Article

Fuzzy ARTMAP-Based Fast Object Recognition for Robots Using FPGA

Victor Lomas-Barrie [1], Mario Pena-Cabrera [1], Ismael Lopez-Juarez [2,*] and Jose Luis Navarro-Gonzalez [3]

[1] Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México, Mexico City 04510, Mexico; victor.lomas@iimas.unam.mx (V.L.-B.); mario.penia@iimas.unam.mx (M.P.-C.)
[2] Centro de Investigación y de Estudios Avanzados del Instituto Politecnico Nacional (CINVESTAV), Ramos Arizpe 25900, Mexico
[3] IJ Robotics S.A. de C.V., Saltillo 25000, Mexico; jnavarro@ij-robotics.com
* Correspondence: ismael.lopez@cinvestav.edu.mx

Abstract: Fast object recognition and classification is highly important when handling operations with robots. This article shows the design and implementation of an invariant recognition machine vision system to compute a descriptive vector called the Boundary Object Function (BOF) using the FuzzyARTMAP (FAM) Neural Network. The object recognition machine is integrated in the Zybo Z7-20 module that includes reconfigurable FPGA hardware and a RISC processor. Object encoding, description and prediction is carried out rapidly compared to the processing time devoted to video capture at the camera's frame rate. Benefiting from parallel computing, we calculated the object's centroid and boundary points while acquiring the progressive image frame; all that was done with the intention of readying it for neural processing. The remaining time was devoted to recognising the object, this caused low latency (1.47 ms). Our test-bed also included TCP/IP communication to send/receive part location for grasping operations with an industrial robot to evaluate the approach. Results demonstrate that the hardware integration of the video sensor, image processing, descriptor generator, and the ANN classifier for cognitive decision on a single chip can increase the speed and performance of intelligent robots designed for smart manufacturing.

Keywords: embedded system; FPGA; Fuzzy ARTMAP; robot vision

Citation: Lomas-Barrie, V.; Pena-Cabrera, M.; Lopez-Juarez, I.; Navarro-Gonzalez, J.L. Fuzzy ARTMAP-Based Fast Object Recognition for Robots Using FPGA. *Electronics* **2021**, *10*, 361. https://doi.org/doi:10.3390/electronics10030361

Academic Editor: Teen-Hang Meen
Received: 24 December 2020
Accepted: 29 January 2021
Published: 2 February 2021

1. Introduction

According to the International Federation of Robotics (IFR), in demand operations for industrial robots include handling applications, welding and assembly, with the first being the most sought after one [1]. Material handling makes extensive use of the robot's simple capability to transport objects using an appropriate end-of-arm tool (e.g., gripper), the robot can efficiently and accurately move products from one place to another. That is, if a precise location and orientation is given. However, due to fluctuating raw material prices, labour constraints, transportation/logistics, and the idiosyncrasy of clients, most industries are heavily influenced by agile production. Such production systems demand intelligent robots equipped with smart machine vision systems capable of fast and invariant object recognition applied to usage in unstructured environments. However, when applying artificial intelligence techniques to the area of industrial robots there are still challenges to overcome. As mentioned in [2], these challenges include the successful learning of complex dynamics; control policies that work in dynamic, unstructured environments; manipulation; and advanced object recognition. Intelligent robots should be capable of recognising the part to grasp quickly and accurately to rapidly reconfigure itself on-line and improve production times.

The artificial vision system presented in this article aims to contributes to the development of intelligent robots via digital machine (characterised by reconfigurable hardware

(FPGA) and an RISC processor to extract characteristics) integration. The RISC processor would help us extract characteristics and form a descriptive vector that in conjunction with Artificial Neural Networks (ANN's), is in charge of the classification of parts through the BOF descriptor (Boundary Object Function) as originally described in [3]. This object descriptor is used to predict the recognition of work-pieces using the FuzzyARTMAP [4] (FAM) Neural Network learning mechanism (implemented in the FPGA). The embedded system optimises resources, improves computing speed and provides a fast determination of the object's POSE.

The BOF descriptive vector is used for its invariability to scaling, rotation, translation, and its ability to describe the work-pieces with the same amount of data, so that, its vector length is always the same. In comparison, other methods based on feature extraction such as SIFT, SURF or ORB, rely on the number of points of interest and are scene dependent. The domain of recognition of the described approach is constrained to rigid objects and the following conditions: (1) The centroid should be contained within its solid surface for grasping, and (2) The generalisation capability of the ANN only considers previously learned objects. These conditions are easily met since most manufactured parts are regular, previously known objects.

Related Work and Original Contribution

Some of the most reliable descriptors that represent outlines of rigid shapes (silhouettes) are generated with tools/procedures such as Shape Number, where a unique number is used to identify the outline of a silhouette [5]; Run Length Codes, which is a type of information compression where a string of equal data is replaced by a code that indicates the value and the number of times it occurs [6]; and Chain Codes, a tool that represents a path in two or three dimensions [7]. They have a reasonable computational cost for offline applications, however, they are not intended for usage in embedded systems.

Additionally, there are several tools/procedures based on detection of characteristic points and their descriptors such being the case of SIFT [8] and its version in FPGA [9]. With the Bouris-SURF [10], which is a variant of the SURF method [11], a 640 × 480 pixel image is processed in 18 ms without object detection. Another modified procedure is the Flex-SURF [12], which had processing times of 2.9 s for the detection of characteristic points and 4s for the calculation of the descriptor; to be noted are the remarkable saving times with BRAM. Another case is the HR-SURF [13] which is used to process images in 335 ms while not excluding object detection. In [14] improvement was presented which allowed 400 features to be detected in a 640 × 480 image at 6.36 ms. Implementations of the ORB method and some variants known as FAST, FAST + BRIEF and FAST + BRISK have also been utilised [15] to result in 6.8 ms of processing time for a video frame. This last work makes interesting comparisons to the algorithm solution in software using a GPU (Jetson TK1), for which times of 13.8, 18.2 and 27.6 ms were obtained, respectively. Alternative approaches are based on the design of digital machines for ANN implementation on reconfigurable hardware such as the FuzzyARTMAP as described in [16]. Early design of VLSI circuits based on adaptive resonance theory were reported in [17]. Other methods such as convolutional neural networks (CNNs) have also been reported [18]; support vector machines (SVM) [19] and IoT approaches are also reported in [20,21]. Late developments in hardware have culminated with Dynamic Vision Sensors (DVSs) which provide a new strategy for representing real-time visual information as event-flows such as spike Neural Networks [17]. Rapid approaches to asynchronous, neuromorphic DVSs have reported very short latency times for robotic control as reported in [22] where pixel data is generated and transmitted asynchronously only from pixels with varying brightness. This allows latency to go as low as 2.2 ms when detecting a moving ball and it triggering a motor command for the goalie robot.

Compared to other approaches, our proposal provides a fast determination of the POSE of the object by optimising resources and improving computational speed. This is basically due to three main reasons: The use of FuzzyARTMAP's fast incremental learning capability, the effective use of memory resources in the FPGA, and the robustness of the BOF algorithm. Furthermore, the BOF is calculated while the pixels of the image frames are being received, resulting in part recognition times (including the object's centroid and its orientation angle) as low as 1.466 ms, which It is a lower value than the values reported in the literature. This timing was also made possible by efficient Block RAM (BRAM) memory management and the inherent parallel processing of the FPGA and the neural architecture. These advantages reinforce our aim to provide industrial robots with the ability to quickly recognise objects by using a robust, yet simple algorithm.

This paper is organised as follows: After a brief introduction, in Section 3 the Boundary Object Function (BOF) and the classification process of the FuzzyARTMAP (FAM) are explained. Section 4 introduces the embedded system architecture, the BOF extractor and the FAM classifier. In Section 5, experiments and results are provided. Finally, in Section 6, conclusions are given.

2. Experimental Test-Bed

The experimental test-bed is integrated via a KUKA KR16 industrial robot equipped with a hand-in-eye camera. The communication framework was based on the open source KUKAVARPROXY server application as it is shown in Figure 1. With this server application running in the robot controller, the read/write system and user-defined variables can be written. This framework allows full robot control that is translated to a Zybo Z7-20 module (Pullman, WA, USA). The Zybo Z7-20 is also used as a smart distributed module since it works as a TCP client and an interface to the KUKA Robot using OpenShowVar in a similar way to that described by [23]. Lastly, the client application is programmed in Zybo Z7-20 to supervise the robotic system status.

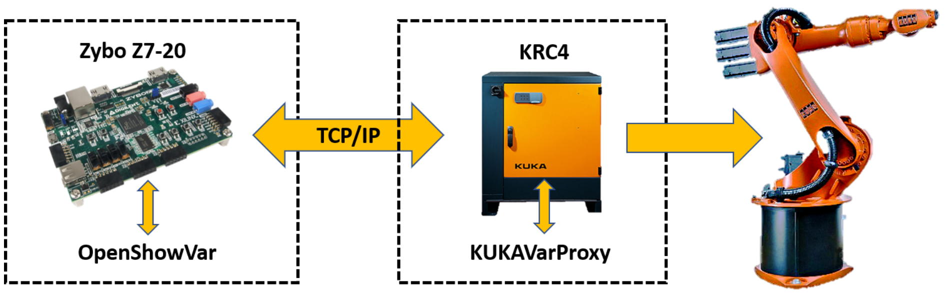


Figure 1. Robot Communications.

Additionally, the test-bed is comprised of a belt conveyor and a working table as shown in Figure 2. The intention is to train the FuzzyARTMAP network with simple objects' geometries so that the industrial robot may be able to grasp for further handling operations. In Section 3, a comprehensive explanation of the BOF and the network's training can be found.

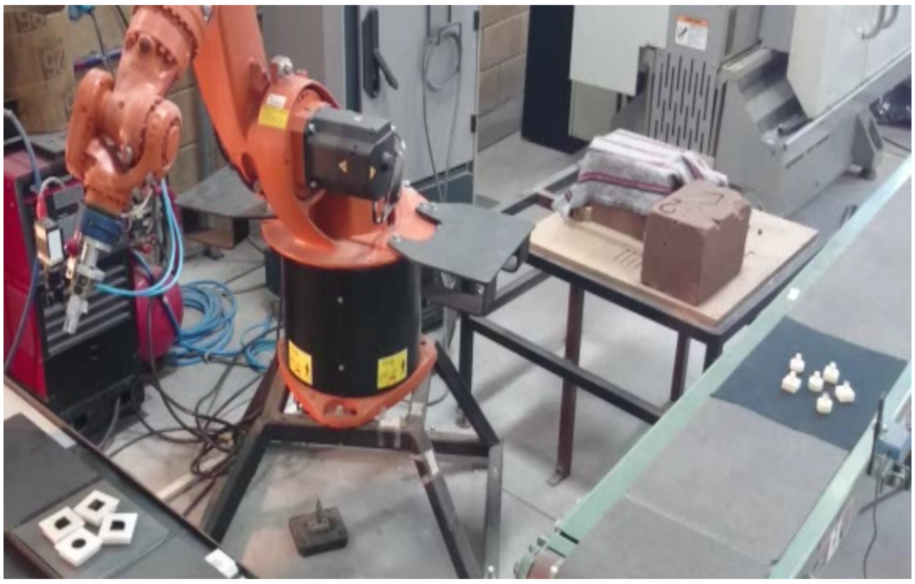


Figure 2. Robotic Testbed.

3. The BOF Descriptor and the Fuzzy ARTMAP

This section addresses the method described in [3] to generate the BOF vector descriptor. The FuzzyARTMAP ANN in charge of classifying the BOF vectors [4] is also described. As explained earlier, the proposed methodology is intended for fast object recognition which requires the training of the FuzzyARTMAP to occur first; additionally, the training would have to happen offline and using sets of images from the work-pieces as input. This procedure was carried out with a Matlab script in an external computer. Once the training was completed, the object's categorical representation was implemented in the Zybo Z7-20 module.

3.1. BOF

The BOF descriptor of an object is invariant to rotation, scaling, and translation, making it well suited for assembly and part-ordering applications. It is a representation of the contour of the object through a one-dimensional function, through the distance between the centroid of the body to each of the points that belong to the perimeter, and as a function of the angle as shown in Figure 3.

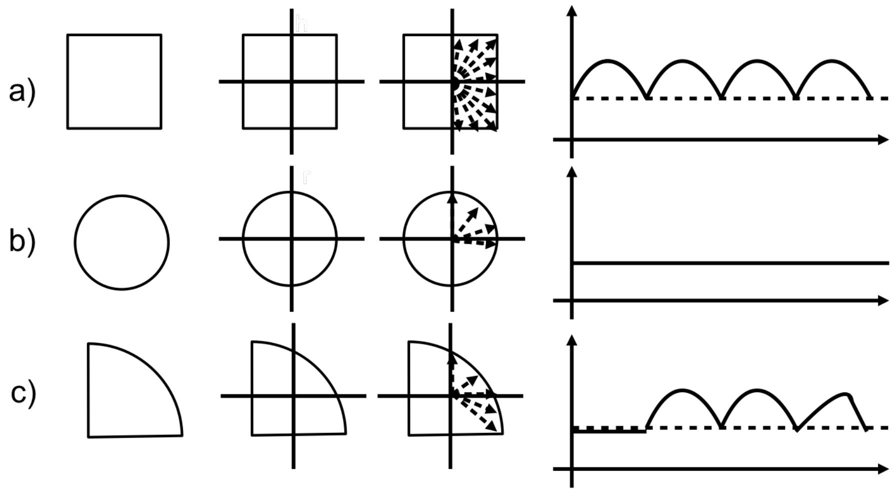


Figure 3. Boundary Object Function (BOF) Descriptor of some basic shapes. (**a**) squared; (**b**) circular; (**c**) Arc.

The process begins when the objects are segmented from the scene and the Regions Of Interest (ROI) are formed. In this work, it is assumed that there is a single object in the scene. A previous process performs the ROI segmentation from the image, histogram equalisation, and background removal.

The algorithm for its extraction is as follows:

1. The region of interest is segmented from the original image and a new image is formed.
2. The image is converted into a binary image (Img_{bin}) from a threshold (THD_{bin}). The background is assigned a logical '0' and the object a logical '1'.
3. The centroid of the object is calculated with:

$$X_c = \frac{\sum A_x}{\sum A} \quad (1)$$

$$Y_c = \frac{\sum A_y}{\sum A} \quad (2)$$

4. A two-dimensional arrangement known as a weight matrix is obtained I_{WM}.

$$I_{WM}(i,j) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} Img_{bin}(i+l, j+k)K(i,j) \quad (3)$$

where: $0 \leq I_{WM}(i,j) \leq 9$; $\forall i,j \in 0 \cup \mathbb{N}$; $-1 \leq k \leq 1$; $-1 \leq l \leq 1$; $K[i,j] = 1 \forall i,j = 1,2,3$.

5. The Boundary Points (BP) are obtained from two thresholds THD_BPM and THD_BPm.

$$BP = \{i,j \mid THD_{BPM} \leq I_{WM}(i,j) \leq THD_{BPm}\} \quad (4)$$

6. The Euclidean distance between the centroid and each boundary point is calculated.

$$D_n = \sqrt{(X_c - x)^2 + (Y_c - y)^2} \quad (5)$$

7. Each distance obtained in the previous step is normalised with the maximum distance between the centroid and a boundary point. The product from this process is the BOF vector.

3.2. Fuzzy ARTMAP

The Adaptive Resonance Theory (ART) developed by Stephen Grossberg at Boston University is a well-established, associative brain and competitive model. It was introduced as a theory of the human cognitive process. ART is applied and working in situations where it is called the stability-plasticity dilemma suggesting that connectionist models should be able to adapt and switch between its plastic and stable modes. That is, a system should exhibit plasticity to accommodate new information regarding unfamiliar events while still being able to remain in a stable condition if familiar or irrelevant information is being presented. The theory has evolved into many real-time architecture variants for unsupervised learning; the ART-1 algorithm for binary input patterns being one [24]. Supervised learning is also possible through ARTMAP [25] which uses two ART-1 modules capable of learning the correspondence between input patterns and desired output classes through training, this is the case seen in our research with the object's representation. For analogue values, an appropriate ANN for supervised learning is the FuzzyARTMAP (FAM) [4]. The learning process and subsequent selection of the BOF descriptor of an object are carried out through this network.

All objects to be recognised are learned by the FAM and represented categorically. Our research is focused on the classification of categories, that is, on the recognition of objects, however the learning is done off-line. The set of weights from all learned categories is transferred to the embedded system from a personal computer. It is important to mention that our BOF descriptor not only represent salient object features but also the manufactured part to be manipulated by the robot.

The procedure for FAM classification is as follows:

1. The input vector I uses the complement coding as given in Equation (6).

$$I = (a, a^c) = (a_1, a_2, ..., a_M, a_1^c, a_2^c, ..., a_M^c) \tag{6}$$

a are the BOF elements where: $a_i \in \mathbb{R}; \forall i \in \mathbb{N}$. $a_i^c \equiv 1 - a_i$

2. Activation T_j is obtained from Equation (7).

$$T_j(I) = \frac{|I \wedge w_j|}{\alpha + |w_j|} \tag{7}$$

where the fuzzy AND, or intersection, operator $\wedge$ is defined by $(p \wedge q)_i \equiv min(p_i, q_i)$ and the norm $|\cdot|$ is defined as $|p| \equiv \sum_{i=1}^{M} |p_i|$ to any vector p or q with dimension M. w_j are the weight vector elements, where $w_j \in [0,1]$, $I_i \in [0,1]$, $\alpha > 0$ and $T_j \in \mathbb{R}^+$. The system is said to make a category choice when, at most, one F2 node can become active at a given time. The category choice is indexed by J, where

$$T_J = max\{T_j : j = 1, ..., N\} \tag{8}$$

were T_J is the winner category. In case more than one Tj is maximal, the category j with the smallest index is chosen. In particular, nodes become committed in order $j = 1, 2, 3, \ldots$. When the Jth category is chosen, $y_J = 1$ and $y_j = 0$ for $j \neq J$.

3. Resonance occurs if the match function of the chosen category J meets the vigilance criterion ρ given by Equation (9).

$$\frac{|I \wedge w_J|}{|I|} \geq \rho \tag{9}$$

were $\rho \in [0,1]$. If Equation (9) is not satisfied, the next category is chosen and ρ verified again. Mismatch reset occurs otherwise.

4. Embedded System Architecture

This section describes the architecture for the embedded system that integrates the digital machine for the extraction of the BOF descriptor and the digital machine for the FAM classifier, namely, the ES-BOF&FAM.

The electronic board used for design, testing and final implementation was the Zybo Z7-20 from Digilent. It contains the 7z020clg400-1 integrated circuit based on the Zynq-7000 family (AP SoC, Xilinx All Programmable System-on-Chip (San Jose, CA, USA)). The AP SoCs has a hybrid architecture that integrates an ARM Cortex-A9 processor (Cambridge, Cambs, UK), the Processing System (PS), and a Field Programmable Gate Array (FPGA) from the Xilinx 7-series(San Jose, CA, USA) family which acts as the Programmable Logic part. (PL). Viviado from Xilinx was used as a tool for the design, routing, synthesis, simulation and generation of the bitstream. The hardware description language to create the system in programmable logic (PL) was VHDL. Some IP modules (Intellectual Property) designed by Xilinx were also integrated. For the programming and debugging of the algorithm that runs in the PS, Xilinx SDK and the C programming language were used.

The acquisition of video frames is carried out through a low-cost sensor, the OV7670 CCD (Santa Clara, CA, USA), with a maximum resolution of 640 $\times$ 480 pixels, a 1/6″ lens and a frame rate of 30 fps in progressive mode. Pixels are sent to the ES-BOF&FAM through an 8-bit parallel port and the I^2C port for control and configuration.

Due to the parallelisation characteristics that FPGAs allow, some parts of the process, for the extraction of the BOF and for the classifier (the FAM network), were implemented in the PL of the Zynq chip. Other procedures are run on the PS taking advantage of specific algorithm libraries.

Processes such as communication and control of the video sensor; conversion of colour pixels to grayscale; image binarisation; histogram calculation; extraction of boundary points from the object; calculation of the object centroid, the orientation angle, distance; and the complement coding to conform the BOF vector are carried out in the FPGA. In the case of the FAM classifier, there is the modulus of the minimums of the inputs and weights. The accumulator and the multiplication of the factor $\frac{1}{\alpha+|w_j|}$ is calculated offline.

The processes that are executed in the PS of the Zynq programmed in C are the highest-to-lowest sorting of the T_j; the integration of the weights of new categories to the FAM module; the main control of the system; the control and communication of the Ethernet network interface; those defined by commands for the control of the video sensor and, in general, the data management of the PL part. The algorithm for obtaining the ROI is performed by another method not addressed in this document.

4.1. General Architecture

In the Figure 4, the architecture of the embedded system is shown. The CCD sensor is connected to a PMOD port on the Zybo card via the GPIO pins. A client sends the data from the Ethernet interface to control the system parameters.

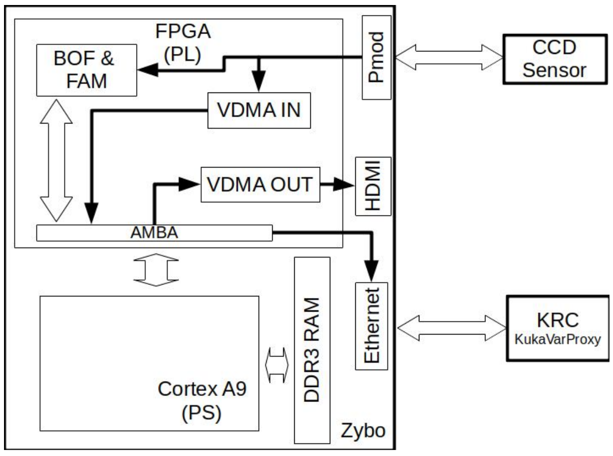


Figure 4. ES-BOF&FAM System architecture.

As mentioned earlier, the BOF descriptor and the FAM classifier are in the PL part of the Zynq and are connected via the AMBA bus (Advanced Microcontroller Bus Architecture) to the PS. In particular, communication is carried out using the AXI4lite (Advanced eXtensible Interface) standard; the reason for using AXI4lite being the amount of data to be transmitted as it does not warrant using another protocol for information burst messaging. Moreover, the CCD transmits the raw pixels to the BOF extractor, which transfers the information to the FAM classifier module.

4.2. Digital Machine for the BOF Extraction

To calculate the BOF, a pipeline process is carried out for each frame. In the first stage, the conversion from RGB to gray scale, the calculation of the histogram, and the determination of the binarisation threshold are carried out. In the second stage, the ROI is binarised and the centroid is calculated. In the third stage, the BOF is generated and stored. Starting from the fourth frame, the BOF is obtained in each subsequent frame.

In Figure 5, the digital machine for BOF extraction is shown. The input is the binary image of the object's ROI and the output is the BOF vector. The BOF vector is stored in a BRAM (Block RAM) that is subsequently accessed from the FAM classifier module, although it can also be read from the PS.

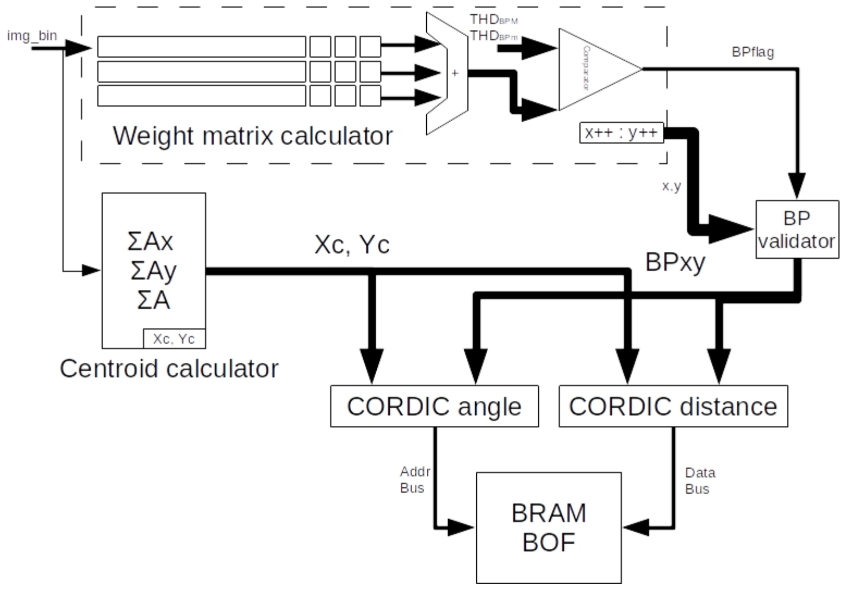


Figure 5. Digital machine for BOF extraction.

4.2.1. Centroid Calculator

To obtain the X_c y X_c values for Equations (1) and (2), it is necessary to obtain the object's area first. The area is calculated by increasing the variable sum_A by 1 each time clk_pixel changes while the object is in the scene, which is a condition verified by the WMC (Weight matrix calculator) module. Likewise, the $\sum A_x$ and the $\sum A_y$ values are calculated increasing its value x and y, respectively; where x and y are pointers along the ROI.

4.2.2. Boundary Points (BP) Extractor

In the third stage, I_WM is calculated according to Equation (3). For this, we have the WM calculator shown in the Figure 6. It is a combination of 3 shift registers (wmr_1, wmr_2 and wmr_3) (wmr is a weight matrix register) and a 4-bit parallel adder. The input to the module is img_bin and the output is a nibble in a range from 0 to 9. The size of each wmr is 1 bit and the width depends on the ROI. This is wmr_{1,2 y 3}$_{width}$ = ROI$_{width}$. In order to know the maximum of the necessary resources after the synthesis, it is considered that the ROI width is equal to 640 pixels.

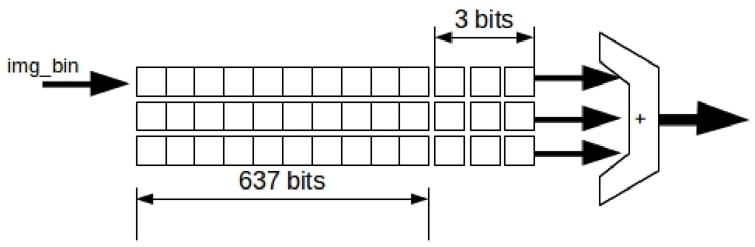


Figure 6. Weight matrix calculator.

All elements of the three wmr registers are initialised with '0'. The first pixel in the image is placed at position wmr_1 (639). In the next cycle, the bits of wmr_1 are shifted one position to the right and so on until the first pixel of the image reaches the first position of the register, that is wmr_1 (0). In the next loop, wmr_1 (0) passes its value to the second register, wmr_2 (639). The same thing happens when wmr_2 (0) is reached and passed to wmr_3 (639).

When the first pixel of the image reaches the wmr_3 (0) register, a sum of the last 3 bits of the three wmr registers is performed. The BP of the object is obtained with the Equation (4). This is done through a two-level comparator segmented by the thresholds THD_BPm and THD_BPM. Any pixel in img_wmr that matches this constraint is accepted as border. The values for THD_BPm and THD_BPM are determined to be 2 and 4, respectively. However, these values can be modified from the PS; we would modify the values because, in some cases, especially with small ROIs, it is necessary to reduce this margin. It should be noted that the value of nine in img_wmr represents a pixel that is exactly on the solid part of the object and a value less than 9 is close to the borders of the object.

The comparator's output, BP_flag, is connected to the BP validator module, which determines if the coordinates x and y are a boundary point. To generate the coordinates x and y, two counters are used: coord_x that goes from 0 to ROI $_{width}$ and coord_y that goes from 0 to ROI $_{height}$. The width depends on the concatenated value of the maximum of coord_x and coord_y for a maximum ROI, this is 10 bits for coordinates in x axis and 9 bits for coordinates in y axis.

4.2.3. BOF Vector Extraction

Each time a boundary point is detected, both the angle θ and the distance D (calculated with Equation (5)) of the vectors [(Xc, Yc), (X$_{BP_n}$, Y$_{BP_n}$)] are calculated. This is precisely the information to be sent to the robot, the angle θ and the centroid of the part to be grasped. In terms of HDL synthesis, the elements $(X_c - x)^2$ and $(Y_c - y)^2$ are implemented directly with basic arithmetic operations. To obtain the angle θ and the square root of (5), the CORDIC method was used, which has already been implemented in reconfigurable hardware with a low error rate [26].

Each element of the BOF vector is stored according to the corresponding position in its angle, that is, the element that corresponds to the 0° angle, corresponds to the first direction of BRAM_BOF. The distance between the centroid and the n_{th} BP is what is stored in that location. Thus, in the first memory address, we have the easternmost vector of the object with respect to the image at $\theta = 0°$.

It was found that 180 elements making up the BOF vector was enough for the recognition process [3]. Hence, the BRAM_BOF contains 180 locations, 2° of range for each element of the BOF. Since there is a probability that there is more than one BP with a separation smaller than 2°, it is taken as a general rule that the last value is overwritten to the previous one.

BOF values are normalised using the BOF max. This value is obtained by comparing the elements of the vector one by one with the last largest one. The Figure 7 shows 4 objects with their centroid and BP superimposed. In addition to their corresponding BOF, all of them are obtained by the digital BOF extraction machine.

It is necessary to choose a suitable number type in VHDL to consume less hardware resources and to obtain acceptable precision based on our specific application for object recognition in the robot's workspace. In ref. [27], it is stated that floating point is more accurate for ANN calculations; however, more hardware resources are unnecessary if we consider the ROI corresponding to the workspace (object to be recognised). As mentioned, the spatial resolution of the camera was 640 × 480 pixels and in practice it would never be necessary to use this resolution for the workpiece. Therefore, it was considered appropriate to work with a spatial resolution of up to 80%, of that resolution, which corresponds to 512 pixels for the ROI. This value was also considered as the maximum distance value to be processed whereas the minimum value would be $\frac{1}{512}$ which corresponds to 0.000000001 in base 2. This value can be represented by 9-bit fixed point and guarantee the best performance and resource usage for the FAM classification.

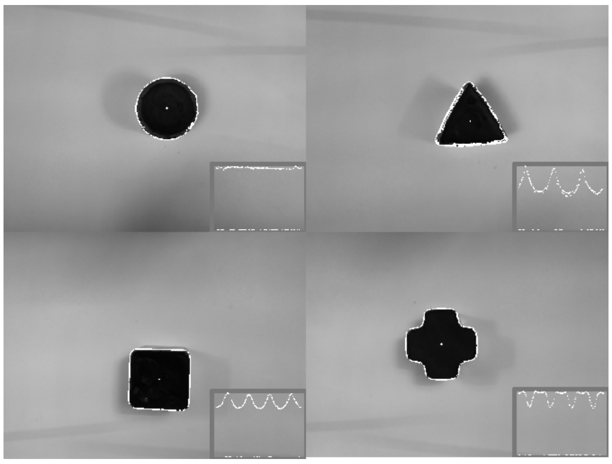


Figure 7. BOF extraction using the ES-BOF&FAM.

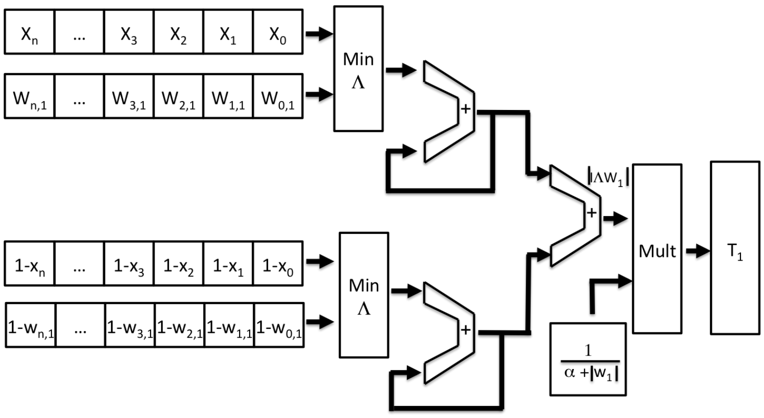
4.3. Digital Machine of the FAM Classifier

The BOF uses complement coding according to the Equation (6). Since the learning of the FAM is made offline, the weight vector W for each category T_j is initialised from the PS program. Either way, the weight vectors can be modified while the system is running, especially when there is a new category, although the total number of classifier structures are fixed from the biststream load to the PL. Figure 8 shows the architecture of the FAM classifier corresponding to a single category as given by Equation (7). The inputs X in this case, the BOF to be recognised, the weights and its complement vector are divided into two groups to make the fuzzy minimum comparison "$\wedge$" faster. Thus, the inputs and the weights are compared by the Min_$\wedge$ module. Simultaneously, the complement of both vectors (the input one and the weights) are compared by a module identical to Min_$\wedge$. The length of the n inputs is the size of the BOF vector, 180.

In each cycle, a pair of vector elements are processed and accumulated with the last pair. At first, the adder's feedback "+" is set to zero. After 180 clock cycles, the equation $|I \wedge w_1|$ is completed.

The factor $\frac{1}{\alpha+|w_j|}$ is calculated offline and registers on the PL. As mentioned, it is also accessible from the PS. In another module, also implemented in VHDL, the previous factor is multiplied by $|I \wedge w_1|$ which corresponds to the category T_1.

T_1 corresponds to a single object learned by the FAM ANN. Thus, for the recognition of more objects, as many T_j as desired are required. One of the advantages of using reconfigurable hardware is the possibility of generating one or more instances of any entity, provided there are free resources in the FPGA. The Figure 9 shows multiple FuzzyARTMAP classifiers corresponding to each learned object.



X_n … X_3 X_2 X_1 X_0
$W_{n,1}$ … $W_{3,1}$ $W_{2,1}$ $W_{1,1}$ $W_{0,1}$
Min $\wedge$
+
$|\wedge W_1|$
$1\text{-}x_n$ … $1\text{-}x_3$ $1\text{-}x_2$ $1\text{-}x_1$ $1\text{-}x_0$
$1\text{-}w_{n,1}$ … $1\text{-}w_{3,1}$ $1\text{-}w_{2,1}$ $1\text{-}w_{1,1}$ $1\text{-}w_{0,1}$
Min $\wedge$
+
Mult
T_1
$\frac{1}{\alpha+|w_1|}$

Figure 8. FAM classifier T_j, $j = 1$.

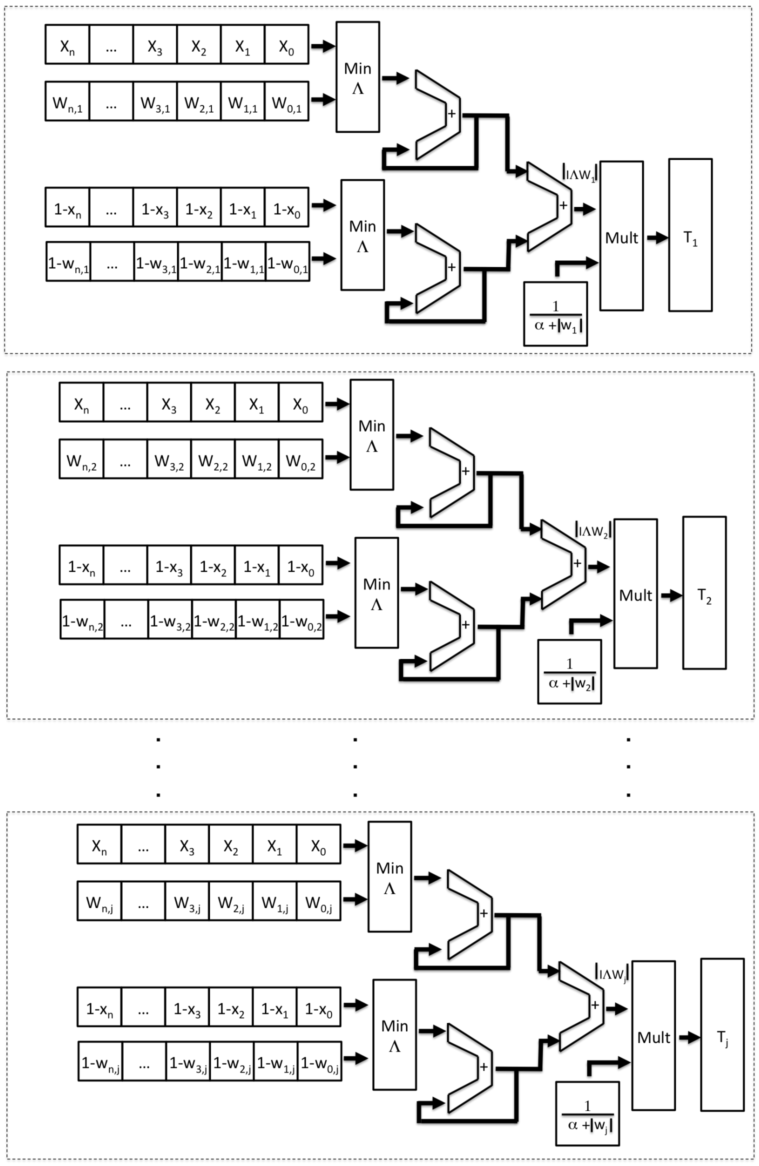

X_n ... X_3 X_2 X_1 X_0
W_n,1 ... W_3,1 W_2,1 W_1,1 W_0,1
Min Λ
1-x_n ... 1-x_3 1-x_2 1-x_1 1-x_0
1-w_n,1 ... 1-w_3,1 1-w_2,1 1-w_1,1 1-w_0,1
+
|I∧w_1|
Mult
T_1
1 / α +|w_1|
X_n ... X_3 X_2 X_1 X_0
W_n,2 ... W_3,2 W_2,2 W_1,2 W_0,2
1-x_n ... 1-x_3 1-x_2 1-x_1 1-x_0
1-w_n,2 ... 1-w_3,2 1-w_2,2 1-w_1,2 1-w_0,2
|I∧w_2|
T_2
1 / α +|w_2|
X_n ... X_3 X_2 X_1 X_0
W_n,j ... W_3,j W_2,j W_1,j W_0,j
1-x_n ... 1-x_3 1-x_2 1-x_1 1-x_0
1-w_n,j ... 1-w_3,j 1-w_2,j 1-w_1,j 1-w_0,j
|I∧w_j|
T_j
1 / α +|w_j|

Figure 9. FAM classifiers T_j.

When all the T_j have finished calculating, they are stored in several registers in the PL, which are accessed from the PS by a program that sorts them from highest to lowest using the bubble algorithm. Thus, the term T_j is obtained from the Equation (8) which corresponds to a possible winning category.

To corroborate the above, ρ of the Equation (9) must be satisfied. The upper term has already been calculated, however the lower term $|I|$ is calculated with a sequential accumulator every time we have a new element of BOF. If it is not fulfilled, it is necessary to repeat this process with another T_j.

4.4. The Processing System (PS)

The PS intervenes in most of the functions to read and write data between the different digital machines, acting as an orchestrator. It also performs the following tasks:

- Configuration of ports with peripherals.
- Configuration of digital machines for BOF; FAM and their parameters ρ; THD_BPm and THD_BPM; and others.
- TCP/IP communication over the stack lwIP (lightweight IP) with other nodes for sending and receiving commands. This is achieved through sockets.

- With an OpenShowVar implementation in C, one is able to send commands and receive status to/from the KUKA robot.
- Asynchronous serial communication for program loading and algorithm debugging. This is only applicable to certain occasions.
- Control of interruptions triggered by the socket.
- Reading the registers T_j from the PL through the AXI4Lite protocol and later their sorting using the bubble algorithm.
- Configuration of the VDMA IN and VDMA OUT blocks in case it is required to use them and they are included in the reconfigurable hardware.
- Configuration and control of the CCD OV7670 (e.g., image equalisation).

5. Tests and Results

The tests to compare the time taken for each of the steps, both for the extraction of the BOF and the FAM classifier, are CPU A and CPU B. Both computers have a Intel Core i7 processor (Santa Clara, CA, USA)with different model (3610QM and 2600S) and 8 GB RAM. CPU A runs at 2.3 GHz while CPU B runs at 2.8 GHz. For both computers, the method was developed in two programming languages, C++ and Matlab. The C++ compiler version was gcc 4.9 and, for Matlab, version 18 was used.

The Table 1 shows the time elapsed in each stage of the process for each platform, the two CPUs and the ES-BOF&FAM.

The test at the workstations was synchronised with a time measurement function. However, since the operating system uses the CPU extensively, the time obtained averaged over 1000 iterations. In addition, to lessen the effect of the operating system on the test, GNU/Linux was used since it is more likely to reduce the number of unnecessary processes at the time of the experiment.

Regarding what was implemented in the FPGA, the time measurement was performed in each stage by counting the number of clock cycles between the frequency of the main clock. This clock is 50 MHz, a value that depends on the speed of the pixel capture for the CCD sensor. However, the classification runs at 125 MHz.

The tests were carried out using the same image captured by the CCD sensor with a resolution of 640 $\times$ 480 pixels. A copy of this image was transmitted to the two CPUs. The used libraries are: OpenCV for C++ to load and process the image and the Graphics library for Matlab. In both cases, the image is stored in RAM. In contrast, the ES-BOF&FAM does not store any data in the RAM of the PS, nor in a BRAM of the PL; the processing is online, so that the capture of video frames is ahead one clock cycle to the calculation of the BP and the centroid. The difference between the performance of both CPUs, whether in C++ or Matlab, is very small. While it is observed that what was developed in C++ ends earlier than in Matlab. Note that the total elapsed time for object recognition in ES-BOF & FAM is less than a fourth part of the elapsed time using any CPU running the C++ program.

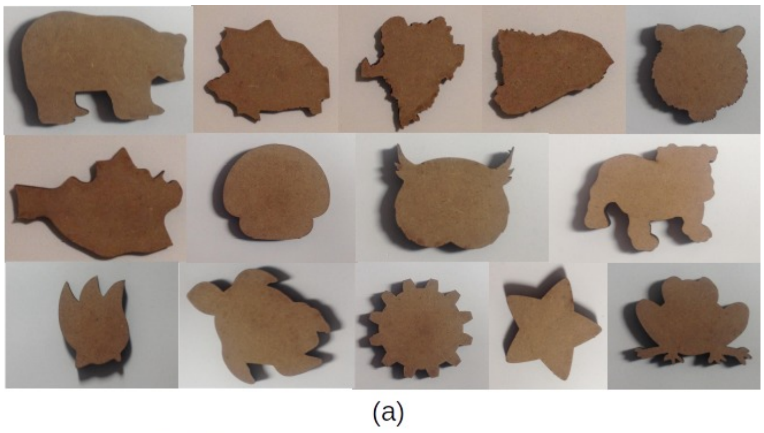
With the 15 objects shown in Figure 10, the ES-BOF&FAM was proven from which the mean of the MSE (Mean Squared Error) between results from ES-BOF & FAM and from the CPU with Matlab was 0.0042 $\sigma = 0.00014$. The logic resources are indicated in Table 2; here, it is shown that 9.09% of LUTs and 36.07% of available BRAM was used for the PL of the XC7Z010-1CLG400C. This utilisation includes raw image processing, BOF extractor (BOF module), and 100 FAM classifiers (100 objects). In order to figure out the amount of resources in case the FAM classifier increases, three iterations with 20, 100 and 200 FAM classifiers were calculated.

The orchestrator code uses 0.19% of the ZYBO's RAM (DDR 512 MB). It should be noted that the PS RAM is not used to store temporary data for image processing or the FAM classifier; this is done in the PL. OpenCV, in C ++, uses approximately 3.7 MB to store such an image. However, to calculate the I_WM, the PL only requires 3 BRAMs of 640 by 1 bit wide. This represents 0.7% of ZYBO's total BRAM.

Table 1. Elapsed time for each platform (time in µs).

	CPU A		**CPU B**		**ES**
	C++	**Matlab**	**C++**	**Matlab**	**BOF&FAM**
Video capture	9830	15,000	9600	14,000	6144
BP calculation and (Xc, Yc)	10,000	15,000	9980	15,000	6144 [†]
D y θ BOF	48	60	47	59	8
T_j	13,000	48,000	12,880	47,000	1440
ρ	39.5	50	39	49	18.43
Elapsed time	32,917.5	78,110	32,546	50,008	7610.43
Power consumption [W]	119		150		11.1
Estimated cost [USD]	$800		$1000		$301

[†] These processes are performed in parallel consuming the same time.



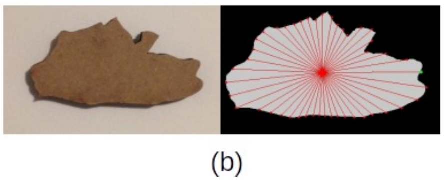


Figure 10. (**a**) Examples of objects for testing purpose. (**b**) Object and its BOF vector.

Table 2. Logic resources per module and total utilisation.

Logic Resources	**BOF**	**FAM_20**	**FAM_100**	**FAM_200**	**ES-BOF&FAM**	**Utilisation (Available)**
LUTs	1075	773	3763	7454	4838	9.09% (53,200)
Registers	468	688	3408	6818	3876	3.64% (106,400)
DSP blocks (DSP48E1)	2	1	1	1	3	1.36% (220)
BRAM (kbit)	35	350	1750	3500	1768	36.07% (4900)

In terms of the use of detectors, it is important to make a comparison with detectors such as SURF + BRIEF and the one presented in this work. These detectors are similar to the BOF, in terms that they are used for feature detection in object recognition tasks. In comparison, the BOF encodes on its own the object description.

Table 3 shows a comparison between the performance of a SIFT-based and SURF-based method for object detection implemented in FPGA + CPU and the one presented in this work.

Table 3. Performance comparison for object detection.

Logic Resources	[28]	[14]	**ES-BOF & FAM**
Algorithm	SIFT + SVM	SURF + BRIEF	BOF & FAM
Device	V5 LX110T [1]	Cyclone V [1]	XC7Z020 [1]
Board	ML509 [1]	DE0-Nano-SoC [2]	Zybo Z7-20
CCD sensor	Stored in CFcard	Aptina MT9V034 [1]	Omnivision OV7670
Resolution	640 × 480	640 × 480–1920 × 1080	640 × 480
Operating frec. [MHz]	50	136.3 *	50–125 ***
Registers	23,700	8177–8238	3876
DSP	64 of 64 (100%)	24 of 84 (28%)	3 of 20 (15%)
BRAM [kbit] (%)	4608 (46%)	1544 (55%)–2527 (91%)	1768 (36.07%)
# classes (# images)	5 (100)	5 (100)	100 (100)
Processing time [ms]	6.144	Det. 0.197 Desc. 6.36 **	1.466

* Maximum throughput speed after timing analyser, not the board frequency; ** Classification step is not included; *** CCD sensor and board speed. [1] Location: San Jose, CA, USA. [2] Location: Hsinchu, Taiwan.

General aspects to highlight in Table 3 is the use of less logic resources as follows:

- Registers: 3876 compared with 23,700 in [28] and 8238 in [14] for only 5 categories.
- DSP: 15% usage compared to 28% in [14] and 100% in [28]
- BRAM: 36% usage compared to 46% in [28] and in comparison up to 91% in [14]

Also, our solution does not use the PS RAM to process data. In ref. [14], the performance to detect characteristic points and calculate their descriptors is presented. However, it does not include the match and recognition time. Another advantage in our approach is that more information is obtained in a single step, such as the centroid of the object and the angle of rotation. In other cases, this information would still have to be obtained, increasing latency and overall processing time during object recognition for robot tasks.

6. Conclusions

The speed process has been faster with the Zybo Z7-20 module using software and digital machines, compared to the algorithm implemented in an ordinary computer. In the embedded system, the whole process takes a fourth the time spent on a computer with the same algorithm developed in C++. It is easy to observe that there is a big difference between the time required to calculate the distances D and the angle θ in the ES-BOF & FAM compared to the time for those same actions in the CPU. This time could even be shortened if a CCD sensor with higher capture rate were used. In terms of computational resources, the use of the RAM in the CPU for this method is not efficient because, in the best case, each video frame is stored only once and the space is overwritten in each processing step requiring 3.7 MB. In comparison, the only data stored temporarily in our BOF method is in the PL and it only needs $3 \times 640 \times 1$ bit of BRAM type memory. The communication framework with an industrial robot controller was tested, the robot responded to motion commands by moving the arm onto the respective object according to its centroid values. The gripper can be oriented for grasping purposes taking the orientation angle θ as a reference. The results presented in this paper were focused on the fast extraction of these parameters as well as the object learning through the BOF learning. The object's centroid and boundary points were calculated while acquiring the progressive image frame; all that was done with the intention of readying it for neural processing. The remaining time was devoted to recognise the object, which was always achieved with learning times as low as 1.466 ms which is rather encouraging for real-time applications with robots.

Multimodal learning processing will require smart robots to respond within the human average response time, about 200 ms, for a safe operation. In this regard, new collaborative robots in the shop floor would demand an increased capability to work with human operators in multimodal scenarios where vision, auditory and somatosensory information would fuse for processing motor coordination actions within this time frame. The results in this paper are a step forward to this direction demonstrating that the hardware integration of the video sensor, image processing, descriptor generator and the ANN classifier can accomplish cognitive decisions on a single chip.

Author Contributions: V.L.-B. and I.L.-J. contributed to the overall conceptualisation, design, implementation, programming of the FPGA as well as the training of the ANN. M.P.-C. conceptualised the adaptation of the BOF to its implementation in the FPGA. I.L.-J. and J.L.N.-G. contributed to the implementation of the ANN in the FPGA as well as its connectivity to the industrial robot. All authors contributed to writing-up the paper and agreed to the published version of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by UNAM-PAPIIT grant number UNAM-PAPIIT TA100619.

Acknowledgments: Thanks are sincerely due to Ricardo Garcia for his kind support in testing the system.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AGC	Address generator counter
ANN	Artificial Neural Network
BOF	Boundary Object Function
BP	Object boundary point
BRAM	Block RAM
ES-BOF&FAM	BOF&FAM Embedded System
I_WM	Weight matrix image
FAM	Fuzzy ArtMAP ANN
PL	Programmable Logic
PS	Processing system
THD_bin	Threshold image binarization
THD_BPM	Threshold boundary point max
THD_BPm	Threshold boundary point min
WMC	Weight matrix calculator
wmr	Weight matrix register

References

1. International Federation of Robotics. IFR Presents World Robotics Report 2020. 2020. Available online: https://ifr.org/ifr-press-releases/news/record-2.7-million-robots-work-in-factories-around-the-globe (accessed on 23 January 2021).
2. Pierson, H.A.; Gashler, M.S. Deep Learning in Robotics: A Review of Recent Research. *arXiv* **2017**, arXiv:1707.07217.
3. Peña-Cabrera, M.; Lopez-Juarez, I.; Rios-Cabrera, R.; Corona-Castuera, J. Machine vision approach for robotic assembly. *Assem. Autom.* **2005**, *25*, 204–216. [CrossRef]
4. Carpenter, G.A.; Grossberg, S.; Markuzon, N.; Reynolds, J.H.; Rosen, D.B. Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps. *IEEE Trans. Neural Netw.* **1992**, *3*, 698–713. [CrossRef] [PubMed]
5. Bribiesca, E.; Guzman, A. Shape Description and Shape Similarity Measurement for Two-Dimensional Regions. In Proceedings of the Fourth International Joint Conference on Pattern Recognition, Kyoto, Japan, 7–10 November 1978; pp. 608–612.
6. Chen, K.Y.; Hsu, P.H.; Chao, K.M. Hardness of comparing two run-length encoded strings. *J. Complex.* **2010**, *26*, 364–374. [CrossRef]
7. Bribiesca, E. A new chain code. *Pattern Recognit.* **1999**, *32*, 235–251. [CrossRef]
8. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
9. Bonato, V.; Marques, E.; Constantinides, G.A. A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection. *IEEE Trans. Circuits Syst. Video Technol.* **2008**, *18*, 1–11. [CrossRef]
10. Bouris, D.; Nikitakis, A.; Papaefstathiou, I. Fast and efficient FPGA-based feature detection employing the SURF algorithm. In Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines, FCCM 2010, Charlotte, NC, USA, 2–4 May 2010; pp. 3–10. [CrossRef]
11. Herbert, B.; Andreas, E.; Tinne, T.; Luc, V.G. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359.
12. Schaeferling, M.; Kiefer, G. Flex-SURF: A flexible architecture for FPGA-based robust feature extraction for optical tracking systems. In Proceedings of the 2010 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2010, Quintana Roo, Mexico, 13–15 December 2010; pp. 458–463. [CrossRef]

13. Battezzati, N.; Colazzo, S.; Maffione, M.; Senepa, L. SURF algorithm in FPGA: A novel architecture for high demanding industrial applications. In Proceedings of the Design, Automation and Test in Europe, DATE, Dresden, Germany, 12–16 March 2012; pp. 161–162. [CrossRef]
14. Cizek, P.; Faigl, J. Real-Time FPGA-Based Detection of Speeded-Up Robust Features Using Separable Convolution. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1155–1163. [CrossRef]
15. Ulusel, O.; Picardo, C.; Harris, C.B.; Reda, S.; Bahar, R.I. Hardware acceleration of feature detection and description algorithms on low-power embedded platforms. In Proceedings of the FPL 2016—26th International Conference on Field-Programmable Logic and Applications, Lausanne, Switzerland, 29 August–2 September 2016; pp. 1–9. [CrossRef]
16. Azouaoui, O.; Chohra, A. Soft computing based pattern classifiers for the obstacle avoidance behavior of Intelligent Autonomous Vehicles (IAV). *Appl. Intell.* **2002**, *16*, 249–272. [CrossRef]
17. Serrano-Gotarredona, T.; Linares-Barranco, B.; Andreou, A.G. ART1 and ARTMAP VLSI Circuit Implementation. Volume 456. Available online: https://doi.org/http://doi-org-443.webvpn.fjmu.edu.cn/10.1007/978-1-4419-8710-5_3 (accessed on 23 January 2021).
18. Wang, Y.; Xia, L.; Tang, T.; Li, B.; Yao, S.; Cheng, M.; Yang, H. Low power Convolutional Neural Networks on a chip. In Proceedings of the IEEE International Symposium on Circuits and Systems, Montreal, QC, Canada, 22–25 May 2016; pp. 129–132. [CrossRef]
19. Kyrkou, C.; Bouganis, C.S.; Theocharides, T.; Polycarpou, M.M. Embedded Hardware-Efficient Real-Time Classification With Cascade Support Vector Machines. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 99–112. [CrossRef] [PubMed]
20. Han, Y.; Oruklu, E. Real-time traffic sign recognition based on Zynq FPGA and ARM SoCs. In Proceedings of the IEEE International Conference on Electro Information Technology, Milwaukee, WI, USA, 5–7 June 2014; pp. 373–376. [CrossRef]
21. Damljanovic, A.; Lanza-Gutierrez, J.M. An embedded cascade SVM approach for face detection in the IoT edge layer. In Proceedings of the IECON 2018—44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 21–23 October 2018; Volume 1, pp. 2809–2814. [CrossRef]
22. Camuñas-Mesa, L.A.; Linares-Barranco, B.; Serrano-Gotarredona, T. Low-power hardware implementation of SNN with decision block for recognition tasks. In Proceedings of the 2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Genoa, Italy, 27–29 November 2019; pp. 73–76. [CrossRef]
23. Sanfilippo, F.; Hatledal, L.I.; Zhang, H.; Fago, M.; Pettersen, K.Y. Controlling Kuka Industrial Robots: Flexible Communication Interface JOpenShowVar. *IEEE Robot. Autom. Mag.* **2015**, *22*, 96–109. [CrossRef]
24. Carpenter, G.A.; Grossberg, S. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Comput. Vision Graph. Image Process.* **1987**, *37*, 54–115. [CrossRef]
25. Carpenter, G.A.; Grossberg, S.; Reynolds, J.H. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Netw.* **1991**, *4*, 565–588. [CrossRef]
26. Chinnathambi, M.; Bharanidharan, N.; Rajaram, S. FPGA implementation of fast and area efficient CORDIC algorithm. In Proceedings of the 2014 International Conference on Communication and Network Technologies, ICCNT 2014, Sivakasi, India, 18–19 December 2014; pp. 228–232. [CrossRef]
27. Nupoor, P.; Patil, D.R. Implementation of Artificial Neural Network on Temperature Control Process. *i-Manag. J. Electr. Eng.* **2018**, *12*, 26. [CrossRef]
28. Qasaimeh, M.; Sagahyroon, A.; Shanableh, T. FPGA-Based Parallel Hardware Architecture for Real-Time Image Classification. *IEEE Trans. Comput. Imaging* **2015**, *1*, 56–70. [CrossRef]