Undefined 1 (2009) 1–5
IOS Press

Deep learning-based video surveillance system managed by low cost hardware and panoramic cameras

Jesus Benito-Picazo [a], Enrique Domínguez [a], Esteban J. Palomo [a] and Ezequiel López-Rubio [a],
[a] *Department of Computer Languages and Computer Science. University of Málaga.*
Bulevar Louis Pasteur, 35. 29071 Málaga, Spain.
E-mail: {jpicazo,enriqued,ejpalomo,ezeqlr}@lcc.uma.es
Biomedic Research Institute of Málaga (IBIMA)
C/ Doctor Miguel Díaz Recio, 28. 29010 Málaga, Spain

Abstract. The design of automated video surveillance systems often involves the detection of agents which exhibit anomalous or dangerous behavior in the scene under analysis. Models aimed to enhance the video pattern recognition abilities of the system are commonly integrated in order to increase its performance. Deep learning neural networks are found among the most popular models employed for this purpose. Nevertheless, the large computational demands of deep networks mean that exhaustive scans of the full video frame make the system perform rather poorly in terms of execution speed when implemented on low cost devices, due to the excessive computational load generated by the examination of multiple image windows. This work presents a video surveillance system aimed to detect moving objects with abnormal behavior for a panoramic 360°surveillance camera. The block of the video frame to be analyzed is determined on the basis of a probabilistic mixture distribution comprised by two mixture components. The first component is a uniform distribution, which is in charge of a blind window selection, while the second component is a mixture of kernel distributions. The kernel distributions generate windows within the video frame in the vicinity of the areas where anomalies were previously found. This contributes to obtain candidate windows for analysis which are close to the most relevant regions of the video frame, according to the past recorded activity. A Raspberry Pi microcontroller based board is employed to implement the system. This enables the design and implementation of a system with a low cost, which is nevertheless capable of performing the video analysis with a high video frame processing rate.

Keywords: Foreground detection and feed forward neural network and panoramic camera and convolutional neural network.

1. Introduction

Increasing public awareness about security issues is caused by the abundance of social conflicts appearing in the media. Research on video surveillance systems has attracted more interest as a consequence of this. Therefore, more reliable and accurate systems are sought. The source of the data for these video surveillance systems is often obtained from static and pan-tilt-zoom (PTZ) cameras. For example, In [1], a novel salient motion detection method for non-stationary footage supplied by PTZ cameras is developed. [2] presents a new background subtraction algorithm designed for PTZ cameras capable of performing this task without the need for explicit image registration, and [3] illustrates a novel method for detecting abnormal behavior in crowded video scenes. The successful operation of such systems depends on their capability to attain real time execution, such as in [4], where a faster patch-based version of Speed-Up Robust Features detector (SURF), named BLS, is introduced as a saliency detection method, or the work illustrated in [5], where a tracking-by-detection system that works under important computational power constraints is presented.

The employ of PTZ cameras is commonplace in computer vision systems. A good example is the work presented in [6], where a low-power, omnidirectional tracking system (LOTS) is described. Another good instance is the dynamic calibration of PTZ cameras for traffic monitoring that can be found in [7]. Other designs feature systems intended to be deployed in PTZ camera networks. In [8], several cooperative localization and tracking methods to be deployed in PTZ camera networks are presented. [9] presents an integrated analysis and control framework for a PTZ camera network using dynamic camera-to-target assignment and efficient feature acquisition to achieve a better scene understanding. There are also some proposals such as the one presented in [10] where optimization strategies, along with a distributed implementation, are proposed in order to decide to focus the attention of a PTZ camera network components on individuals or groups of them, aiming to properly understand scenes displaying people interaction. Some other research involving PTZ cameras focus on the background-foreground segmentation applied to images supplied by PTZ cameras [11]. One example can be found in [12], where a neural-based background subtraction approach to moving object detection using self-organized models, is presented. The work described in [13] goes along the same lines, as it describes a method for compensating the panning and tilting movements of a PTZ camera in order to perform the background segmentation of the scene.

There is no established and comprehensive theoretical framework that establishes the foundations to develop practical video surveillance systems. Moreover, conventional and PTZ surveillance cameras have significant limitations in their coverage due to their restricted field of view. Taking this into account, we have focused our attention on panoramic (360°) video cameras in order to detect abnormally behaving objects in a scene. This way, the broadest possible area is available for the video surveillance procedure. Even though the use of 360° images implies a larger frame size that can slow down the processing times, panoramic video cameras have been satisfactorily employed in computer vision systems in recent years [14]. Some of them present combinations of hardware devices to generate omnidirectional systems such as the one presented in [15], where a novel multi-camera integrated video-sensor, based on an omnidirectional imaging device in conjunction with a PTZ camera, is proposed to design surveillance applications, or the work developed in [16], where a novel video surveillance system, based in the combination of omnidirectional and network controlled cameras, is developed. Other works such as [17], go more deeply into the theoretical part of the omnidirectional surveillance topic by describing a formulation and application of parametric egomotion compensation for omnidirectional vision sensors (ODVS) that allow avoiding false alarms due to irrelevant features.

Deep neural networks have been successfully applied to many different research areas [18]. One of the most important is medicine. For example, [19] proposes an ensemble deep-learning architecture for nonlinearly mapping scalp to intracranial electroencephalography (iEEG) data, intended to circumvent the unavailability of iEEG and the limitations of scalp electroencephalography (sEEG). In [20] and [21], stacked autoencoders are used to compute functional brain connections in order to detect proficiency. Some deep convolutional network-based proposals are oriented to detect and diagnose several types of medical conditions: In [22], deep convolutional networks are used to detect and diagnose seizure in neonatal children, and [23] presents a method, using convolutional neural networks, for electroencephalography (EEG) signal analysis aimed to detect normal, preictal, and seizure classes. Finally, there are some other deep convolutional network-based applications to medicine such as the one described in [24], which presents a deep learning-based approach where deep convolutional networks are utilized to identify Parkinson disease in 3D nuclear imaging data.

Engineering is another relevant research field that can improve from deep neural network applications. More precisely, in civil engineering, we have works such as [25], where a novel method for concrete properties estimation, based on mixture proportions, is developed by using a deep restricted Boltzmann machine. In [26], a novel model for detecting damage in high-rise building structures, based in a restricted Boltzmann machine and a neural dynamics classification algorithm (NDC), is presented. In the same line, [27] describes a new methodology for assessing the local and global condition of structures, featuring synchrosqueezed wavelet transform, Fast Fourier Transform, and deep Boltzmann machines to extract features from the signals provided by the sensors. Finally, the work presented in [28] illustrates a construction cost estimation model using advanced deep learning concepts such as the combination of a deep Boltzmann machine approach along with a softmax layer and some regression models.

Of course, there are many other areas that benefit from deep neural network-based machine learning techniques. In the field of big data science, we can find works such as [29], where it can be found a deep learning based method focused on dealing with big data time series. In the field of applied economics, the research presented in [30] shows a model for estimating the price of new housing at the design phase by integrating a deep belief restricted Boltzmann machine and genetic algorithms. Deep neural networks are also a potent tool when it comes to computer vision. Thus, in [31] we can find a deep learning-based method for haze removal from a single input image, and in [32] a new algorithm, based in the well-known VGG-16 convolutional neural network, is designed aiming to detect splicing in digital images.

Object recognition and image classification stand as typical applications of deep neural networks. These sorts of activities involve complex computational tasks where many obstacles must be faced. Thus, in [33] we can find a review on different techniques and uses of convolutional neural networks to solve inverse problems in imaging such as denoising, deconvolution, superresolution, and medical image reconstruction. Also, [34] illustrates the problem supposed by the data classification in the presence of noise and possible strategies to tackle such a problem by utilizing convolutional neural networks. These techniques are especially useful in the field of automated video surveillance systems covering a wide variety of applications. Two important ones are the verification of civil engineering structure condition and human behavior monitoring. In the first category, we can find works like the one presented by [35], where a recurrent deep neural network is proposed for fully automated crack detection on 3D asphalt pavement surfaces. The research developed in [36] offers an image-based approach for reinforced concrete bridge inspection using convolutional neural networks. Also, inside the civil engineering structure condition monitoring area, we can find the work presented in [37], where the authors propose a pixel-level detection method for identifying road cracks using a deep convolutional encoder-decoder network. In the same field of study is the work presented in [38], where the authors offer a distress classification method for road structures featuring a new network architecture named "convolutional sparse coding deep random network" or (CSDRN).

When it comes to deep neural networks-based human behavior monitoring systems, some relevant studies have been developed. The work performed by the authors of [39], proposes a hierarchical statistical method for recognizing the activities of workers in far-field surveillance videos. [40] presents a convolutional neural network based on multi-scale features for thermal infrared face identification. The research presented in [41] proposes a new network model utilizing stacked multicolumn convolutional neural networks (CNNs) for pedestrian counting. In [42], authors show how to improve the process of detection and classification of vehicles in traffic sequences by using ensembles of convolutional neural networks to overcome the limitations caused by the low resolution of the images supplied by surveillance cameras.

Deep learning based surveillance systems have heavy computational demands which are addressed with GPU acceleration. This incurs in large power consumption. Moreover, high performance computation is associated with expensive hardware. A possible solution to these inconveniences is the use of microcontroller boards, which can be deployed in motion detection systems, given their reduced energy requirements and affordable cost. In other words, microcontrollers constitute economic, small, and flexible hardware. However, these surveillance systems must be optimized to be deployed in such microcontrollers, which do not usually feature a high computing power. In this respect, essential works could be highlighted. In [43] the authors propose a low computation moving object detection method combined with a video encoder, and in [44], we can find the design and implementation of a computationally efficient system for detecting moving objects, ready to be deployed on small, lightweight, low-cost and power-efficient hardware. Some of the proposals in this area include the use of a specific hardware platform such as in [45], where a tracking pipeline designed for fixed smart cameras is presented. This system is able to handle occlusions between objects and can be successfully deployed in a Raspberry Pi board equipped with a RaspiCam camera. Other proposals include the implementation of deep backpropagation learning algorithms to be deployed in FPGAs, which are fast and extreme efficient hardware devices, but also more constrained from a programmer's point of view. Such work is illustrated in [46].

Motion and proximity can be estimated by microcontrollers in several ways. Proper examples of this can be found in [47], were a novel class of flexible linear vision sensor dedicated to motion extraction and proximity estimation named "Vision Tape" is described; and in [48], where the authors present a compact and economic system for measuring cloud shadow

motion vectors, constructed using an array of luminance sensors and a high-speed data acquisition system. Energy savings for street lights can be attained, which is relevant for smart cities [49]. Recently, Self-Organizing Maps (SOMs) were applied to build a motion detection procedure that was implemented on an Arduino DUE board [50]. A static, conventional camera was employed for motion detection in that SOM based video surveillance system.

In this paper, a new video surveillance system for the detection of moving anomalous objects is proposed. Its main difference with respect to the other state of the art systems relies on its probabilistic candidate window generation algorithm for potentially anomalous object detection that uses three new different mixture-based probability distributions. Besides, even though it incorporates a deep learning neural networks-based classifying system, it is still capable of being implemented with microcontrollers and 360° panoramic cameras, in order to attain a low energy consumption and a low hardware cost and yet, avoiding the processing time penalty caused by the size of the panoramic images.

The remainder of the paper is organized as follows: the proposed detection methodology is defined in the next section, the architecture of the proposed system is presented in section 3, the experimental results are provided in section 4 complemented by a comparison with other important avant-garde detection method, and section 5 concludes with some remarks and conclusions.

2. Methodology

For the kind of scenarios that have been described in the previous section, an object is understood to be anomalous if it is not associated to the commonly found object classes in the scene. Under these circumstances, an alarm should be triggered in the video surveillance system.

Next, an anomaly detection method is presented which aims to solve the above defined problem. The foundation of this model is a set comprised by the detections which are active. This set is associated to those objects which have been recently spotted by the surveillance device. A detection is defined as a four dimensional vector (π_i, x_1, x_2, x_3) where:

- π_i is the *a priori* probability that the object is observed.
- (x_1, x_2) are the detected object vertical and horizontal coordinates, computed with respect to the panoramic coordinate system of the video frame.
- x_3 is the length of the window which encloses the object, expressed in pixels.

A forgetting rate α is applied to the a priori probability π_i. Whenever a detected object goes out of sight, i.e. it does not appear in the camera field of view, the detection associated to that object becomes inactive.

In order to simplify the presentation of our method, let us note $\mathbf{x} = (x_1, x_2, x_3)$. Equipped with this abbreviated definition, the valid ranges for $\mathbf{x}$ can be expressed as follows:

$$\mathcal{V} = [1, N_{rows}] \times [1, N_{cols}] \times [S_{min}, S_{max}] \subset \mathbb{R}^3 \quad (1)$$

where $N_{rows} \times N_{cols}$ stands for the height and width of the incoming frame expressed in pixels, while it is assumed that the potential sizes of the detections are lower and upper bounded by are S_{min} and S_{max}, respectively.

Next a probabilistic model is proposed to capture the potential locations of the detected objects:

$$p(\mathbf{y}) = qU_{\mathcal{V}}(\mathbf{y}) + (1-q)\frac{1}{M}\sum_{i=1}^{M}\pi_i K(\mathbf{y}, \mathbf{x}_i, \sigma) \quad (2)$$

where $U_{\mathcal{V}}(\mathbf{y})$ stands for the uniform probability distribution on $\mathcal{V}$, $K(\mathbf{y}, \boldsymbol{\mu}, \sigma)$ denotes a multivariate homoscedastic distribution with mean vector $\boldsymbol{\mu}$ and spread parameter σ, M is the count of active detected objects, $q \in (0, 1)$ is a mixing weight (which is tunable) and σ is the spread parameter (which is also tunable).

As it was pointed out in section 1, in this work, three multivariate homoscedastic distributions are considered in order to implement the probabilistic window-based potential detection generator, namely Gaussian, Student-t and triangular, as given in Table 1, where $\|\cdot\|$ stands for the Euclidean norm of a vector, and ν is the degrees of freedom parameter of the Student-t distribution (which is a tunable parameter). Please note that for the Gaussian and Student-t distributions, the spread parameter σ is also the standard deviation of the distribution. These three distributions have been chosen because they are unimodal multivariate distributions, and their probability density functions are relatively easy to evaluate, which speeds up the computation.

$$K_{Gaussian}\left(\mathbf{y},\boldsymbol{\mu},\sigma\right)=\left(2\pi\right)^{-\frac{3}{2}}\sigma^{-3}\exp\left(-\frac{1}{2\sigma^{2}}\left\|\mathbf{y}-\boldsymbol{\mu}\right\|^{2}\right) \tag{3}$$

$$K_{Student}\left(\mathbf{y},\boldsymbol{\mu},\sigma\right)=\frac{\Gamma\left(\frac{\nu+3}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)\nu^{\frac{3}{2}}\pi^{\frac{3}{2}}\sigma^{3}}\left(1+\frac{1}{\nu\sigma^{2}}\left\|\mathbf{y}-\boldsymbol{\mu}\right\|^{2}\right)^{-\frac{\nu+3}{2}} \tag{4}$$

$$K_{Triangular}\left(\mathbf{y},\boldsymbol{\mu},\sigma\right)=\prod_{j=1}^{3}k_{Triangular,j}\left(y_{j},\mu_{j},\sigma\right) \tag{5}$$

$$k_{Triangular,j}\left(y_{j},\mu_{j},\sigma\right)=\begin{cases}0 & \text{for } y_{j}<\mu_{j}-\sigma\\ \frac{y_{j}-\mu_{j}+\sigma}{\sigma^{2}} & \text{for } \mu_{j}-\sigma\leq y_{j}<\mu_{j}\\ \frac{1}{\sigma} & \text{for } y_{j}=\mu_{j}\\ \frac{\mu_{j}+\sigma-y_{j}}{\sigma^{2}} & \text{for } \mu_{j}<y_{j}\leq\mu_{j}+\sigma\\ 0 & \text{for } y_{j}>\mu_{j}+\sigma\end{cases} \tag{6}$$

Table 1
Gaussian, Student-t, and triangular multivariate homoscedastic distributions.

The rationale behind this model is that the object search should be directed towards those areas of the incoming frame where detections have been recorded previously. This is managed by the multivariate homoscedastic distribution. However, the other regions of the frame must also be queried to look for objects, at a lower rate, which is managed by the uniform distribution.

In the light of the above, an algorithm can be defined so as to detect anomalous objects with the help of a panoramic camera. The algorithm is detailed as follows:

1. Initialize the set of current detected objects $\mathcal{A}$ to the empty set.
2. Load the next frame from the panoramic camera.
3. Refresh the active detected objects applying the forgetting rate α to the a priori probabilities π_i. The updated objects which are out of sight, i.e. they are outside $\mathcal{V}$ are deleted because they are no longer active.
4. Randomly draw a set of M samples from the probability distribution (2). Locate the frame window associated to each sample, and resize it to the size that the convolutional neural network (CNN) requires. Then, the resized window is supplied to the CNN. If the output vector indicates a high likelihood that an object has been detected, then add the current sample into $\mathcal{A}$, and associate the sample with the probability that the detection is reliable.
5. Go to step 2.

In the next section, a proposal for a working implementation of the described methodology on a low cost, low energy consumption microcontroller system is detailed.

3. System architecture

Detection and classification of foreground objects in digital images usually require the processing of large amounts of information in short periods. Under normal circumstances, these jobs would require the use of a high performance hardware architecture integrated by fast computers featuring powerful GPU devices so all the required calculations are performed in time so the system can carry out the jobs fast enough and as accurately as possible.

However, there are some occasions where the environmental conditions make it very difficult or just unfeasible to install an automatic video surveillance system that requires an expensive high performance hardware system such as the one described in the paragraph above. This led the authors of this work to explore the possibility of designing and implementing a system capable of performing detection and classification

of foreground objects in digital images but at a small fraction of the price and electric power consumption traditional CNN-based systems do. This system would present an architecture integrating a potential detection generator based in a multivariate homoscedastic distribution and a CNN-based classification module conveniently optimized to locally[1] achieve acceptable results when deployed in cheap and low power demanding microcontroller-based hardware devices.

Attending to the reduced computing power of the hardware it is going to be deployed in, this system should present a balance between speed and accuracy. In this respect, the choice of the CNN selected for implementing the classification module is critical, as this is the bottleneck of the system in terms of time consumption. After an extensive research process, the authors have considered the convolutional neural network designed by the Microsoft Embedded Learning Library (ELL) team whose architecture can be seen in Table 2. This CNN architecture is based in the VGG-16 network architecture [51] and was selected because it presented the best balance between speed and accuracy after performing a speed and accuracy test in a Raspberry Pi 3 Model B board to the fastest networks offered by the Microsoft ELL team.

The detection and classification system presented in this article consists of two main parts: the hardware platform and the software program that manages all the processes. Thus, both the hardware and the software architectures of the system are presented in the next two subsections.

3.1. Software architecture

The software architecture of the system is illustrated in Fig. 1. As can be seen, it consists of a program developed in C++ language composed of three different modules. The first one is a module that supplies a continuous stream of images shot by a *Point Grey Ladybug 3 Spherical camera*. It is important to remark that, in a regular basis, because of the reduced capabilities of the hardware device used for the project, the system is not capable of processing all the frames coming from the 360° camera at the speed it can supply them, so the system will accept one frame and it will start processing it. Meanwhile, every frame provided by the camera will be discarded until the current frame processing is finished. At that point, the program will accept a new frame from the camera for processing. This way, the system is always searching for anomalous objects in a recent frame avoiding the crescent lag that would happen otherwise.

The second module is dedicated to identifying the potential anomalous objects that may appear in the scene watched by the 360° surveillance camera using a pre-trained convolutional neural network that will be in charge of processing the information found in each one of the windows generated by the potential detection generator.

Working with CNNs can be problematic and inefficient unless they are implemented using a Deep Learning framework. At the same time, it is convenient to have in mind that we are operating with low computing power hardware, so an optimized library for deploying CNNs in microcontrollers is also very recommendable in order to get the maximum performance from the hardware. These reasons led us to select the *Microsoft Cognitive Toolkit* (CNTK) deep learning framework combined with the *Embedded Learning Library* (ELL) also developed by Microsoft. Microsoft CNTK is a framework intended for designing, training, and testing Convolutional Neural Networks, whilst the Microsoft Embedded Learning Library is a special library mainly used to pre-compile the code optimizing it so it can extract the highest performance rates of multi-core microcontroller-based hardware architectures.

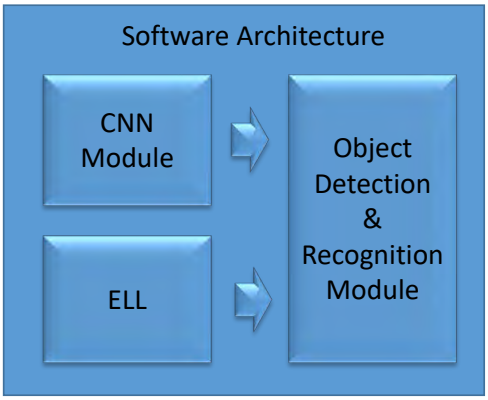


Fig. 1. Overview of the software architecture

The last part of the software architecture is the object detection module. This module is also the main program of the surveillance system and is in charge of interacting actively with the 360° camera module and the identification module when needed, processing the results incoming from it and triggering an alarm when detecting an anomalous object in the scene. However,

[1] without any network connection that would supply the possibility of using any cloud computing services.

the main task of this module is to generate the windows corresponding to the potential detections following the mathematical model explained in section 2.

As it is illustrated in Figure 2, this program is designed to receive a video frame from the 360° camera and generate a fixed number of potential detections whose coordinates in the 360° frame will be generated according to the result of the uniform probability distribution, the Gaussian-uniform, the Student-t-uniform and the triangular-uniform mixture functions proposed in section 2. Next, the areas of the frame enclosed by the cited windows will be fed to the detection module who will determine whether the potential detection contains any object appearing in the list of anomalous objects. May this be the case, the new detection would be added to the detections set (A), and the potential detection will become an actual detection. In the end, the anomalous object alarm will be triggered by drawing a bounding box around the object, spotting it in the current frame.

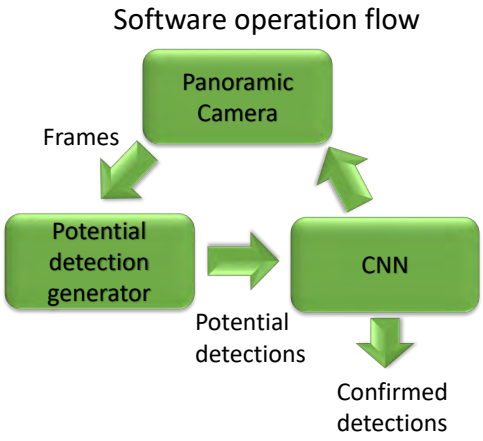


Fig. 2. Overview of the system operation flow

3.2. *Hardware architecture*

Hardware choice is a critical issue when it comes to Deep Learning applications powered by microcontrollers. This hardware should offer a good compromise between performance, energy consumption, and price. Moreover, in order to reduce the development time, it would be desirable that the chosen hardware counts on ample documentation that is easily accessible online.

These reasons suggest the choice of a system-on-chip architecture-based system, such as Raspberry Pi-based boards, to be used in our project. More precisely,

Input	244 x 244 x {B,G,R}
Architecture	Convolution, 224x224x16, size=3x3, stride=1
	Pooling, 112x112x16 size=2x2, stride=2
	Convolution 112x112x64 size=3x3, stride=1
	Pooling 56x56x64 size=2x2, stride=2
	Convolution 56x56x64 size=3x3, stride=1
	Pooling 28x28x64 size=2x2, stride=2
	Convolution 28x28x128 size=3x3, stride=1
	Pooling 14x14x128 size=2x2, stride=2
	Convolution 14x14x256 size=3x3, stride=1
	Pooling 7x7x256 size=2x2, stride=2
	Convolution 7x7x512 size=3x3, stride=1
	Pooling 4x4x512 size=2x2, stride=2
	Convolution 4x4x1024 size=3x3, stride=1
	Convolution 4x4x1000 size=1x1, stride=1
	Pooling 1x1x1000 size=4x4, stride=1
	Softmax 1x1x1000
Output	ILSVRC2012 1000 classes

Table 2
Architecture of the CNN

the platform used is a Raspberry Pi 3 Model B presenting a Broadcom BCM2837 microcontroller, featuring a CortexV8 Quad Core CPU running at 1200 Mhz from ARM, 1GB of RAM memory and a microSD data storage card. It can be powered by a 5.1 V power source, and its max power consumption is up to 2.5 A approximately at max operating load using plenty of USB external devices. The reasons for using this hardware platform instead of other system-on-chip based ones are its trade-off between price and computing power and the large amount of information referring to Raspberry Pi online that brings efficiency to the developing process. Other systems feature a similar computing power just as *Gumstix Pi* or *Orange Pi*, but they feature more capabilities to the system that increase its price or their online support is not as profuse as the Raspberry Pi resulting in higher development times.

4. Experimental results

Briefly explained, the detection system detailed in this document is founded in an algorithm that is capable of detecting and identifying certain objects whose categories are considered anomalous for some rea-

son in a particular environment by analyzing a video stream of it, supplied by a 360° static camera. The video stream is separated into frames and fed to an object detection module that will formulate a certain number of potential detections consisting of various areas of the frame following the model described in section 2. These potential detections will be sent to an identification module that will alert the user in the case of positive identification.

4.1. Experiments design

Because of its balance between accuracy and speed, for these experiments we have selected a CNN designed and trained using the convolutional neural networks implementation from Microsoft Cognitive Toolkit (Microsoft CNTK)[2] combined with the Embedded Learning Library[3], also from Microsoft, whose mission is to optimize neural networks so they can fit properly in a microcontroller-based architecture such as the one presented by Raspberry Pi boards.

In order to set up the experiments, a test program that integrates all the system modules has been implemented. This program emulates the video stream incoming from the 360° camera by supplying frames extracted from a video file recorded using an actual 360° camera. The disposal of 360° videos is an important resource as we can use the same frames through all the experimentation process, allowing us to perform multiple tests in an automatized way. The operating of the program follows the flow described in Fig. 2.

According to it, first of all, a new frame is acquired from the 360° frames pool. Next, the potential detection generation engine, implemented in the detection module of the software, scatters a certain amount of windows representing potential object detections over the frame. Those would be localized in the coordinates determined by the mixture of a random function with each of the three multivariate homoscedastic probability distributions determined by the p function expressed in equations 2, 3, 4, 5 and 6. The potential detection generator also uses a pure random probability density distribution as a control.

Then, the areas of the frame enclosed by each one of those possible detections are fed to the identification module where its pre-trained CNN will state whether the frame section delimited by certain window contains any of the objects appearing in the anomalous object category list. When the CNN finds some anomalous object in any of the potential detections, this detection will become a real detection of an anomalous object that will be added to the set of active detections, referenced in the model illustrated in section 2 as $\mathcal{A}$.

Finally, the user will be informed about the new anomalous object detection by drawing the window as a bounding box around the anomalous object that has been found in the original input image in different colors, using green for confidence value above 70%, yellow for a value between 40% and 70% and red for a value under 40% (Figure 3).

In order to test the accuracy and performance of the system described, a series of experiments have been performed by using the four different probability distribution-based functions described in section 2. Those will be used as the basis of the potential detection window generator. Aiming to make the experiments as rigorous as possible, it is very important to recreate the same conditions through all the tests. Thus, for all the experiments, it has been used a controlled scenario where a certain set of frames coming from a 360° video has been modified by introducing random moving objects, using a video editor. All of these objects belong to categories included in the ILSVR2012 dataset that have arbitrarily been stated as anomalous for that particular environment attending to diversity criteria. Categories are: “aegyptian cat”, “golden retriever”, “soccer ball”, “sunglasses”, “laptop”, “sombrero”, “bald eagle”, “banana”, “wall clock” and “chainsaw”.

Experiments consisted of counting the number of objects detected by the system by performing 10 recognition passes to 300 panoramic 1920x960 frames from six 360° videos supplied by the public LITIV dataset[4], modified artificially by introducing respectively 1, 2, 3, 5, 7 and 10 moving objects considered as anomalous. The objects have been distributed among the six videos as follows (Figure 4):

- Video 1: Chainsaw
- Video 2: Chainsaw, soccer ball.
- Video 3: Chainsaw, soccer ball, golden retriever.
- Video 4: Chainsaw, soccer ball, golden retriever, bald eagle, clock.
- Video 5: Chainsaw, soccer ball, golden retriever, bald eagle, clock, banana, aegyptian cat.

[2] `https://www.microsoft.com/en-us/cognitive-toolkit/`

[3] `https://microsoft.github.io/ELL/`

[4] `https://bitbucket.org/pierre_luc_st_charles/virtualptz_standalone`

Fig. 3. Working diagram of the algorithm's regular operation mode.

– Video 6: Chainsaw, soccer ball, golden retriever, bald eagle, clock, banana, aegyptian cat, sunglasses, laptop, sombrero.

It is important to remark that the mentioned objects move randomly over the frame without any pose or scale changes. The reason for this is to obtain more complete statistics about the system performance with different amounts of objects having different behaviors but without restraining the response of the system and being careful that the number of experiments would not grow to unmanageable dimensions. Besides, all the tests were performed for a number of potential detection windows that goes from 1 to 10 and all these operations were performed for each potential detections generation methods considered in this document: A Gaussian-uniform mixture, a Student-t-uniform mixture, a triangular-uniform mixture and a pure uniform distribution that has been used as a control to demonstrate the performance of the mixture-based mathematical model.

When it comes to test execution, with the purpose of reducing the amount of the possible values that the variables can have, so the amount of cases to test is manageable, we have fixed some values from the mathematical model described in Section 2, leaning on an automatic empirical parameter adjusting process *ad hoc*. Therefore, the potential detections' size will oscillate between 100x100 and 244x244 pixels, the damping factor, α, is set to 0.7, σ, is set to 0.3 and q will also be fixed to 0.7. The results obtained represent the mean number of anomalous objects detected for each number of potential detections and each probability distribution function.

As for the dataset to train, validate and test the system performance levels, it has been used the Large Scale Visual Recognition Challenge 2012 (ILSVRC 2012) from ImageNet [5] . It is important to remark that due to efficiency reasons, only the detection and classification stages are performed on-the-fly by the Raspberry Pi 3 Model B. The training of the network has been achieved offline using a *NVIDIA TITAN X* GPU.

[5] http://www.image-net.org/challenges/LSVRC/

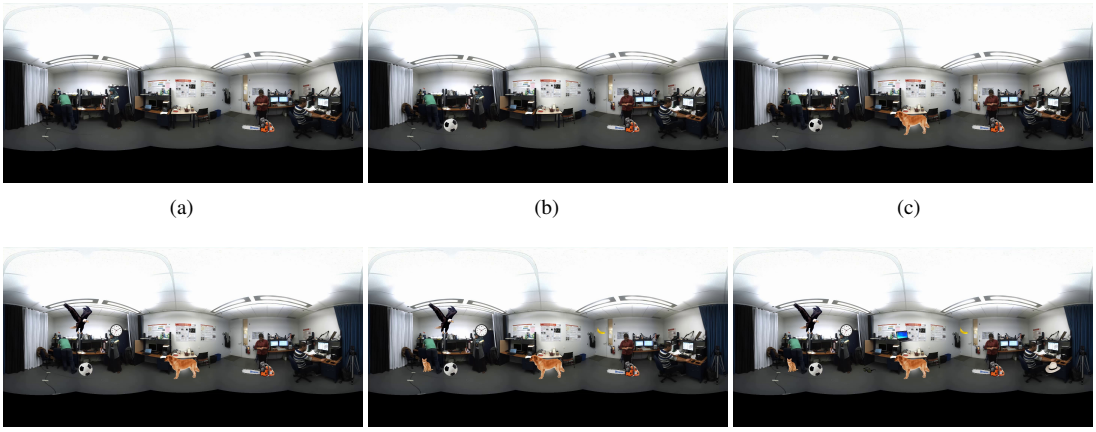
4.2. Accuracy results

Under the conditions exposed above, the mean number of objects detected by the system for scenes where 1,2,3,5,7 and 10 objects have been introduced, for each one of the three mixtures described in the mathematical model and the pure uniform distribution can be checked in the Figure 5.

Globally, we can observe that for all amounts of objects in the experiments, the performance of the mixture based models is better than the pure uniform distribution. Besides, it is easy to observe that the triangular-uniform mixture model performs significantly better than any other of the models that have been tested in all the experiments. Below, a detailed analysis of each chart can be found.

For 1 object, the figure illustrates that even for a small number of potential detections generated by the system, the triangular-uniform mixture's performance clearly outcomes all the others. It is followed by the Gaussian-uniform distribution based model that achieves slightly better results than the Student-t distribution based model. Both performances are very similar, though. The worst performance corresponds to the uniform distribution that we use as a control.

In the case of 2 objects we can observe that even though the triangular-uniform mixture model still outcomes every other model, the difference is not as important as in the 1-object chart. Near the 8 potential detections, we can observe an important oscillation of the triangular-uniform mixture performance. Based on empirical observations, it can be concluded that the reason behind this strong oscillation seems to be the particular distribution of some objects in the image that the network often finds it difficult to identify. However, a deeper statistic study may be required in order to find out the real causes of this phenomenon. Again, the Gaussian-uniform and Student-t-uniform mixture models offer a very similar behavior that gets reflected in a soft curve that for the maximum number of poten-



(a) (b) (c)

(d) (e) (f)

Fig. 4. Frames corresponding to each of the six videos used to perform the experiments.

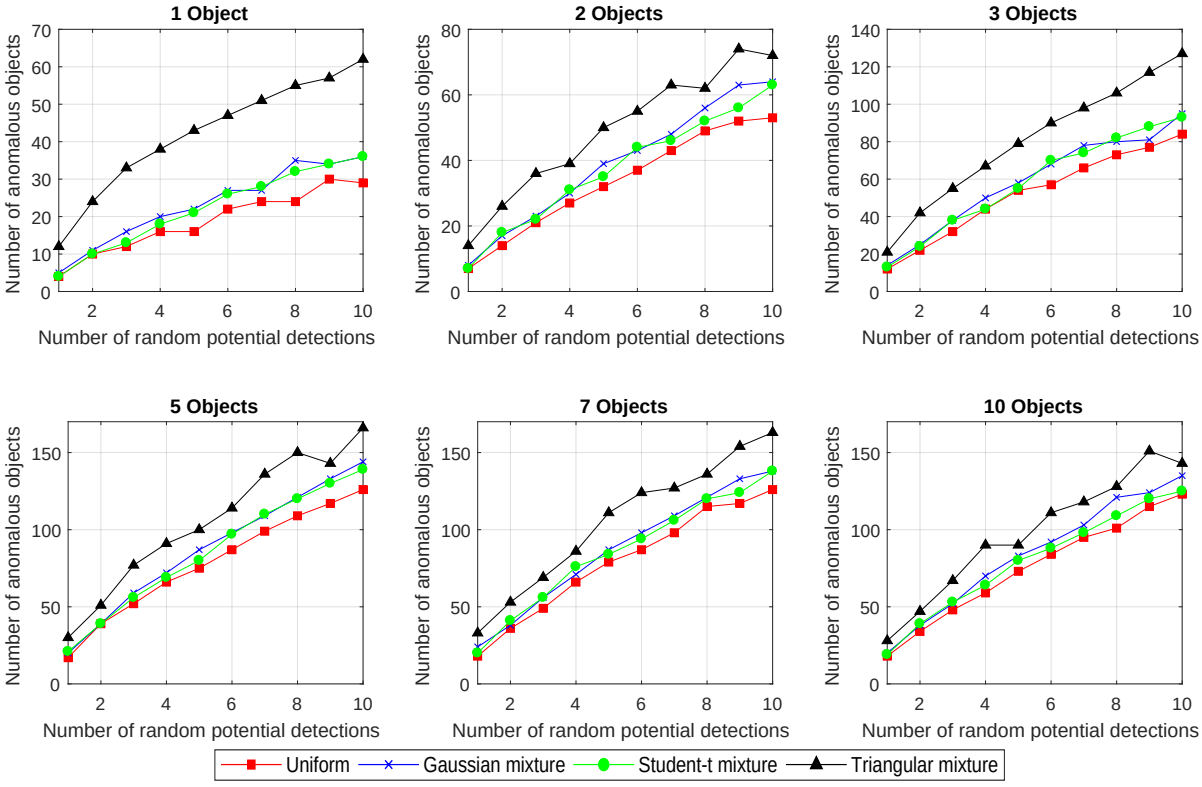


Fig. 5. Mean number of actual anomalous object detections vs number of potential detections generated for each frame.

tial detections almost reaches the number of detections of the triangular-uniform mixture model.

In the case of the video featuring 3 anomalous objects, it can be noticed a very significant increase in the global number of detections. This can be explained because the third object introduced is from the "golden retriever" class which, attending to other experiments that have been performed and empirical observations, seems to be quite easy to be detected by the CNN. Thus, the increase in the number of detections. This

chart illustrates once again that the triangular-uniform model outperforms the others. In this case, for a number of 9 potential detections, it can be noticed a slight drop in the number of detections for the Gaussian model that does not seem to be affecting the Student-t mixture based model performance.

The chart corresponding to the video with 5 anomalous objects indicates the higher amount of detections achieved in the whole experimentation process. This chart illustrates one more time how the triangular-uniform distribution performs better than the others, having a strong oscillation between the 8 and 9 windows. Here, the Gaussian and Student-t models describe a soft and stable ascending curve that is the expected behavior as the number of potential detections grows high. Just as in every chart we have already analyzed, the worst performance is achieved by the pure uniform distribution based model.

The chart presenting the detection and identification performance values for the video where 7 anomalous objects have been introduced, illustrates a very similar trend to the one presented by the chart generated for the 5 objects video. However, the number of detections is not higher than in the 5-object video, which might be something unexpected. One more time, it also can be checked that the best performance is executed by the triangular-uniform mixture model, corresponding the worst performance to the pure uniform distribution model.

The last chart represents the number of object detections in the video where 10 objects have been introduced. Besides the fact that the triangular-uniform model again outcomes all the others, a non-expected phenomenon can be observed: The number of detections when generating 10 potential detections is lower than in the 7-object video. The reason for this is the overpopulation of objects in the scene. The potential detection generation engine is equipped with a routine that avoids the generation of potential detections in places where the percentage of occlusion with the real detections is too high. So, given the abundance of objects in this scene that are considered as real detections by the system, the possible places for generating potential detections by the potential detection generator are very reduced. The consequence is that from one point on, it will not be capable of generating new potential detections so it cannot detect more objects. Except for one important oscillation in the triangular-uniform model, the behavior of the rest of them is very similar to the charts we have already analyzed, with the pure uniform distribution again in the last place.

4.3. Time performance results

Time performance is a critical issue when facing the design of a deep learning-based in-motion object detection and categorization software, more especially when this software is going to be deployed in a low computing capability hardware such as the one described in Section 3. Hence, several time performance tests have been executed after deploying the described system in the Raspberry Pi 3 Model B board.

In the case of the Raspberry Pi, performing the same amount of tests as it was done to test the accuracy of the system would be impractical, as the execution of those tests would last too much. So, in order to test the time performance of the system in a Raspberry Pi 3 Model B, it has been designed a lighter version of the experiments explained above that, without loss of generality, will yield the time performance values for this hardware platform. All the parameters of the probability distribution models described in section 2 will remain the same, and so will do the number of potential detections. Therefore, the experiments consisted of counting the number of objects detected by the system by performing 5 recognition passes to 10 360° video frames for a number of random windows that go from 1 to 10. These operations were performed for each one of the mixture distributions based models described in Section 2, namely Gaussian-uniform, Student-t-uniform, and triangular-uniform mixture based models, and the uniform distribution. In Table 3 the time performance results obtained after the experiments are detailed.

In general, time statistics obtained reveal that all four detection models considered in this research have similar frame processing speeds which go approximately from 2 frames per second, in the case of generating just 1 potential detection per frame, to 0.2 frames per second, in the case of generating 10 potential detections per frame. Looking deeply into these results, slight differences can be observed in the frame processing time, depending on the model we choose for the potential detections generation. Thus, the information displayed in Table 3 indicates that the fastest model is the one based in the Student-t-uniform mixture whilst the slowest ones are the Gaussian-uniform mixture and the pure uniform models. On the other hand, the triangular-uniform mixture model presents an interesting balance between time performance and accuracy.

Results obtained from this experimentation process have three important consequences: The first one is

# Windows	1	2	3	4	5	6	7	8	9	10
Uniform (fps)	1.97239	1.00807	0.677232	0.508079	0.407167	**0.339628**	0.29163	0.253476	0.226378	0.20419
Gaussian mixture (fps)	1.9687	1.01408	**0.67866**	**0.509737**	**0.407199**	**0.339628**	**0.292478**	**0.274575**	**0.243356**	**0.209996**
Student-t Mixture (fps)	**2.00344**	1.00788	0.675768	0.507409	0.405252	0.337169	0.290884	0.254842	0.226583	0.204107
Triangular Mixture (fps)	1.99063	**1.00604**	0.674766	0.506079	0.406802	0.339006	0.290918	0.254118	0.226665	0.204132

Table 3

System performance expressed in mean fps. vs number of potential detection generations for the three mixture models and uniform model.

that from all the window generation models presented in this research the triangular-uniform mixture arises as to the best in terms of time performance and detection accuracy because, even though its frame processing speed is not very different from the other models', its performance in object detection is better under the parameters we have used for this study.

The second consequence is that all the new potential detection generation models described in section 2 seem to perform better than a pure uniform distribution-based model.

The third consequence is that even though the video surveillance system described in this document is not capable of real-time object detection, it is actually capable of detecting foreground objects which are in motion in a non-controlled environment in half a second approximately when deployed in a Raspberry Pi 3 Model B. For these reasons we think our proposal is justified in terms of autonomy and price/performance relation, as it can be deployed in hardware that costs approximately 25$.

4.4. Comparison with the Tiny-YoloV3 model

In order to highlight the performance of the system described in this article, it is important to compare it against some other detection and classification systems belonging to state of the art. In order to be considered as a valid competitor for the proposal presented in this article, the system must have the capabilities of detecting and classifying objects belonging to the ILSVRC2012 dataset in a frame supplied by a panoramic camera just as the system presented in this work does. Of course, it has to be capable of being deployed in a Raspberry Pi 3 Model B without any other computing hardware assistance. These reasons led the authors of this work to consider the Yolov3 detection and classifying system by Joseph Redmon and Ali Farhadi [52] as a fair competitor as it is one of the best in the state of the art when it comes to accuracy in detection and frame processing speed. From all the YoloV3-based networks designed by the Darknet team and other authors, the most powerful version of YoloV3 that it has been possible to be deployed in a Raspberry Pi 3 Model B is the Tiny-YoloV3 [53] which is a reduced version of YoloV3.

Aiming to perform a comparison as fair as possible with the system presented in this work, the Tiny-YoloV3 network also has been trained by following the instructions of the authors in [52] with the ILSVRC2012 dataset.

Even though the YoloV3 algorithm operating is notably different from the system described in this article, the experiments with the Tiny-YoloV3 have been designed to allow the reader to have a clear idea of the performance of both systems. Thus, the experiments consisted in testing our dataset against the Tiny-YoloV3 by performing 10 detection passes through each one of the 300-frame videos conforming the dataset, namely videos containing 1, 2, 3, 5, 7 and 10 in-motion objects that we have considered as anomalous in the environments represented in the cited videos.

Figure 6 illustrates individually the results obtained from the tests. In this figure it can be observed that the Tiny-YoloV3 performs a mean number of detections between 0 and 31 objects. In this figure also can be observed that the Tiny-YoloV3 seems to perform more detections in the videos with less amount of anomalous objects. This could look unexpected at first but considering that the objects are moving around the scene, the reason behind this behavior might be that there are some areas in the frame that are more convenient for the Tiny-YoloV3 for performing detections and the objects get detected as they enter those areas of the frame. It must also be remarked that the network has been trained with 160000 iterations because it was observed that beyond that point, it was not improving its aver-

age loss. At the same time, in order to allow the Tiny-YoloV3 to count as many anomalous object detections as possible, we have set the detection threshold to a confidence rate of 1% so whenever the network discovers any trace of the existence of an anomalous object in the scene, it will be counted immediately.

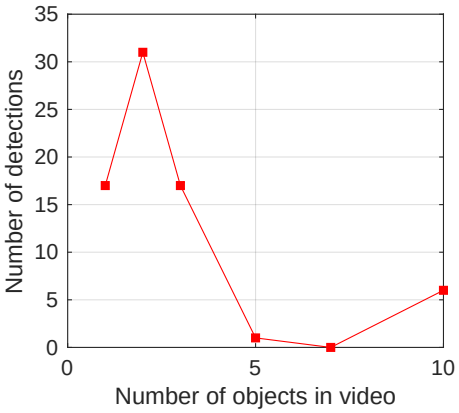


Fig. 6. Mean number of actual anomalous object detections by the Tiny-YoloV3 model vs number of anomalous objects existing in the video file.

Figure 7 illustrates the comparison for the number of anomalous objects detected by the Tiny-YoloV3 against the four detection probabilistic window generation models presented in this paper for 1, 2, 3, 6, 8 and 10 windows. It reveals how the system presented in this paper in any of its three versions outperforms, in any case, the Tiny-YoloV3-based system from 6 potential detection windows on. It also can be observed how the triangular distribution-based version of the system outperforms the Tiny-YoloV3 from 3 windows on.

As this is a system that must be deployed in a System-On-Chip based system such as the Raspberry Pi, time performance is a critical issue in order for the system to be useful enough. Table number 4 represents the mean test speed of the Tiny-YoloV3 in frames per second after performing 10 tests to a 300 frames video.

Results point out how the speed of this system does not depend on the number of anomalous objects in the frame, maintaining the test speed around 0.06 fps. The most important result that can be found in this table is the fact that the processing speed of the Tiny-YoloV3 in a Raspberry Pi 3 Model B is more than 3 times slower than the test speed presented by the probabilistic candidate window algorithm-based system presented in this paper. This fact has a notable relevance in order to illustrate the performance of the system presented in this work against one of the most popular detection systems from state of the art such as Tiny-YoloV3.

5. Conclusions

In this paper, a novel anomalous video surveillance system managed by microcontrollers and 360° cameras has been proposed. Its purpose is to track and identify objects that can be static or in-motion in an environment where these objects are considered anomalous or potentially dangerous. This system uses a Convolutional Neural Network (CNN) based module, optimized for microcontroller architectures, for detecting and characterizing anomalous objects present in the scene. Moreover, a new mathematical model has been proposed to implement a potential detection generator that is in charge of generating a fixed number of potential detections in the video frame, which will be used to detect and track anomalous objects. This mathematical model design consists of three submodels, each one of them relying on one of the following probability distributions: a Gaussian-uniform mixture, a Student-t-uniform mixture, and a triangular-uniform mixture. Experimental results reveal that the system detailed in this document is capable of accomplishing video surveillance tasks at frame rates up to 2 frames per second approximately, with an object tracking accuracy improved to acceptable levels by the described mathematical model.

Even though object detection and categorization are highly complex tasks that usually require large amounts of computing power and energy consumption, the proposal detailed in this document features a low energy, low cost system optimized for a Raspberry Pi 3 Model B microcontroller. Experimental results confirm the excellent performance of our proposal against other new detection systems from state of the art such as Tiny-YoloV3 achieving better results in both accuracy and time performance. These facts support the proposal described in this paper as a valid video surveillance system even considering the low cost and energy saving intrinsic nature of this entire project.

Regarding future work, the objective is to improve the system performance by achieving more accuracy in the detection, categorization, and tracking of anomalous objects, while maintaining the frame processing time within acceptable values. This will involve exhaustive research to find the correct parameter values

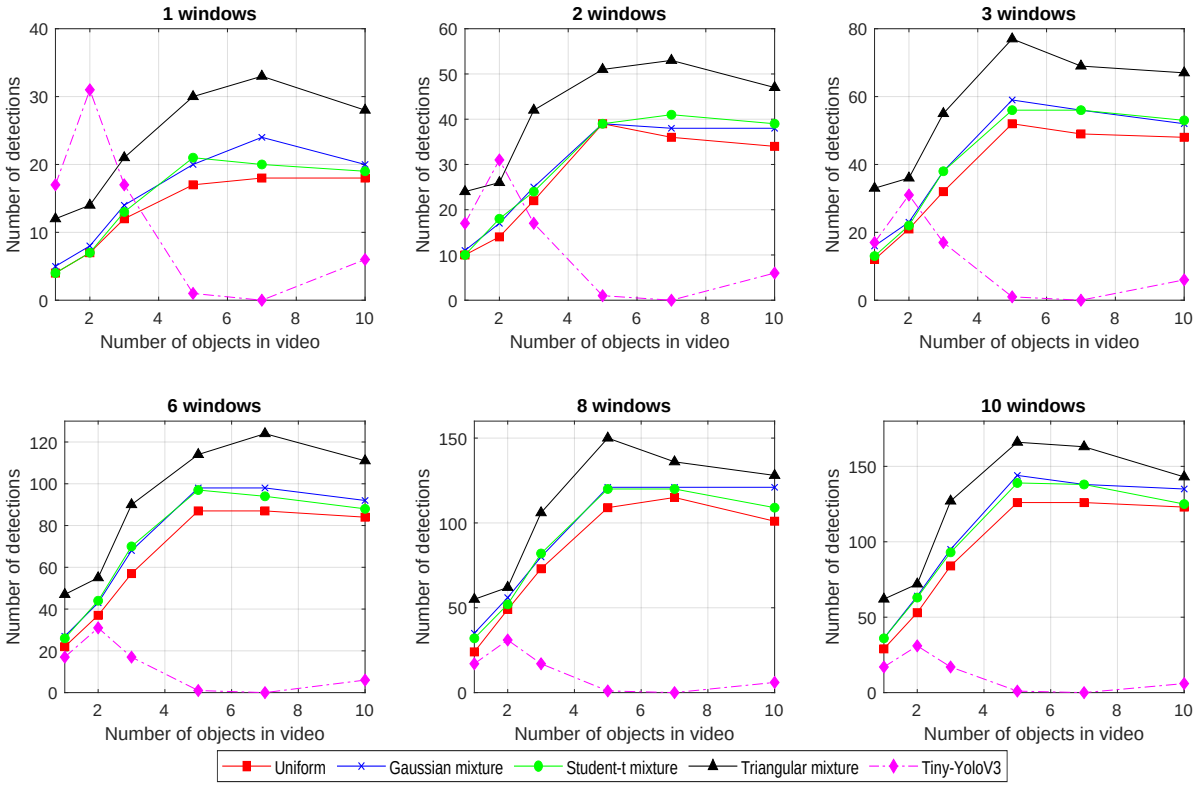


Fig. 7. Mean number of anomalous object detections vs number of objects inserted in the video for all the mathematical models in this work and the Tiny-YoloV3 detection system.

# anomalous objects in video sequence	1	2	3	5	7	10
Tiny-YoloV3 (fps)	**0.066989**	0.067051	0.067072	0.067085	0.067089	0.067090

Table 4

Tiny-YoloV3 performance measured in the Raspberry Pi expressed in mean fps. vs number of anomalous objects introduced in the video.

for tuning the mathematical model so it can perform at its best.

Acknowledgments

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under grants TIN2016-75097-P and PPIT.UMA.B1.2017. It is also partially supported by the Ministry of Science, Innovation and Universities of Spain (grant number RTI2018-094645-B-I00), project name Automated detection with low cost hardware of unusual activities in video sequences. It is also partially supported by the Autonomous Government of Andalusia (Spain) under project MA18-FEDERJA-084, project name Detection of anomalous behavior agents by deep learning in low cost video surveillance intelligent systems. All of them include funds from the European Regional Development Fund (ERDF). The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They have also been supported by the Biomedic Research Institute of Málaga (IBIMA).They also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Titan X GPUs used for this research. The authors also thankfully acknowledge the grant of the Universidad de Málaga. Karl Thurnhofer-Hemsi (FPU15/06512) is funded by a PhD scholarship from the Spanish Ministry of Ed-

ucation, Culture and Sport under the FPU program. The authors acknowledge the funding from the following grants, which was used to develop the OASIS database by its creators: P50 AG05681, P01 AG03991, R01 AG021910, P50 MH071616, U24 RR021382, R01 MH56584.

References

[1] C. Chen, S. Li, H. Qin and A. Hao, Robust salient motion detection in non-stationary videos via novel integrated strategies of spatio-temporal coherency clues and low-rank analysis, *Pattern Recognition* **52** (2016), 410–432.

[2] H. Sajid, S.-C.S. Cheung and N. Jacobs, Appearance based background subtraction for PTZ cameras, *Signal Processing: Image Communication* **47** (2016), 417–425.

[3] J. Huo, Y. Gao, W. Yang and H. Yin, Multi-Instance Dictionary Learning for Detecting Abnormal Events in Surveillance Videos, *International Journal of Neural Systems* **24**(03) (2014), 1430010.

[4] R.G. Mesquita and C.A.B. Mello, Object recognition using saliency guided searching, *Integrated Computer-Aided Engineering* **23**(4) (2016), 385–400.

[5] B. Lacabex, A. Cuesta-Infante, A.S. Montemayor and J.J. Pantrigo, Lightweight tracking-by-detection system for multiple pedestrian targets, *Integrated Computer-Aided Engineering* **23**(3) (2016), 299–311.

[6] T.E. Boult, X. Gao, R. Micheals and M. Eckmann, Omnidirectional visual surveillance, *Image and Vision Computing* **22**(7) (2004), 515–534.

[7] K.-T. Song and J.-C. Tai, Dynamic calibration of pan-tilt-zoom cameras for traffic monitoring, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **36**(5) (2006), 1091–1103.

[8] C. Micheloni, B. Rinner and G.L. Foresti, Video analysis in pan-tilt-zoom camera networks, *IEEE Signal Processing Magazine* **27**(5) (2010), 78–90.

[9] C. Ding, B. Song, A. Morye, J.A. Farrell and A.K. Roy-Chowdhury, Collaborative sensing in a distributed PTZ camera network, *IEEE Transactions on Image Processing* **21**(7) (2012), 3282–3295.

[10] C. Ding, J.H. Bappy, J.A. Farrell and A.K. Roy-Chowdhury, Opportunistic Image Acquisition of Individual and Group Activities in a Distributed Camera Network, *IEEE Transactions on Circuits and Systems for Video Technology* **27**(3) (2017), 664–672.

[11] E. Komagal and B. Yogameena, Foreground segmentation with PTZ camera: a survey, *Multimedia Tools and Applications* **77** (2018), 22489–22542.

[12] A. Ferone and L. Maddalena, Neural Background Subtraction for Pan-Tilt-Zoom Cameras, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **44**(5) (2014), 571–579, ISSN 2168-2232.

[13] G. Allebosch, D. Van Hamme, P. Veelaert and W. Philips, Robust pan/tilt compensation for foreground-background segmentation, *Sensors* **19**(12) (2019), 27, ISSN 1424-8220.

[14] Y. Yagi, Omnidirectional Sensing and Its Applications, 1999.

[15] G. Scotti, L. Marcenaro, C. Coelho, F. Selvaggi and C.S. Regazzoni, Dual camera intelligent sensor for high definition 360 degrees surveillance, *IEE Proceedings - Vision, Image and Signal Processing* **152**(2) (2005), 250–257, ISSN 1350-245X.

[16] Y. Sato, K. Hashimoto and Y. Shibata, A New Networked Surveillance Video System by Combination of Omni-Directional and Network Controlled Cameras, in: *Network-Based Information Systems*, M. Takizawa, L. Barolli and T. Enokido, eds, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 313–322. ISBN ISBN 978-3-540-85693-1.

[17] T. Gandhi and M.M. Trivedi, Motion analysis for event detection and tracking with a mobile omnidirectional camera, *Multimedia Systems* **10**(2) (2004), 131–143, ISSN 1432-1882.

[18] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu and F.E. Alsaadi, A survey of deep neural network architectures and their applications, *Neurocomputing* **234**(November 2016) (2017), 11–26.

[19] A. Antoniades, L. Spyrou, D. Martin-Lopez, A. Valentin, G. Alarcon, S. Sanei and C.C. Took, Deep Neural Architectures for Mapping Scalp to Intracranial EEG, *International journal of neural systems* **28**(8) (2018), Cited By :22. www.scopus.com.

[20] C. Hua, H. Wang, S. Lu, C. Liu and M. Khalid Syed, A Novel Method of Building Functional Brain Network Using Deep Learning Algorithm with Application in Proficiency Detection, *International Journal of Neural Systems* (2018).

[21] C. Hua, H. Wang, S. Lu, C. Liu and S.M. Khalid, A Novel Method of Building Functional Brain Network Using Deep Learning Algorithm with Application in Proficiency Detection, *International journal of neural systems* **29**(1) (2019), Cited By :8. www.scopus.com.

[22] A.H. Ansari, P.J. Cherian, A. Caicedo, G. Naulaers, M. De Vos and S. Van Huffel, Neonatal Seizure Detection Using Deep Convolutional Neural Networks, *International journal of neural systems* **29**(4) (2019), Cited By :25. www.scopus.com.

[23] U.R. Acharya, S.L. Oh, Y. Hagiwara, J.H. Tan and H. Adeli, Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals, *Computers in Biology and Medicine* **100** (2018), 270–278, ISSN 0010-4825.

[24] M.O. Manzanera, S.K. Meles, K.L. Leenders, R.J. Renken, M. Pagani, D. Arnaldi, F. Nobili, J. Obeso, M.R. Oroz, S. Morbelli and N.M. Maurits, Scaled subprofile modeling and convolutional neural networks for the identification of ParkinsonâĂŹs disease in 3D nuclear imaging data, *International Journal of Neural Systems* **29**(9) (2019), 1950010.

[25] M.H. Rafiei, W. Khushefati, R. Demirboga and H. Adeli, Supervised Deep Restricted Boltzmann Machine for Estimation of Concrete, *ACI Materials Journal* (2017).

[26] M.H. Rafiei and H. Adeli, A novel machine learning-based algorithm to detect damage in high-rise building structures, *The Structural Design of Tall and Special Buildings* **26**(18) (2017), 1400.

[27] M.H. Rafiei and H. Adeli, A novel unsupervised deep learning model for global and local health condition assessment of structures, *Engineering Structures* **156** (2018), 598–607, ISSN 0141-0296.

[28] M.H. Rafiei and H. Adeli, Novel Machine-Learning Model for Estimating Construction Costs Considering Economic Variables and Indexes, *Journal of Construction Engineering and Management* **144**(12) (2018), 04018106.

[29] J. Torres, A. Galicia de Castro, A. Troncoso and F. Martínez-Álvarez, A scalable approach based on deep learning for big data time series forecasting, *Integrated Computer-Aided Engi-*

neering **25** (2018), 1–14.

[30] M.H. Rafiei and H. Adeli, A Novel Machine Learning Model for Estimation of Sale Prices of Real Estate Units, *Journal of Construction Engineering and Management* **142** (2015), 04015066.

[31] S. Zhang, F. He, W. Ren and J. Yao, Joint learning of image detail and transmission map for single image dehazing, *The Visual Computer* (2018), ISSN 1432-2315. doi:10.1007/s00371-018-1612-9. https://doi.org/10.1007/s00371-018-1612-9.

[32] A. Kuznetsov, Digital image forgery detection using deep learning approach, *Journal of Physics: Conference Series* **1368** (2019), 032028. doi:10.1088/1742-6596/1368/3/032028. https://doi.org/10.1088%2F1742-6596%2F1368%2F3%2F032028.

[33] M.T. McCann, K.H. Jin and M. Unser, Convolutional Neural Networks for Inverse Problems in Imaging: A Review, *IEEE Signal Processing Magazine* **34**(6) (2017), 85–95.

[34] M. Koziarski and B. Cyganek, Image recognition with deep neural networks in presence of noise - Dealing with and taking advantage of distortions, *Integrated Computer-Aided Engineering* **24** (2017), 337–349.

[35] A. Zhang, K.C.P. Wang, Y. Fei, Y. Liu, C. Chen, G. Yang, J.Q. Li, E. Yang and S. Qiu, Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces with a Recurrent Neural Network, *Computer-Aided Civil and Infrastructure Engineering* **34**(3) (2019), 213–229, Cited By :28. www.scopus.com.

[36] X. Liang, Image-Based Post-Disaster Inspection of Reinforced Concrete Bridge Systems Using Deep Learning with Bayesian Optimization, *Computer-Aided Civil and Infrastructure Engineering* **34**(5) (2019), 415–430.

[37] S. Bang, S. Park, H. Kim and H. Kim, EncoderâĂŞdecoder network for pixel-level road crack detection in black-box images, *Computer-Aided Civil and Infrastructure Engineering* **34**(8) (2019), 713–727.

[38] K. Maeda, T. Ogawa, M. Haseyama, and S. Takahashi, Convolutional Sparse Coding-based Deep Random Vector Functional Link Network for Distress Classification of Road Structures, *Computer-Aided Civil and Infrastructure Engineering* **34**(8) (2019), 654–676.

[39] X. Luo, H. Li, X. Yang, Y. Yu and D. Cao, Capturing and Understanding WorkersâĂŹ Activities in Far-Field Surveillance Videos with Deep Action Recognition and Bayesian Nonparametric Learning, *Computer-Aided Civil and Infrastructure Engineering* **34**(4) (2019), 333–351, Cited By :16. www.scopus.com.

[40] P. Wang and X. Bai, Regional parallel structure based CNN for thermal infrared face identification, *Integrated Computer-Aided Engineering* **25** (2018), 1–14.

[41] J. Shen, X. Xiong, Z. Xue and Y. Bian, A Convolutional Neural Network-Based Pedestrian Counting Model for Various Crowded Scenes, *Computer-Aided Civil and Infrastructure Engineering* **34**(10) (2019), 897–914.

[42] M.A. Molina-Cabello, R.M. Luque Baena, E. López-Rubio and K. Thurnhofer-Hemsi, Vehicle type detection by ensembles of convolutional neural networks operating on super resolved images, *Integrated Computer-Aided Engineering* **25** (2018), 1–13.

[43] L. Tong, F. Dai, D. Zhang, D. Wang and Y. Zhang, Encoder Combined Video Moving Object Detection, *Neurocomput.* **139** (2014), 150–162.

[44] P. Angelov, P. Sadeghi-Tehran and C. Clarke, AURORA: Autonomous Real-time On-board Video Analytics, *Neural Comput. Appl.* **28**(5) (2017), 855–865.

[45] A. Dziri, M. Duranton and R. Chapuis, Real-time multiple objects tracking on Raspberry-Pi-based smart embedded camera, *Journal of Electronic Imaging* **25** (2016), 041005.

[46] F. Ortega-Zamorano, J.M. Jerez, I. Gómez and L. Franco, Layer multiplexing FPGA implementation for deep back-propagation learning, *Integrated Computer-Aided Engineering* **24**(2) (2017), 171–185.

[47] M.K. Dobrzynski, R. Pericet-Camara and D. Floreano, Vision Tape-A Flexible Compound Vision Sensor for Motion Detection and Proximity Estimation, *IEEE Sensors Journal* **12**(5) (2012), 1131–1139.

[48] V. Fung, J.L. Bosch, S.W. Roberts and J. Kleissl, Cloud shadow speed sensor, *Atmospheric Measurement Techniques* **7**(6) (2014), 1693–1700.

[49] L.H. Adnan, Y.M. Yussoff, H. Johar and S.R.M.S. Baki, Energy-saving street lighting system based on the waspmote mote, *Jurnal Teknologi* **76**(4) (2015), 55–58.

[50] F. Ortega-Zamorano, M.A. Molina-Cabello, E. López-Rubio and E.J. Palomo, Smart motion detection sensor based on video processing using self-organizing maps, *Expert Systems with Applications* **64** (2016), 476–489.

[51] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, *CoRR* **abs/1409.1556** (2014).

[52] J. Redmon and A. Farhadi, YOLOv3: An Incremental Improvement, *arXiv* (2018).

[53] W. He, Z. Huang, Z. Wei, C. Li and B. Guo, TF-YOLO: An improved incremental network for real-time object detection, *Applied Sciences (Switzerland)* **9**(16) (2019), Cited By :6. www.scopus.com.