Combined Weightless Neural Network FPGA Architecture for Deforestation Surveillance and Visual Navigation of UAVs

Vitor A.M.F. Torres[a], Brayan R.A. Jaimes[a], Eduardo S. Ribeiro[a,b], Mateus T. Braga[a], Elcio H. Shiguemori[c], Haroldo F. C. Velho[d], Luiz C.B. Torres[a,b], Antonio P. Braga[a]

[a]*Federal University of Minas Gerais, Department of Electronics Engineering, Brazil*
[b]*Federal University of Ouro Preto, Department of Computing and Systems, Brazil*
[c]*Institute of Advanced Studies - IEAv, Ministry of Defense, Brazil*
[d]*National Institute of Space Research - INPE, Brazil*

Abstract

This work presents a combined weightless neural network architecture for deforestation surveillance and visual navigation of Unmanned Aerial Vehicles (UAVs). Binary images, which are required for position estimation and UAV navigation, are provided by the deforestation surveillance circuit. Learned models are evaluated in a real UAV flight over a green countryside area, while deforestation surveillance is assessed with an Amazon forest benchmarking image data. Small utilization percentage of Field Programmable Gate Arrays (FPGAs) allows for a higher degree of parallelization and block processing of larger regions of input images.

Keywords: Classification, Weightless Neural Network, Artificial Neural Networks, UAVs

1. Introduction

Weightless Neural Systems (WNS) [1, 2, 3] are an attractive solution for implementing in-circuit pattern recognition systems [4, 5], since Boolean functions represented by look-up tables can be directly implemented in hardware with RAMs (Random Access Memories). Although WNS are limited to binary inputs, many pattern recognition problems are inherently Boolean or can be represented directly in the Boolean domain without relevant loss of representativeness. Actually, image recognition problems often require binarization prior to recognition, particularly those that consider images after they are processed by edge filters, which is the case of the image navigation problems discussed in the next sections.

WNS are applied in this paper to implement embedded hardware of UAVs (Unmanned Aerial Vehicle) in two applications that require real-time and embedded processing, which aim at autonomous visual navigation and aerial deforestation surveillance of dense forest areas. Proposed architecture combines two different WNS in order to process the images, one that is aimed at deforestation surveillance and the other at UAV localization. The two problems are complimentary, since localization takes binary edge detected images as inputs, while deforestation surveillance actually generates binary images as outputs. The proposed architecture combines the two WNS and explores potential parallelization of embedded Field Programmable Gate Array (FPGA) implementation.

Although Global Navigation Satellite System (GNSS) and embedded systems, such as Inertial Navigation System (INS), are the main UAV navigation systems, signal quality depends on weather conditions and may fail or become

unavailable due to many sources of interference [6]. This is particularly important in the Amazon region, where ionosphere phenomena are quite common and GNSS signal can be degraded. The region can also be cloudy in many months of the year, which also interferes with GNSS and makes satellite monitoring more difficult to be accomplished. In the case of INS, especially the low cost ones, the correction of a complementary system such as GNSS is required, otherwise the error increases rapidly.

Visual navigation of small UAVs has become a viable alternative to sole GNSS and INS navigation due to recent advances in image recognition systems and lower cost of embedded hardware [7]. However, it should be considered, that, in general, such systems are based on small vehicles, which have load capacity limitations and require a careful selection of lightweight sensors and on-board computer. Another point to consider is the requirement for low-power consumption. It is, therefore, particularly important to search for alternatives using computer vision and pattern recognition systems that do not demand a high computational power and that are feasible to hardware implementation. Since feed-forward WNS can be directly implemented as combinational circuits, this work explores its use in both applications. Simplified Boolean functions, corresponding to edge detection filters and deforestation recognition, are possible due to the sparsity of look-up tables. Resulting combinational circuits for pixel-level processing occupies a very small percentage of the chosen FPGA, which facilitates parallelization and further massive implementation of WNS to process larger regions of input images. Deforestation detection is formulated as a binary image classification problem with WISARD (Wilkie, Stonham and Aleksander's Recognition

Device) Discriminators [8, 9, 10], while edge detector filters are implemented as RAM-based binary classifiers [4].

Convolutional neural networks have also been applied recently to aerial image classification with UAVs using other types of sensors, such as micro-Dopplers [11]. In another application, image detection of non-authorized UAV flights in wild and protected areas have also been accomplished with a deep learning approach [12]. The authors considered data augmentation with synthetic images and video sequences extracted from the Internet to overcome the lack of training data. Random Forests, SVMs (Support Vector Machines) and Decision Trees have been also applied to the classification of land-cover classes [13]. The work presented in this paper aims not only at the image classification problem, but also at the construction of a feasible embedded system with programmable hardware implementation of the learning and classification systems.

The paper is organized as follows: in Section II the Ram Discriminator is detailed. Section III presents the UAV position estimation through images. The methodology and results are presented in Section IV. At last, conclusions is presented in Section V.

2. RAM Discriminator

The WISARD RAM discriminator [4] shown in Fig. 1 is basically formed by a single layer of k n-inputs RAMs, which are connected to the N-dimensional input field according to a pre-established connection criteria. The n-tuples corresponding to each RAM address are read directly from each input field to which the address lines are connected. Since connections are usually

sparse ($n \ll N$) and randomly chosen from the input field, each RAM learns a (different) randomly selected Boolean function $h_j(\mathbf{x})$. Their outputs are then summed-up in order to produce discriminator's response, so the output of a RAM discriminator $r_i(\mathbf{x})$ to a given input pattern $\mathbf{x}$ can be generally represented as $r_i(\mathbf{x}) = \sum_{j=1}^{k} h_j(\mathbf{x})$, where $h_j(\mathbf{x})$ is the Boolean function performed by RAM neuron j.

Training a RAM discriminator is accomplished with a one-against-all strategy, so each RAM is trained only with examples of the class associated to the discriminator it belongs to. Recognition occurs by presenting the input pattern $\mathbf{x}$ to all discriminators, which respond with the number of matches with previously trained examples. Final classification is then accomplished with a Winner-Takes-All (WTA) strategy, that assigns to $\mathbf{x}$ the class of the discriminator with the largest response $r_i(\mathbf{x})$.

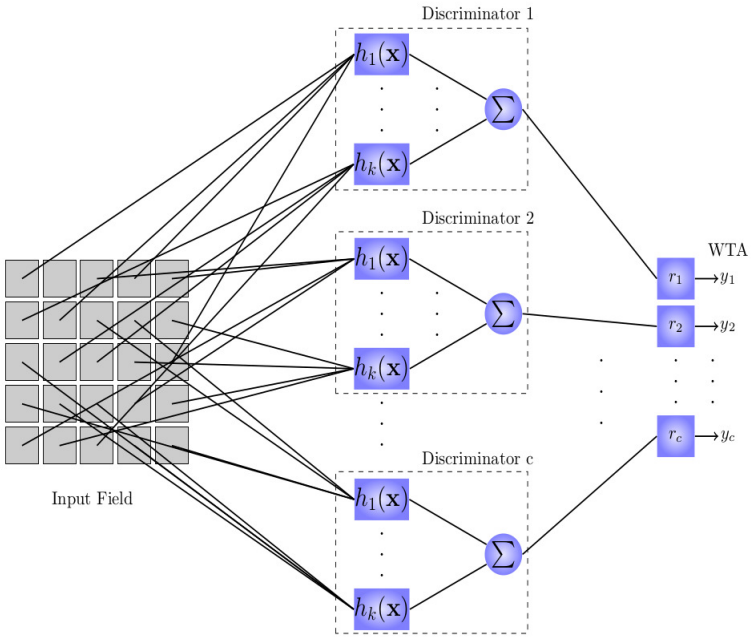


Figure 1: Schematic representation of a RAM discriminator.

3. UAV Position Estimation by Images

In recent years, UAVs have been considered in many applications, among them remote sensing, which requires real-time decision making by humans or computer systems during the flight. Visual image navigation, one of such applications, is addressed in this work. In order to treat the position estimation problem using in-flight images, different techniques have been employed, among them template matching [14], which uses a reference image of the area to be covered, obtained by satellites or previous flights. It has the advantage of being able to estimate the position in several points of the overflown location. It is most suitable when there are previous images of the region to be flown by the aircraft, which is the case of the problem treated

in this work. Other approaches consider on-earth landmark recognition for position estimation, which to not require to embed reference images of the flight area [15]. This approach, however, requires that representative landmarks are availble and sampled in advance in order to train the recognition system. A third approach, which can be employed when there are no previous images or landmarks available, is based on in-flight image odometry [16, 17] and SLAM (Simultaneous Localization and Mapping) [18, 19, 20]. The approach considers a reference image at the beggining of navigation and location estimation according to the differences between en-route sequential images. This approach suffers, however, with accumulated error estimation, mainly due to image resolution constraints and other sources of distortions due to sensor limitations and external interferences.

In this work, images captured during a UAV flight over the city of São Carlos, Brazil, were employed. A picture of the UAV used to capture the images with its specifications is presented in Figure 2. The captured images contain different soil coverings, among them, plantations, native vegetation, rivers, exposed soil and highways. Images have been selected in order to test techniques considering variations in texture, color and borders. Examples of images employed in the tests are illustrated in Figure 5. The conditions for capturing the images were clear sky and flight height at 120 m.

The goal was to estimate visual UAV position having as input in-flight captured images and a previously georeferenced image of the whole navigation area (a map). Position estimation is accomplished by locating the current image captured by the UAV, with a Template Matching [21, 22, 23, 7, 24] approach, in the geo-referenced image.

Figure 2: UAV used in the experiments. Specifications are: Model Echar, Manufacturer XMobots, Maximum Takeoff Weight is 7.8kg, Maximum Payload Weight is 0.9kg, Wingspam is 2.13 meters, Maximum Autonomy is between 120 and 180 minutes and Cruise Speed is 32 Km/h.

3.1. The Template Matching Approach

Template Matching compares two images considering spatial correlation indices [14, 25]. For this, the UAV image moves over the georeferenced image by calculating the sum of the products for each position $(\mathbf{s}, \mathbf{t})$ as presented in Eq. (1). The term $c(s,t)$ denotes the correlation c at position (s,t), with $s = 0, 1, ..., M-1$ and $t = 0, 1, ..., N-1$; where M and N are the dimensions of the matrix f that represent the geo-referenced image. Matrix $\mathbf{W}$ with dimensions $J \times K$, and restrictions $J \leq M$ and $K \leq N$, corresponds to the UAV image.

$$c(\mathbf{s}, \mathbf{t}) = \sum_{x} \sum_{y} f(x,y)\mathbf{W}(\mathbf{x} - \mathbf{s}, \mathbf{y} - \mathbf{t}) \tag{1}$$

Thus, for every position $(\mathbf{s}, \mathbf{t})$ within $f(\mathbf{x}, \mathbf{y})$ is assigned a value $c(\mathbf{s}, \mathbf{t})$ generated from matchig with $\mathbf{W}(\mathbf{x} - \mathbf{s}, \mathbf{y} - \mathbf{t})$. The identification of the maximum value of $c(\mathbf{s}, \mathbf{t})$, indicates the UAV image position on the georeferenced image, as shown in Fig. 3.

Captured images may present distortions due to camera resolution, illu-

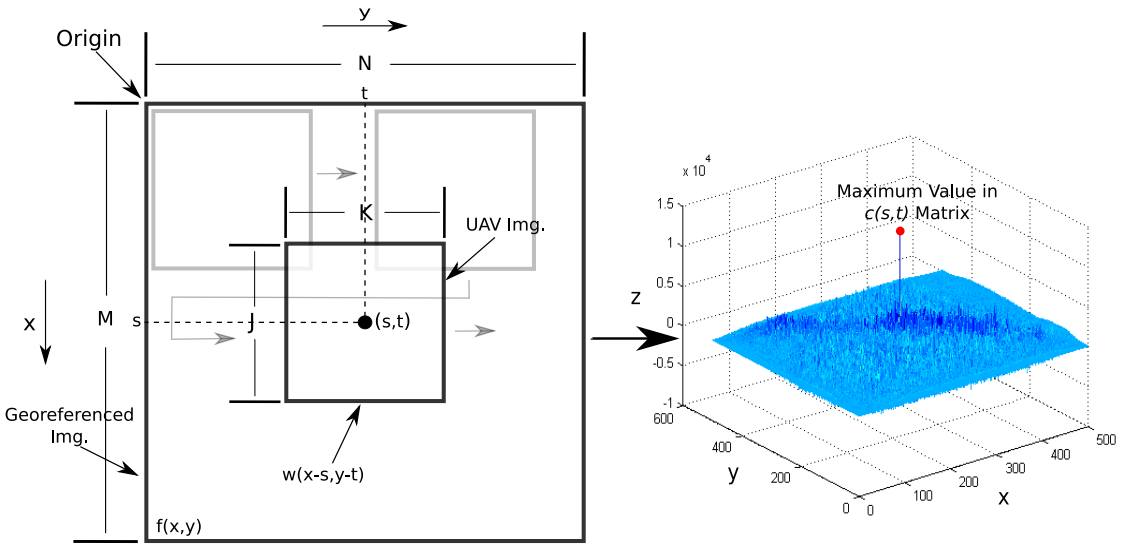


Figure 3: Calculation of correlation between $f(x, y)$ and $w(x, y)$.

mination and perspective deformations due to UAV axes of rotation, that may also affect position estimation [26]. Therefore, edge detection is used prior to Template Matching in order to reduce differences between images, which were captured at different time slots, under different conditions and with different image sensors.

Perspective correction considers UAV's axes angles (Yaw (ψ), Pitch (θ) and Roll (ϕ)) and the intrinsic camera parameters in order to obtain the homographic matrix $\mathbf{H}$ (Eq. 2), which is applied to the image in order to obtain the projection transformation. For rotation adjustment, only Yaw angle is considered [26].

$$\mathbf{H} = \mathbf{A}' \cdot (\mathbf{R} - t_p \mathbf{n}_p^T / d) \cdot \mathbf{A}^{-1} \tag{2}$$

where $\mathbf{A}$ is the calibration matrix of the camera (Eq. 3), $\mathbf{R}$ is the rotation matrix (Eq. 4), t is the position of the vision of the scene, $\mathbf{n}$ is a vector normal

to the plane of the scene, d is the distance from the point in the plane of the image to the point in the plane of the world, f is the focal length of the camera, k_u and k_v are the horizontal and vertical scale factors respectively, and u_o and v_o are the coordinates of the center point of the camera.

$$\mathbf{A} = \begin{bmatrix} f \cdot k_u & 0 & u_0 \\ 0 & f \cdot k_v & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$\mathbf{R}_{\psi,\theta,\phi} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{4}$$

Prior to template matching, edge detection is accomplished with a 3×3 kernel filter implemented with a single layer WNS, that learns from pre-established edge patterns [23]. For the possible binary edge patterns (3×3 kernel), there are 512 possible combinations to define the existence or not of edges that results in a LUT (Lookup Table) of 512×9, with each 1×9 binary vector representing a possible edge. In this experiment, 75 edge-filter patterns were defined and pre-loaded in the RAMs.

Figure 4 presents a general overview of the proposed methodology. Maximization of the spatial correlation between images, after perspective deformation correction and edges detected, is used to estimate position. In parallel with edge detection, deforestation detection via WNS is also performed. In the experiments that follow, Canny operator [27] was also used in order to

extract edges and compare the WNS results with another edge detection approach.

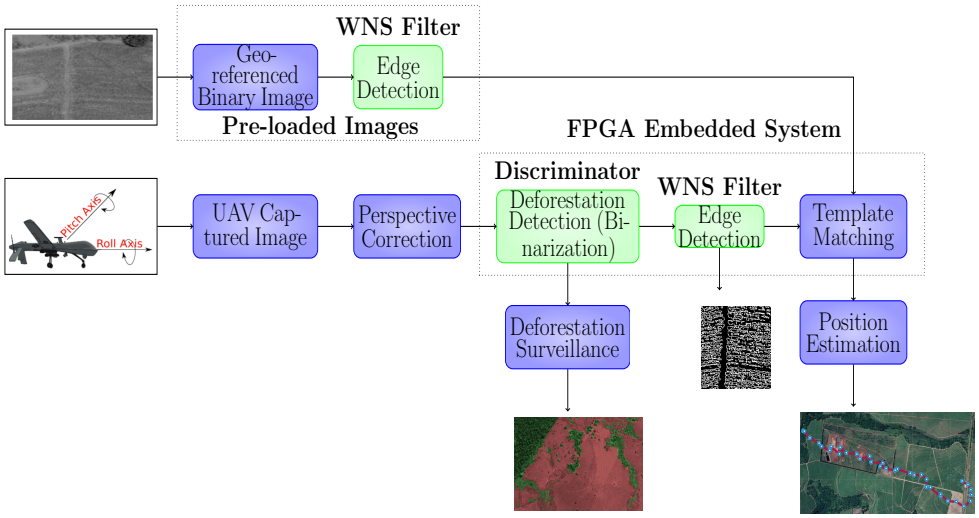


Figure 4: Schematic view of the proposed methodology for in-circuit deforestation surveillance and visual navigation of UAVs.

In order to test the WNS Network as a new edge detector and compare with other approaches proposed in the literature, a real flight was performed on a planned UAV route, and the images were collected. The measure of accuracy in the geographic position estimation with template matching is defined by the Euclidean distance between the position estimated by the proposed methodology and the actual position of the UAV in each frame. Distance error is defined in Eq. (5).

$$D = R_T \ \arccos\left(\cos(lat_1)\cos(lat_2)\cos(lon_1 - lon_2) + \sin(lat_1)\sin(lat_2)\right) \qquad (5)$$

where R_T is the mean radius of the Earth's circunference, lat_1 and lon_1 are the latitude and longitude of the real position, lat_2 and lon_2 correspond to the

latitude and longitude of the position estimated by the proposed methodology. D represents the separation distance in meters between the estimated position and the actual position.

Figure 5: Samples of UAV images collected.

The resolution of the georeferenced embedded image was 0.5 $m/pixel$ and size 9872×10386 $pixels$, so the covered flight area had aproximately 5×5 kms. The size of the original UAV in-flight captured images was reduced to approximately 13% of its original resolution resulting in 220×220 $pixels$. The flight for the experiments was accomplished in a countryside region with predominance of green vegetation, plantations and forest with river. Some of these images can be observed in Fig. 5. Images sampled from the flight video were used in visual navigation experiments that follow.

It is presented in Table 1 the performance of all implemented methods considering distance estimation error, standard deviation, False Positive

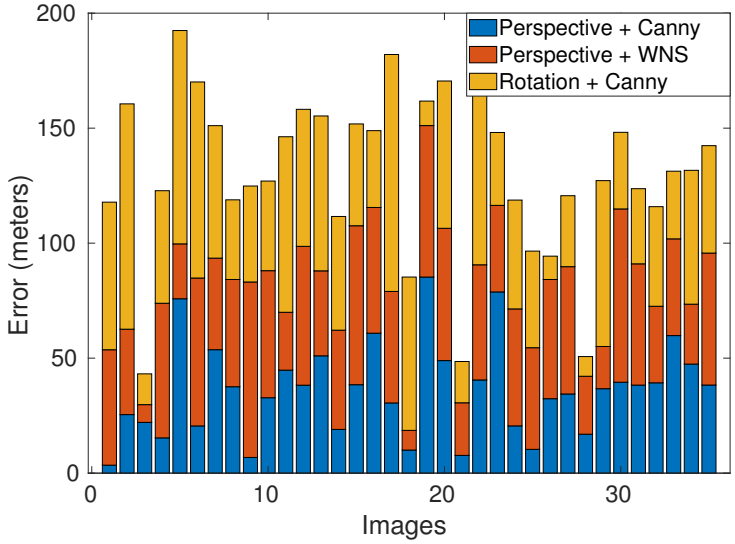


Figure 6: Error in meters of the three methods in relation to the correct trajectory.

Figure 7: Map with the position estimates of each method tested. Black line, real position; Red line, Perspective + WNS; Yellow line, Perspective + *Canny*; Blue line, Rotation + *Canny*.

($\{FP_{img} = 1\}$ **if** $\{Dist.\ Error_{img} \geq 100\ m\}$) and True Positive $\{TP_{img} = 1\}$ if $\{Dist.\ Error_{img} < 100\ m\}$ errors. The error in meters of the three methods considering 35 images sampled from in-flight video is presented

in the stacked barplot of Fig. 6. The position estimates (*lat, lot*) of each method used in the experiment are shown in Fig. 7. As can be observed, the best performance was achieved with perspective correction followed by edge detection with Canny operator [27]. WNS approach, however, had a competitive performance, particularly considering standard deviation and simplicity of implementation as will be discussed in the next sections. The set of images for WNS edge detector and the WNS deforestation Surveillance detector were made available in a public repository[1].

Table 1: Performance considering the distance to the real UAV position: FP (False Positive) and TP (True Positive), when the position estimation of the compared algorithms exceeds or not an error greater than 100 meters.

Case	Av. Dist. Error	Std Dev.	Variance	FP(%)	TP(%)
Perspective.+Canny	**36.04**	20.27	410.86	**0 %**	**100 %**
Rotation+Canny	50.01	25.67	660.44	2.86 %	97.14 %
Perspective+WNS	44.94	**17.50**	**305.89**	**0 %**	**100 %**

4. Methodology and Results

Likewise aerial image navigation, image deforestation surveillance by UAVs can also become more effective with the use of real-time systems and on-board processing. In most applications, however, the problem has been handled post-flight with the aid of human specialists.

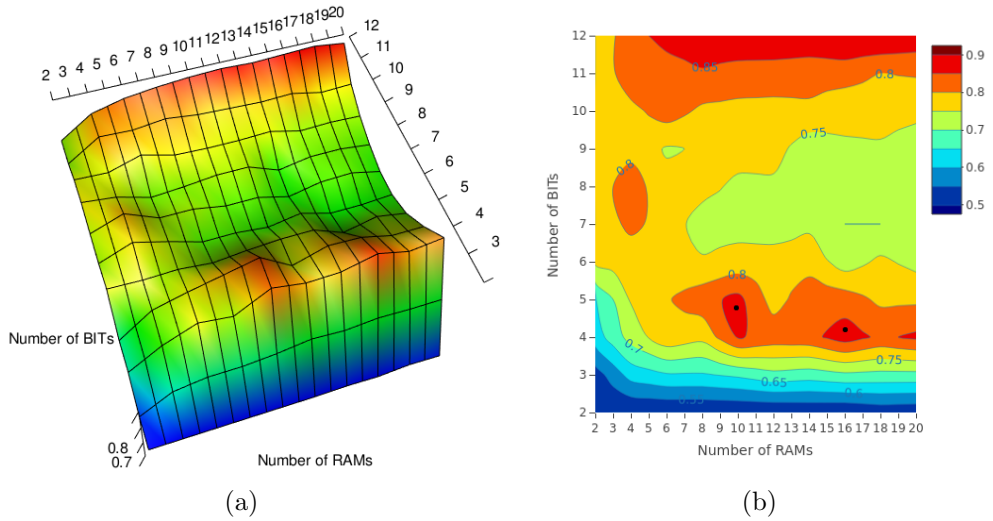
In order to evaluate a combined approach of a fully on-board computer system for this application, a previously tagged set of RGB satellite images of the Amazon region [28], Fig. 11, which is often used for benchmarking

[1]https://github.com/P4yo/Combined-Weightless-Neural-Network-FPGA-Architecture-with-application-in-aerial-images.git

deforestation surveillance methods, was selected for the experiments. Classification was accomplished on pixel level, so each image channel was discretized and represented with 4 bits, resulting on 12 bits per pixel. Since the target classes are well defined, discretization level was not too critical, so 4 bits per channel was enough to properly represent the two classes. Two discriminators were then trained with randomly sampled examples of pixels extracted from forest and deforested areas. A total of 50 image examples were selected for training, with 25 examples of deforested and 25 of forest areas, the same size of the test set.

In order to train the discriminator with forest and deforested image samples, discriminator's hyper-parameters n and k were first selected with a grid search approach. Twenty images from Kaggle image set [28] were randomly selected in order to provide examples of the two classes. For each image, 10 square regions with 20×20 pixels of forest and 10 of non-forest areas were selected for the grid search. Ten trials of random connections to the input field were accomplished for each pair of values of n and k, which were assessed from 2 to 20 and from 2 to 12, respectively. The resulting grid-search surface and contour plot are presented in Fig. 8. For each pair of n and k, of every trial, the corresponding FPGA circuit was synthesized and the circuit cost was obtained, resulting also on the cost surface corresposnding to the contour plot of Figure 9. The selected architecture results, therefore, from a trade-off between performance and cost presented in Figures 8 and 9. As can be observed in Figure 8, there are two peaks of performance in the lower cost region of Figure 9, which occur at $n = 5$ and $k = 10$, with accuracy 0.865 and the other one at $n = 4$ and $k = 16$, with accuracy

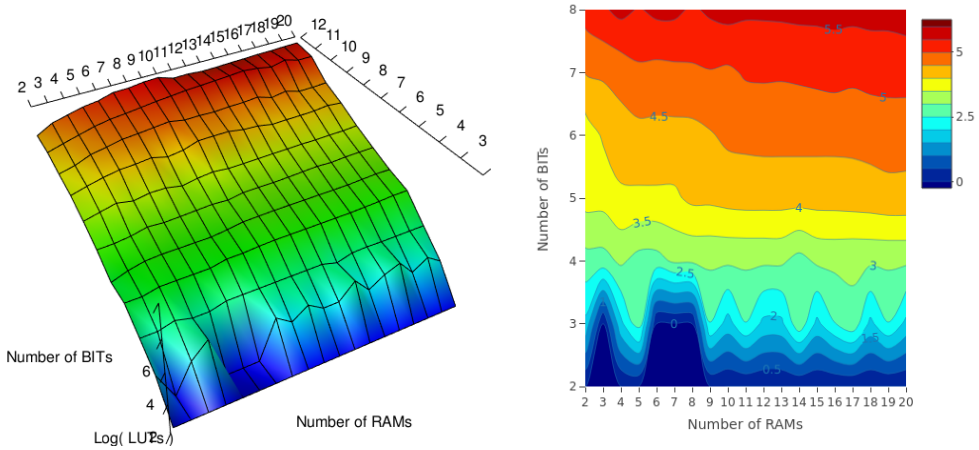
0.900. Since there is trade-off between cost and accuracy for these two circuit configurations, an additional statistical test was accomplished in order to select one of them. An additional experiment was made with 100 trials per each connection configuration between RAMs and input images. The resulting accuracy histograms for the two configurations are presented in Figure 10. The Wilcoxon's Statistical Test applied to the experiment shows that the distribution of accuracy rate of the two configurations are significant diferent with 0.95 confidance level ($p < 0.05$), so the one with the highest accuracy was selected ($n = 4$ and $k = 16$).



(a) (b)

Figure 8: Grid search surface and contour plots of hyper-parameters n and k of the RAM discriminator.

Although discriminators were built as classifiers on the pixel level, images were not scanned on a pixel-by-pixel basis. Since the objective was to identify the presence or not of larger deforested areas, some degree of discretization in the detected regions was acceptable. So, in order to reduce image scanning costs, square regions of the input images were classified instead of individual



(a) (b)

Figure 9: Contour plot of the corresponding grid-search of the logarithm cost surface of the trials of Figure 8.

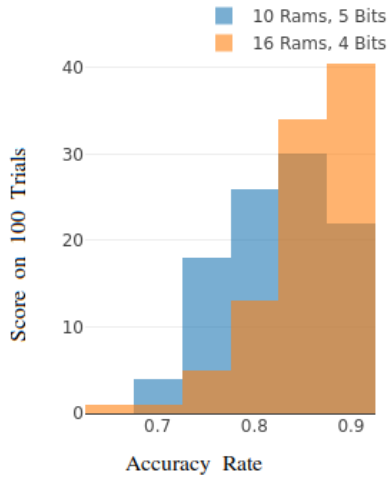


Figure 10: Histograms of the obtained accuracies of the selected discriminator configurations.

pixels. For each square region, 50% of the pixels were randomly selected and classified. Each square region was classified according to a majority voting

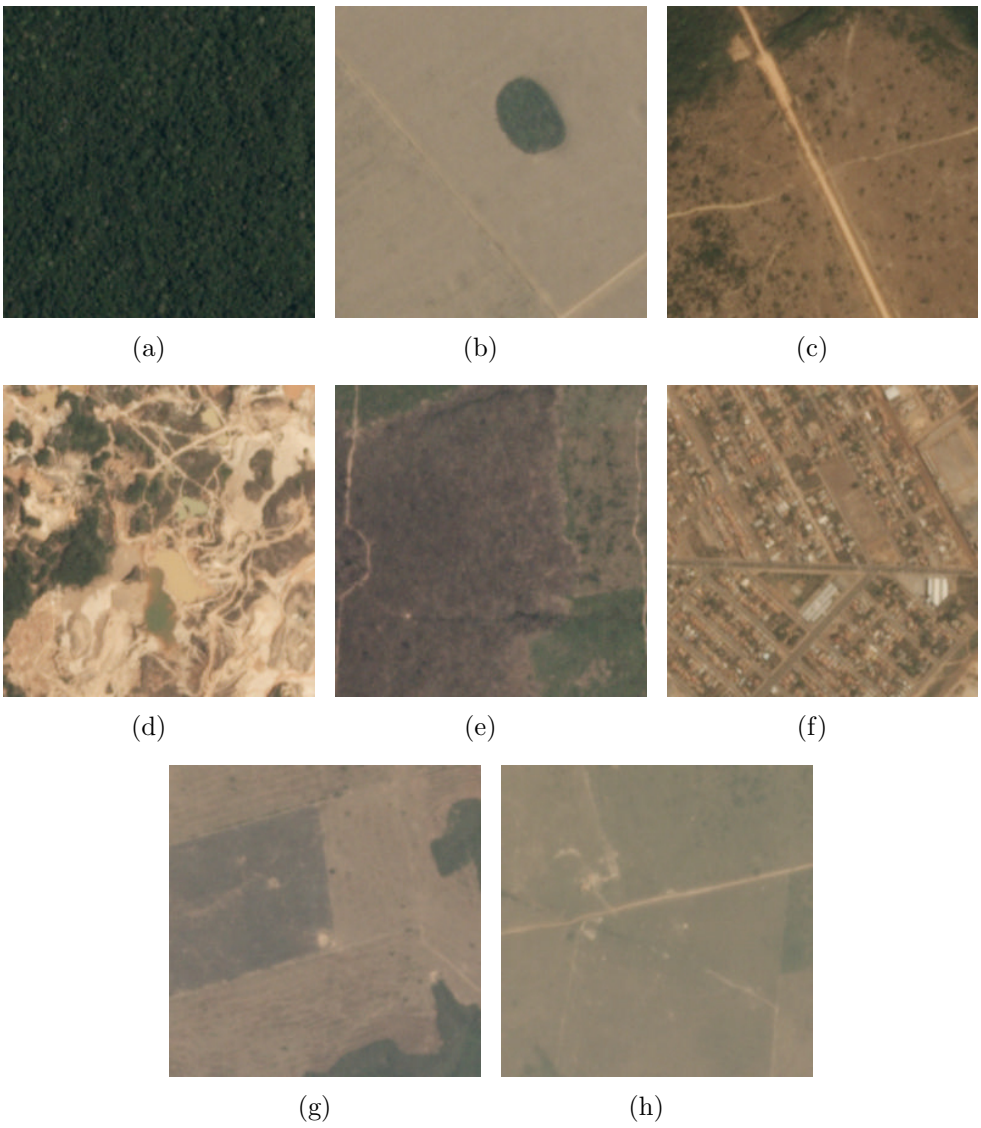
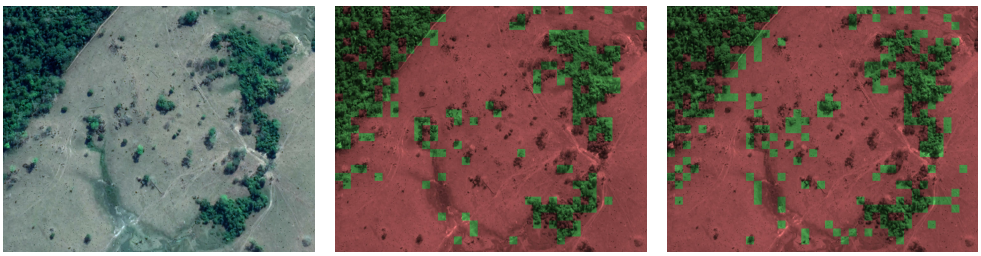


Figure 11: Example of Amazon images - kaggle. The first image represent forest areas and other images represent deforest areas. (a) Forest Image example; (b)-(h) Deforestation Image example.

amongst all pixel classifications, resulting on a certain degree of discretization, depending on the square size, as can be observed in the satellite image example of Fig. 12. It is interesting to notice in the figure that the two regions were

correctly identified and that, even some trees in the deforested area were spot by the classifier. Although the deforestation region has some green tones, it was correctly detected, what is particularly interesting, since some of the deforested areas can be used for cattle breeding and are actually green. This outcome is explained by the fact that training samples were selected from dense forest areas, which are quite characteristic and differ from the pale green tones in deforested areas. Classification accuracy performance in test benchmarking images was 90.0%.



(a) (b) (c)

Figure 12: (a) Original Image from Google Earth, (b) classification with 10 RAMs - 5 bits, (c) 16 RAMs - 4 bits.

4.1. FPGA Implementation

Hardware implementation of WNS is often presented in the literature [8] and FPGAs are an efficient approach for fast prototyping and customization. Previous works present, for example, a system that can be used in training, by employing external memory for storage and a FPGA for address mapping and discriminator selection [29]. FPGAs have also been used to accelerate specific functions of Deep Learning Networks [30]. It has been shown in this works da FPGAs are capable to improve performance, especially when accelerating repetitive and parallelized operations on the same data, as is the case for the CPU (Central Processing Unit) intensive convolutions.

The Sobel Operator [31] edge detection method is also frequently found in optimized applications. From early silicon implementations [32] to modern FPGA accelerators [33, 34], the method remains an interesting option, especially due to its simplicity. Despite being very efficient, the method requires arithmetic operations, as opposed to the pure combinational logic of WNS. It is also a very good comparison baseline for the method proposed here, since its base operation is performed on 3×3 pixel filters.

FPGAs implementations enable applications with low power consumption, which can be easily embedded and is particularly appropriate for UAV onboard image navigation. For our application, there would be no training in the final embedded implementation, which allows us to synthesize the trained RAMs in static combinational circuits. The sparsity of the trained discriminators, and the simplicity of the other circuits allowed very efficient synthesis in terms of internal FPGA resources.

The edge detector, the deforestation discriminators and the template matching circuits were synthesized considering the Xilinx Zynq-7000 family, which integrates ARM® processor cores with an FPGA fabric. This architecture is particularly appropriate for this problem, since the CPU can be dedicated to higher level tasks, while accelerating specific procedures in the FPGA, such as edge detection and image classification. The considered model was the xc7z020clg484, which contains the resources summarized next. It is clear from these specifications that the device has a CPU powerful enough to run a full operating system and that the programmable hardware is not among the high-end devices from the same manufacturer.

- CPU dual-core ARM® Cortex®-A9 (ARMv7-A architecture)

- FPGA Artix®-7 with:
  - 53.200 Look-Up Tables (LUTs).
  - 106.400 Flip-Flops (FFs).
  - 140 RAM units with 36 KB each (4.9 MB in total).
  - 220 arithmetic processors (DSPs).

The basic unit of the deforestation circuit was formed by 9 discriminators in parallel, corresponding to each pixel of a 3×3 square region of the input image. The 9 outputs, which result on the binarization of the 3×3 region are then fed to the edge detection circuit as its input kernel, as shown schematically in Fig. 4. The edge filter circuit has then a single output, which corresponds to the edge detection outcome of the central pixel of the 3×3 kernel.

Due to the tendency of exponential growth of FPGA resources with the address size of the RAMs, the complexity analysis was focused on the two performance peaks presented in Fig. 8. In this region, the sizes of 4 and 5 bits were analyzed with 100 random trials each, resulting in the histograms of Figure 10 and the statistical test presented in Section 4. The final discriminator configuration selected has 16 RAMs with 4 inputs each.

A comparison of the proposed method to recent Sobel Operator FPGA implementations is presented in Table 2. With the chosen parameters (16 RAMs with 4 address lines), excluding the random trials which resulted in connections with saturated memories, the synthetized circuits were very simple. It should also be noted that the 9 discretized outputs are also usable

for the decision between forest or deforestation, which is an extra feature when compared to edge enhancement based on the Sobel Operator.

Table 2: Comparison of FPGA resources of two implementations of the Sobel Operator × edge and deforestation detection circuits for a 3×3 pixels block.

Resource	**[33]**	**[34]**	**Proposed Method**
Slice Registers	49	0	0
Slice LUTs	135	114	21

Template Matching could also be implemented in FPGA, since the 9 outputs of the deforestation discriminators can also be connected to a circuit that matches a block of the captured image to one in the image map. Considering only the combinational part (bit matching and counting), the circuit could be implemented with just 11 LUTs. A segment of the georeferenced image, with dimensions up to 543×543 would fit in a single RAM block of the selected device to be used as a search reference.

5. Conclusions

In this work, two important applications for UAVs, which require in-flight processing for fast decision making, were presented. The two applications have both real-time and embedded processing requirements, which demand a need for high computational performance for image processing and low power consumption.

The FPGA-synthesized WNS architecture presented in this work was capable to efficiently combine the outcomes of deforestation surveillance and edge detection Boolean neural models. Sparsity of learned Boolean functions resulted in simplified circuits and very small utilization percentage of the

FPGA, what allows for several UAV image processing tasks to be executed and implemented in parallel. FPGA utilization can be increased, while still retaining enough capacity to implement high speed bus communication between CPU, FPGA, memory and also to include image scanning sequential circuits. Further developments could include other types of surveillance such as fires, burnt areas and invasion of environmentally protected areas, problems that are feasible to be treated in the Boolean domain with weightless neural systems.

Acknowledgment

This work has been supported by CNPq, Capes and FAPEMIG.

[1] W. W. Bledsoe, I. Browning, Pattern recognition and reading by machine, ACM, New York, NY, USA, 1959, pp. 225–232 (1959). doi:10.1145/1460299.1460326.

[2] I. Aleksander, M. De Gregorio, F. M. G. França, P. M. V. Lima, H. Morton, A brief introduction to weightless neural systems., in: ESANN, 2009, pp. 299–305 (2009).

[3] M. De Gregorio, M. Giordano, Change detection with weightless neural networks, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2014 (June 2014).

[4] I. Aleksander, Self-adaptive universal logic circuits, Electronics Letters 2 (8) (1966) 321–322 (August 1966). doi:10.1049/el:19660270.

[5] A. P. Braga, Predicting contradictions in the storage process of diluted recurrent boolean neural networks, Electronics Letters 30 (1) (1994) 55–56 (Jan 1994). doi:10.1049/el:19940020.

[6] P. M. Kintner, B. M. Ledvina, E. R. d. Paula, Gps and ionospheric scintillations, Space Weather 5 (9) (2007) 1–23 (Sep. 2007). doi:10.1029/2006SW000260.

[7] G. Conte, P. Doherty, An integrated uav navigation system based on aerial image matching, in: 2008 IEEE Aerospace Conference, 2008, pp. 1–10 (March 2008). doi:10.1109/AERO.2008.4526556.

[8] I. Aleksander, W. Thomas, P. Bowden, Wisard - a radical step forward in image recognition, Sensor review 4 (3) (1984) 120–124 (1984).

[9] I. Wickert, F. M. França, Autowisard: Unsupervised modes for the wisard, in: International Work-Conference on Artificial Neural Networks, Springer, 2001, pp. 435–441 (2001).

[10] P. Coraggio, M. De Gregorio, Wisard and nsp for robot global localization, in: J. Mira, J. R. Álvarez (Eds.), Nature Inspired Problem-Solving Methods in Knowledge Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 449–458 (2007).

[11] B. K. Kim, H. Kang, S. Park, Drone classification using convolutional neural networks with merged doppler images, IEEE Geoscience and Remote Sensing Letters 14 (1) (2017) 38–42 (Jan 2017).

[12] Y. Chen, P. Aggarwal, J. Choi, C. . Jay, A deep learning approach to

drone monitoring, in: 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2017, pp. 686–691 (Dec 2017).

[13] B. Kalantar, S. Mansor, M. Al-Zuhairi, B. Pradhan, H. Shafri, Drone-based land-cover mapping using a fuzzy unordered rule induction algorithm integrated into object-based image analysis, International Journal of Remote Sensing (2017) 1–22 (01 2017).

[14] K. Thakar, D. Kapadia, F. Natali, J. Sarvaiya, Implementation and analysis of template matching for image registration on devkit-8500d, Optik-International Journal for Light and Electron Optics 130 (2017) 935–944 (2017).

[15] Y. Li, D. J. Crandall, D. P. Huttenlocher, Landmark classification in large-scale image collections., in: ICCV, IEEE Computer Society, 2009, pp. 1957–1964 (2009).
URL http://dblp.uni-trier.de/db/conf/iccv/iccv2009.html#LiCH09

[16] M. Maimone, Y. Cheng, L. Matthies, Two years of visual odometry on the mars exploration rovers, Journal of Field Robotics 24 (3) (2007) 169–186 (2007).

[17] D. Scaramuzza, R. Siegwart, Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles, IEEE transactions on robotics 24 (5) (2008) 1015–1026 (2008).

[18] P. Corcoran, A. C. Winstanley, P. Mooney, R. H. Middleton, Background foreground segmentation for SLAM, IEEE Trans. Intelligent Transportation Systems 12 (4) (2011) 1177–1183 (2011). doi:10.1109/TITS.2011.2143706.
URL https://doi.org/10.1109/TITS.2011.2143706

[19] T. Bailey, H. Durrant-Whyte, Simultaneous localization and mapping (slam): Part ii, IEEE Robotics & Automation Magazine 13 (3) (2006) 108–117 (2006).

[20] M. Laîné, S. Cruciani, E. Palazzolo, N. J. Britton, X. Cavarelli, K. Yoshida, Navigation system for a small size lunar exploration rover with a monocular omnidirectional camera, in: First International Workshop on Pattern Recognition, Vol. 10011, International Society for Optics and Photonics, 2016, p. 100111M (2016).

[21] G. Conte, P. Doherty, Vision-based unmanned aerial vehicle navigation using geo-referenced information, EURASIP Journal on Advances in Signal Processing 2009 (2009) 10 (2009).

[22] W. da Silva, E. H. Shiguemori, N. L. Vijaykumar, H. F. de Campos Velho, Estimation of uav position with use of thermal infrared images, in: Sensing Technology (ICST), 2015 9th International Conference on, IEEE, 2015, pp. 828–833 (2015).

[23] J. R. Braga, H. F. Velho, G. Conte, P. Doherty, É. H. Shiguemori, An image matching system for autonomous uav navigation based on neural

network, in: Control, Automation, Robotics and Vision (ICARCV), 2016 14th International Conference on, IEEE, 2016, pp. 1–6 (2016).

[24] G. A. M. Goltz, E. H. Shiguemori, H. F. D. C. Velho, Position estimation of uav by image processing with neural networks, in: G. d. A. Barreto, J. A. F. Costa (Eds.), X Brazilian Congress on Computational Intelligence, SBIC, Fortaleza, CE, 2011, pp. 1–8 (2011).

[25] J.-C. Yoo, T. H. Han, Fast normalized cross-correlation, Circuits, systems and signal processing 28 (6) (2009) 819 (2009).

[26] B. Jaimes, C. Castro, Perspective correction in aerial images (12 2018). doi:10.13140/RG.2.2.34885.29926.

[27] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8 (6) (1986) 679–698 (Nov 1986). doi:10.1109/TPAMI.1986.4767851.

[28] Planet and sccon: Understanding the amazon from space (Jun. 2009). URL `https://www.kaggle.com/c/planet-understanding-the-amazon-from-space`

[29] M. H. B. Azhar, K. R. Dimond, Design of an fpga based adaptive neural controller for intelligent robot navigation, in: Proceedings Euromicro Symposium on Digital System Design. Architectures, Methods and Tools, IEEE, 2002, pp. 283–290 (2002).

[30] A. Shawahna, S. M. Sait, A. El-Maleh, Fpga-based accelerators of deep learning networks for learning and classification: A review, IEEE Access 7 (2018) 7823–7859 (2018).

[31] R. O. Duda, P. E. Hart, D. G. Stork, Pattern classification and scene analysis (pp. 271-272), Vol. 3, Wiley New York, 1973 (1973).

[32] N. Kanopoulos, N. Vasanthavada, R. L. Baker, Design of an image edge detection filter using the sobel operator, IEEE Journal of solid-state circuits 23 (2) (1988) 358–367 (1988).

[33] S. Halder, D. Bhattacharjee, M. Nasipuri, D. K. Basu, A fast fpga based architecture for sobel edge detection, in: Progress in VLSI Design and Test, Springer, 2012, pp. 300–306 (2012).

[34] N. Nausheen, A. Seal, P. Khanna, S. Halder, A fpga based implementation of sobel edge detection, Microprocessors and Microsystems 56 (2018) 84–91 (2018).