C.B. Developing a data analytics platform to support decision making in emergency and security management. *Expert Syst. Appl.* **2019**, *120*, 167–184. [CrossRef]
30. Ma, G.; Wu, Z. BIM-based building fire emergency management: Combining building users' behavior decisions. *Autom. Constr.* **2020**, *109*, 102975. [CrossRef]
31. Franke, T.; Lukowicz, P.; Blanke, U. Smart crowds in smart cities: Real life, city scale deployments of a smartphone based participatory crowd management platform. *J. Internet Serv. Appl.* **2015**, *6*, 1–19. [CrossRef]

32. Palmieri, F.; Ficco, M.; Pardi, S.; Castiglione, A. A cloud-based architecture for emergency management and first responders localization in smart city environments. *Comput. Electr. Eng.* **2016**, *56*, 810–830. [CrossRef]
33. Alazawi, Z.; Alani, O.; Abdljabar, M.B.; Altowaijri, S.; Mehmood, R. A smart disaster management system for future cities. In Proceedings of the 2014 ACM International Workshop on Wireless and Mobile Technologies for Smart Cities, Co-Located with MobiHoc, Philadelphia, PA, USA, 11–14 August 2014; pp. 1–10.
34. Astarita, V.; Festa, D.C.; Giofrè, V.P.; Guido, G.; Stefano, G. Mobile for emergencies M4EM: A cooperative software tool for emergency management operations. *Procedia Comput. Sci.* **2018**, *134*, 433–438. [CrossRef]
35. He, Y.; Zhang, D.; Fang, Y. Development of a mobile post-disaster management system using free and open source technologies. *Int. J. Disaster Risk Reduct.* **2017**, *25*, 101–110. [CrossRef]
36. Kaku, K. Satellite remote sensing for disaster management support: A holistic and staged approach based on case studies in Sentinel Asia. *Int. J. Disaster Risk Reduct.* **2019**, *33*, 417–432. [CrossRef]
37. Hiltz, S.R.; Hughes, A.L.; Imran, M.; Plotnick, L.; Power, R.; Turoff, M. Exploring the usefulness and feasibility of software requirements for social media use in emergency management. *Int. J. Disaster Risk Reduct.* **2020**, *42*, 101367. [CrossRef]
38. Loureiro, A.L.D.; Miguéis, V.L.; da Silva, L.F.M. Exploring the use of deep neural networks for sales forecasting in fashion retail. *Decis. Support Syst.* **2018**, *114*, 81–93. [CrossRef]
39. Kor, A.L.; Yanovsky, M.; Pattinson, C.; Kharchenko, V. SMART-ITEM: IoT-enabled smart living. In Proceedings of the FTC 2016—Proceedings of Future Technologies Conference, San Francisco, CA, USA, 6–7 December 2016; pp. 739–749.
40. Zhang, S.; Yao, L.; Sun, A.; Tay, Y. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.* **2019**, *52*, 5. [CrossRef]
41. Kunaver, M.; Požrl, T. Diversity in recommender systems—A survey. *Knowl. Based. Syst.* **2017**, *123*, 154–162. [CrossRef]
42. Di Martino, S.; Rossi, S. An Architecture for a Mobility Recommender System in Smart Cities. *Procedia Comput. Sci.* **2016**, *58*, 425–430. [CrossRef]
43. Lujak, M.; Billhardt, H.; Dunkel, J.; Fernández, A.; Hermoso, R.; Ossowski, S. A distributed architecture for real-time evacuation guidance in large smart buildings. *Comput Sci. Inf. Syst.* **2017**, *14*, 257–282. [CrossRef]
44. Saleem, Y.; Crespi, N.; Rehmani, M.H.; Copeland, R.; Hussein, D.; Bertin, E. Exploitation of social IoT for recommendation services. In Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things, WF-IoT, Reston, VA, USA, 12–14 December 2016; pp. 359–364.
45. Vijayalakshmir, D.; Kalaivani, R. Minimum Cost Spanning Tree using Matrix Algorithm. *Int. J. Sci. Res. Publ.* **2014**, *4*, 1–5.
46. Goyal, A.; Mogha, P.; Luthra, R.; Sangwan, N. Path finding: A* or Dijkstra's? . *Int. J. IT Eng.* **2013**, *2*, 1–15.
47. Khuller, S.; Raghavachari, B. Graph and Network Algorithms. Available online: https://it.mathworks.com/help/matlab/graph-and-network-algorithms.html?s_tid=CRUX_lftnav (accessed on 20 September 2020).
48. NFC Forum NFC Technology. Available online: https://nfc-forum.org/what-is-nfc/ (accessed on 20 September 2020).
49. Freire, S. Modeling of spatiotemporal distribution of urban population at high resolution—Value for risk assessment and emergency management. In *Lecture Notes in Geoinformation and Cartography*; Springer: Berlin, Germany, 2010; pp. 53–67.
50. Chan, A.B.; Liang, Z.S.J.; Vasconcelos, N. Privacy preserving crowd monitoring: Counting people without people models or tracking. In Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Anchorage, AK, USA, 23–28 June 2008.
51. Bacanin, N.; Bezdan, T.; Tuba, E.; Strumberger, I.; Tuba, M. Optimizing convolutional neural network hyperparameters by enhanced swarm intelligence metaheuristics. *Algorithms* **2020**, *13*, 67. [CrossRef]
52. Qu, H.; Wang, M.; Zhang, C.; Wei, Y. A study on faster R-CNN-based subway pedestrian detection with ACE enhancement. *Algorithms* **2018**, *11*, 192. [CrossRef]
53. Solano, F.; Di Fazio, S.; Modica, G. A methodology based on GEOBIA and WorldView-3 imagery to derive vegetation indices at tree crown detail in olive orchards. *Int. J. Appl. Earth Obs. Geoinf.* **2019**, *83*, 101912. [CrossRef]
54. Lanucara, S.; Praticò, S.; Modica, G. Harmonization and interoperable sharing of multi-temporal geospatial data of rural landscapes. *Smart Innov. Syst. Technol.* **2019**, *100*, 51–59.

55. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 386–397. [CrossRef]
56. Zhang, S.H.; Li, R.; Dong, X.; Rosin, P.; Cai, Z.; Han, X.; Yang, D.; Huang, H.; Hu, S.M. Pose2Seg: Detection free human instance segmentation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; Volume 2019, pp. 889–898.
57. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 10778–10787.
58. Bengio, Y.; Goodfellow, I.J.; Courville, A. *Deep Learning: Chapter 9—Convolutional Networks*; MIT Press: Cambridge, MA, USA, 2016.
59. Lecun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *7553*, 436–444. [CrossRef]
60. Fedele, R.; Merenda, M.; Praticò, F.G.; Carotenuto, R.; Della Corte, F.G. Energy harvesting for IoT road monitoring systems. *Instrum. Mes. Metrol.* **2018**, *17*, 605–623. [CrossRef]
61. Merenda, M.; Praticò, F.G.; Fedele, R.; Carotenuto, R.; Corte, F.G. Della A real-time decision platform for the management of structures and infrastructures. *Electronics* **2019**, *8*, 1180. [CrossRef]
62. Merenda, M.; Iero, D.; Carotenuto, R.; Corte, F.G.D. Simple and low-cost photovoltaic module emulator. *Electronics* **2019**, *8*, 1445. [CrossRef]
63. STMicroelectronics IoT board (Model: Kit Discovery B-L475E-IOT01A). Available online: https://www.mouser.it/Search/Refine?Ntk=P_MarCom&Ntt=160178092 (accessed on 20 September 2020).
64. Fedele, R.; Praticò, F.G. Monitoring infrastructure asset through its acoustic signature. In Proceedings of the INTER-NOISE 2019 MADRID, Spain—48th International Congress and Exhibition on Noise Control Engineering, Madrid, Spain, 16–19 June 2019.
65. Fedele, R.; Praticò, F.G.; Carotenuto, R.; Corte, F.G.D. Structural health monitoring of pavement assets through acoustic signature. In Proceedings of the 10th International Conference on the Bearing Capacity of Roads, Railways and Airfields, BCRRA, Athens, Greece, 28–30 June 2017; pp. 869–875.
66. Praticò, F.G.; Fedele, R.; Naumov, V.; Sauer, T. Detection and monitoring of bottom-up cracks in road pavement using a machine-learning approach. *Algorithms* **2020**, *13*, 81. [CrossRef]
67. Praticò, F.G.; Della Corte, F.G.; Merenda, M. Self-powered sensors for road pavements. In Proceedings of the 4th Chinese-European Workshop on Functional Pavement Design, CEW, Delft, The Netherlands, 29 June–1 July 2016; pp. 1365–1374.
68. ThingsBoard. ThingsBoard IoT Open Source Plataform. Available online: https://thingsboard.io/ (accessed on 20 September 2020).
69. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
70. Lukežič, A.; Vojíř, T.; Čehovin Zajc, L.; Matas, J.; Kristan, M. Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *Int. J. Comput. Vis.* **2018**, *126*, 671–688. [CrossRef]