ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **23**, Number S, pp. S53–S66, 2020

An autonomous UAV system for video monitoring of the quarantine zones

Dan-Marius DOBREA[1], Monica-Claudia DOBREA[2]
[1] Electronics, Telecommunications and Information Technology, "Gheorghe Asachi" Technical University, Iași, România
E-mail: mdobrea@etti.tuiasi.ro
[2] Electronics, Telecommunications and Information Technology, "Gheorghe Asachi" Technical University, Iași, România
E-mail: mcdobrea@etti.tuiasi.ro

Abstract. In this paper, a comparative study between two classification approaches was done, namely, between a Support Vector Machine (SVM) neural network - as a classical machine learning approach -, and four different deep learning classification systems, considered today to be the state-of-the-art systems in computer vision. Two embedded companion computers, a Raspberry Pi and a Jetson Nano, placed on a HoverGames quadcopter support the classification systems that were developed and deployed to identify humans. The ultimate goal of this research consists in developing a quadcopter system endowed with the capabilities of following a pre-programmed flight route and simultaneously detecting humans as well as of warning the system operator to reinforce the quarantine zones. The obtained results demonstrated the superior performances provided by the deep learning approach: more than six times faster than the classical approach, and with a correct classification performance higher than 90% on a direct stream of video data.

Keywords: UAV, HoverGames, PX4, QGroundControl, deep learning, HOG, SVM

1. Introduction

At the end of 2019, a new disease based on the SARS-CoV-2 coronavirus (COVID-19) was identified in China. The disease spread all around the world, and currently, the number of people infected by the virus is increasing rapidly.

Up to now, no COVID-19 vaccine has been successfully developed, and more, no effective drugs exist to treat the disease. The most common complications in patients with COVID-19 infection are acute respiratory distress syndrome, anemia, acute heart injuries, and different secondary infections [1]. Therefore, treatments and patients support are usually provided such as antiviral therapy and therapies based on antibiotics, systemic corticosteroids, neuraminidase inhibitors, RNA synthesis inhibitors as well as mechanical ventilation for patients with intractable hypoxemia [2], to name only a few. Nevertheless, all of these drug treatments are empirical, and the efficacy of them still needs to be verified by clinical trials [2].

Due to the absence of effective treatments, the best way to deal with the COVID-19 epidemic is to control the sources of infection [2] through early diagnoses, epidemiological investigation, isolation, social distancing, quarantine, by banning the gathering groups, improving personal hygiene, wearing medical masks and by keeping rooms well ventilated.

In Romania, the government took several preventive measures in order to control the sources and the spread of infection. One of the first measures, taken on February 21, 2020, was to impose 14-day quarantine for all persons that were returning to Romania from the abroad affected regions where quarantine was already in place (e.g., northern Italy). On March 29, the Suceava town, along with eight adjacent communes, were placed under total quarantine due to the COVID-19 outbreak. The Suceava city was the first Romanian city to be placed under the complete lockdown. The effective surveillance of quarantine areas and the national lockdown proved to be and continue to be challenging tasks for both police and gendarmerie, even if they are receiving technical and personnel military support.

In this paper, we propose a solution to sustain and enforce the quarantine zones. The solution is based on an autonomous unmanned aerial vehicle (UAV) able to both detect humans and timely send warnings to a control center. Nowadays, the UAVs are used on a large number of applications [3] like military, scientific, environmental monitoring, product deliveries, infrastructure inspections, entertainment, etc. In the frame of the COVID-19 pandemic, the UAV technology could make an important contribution by exploiting its huge potential in applications that are dealing with delivering medicine and collecting samples [4], spraying crowded urban areas with disinfectant, temperature checking, information delivery or surveillance and monitoring. In all these, the key advantage of using the drones resides in their ability to limit face-to-face contact, which, in turn, will prevent the contamination with the COVID-19 virus.

2. The main concept of the system

The main goal of the quarantine is to separate the people that may have been exposed to a disease in order to monitor their health state evolution over the incubation period of the pathogen. The idea of this research is to start using technology to enforce the quarantine and to protect the public by preventing its exposure to people who were identified as carriers, either symptomatic or asymptomatic, of a contagious disease like COVID-19.

For this purpose, an autonomous drone (UAV system) will carry out pre-programmed flight missions by following a planned path around the quarantine zone, and, at the end of the mission, it will return to the landing point. The drone is designed to be equipped with two video systems. One of these is connected to an embedded system, and its role is to detect and to localize humans in an automatic manner. When a human is localized, a warning is sent, through a radio module, to the base station, see **Figure 1**. All the image processing steps are done on the onboard computer, and no images are sent to the ground station or stored on the onboard computer. Based on this approach, there are no issues regarding the General Data Protection Regulation (GDBR) or any infringement of these rules, either intentionally or by negligence.

The second video system is dedicated both for the remote control of the UAV as well for the validating of the automatic human recognition process. The first function of the UAV system is managed through an RC (remote control) unit, and it is activated whenever the human operator from the ground base station considers as being necessary.

3. The unmanned aircraft system

3.1. The hardware components

The unmanned aircraft system (UAS) used in this research, see **Figure 1**, is composed of a HoverGames UAV, a ground base controller – composed, in its turn, of several different elements –, and the communication system, mainly sustained by MAVLINK protocol.

The UAV system was built based on the HoverGames drone kit, a professional development kit produced by NXP Semiconductors N.V. company. This kit contains all the required components needed to build a quadcopter (see **Figure 2**): the flight management unit (RDDRONE-FMUK66), four BLDC brushless motors, ESCs motor controllers, propellers, an RC unit, a carbon fiber frame, a GPS module, the power management module, the power distribution board, etc.

The human detection function was implemented by using two companion computers: a Raspberry Pi system, that supported the classical classification approach, and a Jetson Nano system, that supported the deep learning classification algorithms – see **Figure 1**, and **Figure 2**, respectively. The communication between the flight management unit (FMU) and each of the above-mentioned companion systems was done by using the I2C protocol, see **Figure 1**.

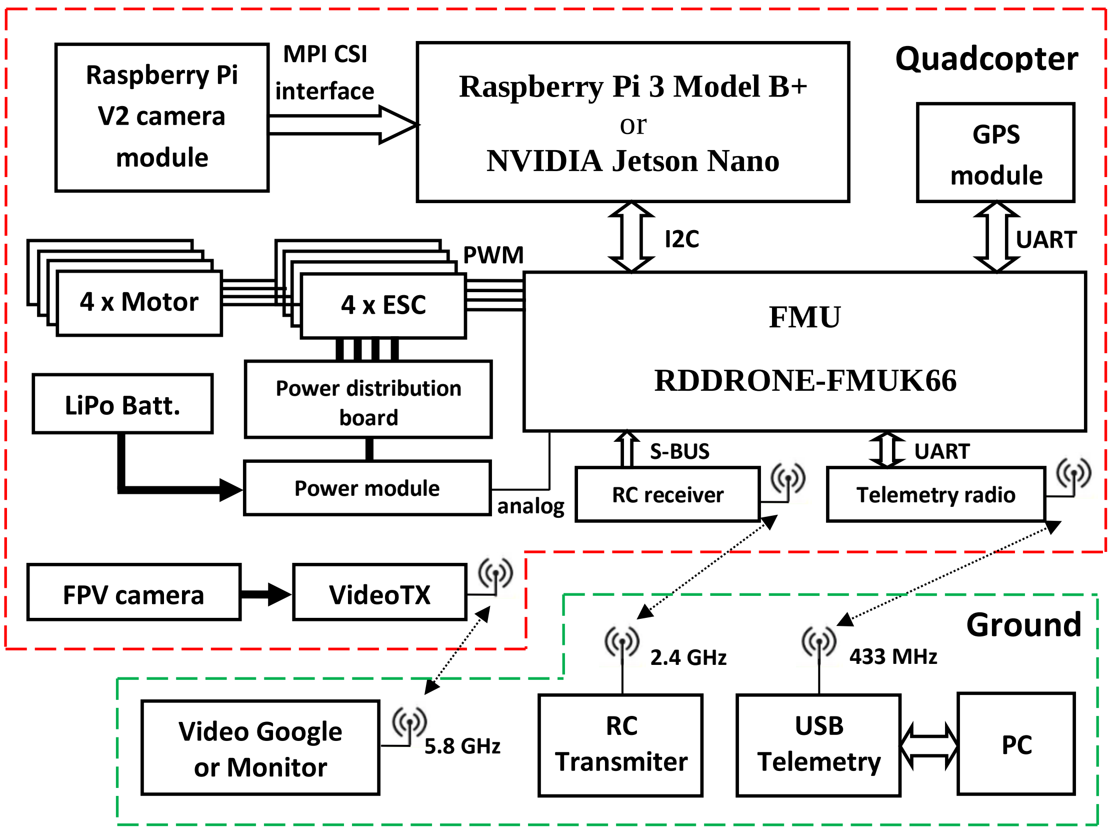


Figure 1. A schematic block diagram of the UAS

The communication between the HoverGames drone and the ground control station was designed and implemented using three different communication links. A telemetry radio transceiver, working on 433 MHz, assures the main link with the ground station and it allows to access the following functions: to configure the flight path, to supervise and change settings and parameters of the FMU without having a USB connection; but, most of all, it allows checking the status of the HoverGames quadcopter while it is in the air.

The second communication link, on 2.4 GHz, is a radio controller (RC) type link through which a ground control operator can take any time control of the drone, switching from the autonomous mode to the manual mode. The FlySky FS-i6S RC transmitter used for this is a highly configurable radio controller, supporting up to 10 control channels at the same time. Both links presented above use frequency hopping techniques to avoid collisions: the AFHDS 2A (Automatic Frequency Hopping Digital System) for the RC transmitter and the FHSS (Frequency Hopping Spread Spectrum) for the telemetry radio unit.

The third control link consists of the First-Person View (FPV) system that is a method through which, based on information provided by the onboard camera, a UAV can be controlled as from the pilot viewpoint, **Figure 2**. In this research, the FPV system was used: (1) to pilot the HoverGames drone in manual mode, as well as (2) to endorse and check the automatic human recognition system.

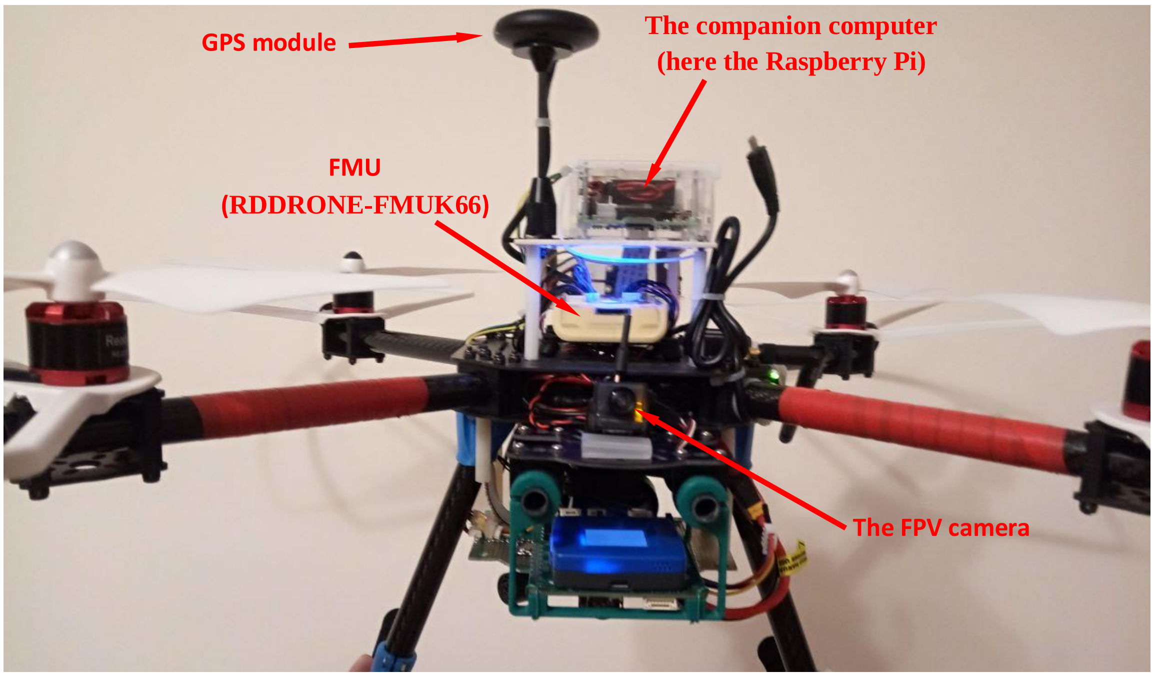


Figure 2. The UAV system build on a HoverGames drone

3.2. The software components

The flight management unit supports the open-source PX4 flight stack. Additionally, the Linux Foundation supports a large number of open-source collaborative projects. One of these projects is Dronecode, the umbrella for both the PX4 autopilot – which is also the "brain" of the HoverGame drone –, and the MAVLINK inter-vehicle communication protocol, which was mainly designed for communications between ground-stations, autopilots, and companion computers.

The PX4 autopilot consists of two main layers. The first layer is the flight stack. This one is composed of several estimation, guidance, and flight control modules, as well as of all the components required to sustain these core blocks – *e.g.*, sensor interface, radio command input, or the motor and servo control functions. The second layer, which is also the middleware, is a general robotics layer that can support any type of code integration envisaged to sustain internal/external communications and any additional hardware integration. At this software level, we developed and integrated a software component designed to take the information from the human detection system and to send it to the ground station.

For each of the companion computers, we developed several software components dedicated to acquiring images, searching for humans in these images, and sending the obtained results to the FMU. On the FMU, the middleware developed component receives the human detection module results and send them further to the base station; to implement this, we made use of the MAVLINK communication protocol.

4. Autonomous flight

The HoverGames drone may operate in an autonomous way or under the remote control of an operator. The PX4 autopilot, the onboard FMU, and the QGroundControl control station are the main components that provide the HoverGames drone with the autonomous flying capacity. More precisely, the host PX4 autopilot allows autonomous drones to take off, carry out predetermined missions, and land entirely autonomously without any human intervention.

The QGroundControl control station can manage multiple drones simultaneously, and it has two main components functionally involved in an autonomous flight: a mission planner and a mission tracker. The mission path planning for an autonomous flight is done mainly through waypoints insertion, see **Figure 3**.

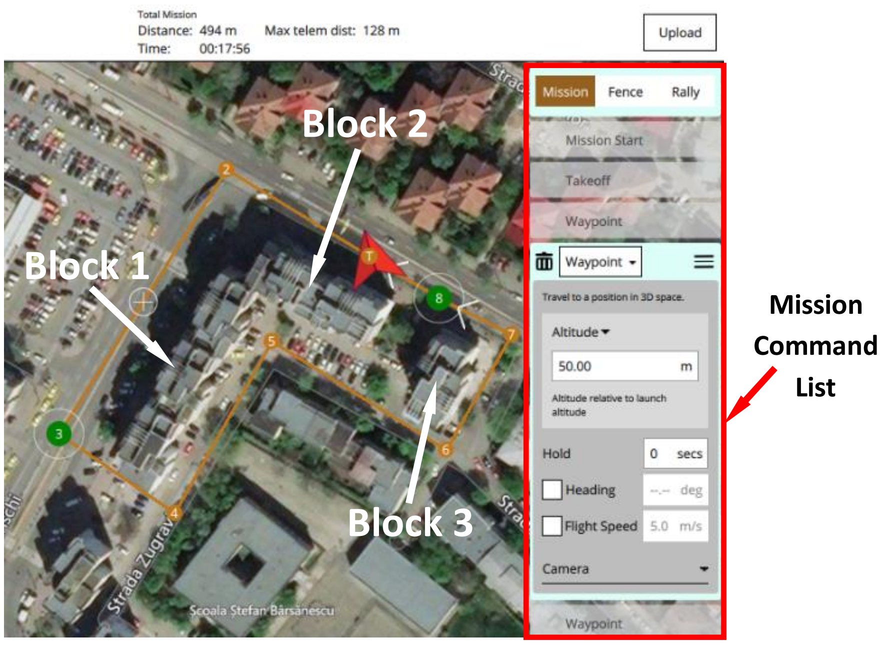


Figure 3. A screenshot showing a mission plan

The screenshot in **Figure 3** shows a mission plan example that follows a polyline in order to survey the perimeters of three structures, namely, three blocks of flats. The mission plan starts with a takeoff that corresponds to the red arrow position, also referred to as the ***Planned Home*** point; then, it continues with flying through seven waypoints, provided in a prescribed order, and finally, it ends with the landing at the ***Return To Launch*** point. In the right, the *Mission Command List* is provided. By selecting a specific waypoint, several of the flight parameters for this particular point become available for editing; these parameters are the altitude, the flight speed (*e.g.*, one can set a particular speed for this waypoint, other than the default mission speed), the option for a specific camera action that should be taken when the drone will fly over the set waypoint (*e.g.*, controlling the gimbal, selecting the photo or video mode), etc.

Once the mission has been planned and uploaded to the HoverGames drone, one can switch from the mission planning mode to the mission tracking mode in order to achieve the mission. The missions tracker, i.e. the Fly View, is used to both command and monitor the drone. The mission control (*e.g.*, the start, continue, pause, and resume commands) and the drone guidance (*e.g.*, the arm/disarm/emergency stop, takeoff/land, change altitude commands) are carried out in real-time by employing the telemetry link (433 MHz).

5. The human detection systems

All the human bodies have the same basic structure – four limbs (two arms and two legs), a head, a neck, and a torso –, regardless of the gender, race, age or ethnicity involved. Beyond all these similarities registered at the structural level, there is also a large variability in between humans given by the physical appearance, which is in direct relation with the body shape, the age, the health, the skin color, the skeletal shape, etc. of each individual. From the similarities and dissimilarities encountered in between humans, the first are the ones exploited within our classification systems. The classifiers primarily extract the features able to best describe and differentiate the human bodies from other statical or dynamical objects and things identified within an image. So, based on these features, a machine learning model can be trained to learn the human body variabilities in order to detect humans in video streams.

In the field of computer vision, a large number of methods are addressing today the challenging task of human detection in video streams. One of the simplest methods applied is the differential motion analysis between frames [5]. Other methods rely on identifying humans based on their detected body parts [6]. The Haar cascade classifiers, based on Viola-Jones detectors, were the first object detection framework able to provide real-time detection [7]. The Viola-Jones detectors supported for many years the advances in the object detection research field. Nevertheless, in recent times, much more powerful methods have been developed, like the histogram of oriented gradients [8], [9], the deformable parts models [10], or the deep learning approach [11], [12].

The human detection task achievable on a UAV system is a very challenging one due to: (1) the computationally expensive algorithms and the power constraints of the quadcopter, (2) the variable imaging conditions, (3) the image distortions introduced by both the actual movement of the quadcopter and its motor vibrations, (4) the variable size of the targets to be identified, etc. This study proposes an evaluation of several algorithms for human detection on video streams captured from UAVs. Two main constraints of the analyzed systems are addressed, the computation time and the detection performance, and two different classification approaches are evaluated in a comparative manner. For *the first approach* – also referred to as the classical one –, a Support Vector Machine (SVM) classification system and the Histogram of Oriented Gradients (HOG) features descriptors were taken into account for analysis. Our choice for this method is justified by both the flexibility of the method that is extensively used in many research fields and, more than that, by its superior classification results [8], [9], [13], [14]. The deep learning classification systems, nowadays considered the state-of-the-art systems in computer vision, represent *the second approach* explored in our study.

5.1. Classification algorithms

In this research, the main goal resides in classifying the input frames in only two classes: **(a)** images without human(s) and (b) frames in which one, two or more humans are detected.

In **the first used methodology**, the features of the human(s) in images were extracted using the Histogram of Oriented Gradients (HOG) and classified by a support vector machine (SVM) network.

The HOG [15] is particularly suited for human detection in images, and it is considered one of the best human descriptors. In this feature extraction method, the input image is divided into small connected regions, of *n x n* pixels, called “cells” and a sliding detector window of fixed size (*l x h* cells) is defined by the algorithm in order to scan the image. On each cell, a histogram of gradient orientations is computed. For best results, in order to reduce the influence of the effects of image lightning changes, these gradients are computed over the normalized color and

Gamma image. The gradient value computed for each pixel of each cell is then used in the cell orientation histogram for voting. Further, in order to obtain the HOG descriptors, to provide better invariance against illumination, shadows, and contrast of the edges, and to decrease the required computation, another processing step, called Local Normalization, is executed, in which normalization is applied to k x k “blocks” of histograms instead on single histograms [15]. In order to detect humans, the sliding detection window scans the entire image; more, it is applied to the input image and to the different scaled variants of it. In the end, the HOG descriptors extracted for all these windows are provided further to the classification system.

The SVM was firstly introduced in 1995 [16], and nowadays, it is considered to be among the best performers in the class of classical machine learning. The main idea of this type of classifier is to endow a neural network with a high generalization capacity by mapping inputs non-linearly to high-dimensional feature spaces. In this new feature space, the linear decision surfaces are constructed and, then, used in the final decision process.

In the **second applied methodology**, in order to classify humans from video streams, we implemented a transfer learning approach with AlexNet, SqueezeNet, ResNet-18, and ResNet-34 deep neural network, respectively.

The AlexNet is the name of a convolutional neural network (CNN) composed of eight layers. The first five layers are convolutional layers, and the last three layers are classically fully connected layers. An Overlapping Max Pooling layer follows after each of the first two convolutional layers as well as after the fifth convolutional layer. Only the third, the fourth, and the fifth convolutional layers are connected directly [17]. Using the convolution, a convolutional layer combines the input data from the previous layer with a convolution kernel (i.e., a filter) in order to extract the features that will, then, feed the next layer. Each convolutional layer uses many kernels of the same size. For example, the first convolutional layer contains 96 kernels (with the size of 11x11x3); the second convolutional layer of the AlexNet network contains 256 kernels, but for this layer, the convolution window is only 5x5. The following three convolutional layers are based on kernels with the 3x3 size. In the AlexNet network, the ReLU nonlinearity is used for neurons in all convolutional and in all fully connected layers. The Overlapping Max Pooling layers are usually used both as a downsample discretization process and to extract the most important features.

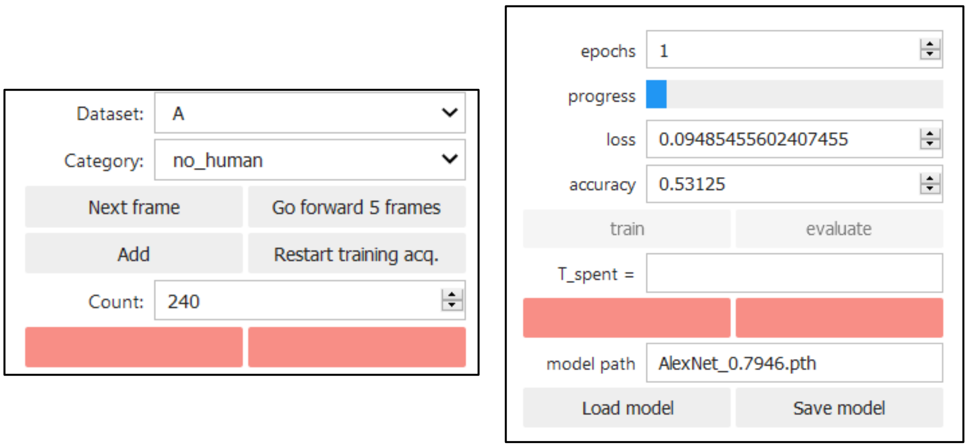
When the SqueezeNet was created, its primary goal was to provide a new neural network smaller than AlexNet and with the same level of accuracy when applied to the ImageNet Large Scale Visual Recognition Challenge database [18]. The SqueezeNet deep neural network has an entirely different architecture than AlexNet. The SqueezeNet has ten layers, out of which only the first one and the last one are convolutional layers; the remaining eight intermediate layers are special layers, called Fire Modules. A Fire module is comprised of two components: (1) a squeeze convolution layer, which contains only 1x1 filters, and whose output feeds (2) an expand layer that has a mix of 1x1 and 3x3 convolution filters. Two of the main concepts of the SqueezeNet are: (1) to replace as many as 3x3 filters with 1x1 filters, and (2) to decrease the number of input channels using the squeezing layers. By applying all these concepts, the SqueezeNet succeeded in obtaining a reduction, by a factor of fifty, in the model size compared to AlexNet.

The residual neural network (ResNet) uses a different concept; namely, it implements a skip connection type that allows for jumping over two or even three layers, modeling in this mode the pyramidal cells from the cerebral cortex. All layers of this network are convolutional layers, consisting mostly of 3x3 filters, with ReLU nonlinearities. The ResNet architectures have a large number of implementations, starting from 18-layers deep up to 152-layers deep. The key concept of this new type of deep network relies on the observation that increasing the depth of a neural network should increase the accuracy of the network, as long as we take care of the over-fitting process. In this research, we took advantage of the two smallest implementations of the ResNet – ResNet-18 and ResNet-34 –, and this is because they provide a good trade-off between accuracy and efficiency.

The PyTorch framework used in this research includes pre-trained models for all the deep neural networks presented above. These models were trained on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) database. ILSVRC has more than 14 million images in the dataset, grouped in more than 21 thousand classes. All the deep neural network models from PyTorch were trained with 1000 classes. The trained neural networks embed layers able to find outlines, lines, curves, and other image features. All these image features can be re-usable for our classification task, namely, the human detection task, by replacing and retraining the last neural layer. The replaced layer has 4096 inputs for AlexNet and 512 inputs for SqueezeNet, ResNet-18, and ResNet-34, and two outputs, defined by our particular problem.

5.2. Sample database

To train the last layer of the deep neural networks, a new training database was created, consisting of 600 images, out of which half are corresponding to the first class (*i.e.*, the class of images that include humans), and the other half of them are corresponding to the second class (*i.e.*, the class of images without humans). These images were extracted from the movies recorded during quadcopter flight, and to do this, a specific interface was designed and developed by us, in Python, **Figure 4(a)**. The selected frames include one or more humans (*e.g.*, men, women, and children), differently dressed, of different ages, being in different positions, and placed in an urban environment as varied as possible.


Dataset: A
Category: no_human
Count: 240

epochs 1
loss 0.09485455602407455
accuracy 0.53125
T_spent =
model path AlexNet_0.7946.pth

Figure 4. The user interfaces for: (a) database creation, and (b) training of the deep neural networks

The training data set was employed only for the deep neural networks, **Figure 4(b)**. The OpenCV embeds in its libraries a pre-trained HOG and a Linear SVM model, both used by us to perform pedestrian detection in video streams.

A second data set, i.e., the test set, was used to analyze the performances of the trained classification models. For this, six short movies (20 fps) were used, and from these, only three included humans. The above-discussed program was developed to have the ability to both extract the frames and process them in an online manner. The images from the test set were different from those in the training set, with quite balanced classes, i.e., with 654 images for no humans class and 656 images for humans class.

The HOG extractor and the Linear SVM model worked with images respecting the standard VGA resolution of 640 columns by 480 lines; the images were acquired by the Raspberry Pi V2 camera module. For all deep neural networks, the input images were resized at 244 x 244 on all the R, G, B channels and were normalized in the [0, 1] range, with the mean set to [0.485, 0.456, 0.406] and the standard deviation to [0.229, 0.224, 0.225].

6. Results

For each of the classification systems, five classification performance metrics, the accuracy, the true positive rate (TPR), true negative rate (TNR), positive predictive value (PPV), and negative predictive value (NPV), were computed.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} \quad (1)$$

$$TPR = \frac{TP}{TP+FN} \quad (2)$$

$$TNR = \frac{TN}{TN+FP} \quad (3)$$

$$PPV = \frac{TP}{TP+FP} \quad (4)$$

$$NPV = \frac{TN}{TN+FN} \quad (5)$$

where, TP = True Positive, FP = False Positive, FN = False Negative, and TN = True Negative. The classification *accuracy* is simply the rate of correct classifications, while the *sensitivity* index (also called the true positive rate or recall) measures, in our particular case, the probability of detection, the percentage of humans that are correctly identified as actually existing in the analyzed image. Mainly because we want to detect, with high precision, humans in a given quarantine zone, it effectively becomes mandatory for our system to achieve high values for the two above-mentioned performance indices, i.e., accuracy and sensitivity. The true negative rate, also called *specificity*, measures the percentage of frames that are correctly identified as not including any human. However, an excellent classification system requires to have the positive predicted value and the negative predictive value as higher as possible. The positive predicted value, also known as *precision*, is the probability that images classified by the system as alert-triggering images truly contain at least one human. The negative predictive value is the probability that images classified by the system as non-alert-triggering images truly not containing any human.

6.1. HOG+SVM results

For this first approach, that includes HOG descriptors and SVM classifier, a C program for human detection was developed. The program used the HOG feature extractor and the Linear SVM model classifier, provided by OpenCV, in order to perform human detection in video streams. The program runs on a Raspberry Pi 3 Model B+ companion computer that has a CPU unit with four cores, all clocked at 1.4 GHz.

The goal of the classification system is to detect human(s) and to send, in such a case, a warning to the ground station. The warning should be sent whenever the system detects one, two, or more human subjects. The obtained classification performances, extracted from the cross-validation matrix computed on the training set, are presented in **Table 1**. In computing the results, several rules were followed. Thus, each time a human was detected not only as one presence but as multiple ones, like in the case presented in **Figure 5(a)**, only a single TP sample was counted. More, even if in the image there were several humans and the system detected only one of them, or only part of them, like in **Figure 5(b)** and **(c)** cases, also, only a single TP sample was counted. **Figure 5(d)** presents another particular case. In this one, there was indeed a human in the image, but the classification system "detected" not the human but an object from the environment; in such a situation, the classifier output was treated in the cross-validation matrix as an error (*i.e.*, an FN sample).

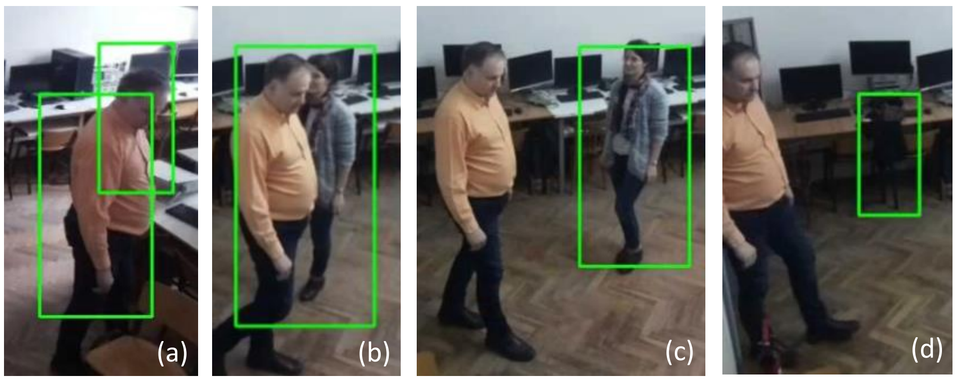


Figure 5. Several situations of human(s) detection

Table 1. The confusion matrix for HOG and SVM classification system

HOG & SVM Performance		**Actual class**	
		Human(s)	No human(s)
Predicted class	Human(s)	601	239
	No human(s)	53	417

For the HOG and SVM approach, the correct detection rate was only 77.71%, and the sensitivity, 91.89%. Nevertheless, the main drawback of this method was by far its low time efficiency. Thus, the feature extraction and the classification processes took a substantial amount of time, somewhere between 2.4 and 2.9 seconds.

Table 2. The statistical performance measures for the HOG and SVM classification system

	Accuracy	**Sensitivity**	**Specificity**	**PPV**	**NPV**
HOG and SVM	77.1%	91.89%	63.57%	71.55%	88.72%

Starting from the confusion matrix, presented in **Table 1**, different statistical performance measures for the HOG and SVM classification system were computed and provided in **Table 2**. A perfect human recognition system is described as 100% sensitive (*i.e.*, all video frames containing humans are correctly recognized) and 100% specific (*i.e.*, the elements of the urban landscape are not recognized as people; such an opposite situation is presented in **Figure 5(d)**). The results of Table 2 reveal that the HOG-SVM classification system has an excellent sensitivity but a quit low specificity. More precisely, the HoverGames drone system will miss roughly 8 frames out of 100 frames containing humans, but at the same time, it will also generate almost 37 false alarms out of 100 situations that do not require triggering the alert.

To improve the execution time, a multi-thread approach was further employed, namely, with three different recognition threads running on three distinct cores of the Raspberry Pi system, **Figure 6**.

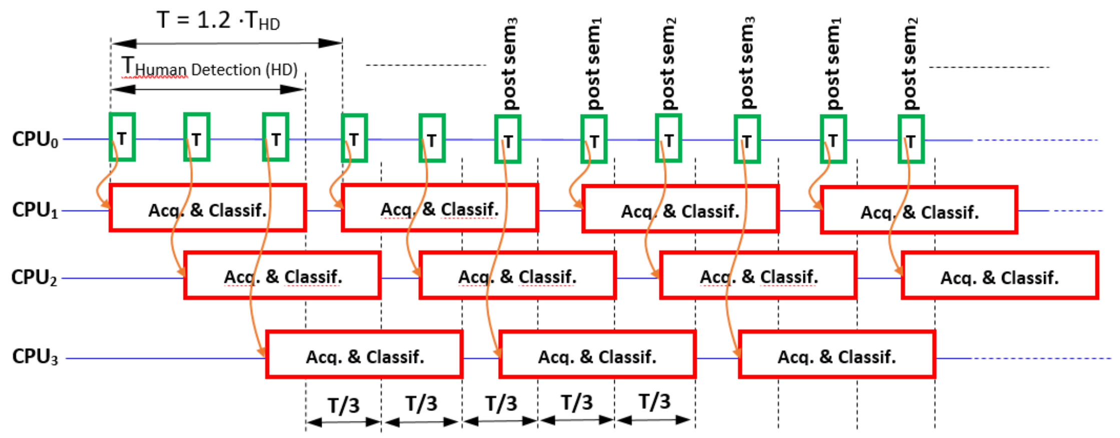
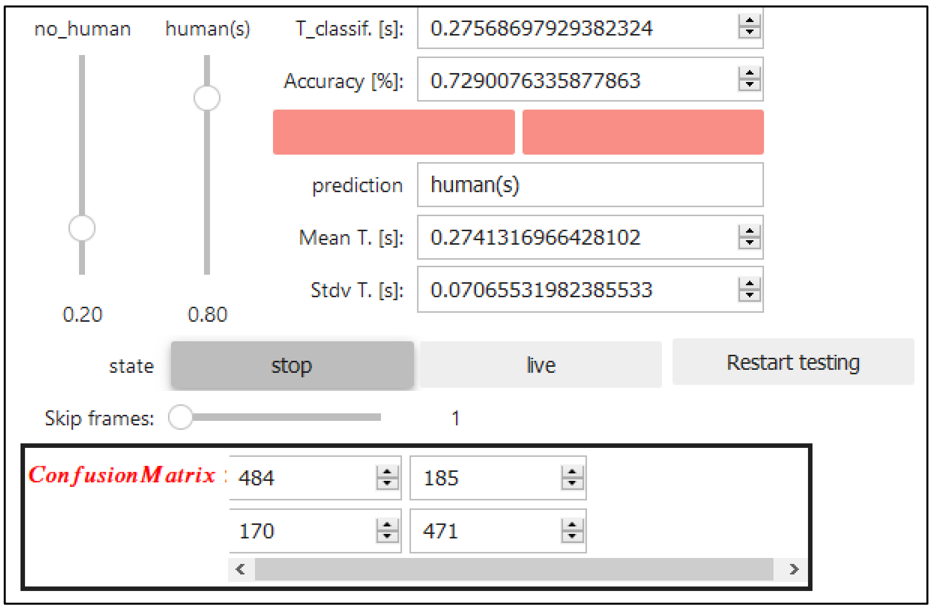


Figure 6. The multi-thread approach used to improve the classification execution time

In **Figure 6**, T_{HD} represents the entire time interval needed to detect human(s) – to acquire an image, extract the features, and run the class0ification system – in the worst case. The main idea, implemented in our application, was to start on three different cores of the Raspberry Pi system, and by using a timer, three different detection processes, delayed each with a time interval of T/3, see **Figure 6**. To have a safety margin, T was set at a value with 20% higher than the worst time required to detect humans, T_{DH}. Based on this approach, at each T/3 seconds, the system could provide the output for the human detection process. This means that, for a real situation, in which a human detection previously took between 2.4 and 2.9 seconds, now, after using the threads as above, improved speed of around one frame/second could be achieved for the entire processing flow - image acquisition, feature extraction, and classification.


no_human 0.20
human(s) 0.80
T_classif. [s]: 0.27568697929382324
Accuracy [%]: 0.7290076335877863
prediction human(s)
Mean T. [s]: 0.2741316966428102
Stdv T. [s]: 0.07065531982385533
state stop live Restart testing
Skip frames: 1
ConfusionMatrix :

484	185
170	471

Figure 7. The user interface used to test the deep neural network – an example of the performance obtained by using the AlexNet network

6.2. Deep learning results

In the development of the deep neural networks, we used the Python language, sustained by the PyTorch backend. Reported in the literature there are other frames too, like the classical TensorFlow, in which amazing applications were developed [19]; however, our choice for using PyTorch was motivated mainly by the better development and the debugging experience provided to the developer. The human detection application was implemented in the Jupyter Notebook computational environment and executed on the Jetson Nano companion computer. The Jetson Nano has a quad-core 64-bit ARM CPU and includes a 128-core NVIDIA Maxwell GPU. It can deliver 472 GFLOPS of computational power and can run a full training deep neural networks; also, it can re-train the deep networks based on the transfer learning paradigm.

In **Figure 7** the testing user interface is presented. In addition to presenting the classification information, i.e. accuracy and confusion matrix, the user interface provides: (a) in real-time and in direct accordance with the presented image (not shown in **Figure 7**) – the frame classification results, and (b) ultimately, when the testing stage was finished – the average classification time as well as the standard deviation.

The results obtained on the test set, by running the four deep neural network models presented-above, are reported in **Table 3**. These performances were achieved for the best classifier architectures obtained after at least twenty different training sessions. The last neural layer was trained with the Adam algorithm [20]. The Adam (adaptive moment estimation) is a gradient-based optimization method used in the deep learning models to update network weights, thus replacing the classical stochastic gradient descent algorithm. In Adam algorithm, each network weight has its learning rate, which is adapted accordingly with the first and second moments of the gradients. Adam outperforms other similar algorithms [20], and it is currently recommended as the default learning algorithm for the deep learning structures [21], [22].

Table 3. The obtained deep neural networks performances

	Accuracy	**Sensitivity**	**Specificity**	**PPV**	**NPV**	**Classification time**	
						Mean [s]	**Std. Dev.**
AlexNet	72.9%	74%	71.79%	72.34%	73.47%	0.274	0.07055
SqueezeNet	62.34%	93.42%	31.29%	57.58%	82.66%	0.263	0.02976
ResNet-18	90.76%	98.77%	82.77%	85.11%	98.54%	0.449	0.01425
ResNet-34	93.36%	100%	86.73%	88.25%	100%	0.732	0.03284

By comparing the data from **Table 3**, one can observe that the SqueezeNet CNN is the fastest neuronal structure from the ones analyzed, with a processing speed of 3.8 frames/second. However, even if the SqueezeNet is almost three-times faster than ResNet-34, its classification performances for the human detection problem are far from the best ones; also, the specificity is very low. On the other hand, the ResNet-18 model seems to be the best choice for our problem. With a high classification accuracy (90.76%), with a very high sensitivity (98.77%), and with a detection time greater than 2 frames/second, ResNet-18 is the natural choice for the human detection system. In the autonomous mode, the HoverGames default mission speed is 5 m/s. If the ground operator sets the speed to be slower, e.g., 1-2 m/s, the ResNet-34 can also be used. The ResNet-34 network models in the best way the human detection problem, and it obtains a classification accuracy of 93.36%, and a sensitivity and NVP of 100%. For this case, the classification time (frame acquisition time + time required to preprocess the frame + frame classification interval) is around 732 ms or, equivalently, 1.36 frames/second. Also, the ResNet-18 and ResNet-34 have the highest positive predictive values (*i.e.*, above 85%), and negative predictive values from all the classification systems analyzed in this paper, with the negative predictive values being 100% or almost 100%. In particular, with a prevalence of about 50% (this metrics shows how often human-containing image category occurs in all analyzed images), the predictive values reported for the ResNet-18 system point out that, out of 2000 analyzed images, about 851 true positives and 985 true negatives are likely, with 149 false positives and 15 false negatives.

In a study analyzing a quadcopter system used for searching and rescuing operations, the same approach employed by us in this research paper, i.e., HOG feature extraction and SVM classifier, led to performances of 78% for sensitivity and 83% for specificity [23]. The performances were similar to the ones reported by us in the classical machine learning approach. However, there is a significant difference between these two studies, materialized in both the way the research was conducted and the way the results were obtained. Thus, in our research, we used an onboard computer (i.e., a Raspberry Pi system). In [23], on the other side, the authors performed an offline analysis done on a personal computer. In another study [24], a real-time human detection system based on a UAV is presented. In this research, the video frames are acquired from a DJI Matrice 100 UAV and sent to a ground station where all the recognition process is done. The reported performances for this system indicate a precision value of 88% and a recall (sensitivity) value of 92%. A comparison between our system performances and the results obtained in [24] reveals that the sensitivity of the HoverGames human detection system proposed in this study outperforms the corresponding performance measure reported in [24], for both of the ResNet neural networks, Table 3. The precision computed for the ResNet-18 is a little bit lower (85.11% versus 88%), but the ResNet-34 network has a comparable precision – 88.25% versus 88%, see Table 3. Nevertheless, unlike the system presented in [24], in this research paper, all the processing steps were done onboard of the HoverGames UAV system.

7. Conclusions

In this paper, we proposed a highly accurate real-time quadcopter human detection system. The HoverGames quadcopter is able to fly autonomously on a predetermined path and, also, to warn in real-time the ground station if a human was detected in its path. In this mode, the autonomous UAV system becomes a very useful and efficient tool in enforcing the COVID-19 quarantine zones.

To make a comparison and to decide, in a knowledgeable way, which is the best approach for the human detection task in video streams, we selected for analysis a Support Vector Machine (SVM) classification system and four deep neural network models, i.e., the AlexNet, the SqueezeNet, the ResNet-18, and the ResNet-34, respectively.

According to the experimental results, the HOG and SVM model produced a 77.71% classification accuracy rate and a 91.89% sensitivity, while the ResNet-18 produced a 90.76% classification accuracy rate and a 98.77% sensitivity. These results reveal that ResNet-18 is the most accurate and robust classifier, and, in consequence, it is very well suited to be applicable in real-world applications.

The present research is a first, but a major step towards achieving a robust system for monitoring quarantined areas. In future research, an important objective will be the analysis performed on an extensive database of images and video streams, as well as the application - on the results obtained - of well-known statistical tests in order to provide a calibrated and robust indication on the ability of the selected model to correctly detect humans.

Acknowledgements. This study was partly funded by a donation received from the NXP Semiconductors N.V Company. Here, I would like to express my gratitude to the NXP company.

References

[1] Huang C., Wang Y., Li X., Ren L., Zhao J., Hu Y., Zhang L., Fan G., Xu J., Gu X., Cheng Z., Yu T., Xia J., Wei Y., Wu W., Xie X., Yin W., Li H., Liu M., Xiao Y., Gao H., Guo L., Xie J., Wang G., Jiang R., Gao Z., Jin Q., Wang J., Cao B., *Clinical features of patients infected with 2019 novel coronavirus in Wuhan,* The Lancet **395**, 2020, pp. 497- 506.

[2] Sun P., Lu X., X., Sun W., Pan B., *Understanding of COVID-19 based on current evidence*, Medical Virology **92**, Issue 6, 2020, pp. 548-551.

[3] Vacca A., Onishi H., *Drones: military weapons, surveillance or mapping tools for environmental monitoring? The need for legal framework is required - World Conference on Transport Research, Shanghai,* Transportation Research Procedia **25**, Elsevier, 2016, pp 51–62.

[4] Lakhani A, *Which Melbourne metropolitan areas are vulnerable to COVID-19 based on age, disability and access to health services? Using spatial analysis to identify service gapsand inform delivery,* Journal of Pain and Symptom Management, Elsevier, 2020.

[5] Lee D., Zhan P., Thomas A., Schoenberger R., *Shape-based Human Detection for Threat Assessment - International Symposium on Defense and Security, Visual Information Processing XIII*, 2004, Orlando, Florida, USA, vol. 5438, pp. 81-91.

[6] Plagemann C., Ganapathi V., Koller D., Thrun S., *Real-time identification and localization of body parts from depth images - IEEE International Conference on Robotics and Automation*, Anchorage, USA, 2010, pp. 3108-3113.

[7] Viola P., Jones M., *Rapid object detection using a boosted cascade of simple features - IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001, Kauai, USA*, 2001, pp. I-I.

[8] Wei Y., Tian Q., Guo J., Huang W., Cao J., *Multi-vehicle detection algorithm through combining Harr and HOG features - Mathematics and Computers in Simulation* **155**, 2019, pp. 130-145.

[9] Savitha G., Jidesh P., *A fully-automated system for identification and classification of subsolid nodules in lung computed tomographic scans - Biomedical Signal Processing and Control* **53**, 2019, pp. 1-14.

[10] Li T., Pang Y., Pan J., Liu C., Weighted Deformable Part Model for Robust Human Detection – International Conference on Intelligent Computing, 2014, Taiyuan, China, 2014, Lecture Notes in Computer Science **8588**, Springer, 2014, pp. 764-775.

[11] Mateus A., Ribeiro D., Miraldo P., Nascimento J.C., Efficient and robust Pedestrian Detection using Deep Learning for Human-Aware Navigation - Robotics and Autonomous Systems **113**, 2019, pp 23-37.

[12] Zhao Z., Zheng P., Xu S., Wu X., Object Detection With Deep Learning: A Review - *IEEE Transactions on Neural Networks and Learning Systems* **30**, no. 11, 2019, pp. 3212-3232.

[13] Kong X., Meng Z., Nojiri N., Iwahori Y., Meng L., Tomiyama H., *A HOG-SVM Based Fall Detection IoT System for Elderly Persons Using Deep Sensor - Procedia Computer Science* **147**, 2019, pp. 276-282.

[14] Žemgulys J., Raudonis V, Maskeliūnas R., Damaševičius R, *Recognition of basketball referee signals from videos using Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM) - Procedia Computer Science* **130**, 2018, pp. 953-960.

[15] Dalal N., Triggs B., Histograms of oriented gradients for human detection - *IEEE Conference on Computer Vision and Pattern Recognition* **1**, San Diego, USA, 2005, pp. 886-893.

[16] Cortes C., Vapnik V., *Support-vector networks – Machine Learning* **20**, no. 3, pp. 273-297.

[17] Krizhevsky A., Sutskever I, Hinton G.E, *ImageNet classification with deep convolutional neural networks - Communications of the ACM* **60**, no. 6, 2012, pp. 84-90.

[18] Iandola F.N., Moskewicz M.W., Ashraf K., Han S., Dally W.J., Keutzer K., *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size – Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1-13.

[19] Franți E., Dascălu M., Ispas I., Tebeanu A.V., E.Z., Branea S., Dragomir. V., *Decoding communication: a deep learning approach to voice-based intention*

detection - Romanian Journal of Information Science and Technology **21**, no. 4, 2018,
pp. 460-474.

[20] Kingma D.P., Ba J.L., *ADAM: a method for stochastic optimization - 3rd International Conference on Learning Representations*, San Diego, USA, 2015, pp. 1-15

[21] Wilson A.C., Roelofs R., Stern M., Srebro N., Recht B., *The Marginal Value of Adaptive Gradient* Methods in Machine Learning, in: Guyon I., Luxburg U.V., Bengio S., Wallach H. Fergus R., Vishwanathan S., Garnett. R.: *Advances in Neural Information Processing Systems* **30** - *International Conference on Neural Information Processing Systems, Long Beach, USA*, 2017, pp. 1-14.

[22] Cole M.R., *Deep Learning with C#, .Net and Kelp.Net: The Ultimate Kelp.Net Deep Learning Guide*, BPB Publications, 2019

[23] Martins F.N., de Groot M., Stokkel X., Wiering M.A., *Human Detection and Classification of Landing Sites for Search and Rescue Drones – European Symposium on Artificial Neural Networks, Computational Intelligence, and Machine Learning*, 2016, pp. 1-6.

[24] Bhattarai N., Nakamura T., Mozumder C., *Real Time Human Detection and Localization Using Consumer Grade Camera and Commercial UAV – Preprints*, 2018, pp. 1-18.