PEOPLE COUNTING USING VISIBLE AND INFRARED IMAGES

A PREPRINT

Martín Biagini[1], Joaquín Filipic[1], Ignacio Mas[2], Claudio D. Pose[3], Juan I. Giribet[3,4,5], and Daniel R. Parisi[6]

[1]Dto. Ing. Informática, Instituto Tecnológico de Buenos Aires (ITBA), Lavardén 315, C. A. de Buenos Aires, Argentina
[2]Dto. de Matemática, Instituto Tecnológico de Buenos Aires (ITBA), CONICET, Av. Madero 399, C. A. de Buenos Aires, Argentina
[3]Dto. Ing. Electrónica, Facultad de Ingeniería, Universidad de Buenos Aires, Av. Paseo Colón 850, C. A. de Buenos Aires, Argentina
[4]Dto. de Matemática, Facultad de Ingeniería, Universidad de Buenos Aires, Av. Paseo Colón 850, C. A. de Buenos Aires, Argentina
[5]Instituto Argentino de Matemática "Alberto Calderón" CONICET, Saaverda 15, C. A. de Buenos Aires, Argentina
[6]Dto. Ing. Informática, Instituto Tecnológico de Buenos Aires (ITBA), CONICET, Lavardén 315, C. A. de Buenos Aires, Argentina

October 2020

ABSTRACT

We propose the use of convolutional neural networks (CNN) for counting and positioning people in visible and infrared images. Our data set is made of semi-artificial images created from real photographs taken from a drone using a dual FLIR camera. We compare the performance between CNN's using 3 (RGB) and 4 (RGB+IR) channels, both under different lighting conditions. The 4-channel network responds better in all situations, particularly in cases of poor visible illumination that can be found in night scenarios. The proposed methodology could be applied to real situations when an extensive databank of 4-channel images will be available.

1 Introduction

Crowd analysis is a labor of known relevance and interest at a global level. It is necessary for its various applications: from the social and political perspective, it is important to obtain data and draw conclusions from demonstrations; from the safety point of view, an accurate calculation of densities can facilitate the design of emergency exits and evacuation methods; for informational or statistical purposes, it is a basic task but does not relegate utility.

There are several methods to process images and calculate the total amount of people present [1, 2, 3]. In particular, the use of AI for this work is already widely known, and is addressed by various investigations starting from data sets of crowds images and using convolutional neural networks (CNN) that obtain favorable results [4, 5, 6]. In [7] accurate density maps are obtained as output, commenting that the difficulty lies in particularly dense congregations. However, this study is limited by the simple fact of relying on the visible information provided by the images, so an extra complexity is added when considering dark environments, where the light is scarce and irregular, or where monochromatic lights are dominant (due to external light sources).

The need to add extra information that helps the network to individually discriminate the position of each person within a crowd comes into play when the visual conditions are not optimal. Starting from [8], in which a method is presented to obtain said positions using a dual-camera (visible + infrared) mounted on an Unmanned Aerial Vehicle (UAV), this study continues and presents the novelty of using a special type of a CNN (U-Net CNN) to evaluate images with their data distributed in 4 channels, 3 for the visible spectrum (RGB) and one for the infrared. The latter, by providing thermal information, will allow a more accurate analysis in these scenarios.

Following the research line of previous works such as [9], which takes advantage of the combination of visual and infrared images, and [10], which performs crowd density estimations on visible images implementing a similar network architecture, this work compares the precision of a neural network that considers only the RGB channels, against one that considers RGB+infrared (from now on, the first network will be referenced as CNN_3 and the second network as CNN_4).

Considering that currently a dataset big enough to train and test the proposed networks does not exist, we computationally generate one using as base a few zenithal images taken at different heights, of a group of2around 20 people arranged at different densities.

These images were taken with a dual FLIR camera mounted on an unmanned aerial vehicle. In this sense, our data set is semi-artificial, because the individual profiles and the backgrounds come from real 4-channel pictures.

2 Methods

2.1 Semi-artificial Data

Naturally, both the CNN_3 and the CNN_4 will use the same synthetic data sets, which, as per the standards in machine learning, it will be divided into the training set, the validation set and the testing set, being the latter a separate subset from the other two.

For the training phase, the data set containing generated images will be divided in the following way: 90% of the imaged will be used for training, and the remaining 10% for validation, each set picked randomly from the pool of images. On the other hand, the testing set has independent generated images, on which different metrics will be analyzed and conclusions will be drawn from their results.

Before detailing the composition of each data set, it is necessary to explain how each image was obtained. Real photographs were taken by a FLIR DUO PRO R 640x512 camera attached to a multirotor-type, unmanned aerial vehicle. Specifications of the camera and the drone can be found in [8].

These photographs were taken at altitudes of 15 m and 30 m of small congregations of people distributed in different ways in an open field during the day, with full visible light. Each of these images is composed of data from four channels, the RGB visible spectrum and and infrared component. Therefore, Using this images as a base, the following method is used to generate the final images to be used by the networks. First, from the photographs, cut-outs of people and backgrounds are taken to use as templates (considering both visible and infrared information). Then, we simulate different crowd configurations (number of people and positions) by performing simulations with the Social Force Model [11]. Different number of simulated pedestrians were randomly generated inside the area, which were then moved toward different targets. Positions during the evolution of trajectories were registered, and different configurations and densities were obtained.

Using these positions, we overprint the cut-outs of the people from the real photographs over a given background. Backgrounds were also taken from real photographs and artificially generated by noise. After that, and in order to erase border artifacts of cut-out images we apply a blurring function and add random noise. Blurring is implemented using a Gaussian function (with a kernel of side $k = 7$ and a standard deviation $\sigma = 1$). The noise added to every image is achieved with a percentage of randomness in each pixel, using as a range $[I_p\ (1 - \beta), I_p\ (1 + \beta)]$, where I_p is the pixel value and $\beta = 0.1$.

The obtained images used in the data sets differ from each other in terms of a series of configurable variables (including the positions of agents):

- The attenuation factor for the intensity of the RGB components ζ_{RGB} (the intensity is the pixel value of each channel, an integer within the range of [0, 255]).
- The attenuation factor for the intensity of the IR component ζ_{IR}
- The type of background B (different visual environments and ambient temperature).
- The distance A that would represent the altitude at which the photo was taken. Associated with this variable is the amount of people present in the image (depending on the altitude, there is certain amount within a range).
- The people distribution D (their positions in the shown area).
- The composition of the background IR channel T (backgrounds are self-generated with pixel values that correspond to certain temperatures, within a range). This variable is considered only for the formation of the testing set, not for the training set.

Every image has its 4 channels, with a resolution of 1024×768 pixels. The four channels for the full image are processed by the CNNs (in the case of the CNN_3, the last channels is ignored).

Examples of real photographs with different people densities and generated images of different altitude and amount of people can be seen in Fig. 1.

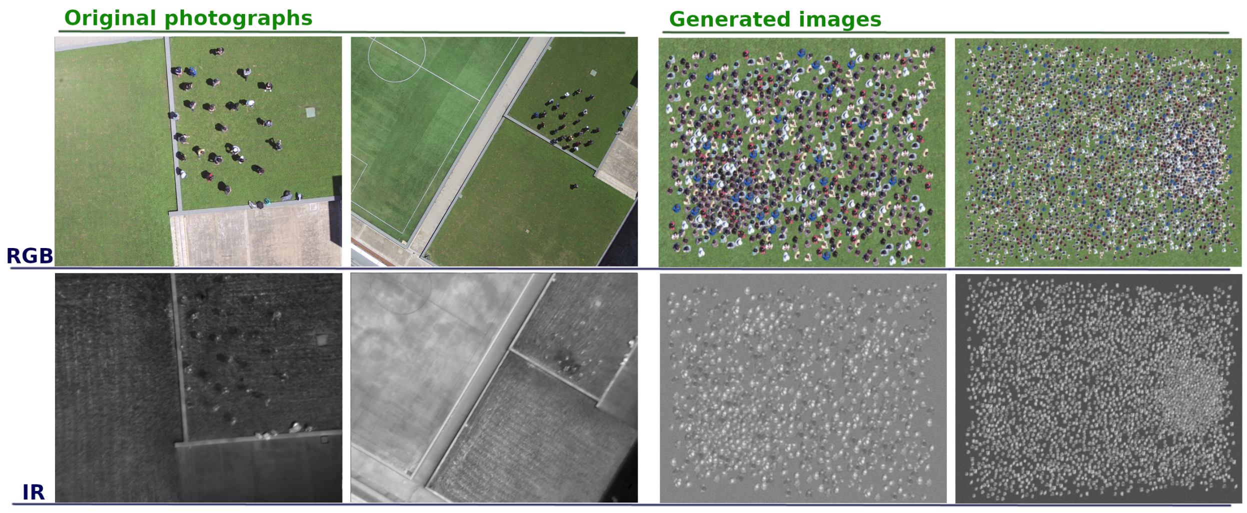

Figure 1: On the left, examples of real photographs taken at 15 m and 30 m altitude, with their corresponding IR channels. On the right, examples of generated images at the same altitudes using the cut-outs from the first ones.

With that being said, on one hand we have the data set for the training and the validation phase of the CNNs. Eq. 1 shows the combinations of the possible values for each variable involved in the generation of this set. Taking into account all combinations between them, we have 1530 images.

The value of the people distribution simply indicates that 5 different instants of the simulation (hence, the people positioned in different places on the ground) were considered for each of the other variable's combinations. For the latter, Table 1 shows their possible values. A few clarifications are made: firstly, cases annotated (*) are repeated twice, and the case annotated with (**) thrice (giving 17); secondly, the total number of people in each image, associated with the altitude, is chosen randomly from the specified range, with a uniform distribution of probabilities

$$S^{tr} = \zeta^{tr}_{RGB} * \zeta^{tr}_{IR} * B^{tr} * A^{tr} * D^{tr} = 17 * 3 * 3 * 2 * 5 = 1530 \tag{1}$$

ζ^{tr}_R	ζ^{tr}_G	ζ^{tr}_B		ζ^{tr}_{IR}	B^{tr}	A^{tr}
1	0	0		0.6	Cement	30 m/[400,2800]
0	1	0		1	Grass	15 m/[100,700]
0	0	1		1.2	Synthetic Grass	
0.01	0.01	0.01	*			
0.05	0.05	0.05	**			
0.2	0.2	0.2				
0.4	0.4	0.4				
0.6	0.6	0.6				
0.8	0.8	0.8				
1	1	1	*			

Table 1: Possible values for each variable of the training and validation set S^{tr}.

On the other hand, as mentioned previously, independent images were used to carry out the prediction tests, obtain results and draw conclusions, once the networks were trained. This testing set is formed by combining the possible values of the variables indicated in Eq. 2, obtaining 3456 images. In the same way as before, table 2 shows the possible values for these variables.

Later on, in Sec. 3, the 6 cases divided by the combination of the RGB attenuation factor ζ will be treated separately. As previously said, the reason for an alternate set is to avoid making a prediction with an image that was previously used to update the weights of the networks during the training phase.

$$S^{ts} = \zeta^{ts}_{RGB} * \zeta^{ts}_{IR} * B^{ts} * T^{ts} * A^{ts} * D^{ts} = 6 * 3 * 3 * 4 * 2 * 8 = 3456 \tag{2}$$

ζ^{ts}_{R}	ζ^{ts}_{G}	ζ^{ts}_{B}	ζ^{ts}_{IR}	B^{ts}	T^{ts}	A^{ts}
1	0	0	0.6	Cement	Real/-	30 m/[400,2800]
0.01	0.01	0.01	1	Grass	Generated/[15 °C,20 °C]	15 m/[100,700]
0.05	0.05	0.05	1.2	Synthetic Grass	Generated/[30 °C,35 °C]	
0.1	0.1	0.1			Generated/[45 °C,50 °C]	
0.2	0.2	0.2				
1	1	1				

Table 2: Possible values for each variable of the testing set S^{ts}.

2.2 Neural Network Model

The implementation carried out consists mainly of the convolutional neural network, which is fed by the images of the data set to be trained first and perform the evaluations later, and a series of functions to transform its output and obtain the desired information.

First of all, it is necessary to have the exact positions of the people present in each of the images to be used as input. As these images are obtained from the previously described method, we can count on these positions throughout the simulations, since they are recorded at all times. So, for each image, we have in a separate file (called "ground-truth"), the position of each person, represented in a matrix of integers as a coordinate map, with value 0 where there are no people and value 100 in the center of each of them. Based on this, the CNN, after processing the information of each image, will be able to compare the predictions of the positions with the actual positions.

Considering the neural network, it was trained using the incremental methodology in each epoch, and its architecture is based on an evolved design of normal convolutional neural networks: the U-Net. The use of this approach was chosen for two main reasons. The first one is that it is aimed both at being able to classify the patterns recognized in the inputs and at being able to locate them spatially. The second, thanks to the series of functions that are applied internally, the size of the training data set can be considerably smaller than in the case of other neural network.

In particular, the implementation used here is based on the article [12] and the code found in [13], consisting of repeated applications of two 3×3 convolutions, each followed by a rectified linear unit ReLU ($f(x) = max(0, x)$, with x being the input of the neuron) and a 2×2 max pooling operation with stride 2 for the downsampling phase. For the upsampling phase, successive techniques are also performed to return the image to its original dimension using a 2×2 transposed convolution, a concatenation with the corresponding image in the downsampling phase (to combine the information of previous phases) and two 3×3 convolutions followed by ReLU.

The differences between this implementation and the original model described in the aforementioned work are the application of a batch normalization of the values of each map between the convolutions and the rectification in the downsampling phase, the use of 3 downsampling/upsampling stages instead of 4 and the inclusion of the 4th channel in the input images. This process is summarized in the Fig. 2.

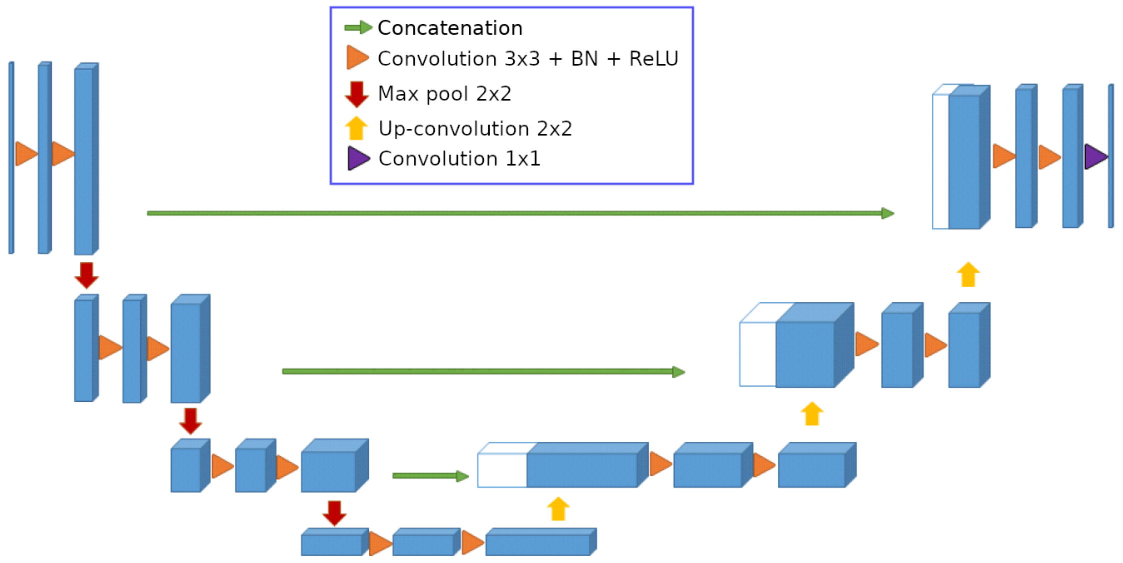

Figure 2: U-Net architecture. First block (top left) is the input image and last block (top right) is the output of the network.

Once established the architecture for the CNN, it remains to define its parameters, which will be the following: Nesterov Momentum $\alpha = 0.9$, learning rate $\eta = 0.02$, weight decay $w = 0$, $\gamma = 0.1$ and step size $s = 10$, the latter being the step of epochs through which η is updated, multiplied by the factor γ.

The CNN will output a matrix of integers, where the predicted positions of the people are indicated, although further processing is necessary. This output matrix has the same dimension as the resolution of the images that are used as input (which are transformed to similar matrices when feeding the network, but with an additional dimension that contains the information of the RGB and IR channels), that is, 1024×768, for the case analyzed.

The additional processing required for the CNN output is to go through the elements of the matrix, each one corresponding to one pixel of the original image. Those which show a value of zero will correspond to a pixel in the image where no people are present, and those with a value different from zero will correspond to pixels where a person, or part of a person, is found. As one person may be seen in the image as a group of pixels, it is necessary to check the neighboring elements every time a value other than zero is found. In those cases cases, if several neighboring elements shown a value different than zero, their values will be added and stored in the element with the original highest value. This unification process can be seen in Fig. 3.

$$\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 23 & 23 \\ \hline 0 & 25 & 21 \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 92 & 0 \\ \hline \end{array}$$

Figure 3: Looping through every element of the network output map (left), neighbours with values higher than 0 must be unified to obtain a final map indicating the predicted positions of the people (right).

Summarizing, the CNN will provide as output the positions of all pedestrians present in the input image, represented in matrix form, where the non-zero elements correspond to the pixels in the original image where the centroid of an individual is located

It is important to mention that, during the training of the network, the unification process does not take place: to determine the completeness of the training, at the end of each epoch a comparison value is obtained to determine its level of precision. This comparison is made on the images conforming the validation set (differentiating it from the training set, which is used specifically to update the weights of the network). It is calculated as expressed by Eq. 3, which is the same function used by the CNN as its loss function.

$$P_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)^2 \tag{3}$$

where $\mathbf{X}$ is the ground-truth matrix, $\mathbf{Y}$ the network output matrix and n the number of elements in each of them. This is the mean squared error taken between each element of the the ground-truth matrix (the one that provides the real people positions) and the matrix of the output map of the networks (with no further processing, as shown in the left panel on Fig. 3).

2.3 Metrics

First, while training the network, the best combination of the connection weights will be recorded. Each epoch is compared with the previous one, keeping the one with least P_{MSE}, as described before in Eq. 3. The cut-off criteria is the passage of 30 epochs without registering a minor error.

Prior to calculating network efficiency with the independent testing set, it is necessary to determine the threshold value μ that must be exceeded to be able to discern whether or not a person is detected in a predicted position on the final matrix (after the unification process). This value is found empirically: for different threshold values, the mean absolute error is calculated between the predicted number of people and the real number of people for each image in the training and validation sets, as noted in Eq. 4. The optimal μ corresponds to the minimum calculated error.

$$N_{MAE} = \frac{1}{m}\sum_{i=1}^{m}|n_{pi} - n_{ri}| \tag{4}$$

where n_p is the predicted number of people (which depends on μ), n_r the real amount and m the number of images in the set.

Finally, for the analysis of results, there are 2 metrics that refer to the precision of the network processing a single image. The first one is the relative error of the number of predicted people versus the real number, which is show in Eq. 5

$$E_{num} = \frac{1}{n_r}|n_p - n_r|. \tag{5}$$

The second metric is the average error of the distance between the predicted position and the real one (in cm), for each position that has its real pair (that is, excluding people who were not counted and those who were ”added”). It is shown in Eq. 6

$$E_{pos} = (\frac{1}{p}\sum_{j=1}^{p}|\mathbf{X}_j - \mathbf{Y}_j^*|) * \psi \tag{6}$$

where p is the number of pairs, $\mathbf{Y}^*$ and $\mathbf{X}$ are the coordinates of the predicted and real positions respectively, and ψ the conversion factor implying the distance per pixel given the altitude of the image, according to the results in [8].

3 Results

3.1 Training and Validation

Since the main objective of this work is to compare the performance of the CNN_3 against the CNN_4, it is taken into account that all measurements were carried out against both of them. The results of the training phase are shown in Table 3. The number of epochs run by each CNN shows the amount of epochs needed to find the lowest P_{MSE}.

CNN	Epochs run	P_{MSE}
CNN_3	29	12.61
CNN_4	19	11.51

Table 3: Training results comparison between CNN_3 and CNN_4: number of epochs run by each network and lower P_{MSE} found on the images of the validation set.

This results shows that the CNN_4 needed fewer epochs to obtain a smaller error and perform better at this stage than the CNN_3.

Then, to determine the optimal threshold μ to be used as a criterion in the detection of a person given the output map, the entire training + validation set was run with the CNN_4 and the N_{MAE} was calculated, using different μ (from 10 to 90, with step 5). The results can be observed in the Fig. 4, and the best value for μ is found to be 30 (being $N_{MAE} = 6\ 10^{-4}$). Also, it can be seen that the selection of μ is robust considering that a wide range can be used to perform the required discrimination.

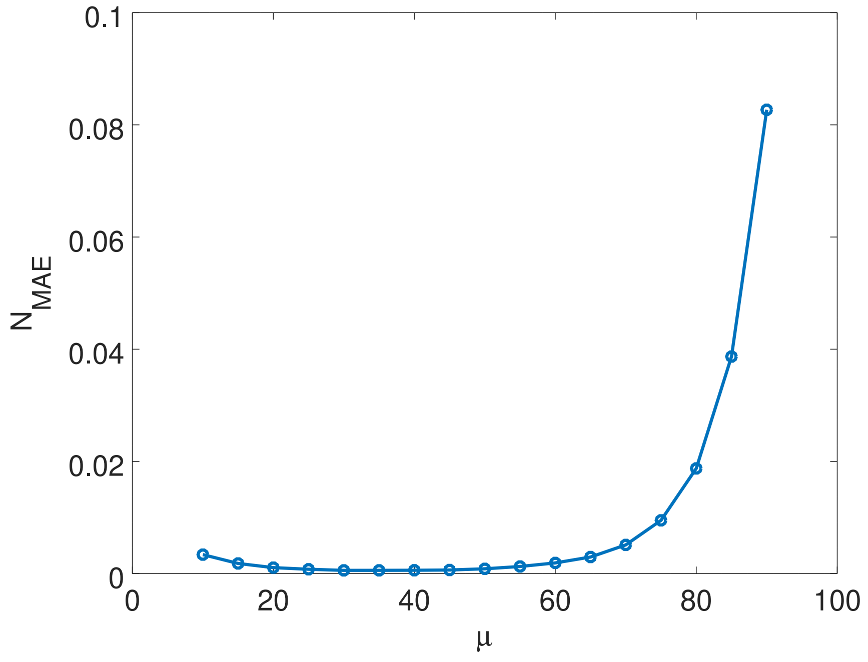

Figure 4: Calculated N_{MAE} of the entire training + validation set S^{tr} for different values of the threshold μ, using the CNN_4.

3.2 Testing

Being able to measure the errors in the predictions of the images of the testing set, using the two metrics described in Sec. 2.3, E_{num} and E_{pos}, and the threshold $\mu = 30$, the testing set is divided into the six cases mentioned in Sec. 2.1 corresponding to the six values of first column of Table 2. Basically, this division is made from the amount of light present in the images, represented by the attenuation factor ζ for the R, G and B channels. For a given value in the first column of Table 2, the rest of the variables are considered in order to have 576 images for each case.

In this way, case 1 is composed of monochromatic images ($\zeta_R = 1$, $\zeta_G = 0$, $\zeta_B = 0$), and from case 2 to case 6 they are composed of images with incremental values of the amount of light: respectively, $\zeta_{RGB} = 0.01$, $\zeta_{RGB} = 0.05$, $\zeta_{RGB} = 0.1$, $\zeta_{RGB} = 0.2$ and $\zeta_{RGB} = 1$ for each consecutive case.

Measurements were taken using the CNN_3 and the CNN_4, seeing the corresponding results on Table 4 and Table 5 in which the values of all the images in each case are averaged. For compute E_{pos} the correspondence conversion factor between pixel and cm was calculated taking into account the distance that represents the altitude of each image.

These factors are 1.25 cm/pixel for height 15 m and 2.5 cm/pixel for height 30 m, based on our previous work with the same acquisition system [8].

	E_{num}		$E_{pos}(cm)$	
Case	Mean	Std. Deviation	Mean	Std. Deviation
1	$1\ 10^{-3}$	$2\ 10^{-3}$	1.1	0.4
2	$2\ 10^{1}$	$2\ 10^{1}$	-	-
3	$1\ 10^{-2}$	$2\ 10^{-2}$	2.1	1.1
4	$1\ 10^{-3}$	$1\ 10^{-3}$	1.0	0.4
5	$2\ 10^{-4}$	$9\ 10^{-4}$	1.0	0.4
6	$2\ 10^{-4}$	$8\ 10^{-4}$	1.0	0.3

Table 4: Prediction results of the CNN_3: mean and standard deviation of E_{num} and E_{pos} for images in each case of the testing set S^{ts} corresponding to the first column of Table 2.

	E_{num}		$E_{pos}(cm)$	
Case	Mean	Std. Deviation	Mean	Std. Deviation
1	$1\ 10^{-3}$	$5\ 10^{-3}$	1.0	0.4
2	$1\ 10^{-2}$	$3\ 10^{-2}$	2.1	1
3	$1\ 10^{-3}$	$4\ 10^{-3}$	1.1	0.4
4	$1\ 10^{-3}$	$1\ 10^{-3}$	1.0	0.4
5	$1\ 10^{-4}$	$5\ 10^{-4}$	1.0	0.4
6	$2\ 10^{-4}$	$6\ 10^{-4}$	1.0	0.4

Table 5: Prediction results of the CNN_4: mean and standard deviation of E_{num} and E_{pos} for images in each case of the testing set S^{ts} corresponding to the first column of Table 2.

Comparing the results of case 1, it can be observed that the two CNNs show similar values in both metrics. Given that this case is considering monochromatic images, this behavior was as expected, because the information provided by this images can be exploited by both CNNs equally.

The greatest difference is observed, clearly and as expected, between the metrics of case 2, corresponding to the dark images. The CNN_3 is unable to predict decent results and throws out of bounds errors for E_{num}. Regarding E_{pos}, the fact that the error in the people amount is so high prevents us from associating the predicted people with the real ones to calculate the error in their distances.

Logically, it is in these scenarios that the advantage to use networks trained with the infrared channel is evident, making an almost exclusive use of it to process the images and make acceptable predictions.

Furthermore, there is a tendency to reduce the error (both in the number of people and in their positions) when comparing the subsequent cases between the two networks (increasing the amount of light in the images). So, the network using four channels CNN_4 does produce better predictions at a general level.

In order to confirm this, we also calculated the relative error in the number of people E_{num} changing the attenuation factor ζ_{RGB}. This calculation involves a larger testing set S_2^{ts}, composed of more number of cases (in addition to the cases of S^{ts}). The comparison between both networks can be seen in Fig. 5, showing a clear advantage of the CNN_4 over the CNN_3, that is wider for dark images but also better for all the range of the studied visible attenuation factor.

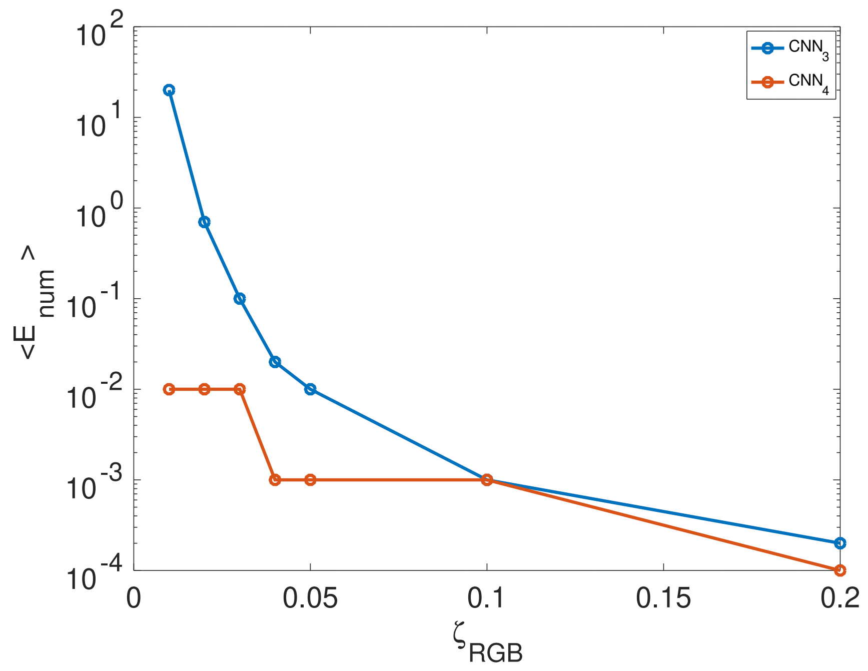

Figure 5: E_{num} mean for each case of the larger testing set S_2^{ts}, depending on ζ_{RGB}.

In order to illustrate the performance of the CNN_4, we chose a random image of the testing data set and displayed the predicted positions in Fig. 6, where it can be seen the precision of the neural network counting and positioning people from an image.

In general, the results presented show high precision when counting and positioning people in the semi-artificial images. Even this is a biased effect produced by the synthetic images it is useful in order to compared the general performance between using three or four channel data.

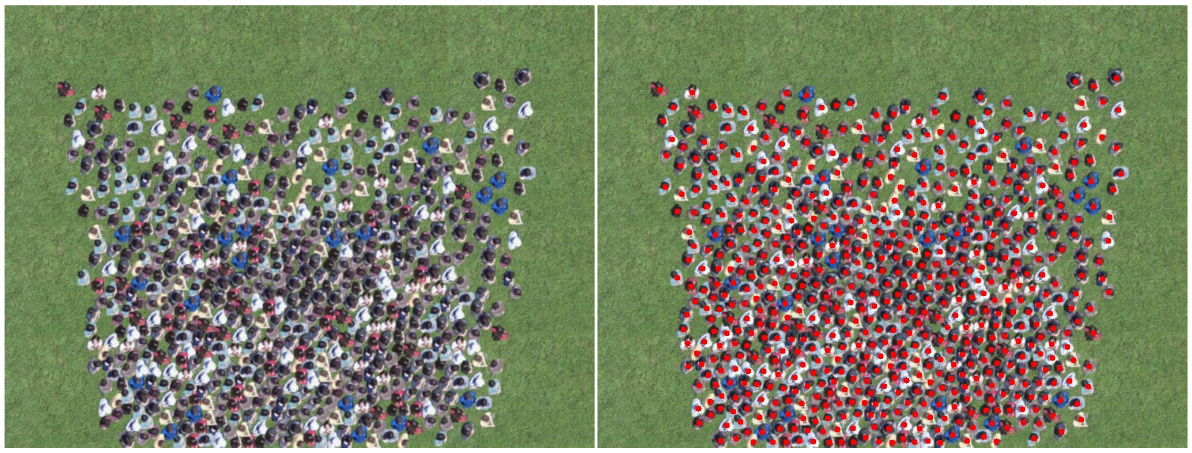

Figure 6: On the left, a random image of the testing set S^{ts} with $\zeta_{RGB} = 1$, grass background and 15 m altitude. On the right, the predicted positions of the people on top of the same image, using the CNN_4.

4 Conclusions

Two convolutional neural networks were created and trained, one taking 3 channels images (RGB) as input and the other 4 channel images (RGB + IR), which can count and provide the positions of people in aerial images.

The performances of these networks were tested with zenith semi-artificial images of crowds at different altitudes and under different lighting conditions, in addition to combining other variables.

It can be clearly observed that in general, for the studied cases, the use of the fourth channel (IR) improves the precision in counting and positioning people in the images. This improvement is much more evident when the intensity of visible light is low.

As a next step, we will collect a big set of real 4-channel images that will allow extending the proposed methodology to real scenarios.

References

[1] David Ryan, Simon Denman, Clinton Fookes, and Sridha Sridharan. Crowd counting using multiple local features. *Digital Image Computing : Techniques and Applications (DICTA)*, pages 81–88, 2009.

[2] Ya-Li Hou and Grantham K. H. Pang. People counting and human detection in a challenging situation. *IEEE Transactions On Systems, Man, And Cybernetics Part A:Systems And Humans*, 41(1):24–33, 2011.

[3] Ke Chen, Chen Change Loy, Shaogang Gong, and Tao Xiang. Feature mining for localised crowd counting. *BMVC*, 2012.

[4] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. *CVPR*, 2015.

[5] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. *CVPR*, 2016.

[6] Viresh Ranjan, Hieu Le, and Minh Hoai. Iterative crowd counting. *ECCV*, 2018.

[7] Lokesh Boominathan, Srinivas S S Kruthiventi, and R. Venkatesh Babu. Crowdnet: A deep convolutional network for dense crowd counting. *MM '16: Proceedings of the 24th ACM international conference on Multimedia*, pages 640–644, 2016.

[8] Daniel R. Parisi, Juan I. Giribet, Claudio D. Pose, and Ignacio A. Mas. Set-up of a method for people-counting using images from a uav. *Traffic and Granular Flow*, 2019.

[9] Imran Amin, A.J. Taylor, Faraz Junejo, Amin Al-Habaibeh, and Robert Parkin. Automated people-counting by using low-resolution infrared and visual cameras. *Measurement*, 41:589–599, 2008.

[10] Van Su Huynh, Vu Hoang Tran, and Ching Chun Huang. Iuml: Inception u-net based multi-task learning for density level classification and crowd density estimation. *IEEE International Conference on Systems, Man and Cybernetics, SMC*, pages 3019–3024, 2019.

[11] Dirk Helbing, Illés Farkas, and Tamás Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention*, 9351:234–241, 2015.

[13] NeuroSYS. Objects counting by estimating a density map with convolutional neural networks. `https://github.com/NeuroSYS-pl/objects_counting_dmap`.