Neural Controller for PTZ cameras based on nonpanoramic foreground detection

Miguel A. Molina-Cabello*, Ezequiel López-Rubio*, Rafael Marcos Luque-Baena*,
Enrique Domínguez* and Karl Thurnhofer-Hemsi*
*Department of Computer Languages and Computer Science
University of Málaga, Bulevar Louis Pasteur, 35, 29071 Málaga, Spain
Emails: {miguelangel,ezeqlr,rmluque,enriqued,karlkhader}@lcc.uma.es

***Abstract*—In this paper a controller for PTZ cameras based on an unsupervised neural network model is presented. It takes advantage of the foreground mask generated by a non-parametric foreground detection subsystem. Thus, our aim is to optimize the movements of the PTZ camera to attain the maximum coverage of the observed scene in presence of moving objects. A growing neural gas (GNG) is applied to enhance the representation of the foreground objects. Both qualitative and quantitative results are reported using several widely used datasets, which demonstrate the suitability of our approach.**

1. Introduction

Most of the former surveillance systems were built with a single stationary camera for many years. However, nowadays it is possible to find different types of cameras and any surveillance system is frequently composed by multiple devices which try to cover the largest possible area. [1]. Among other types or criteria, two of the most used types of camera are the omnidirectional and the pan-tilt-zoom (PTZ) cameras. Conventional surveillance systems usually comprise at least one omnidirectional camera and one PTZ camera.

Surveillance systems are capable of monitoring the entire scene by using a single omnidirectional camera but, due to the limited resolution of these panoramic cameras, detailed information of the objects might not be acquired. As a result of that, a PTZ camera is used for those tasks, which require close-up views at high resolution.

PTZ cameras are well suited for object identification and recognition in far-field scenes. However, the practical use of PTZ cameras in real world scenarios is complicated due to several reasons [2]. A continuous online camera calibration is needed since the absolute pan, tilt and zoom positional values provided by the camera actuators are not synchronized with the video stream in most cases. Moreover, some adaptive background representation becomes necessary to make target tracking, since the scene background is continuously changing due to the camera operation [3].

Conventional camera systems are usually easily customizable and let users deploy the sensing infrastructures according to their needs, by adjusting the various camera parameters such as field of view, resolution, operating mode (night/day vision, indoor/outdoor) [4]. In most cases the design of the infrastructure of a surveillance system is performed manually, although the wide range of configurations and parameter settings leads to suboptimal solutions, which imply an incomplete coverage of the monitored area or, conversely, higher deployment costs to achieve a satisfactory result [5]. As a general rule, we can assume that the goal of a camera planner is to guarantee the maximum coverage of the observed space, minimizing occlusions and obtaining the best visibility of the objects of interest [6].

In this paper, we propose a method for PTZ cameras based on Growing Neural Gas (GNG) models in order to automatically determine the position and pan-tilt-zoom settings to optimize the coverage of the foreground.

Traditional foreground algorithms identify foreground pixels because their features are different from those of the background, but this leads to false detection for moving cameras. Apart of other proposals based on building a panoramic model of the scene, we are focusing on non-panoramic methods [7], [8] which are suitable for free moving cameras. The use of the neural approach filters the noise and spurious objects obtained in the foreground mask. Furthermore the moving objects in the scene are represented with higher accuracy and robustness. These are the key elements to design an effective PTZ controller.

GNG models are a type of self-organizing neural networks and one of the most successful example of unsupervised learning in a graph. The original GNG model was proposed by Fritzke [9] and has become a standard of applications in computer vision [10] and robotics [11], as well as other self-organizing models for foreground detection [12] or object tracking [13] in video sequences.

The rest of the paper is organized as follows. In section 2, the proposed control system for PTZ cameras is presented. In section 3, we report the results achieved with the proposed neural controller. Finally, section 4 includes some concluding remarks.

2. System architecture

In this section the architecture of the proposed PTZ camera control system is described. The system is made of three modules, namely a foreground detection procedure (Subsection 2.1), an unsupervised learning model to learn

978-1-5090-6182-2/17/$31.00 ©2017 IEEE

the distribution of the foreground objects (Subsection 2.2), and a control module to move the camera (Subsection 2.3).

2.1. Foreground detection

The first task to be accomplished is the detection of the pixels which belong to the foreground. This means that a binary mask must be computed for each incoming video frame, so that the foreground pixels are marked as true in that mask. In order to do this, we have used our previous background model for moving cameras [8]. The reader is kindly directed to the reference for more details. At each time instant t, the output of this algorithm is a binary flag $f_{i,j} \in \{true, false\}$ for each pixel, where (i, j) are the pixel coordinates. From this a training set is built, which comprises the coordinates of all foreground pixels:

$$\mathcal{S}_t = \{(i,j) \mid f_{i,j} = true\} \subset \mathbb{R}^2 \tag{1}$$

This training set is provided to a modified version of the GNG neural model, as specified next.

2.2. Neural model

In order to locate the most relevant foreground objects in the scene, we propose to train a GNG [9] online by episodes, so that each episode is an incoming video frame. The GNG features a variable number of neurons H, which are inserted and removed from the network as the learning process is executed. The neurons are connected by undirected links, so that the resulting graph might have several connected components. We modify the GNG to process the incoming input data by episodes, so that at episode t a new training set $\mathcal{S}_t$ is presented, which is made of D-dimensional real valued vectors, $\mathcal{S}_t \subset \mathbb{R}^D$. As seen in Subsection 2.1, for our application $D = 2$. Each neuron $i \in \{1, ..., H\}$ has an associated centroid $\mathbf{w}_i \in \mathbb{R}^D$, an age (which is a natural number) and an error variable $e_i \in \mathbb{R}$, $e_i \geq 0$. Each connection also has an age. We will note A the set of all connections, $A \subseteq \{1, ..., H\} \times \{1, ..., H\}$.

The learning algorithm is given by the following steps:

1) At the initial episode $t = 0$ start with two neurons ($H = 2$) joined by a connection. Each prototype is initialized to a sample drawn at random from $\mathcal{S}_0$. The error variables are initialized to zero. The age of the connection and the neurons are initialized to zero, too.
2) Draw a training sample $\mathbf{x} \in \mathbb{R}^D$ at random from from $\mathcal{S}_t$.
3) Find the nearest neuron q and the second nearest neuron s in terms of Euclidean distance:

$$q = \arg \min_{i \in \{1,...,H\}} \|\mathbf{w}_i - \mathbf{x}\| \tag{2}$$

$$s = \arg \min_{i \in \{1,...,H\} - \{q\}} \|\mathbf{w}_i - \mathbf{x}\| \tag{3}$$

4) Increment the age of all edges departing from q.
5) Add the squared Euclidean distance between $\mathbf{x}$ and the nearest neuron q to the error variable e_q:

$$\Delta e_q = \|\mathbf{w}_q - \mathbf{x}\|^2 \tag{4}$$

6) Update q and all its direct topological neighbors with step size ϵ_b for neuron q and ϵ_n for the neighbors, where $\epsilon_b > \epsilon_n$:

$$\epsilon(i) = \begin{cases} \epsilon_b & \text{iff } i = q \\ \epsilon_n & \text{iff } (i \neq q) \wedge (i,q) \in A \\ 0 & \text{iff } (i \neq q) \wedge (i,q) \notin A \end{cases} \tag{5}$$

$$\Delta \mathbf{w}_i = \epsilon(i)(\mathbf{x} - \mathbf{w}_i) \tag{6}$$

7) If q and s are connected by an edge, then set the age of this edge to zero. Otherwise, create it.
8) Set the age of q and s to zero, and increment the age of all the other neurons.
9) Remove edges with an age larger than a_{max}. Then remove all neurons which have no outgoing edges, and those neurons whose age is larger than a_{max}.
10) If λ samples have been processed since the last neuron creation and the current number of neurons H is lower than the maximum H_{max}, then insert a new neuron as follows. First determine the neuron r with the maximum error and the neuron z with the largest error among all direct neighbors of r. Then create a new neuron k, insert edges connecting k with r and z, and remove the original edge between r and z. After that, decrease the error variables e_r and e_z by multiplying them with a constant α, and initialize the error variable e_k to the new value of e_r. Finally, setup the prototype of k to be halfway between those of r and z, as follows:

$$\mathbf{w}_k = \frac{1}{2}(\mathbf{w}_r + \mathbf{w}_z) \tag{7}$$

11) Decrease all error variables e_i by multiplying them by a constant d.
12) Remove all neurons which have not won during the last N_k steps, and their connections.
13) If the maximum number of samples to be processed for the current episode has been reached, then go to step 13. Otherwise, go to step 2.
14) If the last episode has been processed, then stop. Otherwise, increment the episode counter t, load the next episode and go to step 2.

2.3. Camera control

The last step of the system is the camera control module. At each time instant t, the set of connected components of the directed graph associated to the GNG is computed. From the set of all connections $A \subseteq \{1, ..., H\} \times \{1, ..., H\}$, the set of connected components $\mathcal{S}_t \in 2^{\{1,...,H\}}$ is a partition of

the set of all neurons $\{1, ..., H\}$, so that each set $S_{i,t} \in \mathcal{S}_t$ comprises neurons which are linked by a chain of connections in A. These connected components are associated to foreground objects which appear in the scene at time t. From them we choose the largest one, i.e. the connected component associated to the largest foreground object. Then the centroid of that component is computed:

$$\boldsymbol{\mu}_t = \frac{1}{\left|\hat{S}_t\right|} \sum_{i \in \hat{S}_t} \mathbf{w}_i \tag{8}$$

where $\hat{S}_t$ stands for the largest connected component of $\mathcal{S}_t$. Finally, the camera is moved towards the centroid $\boldsymbol{\mu}_t$.

In addition to this, we have assumed that the size of the connected component is calculated like a circle, considering the centroid as the center of the circle and its radius as the mean distance from each neuron to the centroid:

$$\boldsymbol{r}_t = \frac{1}{\left|\hat{S}_t\right|} \sum_{i \in \hat{S}_t} \|\mathbf{w_i} - \boldsymbol{\mu_t}\| \tag{9}$$

Therefore the size of the connected component is estimated as follows:

$$\boldsymbol{\omega}_t = \boldsymbol{\pi} \boldsymbol{r}_t^2 \tag{10}$$

The camera control module, given the centroid and the size of the largest connected component, sends to the camera the commands that it must execute in order to follow the tracked foreground objects. The commands the module can send to the camera are horizontal, vertical and zoom movements. With the horizontal movement the camera moves to the left or to the right; the camera can move upwards or downwards with the vertical movement; and it can apply zoom in or zoom out with the zoom movement. The amounts of movement of each kind are quantized, so that there is a minimum movement of size ρ_δ for each kind of movement $\delta \in \{pan, tilt, zoom\}$. Furthermore, we have to avoid forbidden movements. For example, the camera can not apply zoom indefinitely because it has defined a maximum and a minimum zoom value. On the other hand, no movement can be applied in each kind of movement. Besides, the camera will do one movement per each kind of movement at each time instant t of the video. Thus, if we represents the horizontal, vertical and zoom position of the camera in these frame as $(\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t, \boldsymbol{\gamma}_t)$ as spherical coordinates, in the $(t+1)$-th frame the position of the camera position will be:

$$(\boldsymbol{\sigma}_{t+1}, \boldsymbol{\beta}_{t+1}, \boldsymbol{\gamma}_{t+1}) = (\boldsymbol{\sigma}_t, \boldsymbol{\beta}_t, \boldsymbol{\gamma}_t) + (\Delta\boldsymbol{\sigma}_t, \Delta\boldsymbol{\beta}_t, \Delta\boldsymbol{\gamma}_t) \tag{11}$$

where the variation of each kind of movement δ is $\delta_t \in \{-\rho_\delta, 0, \rho_\delta\}$ for a decision of the control module to decrease, stop or increase the position of the camera in that kind of movement, respectively.

Furthermore, each kind of movement has a maximum and a minimum possible value as a physical limit of the camera. For example, we cannot apply zoom in or zoom out all we want. These maximum and minimum values are noted $\boldsymbol{\Psi_\delta}$ and $\boldsymbol{\psi_\delta}$, respectively.

In order to carry out as few movements as possible, we have defined some scenarios where no movement is applied to the camera. So that, when the horizontal and vertical coordinates of the centroid of the tracked object are not far from the vertical and horizontal coordinates of the center of the frame, then the control module does not issue a vertical and horizontal movement command, respectively. Moreover, no zoom movement is applied if the size (in pixels) of the target object is between a minimum and a maximum value of percentage respect to the overall number of pixels of the frame. So that, for each kind of movement δ we have specified a minimum and a maximum value: ϕ_δ and Φ_δ, respectively. The values for the horizontal and vertical movements indicate the distance (in degrees) from the center of the frame to the centroid; and the values for the zoom movement indicates the percentage of the number of pixels of the frame which are occupied by the target.

Finally, when no target is found the control module actively tries to find a target by moving the camera in order to avoid system states where the camera will never find a target.

3. Experimental results

In this section we show the computational experiments we have carried out and its results. The software and hardware that have been used are reported in Subsection 3.1. Then, the tested video sequences are specified in Subsection 3.2. The tuned parameters of the software are shown in Subsection 3.3. Finally, the obtained results from the experiments are described in Subsection 3.4.

3.1. Methods

We have used a nonpanoramic foreground object detection algorithm for our control system. The method we have employed is a related work from our research group [8]. This method that we note *nonpan* is implemented in Matlab and it uses MEX files written in C++ for the most CPU time demanding parts. The implementation is available on its website [1]. On the other hand, our implementation of the GNG model is written in Matlab.

The camera control module is based on the *virtualptz library* [14]. It simulates the operation of a PTZ camera from a 360-degree panoramic video, and it is accessible from its website [2]. Its implementation is written in C++ and uses the OpenCV library[3].

The reported experiments have been carried out on a 64-bit Personal Computer with an eight-core Intel i7 3.60 GHz CPU, 32 GB RAM and standard hardware. The implementation of our approach does not use any GPU resources, so it does not require any specific graphics hardware.

1. http://www.lcc.uma.es/~ezeqlr/nonpan/nonpan.html
2. https://bitbucket.org/pierre_luc_st_charles/virtualptz_standalone
3. http://opencv.org/

TABLE 1. CONSIDERED PARAMETER VALUES.

Method	Parameters
Nonpan	Features, $F = \{[19\ 20\ 22]\}$
	Step size, $\epsilon = 0.03$
	Threshold, $\tau = 0.999$
GNG model	Max units, $H_{max} = 100$
	Lambda, $\lambda = 100$
	Number of steps, $N = 20000$
	Epsilon B, $\epsilon_b = 0.2$
	Epsilon N, $\epsilon_n = 0.006$
	Alpha, $\alpha = 0.5$
	A max, $a_{max} = 50$
	D, $d = 0.995$
	Steps to remove non active neurons, $N_k = 1000$
Virtualptz	Variation of horizontal movement, $\rho_\sigma = 2$
	Variation of vertical movement, $\rho_\beta = 2$
	Variation of zoom movement, $\rho_\gamma = 2$
	Minimum horizontal limit, $\psi_\sigma = -180$
	Maximum horizontal limit, $\Psi_\sigma = 180$
	Minimum vertical limit, $\psi_\beta = 0$
	Maximum vertical limit, $\Psi_\beta = 180$
	Minimum zoom limit, $\psi_\gamma = 40$
	Maximum zoom limit, $\Psi_\gamma = 140$
	Minimum horizontal distance, $\phi_\sigma = -10$
	Maximum horizontal distance, $\Phi_\sigma = 10$
	Minimum vertical distance, $\phi_\beta = -20$
	Maximum vertical distance, $\Phi_\beta = 20$
	Minimum zoom distance, $\phi_\beta = 0.10$
	Maximum zoom distance, $\Phi_\beta = 0.40$

3.2. Sequences

Three videos have been used in order to carry out the experiments, which are available on the virtualptz website. The three sequences are indoor scenes and they are named *scenario3*, *scenario4* and *scenario5*. The two first ones correspond to the same location (a spacious hall with people walking on in) and they are very similar, so we have just reported the results for the *scenario3* sequence (3500x1750 pixels and 566 frames). On the other hand, the *scenario5* sequence (3500x1750 pixels and 1957 frames) shows a room with people moving on in and doing different actions.

3.3. Parameter selection

A set of tuned values for the parameters of the methods have been defined to carry out the experiments. These fixed parameters have been chosen from the recommendations of the GNG authors and our experience. They are reported in Table 1. In particular, the values related to the PTZ camera control have been chosen in order to address the low frame rate of the benchmark videos.

3.4. Results

The operation of our proposal with one of the selected videos is reported in Fig. 1. In this figure, a captured frame with the PTZ camera is shown, along with the obtained result after the execution of the nonpan algorithm, the state of the GNG model in that moment and the largest connected component which is to be tracked. This last image also shows the centroid of the largest (tracked) component.

Afterwards, depending of the centroid and the size of the selected component, the control module will indicate the different steps (horizontal, vertical and zoom movements) to the camera.

The GNG state depends directly from its previous state and the nonpan output. The faster the movement of the camera and the actions of the agents, the less close the GNG will be with respect to the desired one. This can be better appreciated in Fig. 2. The evolution of the GNG state keeps connected components of neurons even though they belong to different components. This effect is stronger for increased agent activity. Nevertheless, the own nature of the GNG provides an efficient approximation of the centroid of the target and a suitable approximation to its size.

Furthermore, in the two first rows it can be observed how the result and the evolution of the video can be different depending of the GNG neural network initialization. This has an influence on the controller module and its decision could be different with the same frame. This normally occurs when the frame presents more than one person or a high level of noise in the output produced by the nonpan.

We have chosen some frames as ground truth for the purpose of determining the performance due to the sequences not incorporating a ground truth mask. Because of this we have used a tracking ground truth added to the tested videos, which contains the information about the centroid and the bounding box of a person. Thus, the selected benchmark frames present only one person in different situations. The commands issued by the controller according to the ground truth of the tracked object position and the GNG estimated position are reported.

The approach has been run with the two tested videos and two different initializations for the purpose of having a wider range of benchmark frames, so we have four sequences. With each one we have selected 25 random frames from the benchmark frame set. As it can be observed in Fig. 3 the qualitative results offered by the GNG are similar to the ground truth. In addition, the decision of the control module is quite similar for the ground truth and the GNG data in each benchmark frame.

In order to get a quantitative point of view about the performance of our approach we have selected the accuracy, by computing the hits of the decision and the attempts. The obtained performance is reported in Table 2. We can consider the most important performance in this tested videos is the accuracy in the horizontal movement and, to a lesser extent, the zoom movement. This is because the people are moving on from left to right and vice versa, further away or closer, but almost always around the camera. According to these results, most of the mistakes in the decision of the movement are produced by the noise of the obtained result of the nonpan method and the GNG state. This can be observed in the last two rows of Fig.3: the GNG has some well-distinguished parts but they are connected, so the GNG considers a higher target and its centroid is displaced, producing a different movement decision from the ground

TABLE 2. ACCURACY RESULTS. FIRST COLUMN CORRESPONDS TO A BENCHMARK VIDEO (THE TWO TESTED SEQUENCES WITH TWO DIFFERENT INITIALIZATION) AND REMAINING COLUMNS INDICATE THE ACCURACY (HITS/ATTEMPTS) FOR THE HORIZONTAL, VERTICAL AND ZOOM MOVEMENT CONSIDERING THE 25 SELECTED BENCHMARK FRAMES FOR EACH VIDEO. EACH ROW SHOWS A VIDEO AND ITS ACCURACY PERFORMANCES AND THE LAST ROW INDICATES THE TOTAL ACCURACY CONSIDERING THE FOUR VIDEOS THAT THEY HAVE BEEN CARRIED OUT.

Video	Horizontal	Vertical	Zoom
Scenario3 (1)	16/25 (0.64)	22/25 (0.88)	22/25 (0.88)
Scenario3 (2)	23/25 (0.92)	18/25 (0.72)	19/25 (0.76)
Scenario5 (1)	11/25 (0.44)	12/25 (0.48)	11/25 (0.44)
Scenario5 (2)	17/25 (0.68)	18/25 (0.72)	7/25 (0.28)
Total	**67/100 (0.67)**	**70/100 (0.70)**	**59/100 (0.59)**

truth decision and it produces a lower accuracy. Moreover, the fact that opposite errors (for example, the ground truth indicates a left movement and the GNG a right one) have only appeared a few times must also be highlighted: 3 times for the 100 tested horizontal movement decisions and 2 times for the 100 tested zoom movement decisions.

4. Conclusion

A neural controller for PTZ cameras, which is based in a growing neural gas (GNG) approach, was presented in order to optimize the maximum coverage of the area of the scene in presence of foreground objects. The objects in motion are detected using a nonparametric foreground detection algorithm which yields a foreground binary mask. The GNG model represents the foreground mask with the aim of avoiding noise and spurious objects, in addition to provide higher robustness in the camera control module. The *virtualptz* library was used to simulated the performance of a real PTZ camera. Several publicly available video sequences has been considered in our study. In particular, some quantitative results are obtained in comparison to the ground truth (pan, tilt or zoom movement in each frame) of the scene. Within this scheme some promising results (around 65% of accuracy) are obtained. It must be taken into account that performing the exact same movement as the ground truth is not always necessary to obtain the best coverage of the scene. This means that evaluating the performance of this kind of systems is difficult. In later works, new quantitative measures should be proposed to better capture the particularities of the control process of a PTZ camera.

Acknowledgments

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under grant TIN2014-53465-R, project name Video surveillance by active search of anomalous events. It is also partially supported by the Autonomous Government of Andalusia (Spain) under projects TIC-6213, project name Development of Self-Organizing Neural Networks for Information Technologies; and TIC-657, project name Self-organizing systems and robust estimators for video surveillance. 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 also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

References

[1] Y. Xu and D. Song, "Systems and algorithms for autonomous and scalable crowd surveillance using robotic ptz cameras assisted by a wide-angle camera," *Autonomous Robots*, vol. 29, no. 1, pp. 53–66, 2010.

[2] G. Lisanti, I. Masi, F. Pernici, and A. Del Bimbo, "Continuous localization and mapping of a pantiltzoom camera for wide area tracking," *Machine Vision and Applications*, vol. 27, no. 7, pp. 1071–1085, 2016.

[3] H. Sajid, S.-C. Cheung, and N. Jacobs, "Appearance based background subtraction for ptz cameras," *Signal Processing: Image Communication*, vol. 47, pp. 417–425, 2016.

[4] K. Konda, N. Conci, and F. De Natale, "Global coverage maximization in ptz-camera networks based on visual quality assessment," *IEEE Sensors Journal*, vol. 16, no. 16, pp. 6317–6332, 2016.

[5] E. Hörster and R. Lienhart, "On the optimal placement of multiple visual sensors," in *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks*, ser. VSSN '06. New York, NY, USA: ACM, 2006, pp. 111–120. [Online]. Available: http://doi.acm.org/10.1145/1178782.1178800

[6] Gil Whoan Chu and Myung Jin Chung, "Selection of an optimal camera position using visibility and manipulability measures for an active camera system," in *Proceedings International Conference on Intelligent Robots and Systems (IROS 2000)*, vol. 1. IEEE, 2000, pp. 429–434.

[7] S. Kim, K. Yun, K. Yi, S. Kim, and J. Choi, "Detection of moving objects with a moving camera using non-panoramic background model," *Machine Vision and Applications*, vol. 24, no. 5, pp. 1015–1028, 2013.

[8] F. J. López-Rubio and E. López-Rubio, "Foreground detection for moving cameras with stochastic approximation," *Pattern Recognition Letters*, vol. 68, pp. 161–168, 2015.

[9] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems 7*. MIT Press, 1995, pp. 625–632.

[10] E. J. Palomo and E. López-Rubio, "The growing hierarchical neural gas self-organizing neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1–10, 2016.

[11] P. M. Yanik, J. Manganelli, J. Merino, A. L. Threatt, J. O. Brooks, K. E. Green, and I. D. Walker, "A Gesture Learning Interface for Simulated Robot Path Shaping With a Human Teacher," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 1, pp. 41–54, feb 2014.

[12] R. M. Luque-Baena, E. López-Rubio, E. Domínguez, E. J. Palomo, and J. M. Jerez, "A self-organizing map to improve vehicle detection in flow monitoring systems," *Soft Computing*, vol. 19, no. 9, pp. 2499–2509, 2015.

[13] R. M. Luque-Baena, J. M. Ortiz-de Lazcano-Lobato, E. López-Rubio, E. Domínguez, and E. J. Palomo, "A competitive neural network for multiple object tracking in video sequence analysis," *Neural Processing Letters*, vol. 37, no. 1, pp. 47–67, 2013.

[14] G. Chen, P.-L. St-Charles, W. Bouachir, G.-A. Bilodeau, and R. Bergevin, "Reproducible evaluation of pan-tilt-zoom tracking," in *Image Processing (ICIP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2055–2059.

Figure 1. Graphical description of the operation of the proposed method. From left to right, the columns show a frame of a sequence, the binary mask produced by the nonpanoramic foreground detection method, the state of the neural network model in that moment (red circles to show the neurons and blue line segments to represent the connections among the neurons), the selected component (i.e. the largest connected component) and its centroid (represented with a green asterisk). The first and second rows show frame 78 of scenario3, each row with a different GNG initialization in the first frame.

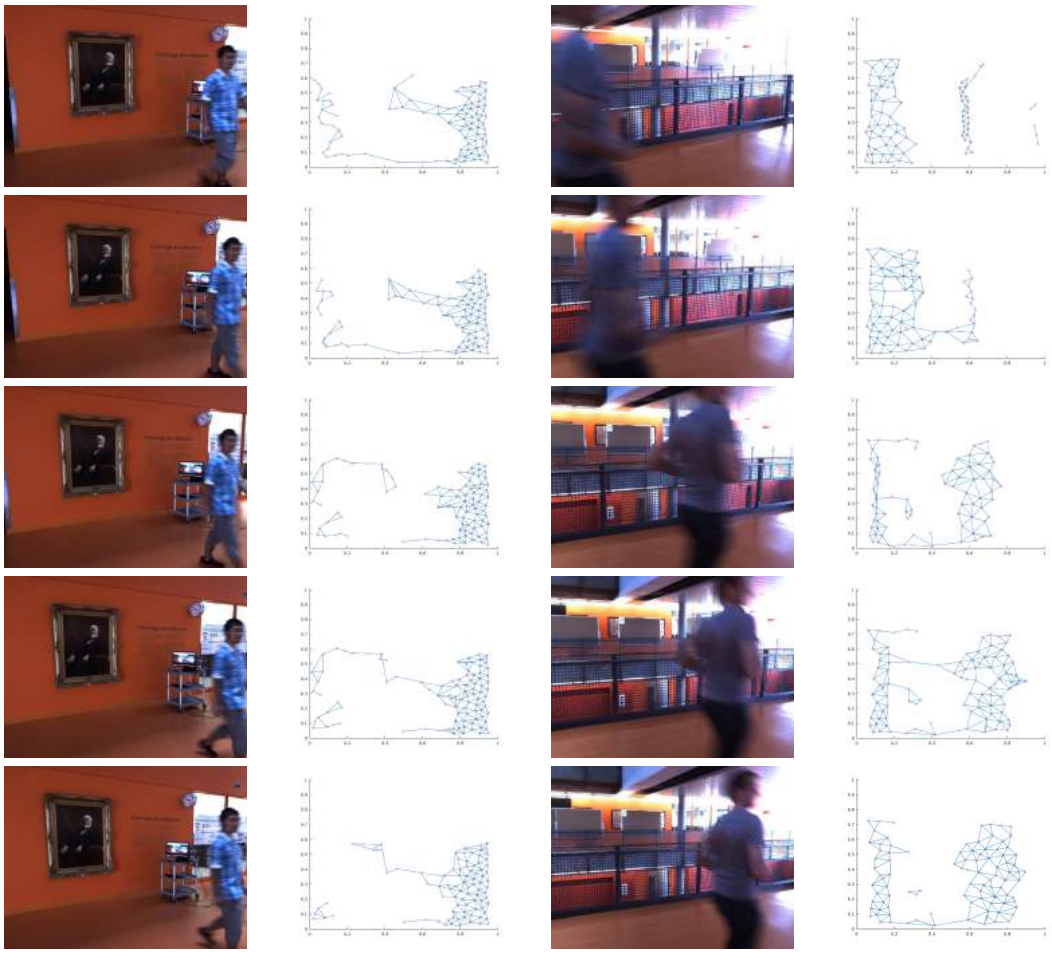


Figure 2. Graphical evolution of the GNG. First and second columns correspond to the frames 176 to 180 of the video scenario3 and its corresponding GNG state. Third and fourth columns correspond to the frames 506 to 510 of the video scenario3 and its corresponding GNG state.

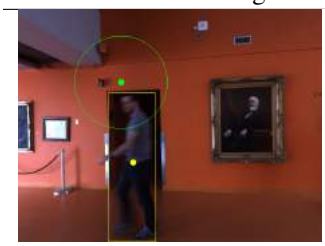
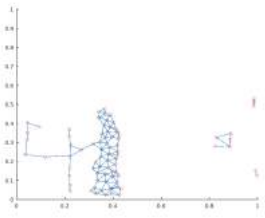
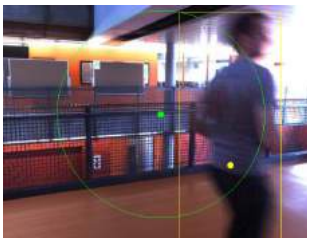
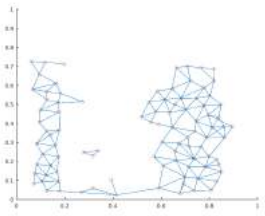
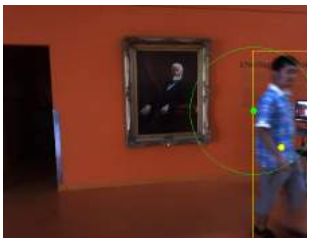
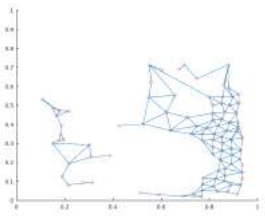
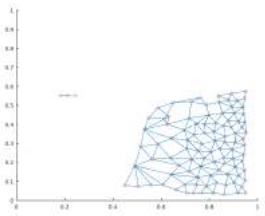
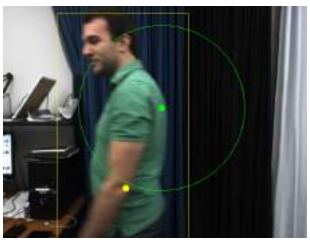
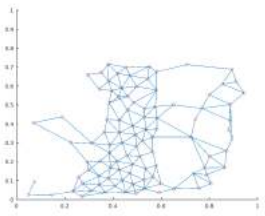
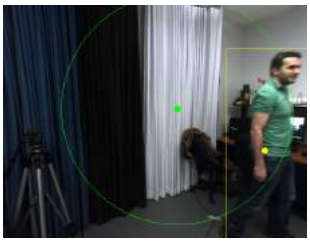
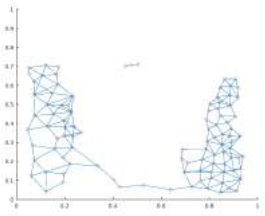
Frame with tracking	GNG	GT decision	Controller decision
		Left move No vertical move Zoom in move	Left move Down move Zoom in move
		Right move Down move Zoom out move	No horizontal move No vertical move Zoom out move
		Right move No vertical move No zoom move	Right move No vertical move No zoom move
		Right move Down move No zoom move	Right move No vertical move No zoom move
		Left move Down move Zoom out move	No horizontal move No vertical move No zoom move
		Right move No vertical move No zoom move	No horizontal move No vertical move Zoom out move

Figure 3. Results for some benchmark frames. First column shows a frame with the centroid and the bounding box of the ground truth target, both coloured in yellow, and the centroid and the size of the target indicated by the control module, in green. Second column represents the state of the GNG at these moment. The last two columns reports the directions given by the controller to the ground truth target and the target detected by the GNG, respectively. First and second rows are corresponding to the frames 58 and 510 of the video scenario3, and the third and fourth rows represent the frames 171 and 192 of the same sequence with a different initialization and ground truth target. The fifth and sixth rows show the frames 165 and 249 of the sequence scenario5.