T.C.

ALTINBAS UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

OBJECT DETECTION AND RECOGNITION IN RGBD IMAGES USING FASTER R-CNN

Ahmed Hameed Neamah AL-NUSSAIRAWI

Master of Science

Supervisor

Asst. Prof. Dr. Abdullahi Abdu IBRAHIM

Istanbul, 2021

OBJECT DETECTION AND RECOGNITION IN RGBD IMAGES USING FASTER R-CNN

by

Ahmed Hameed Neamah AL-NUSSAIRAWI

Electrical and Computer Engineering

Submitted to the Institute of Graduate Studies

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2021

The thesis titled “OBJECT DETECTION AND RECOGNITION IN RGBD IMAGES USING FASTER R-CNN” prepared and presented by “Ahmed Hameed Neamah AL-NUSSAIRAWI” was accepted as a Master of Science Thesis in Electrical and Computer Engineering.

Asst. Prof. Dr. Abdullahi Abdu IBRAHIM

Supervisor

Thesis Defense Jury Members:

Asst. Prof. Dr. Abdullahi Abdu IBRAHIM	School of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Timur INAN	School of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Tareq MOHAMMED	Computer Science and Information Technology, Imam Ja’afar AL-Sadiq University	_____

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Approval Date of Institute of Graduate Studies:
____/____/____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Ahmed Hameed Neamah AL-NUSSAIRAWI

DEDICATION

I would like to thank Allah Almighty for the power of mind, health, strength, guidance, knowledge and skills to complete this study.

This thesis is wholeheartedly dedicated to my parents. There are no words to describe what you mean to me; there is nothing that I can repay for what you have done to me. I will continue to do my best to achieve your expectations.

Lastly, I dedicated this to my wife, brother and sisters, relatives, and friends who have been encouraging me during this study.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Asst. Prof. Dr. Abdullahi Abdu IBRAHIM for all support during my study. Its great pleasure to express my deepest gratitude to my friends who have shared with me best moments during my study for the Master degree.

ABSTRACT

OBJECT DETECTION AND RECOGNITION IN RGBD IMAGES USING FASTER R-CNN

AL-NUSSAIRAWI, Ahmed Hameed Neamah,

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Abdullahi Abdu IBRAHIM

Date: 09/2021

Pages: 63

The process of splitting the image into many segments can be called as image segmentation. The main aim of this process is to recognize the object in an image. Let's take an example a person wants to cross the road the first thing he'll do is he'll look at his right side first then left. If no vehicle is coming then the person will try to cross the road apart from that, this person will also check whether there is any electric poll or dog or is there any whole in the ground etc. The point is we're doing object detection first then we're crossing the road. Same thing we'll train our machine to do but it can't recognize directly the way humans do. We have to train the model first for that we're doing image segmentation. This step is typically used to recognize the objects or other relevant information.

Keywords: RCNN, Object Detection, Segmentation, Deep Learning, RGB.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

1. INTRODUCTION

1.1 BACKGROUND

The process of splitting the image into many segments can be called as image segmentation. The main aim of this process is to recognize the object in an image. Let's take an example a person wants to cross the road the first thing he'll do is he'll look at his right side first then left. If no vehicle is coming then the person will try to cross the road apart from that, this person will also check whether there is any electric poll or dog or is there any whole in the ground etc [7]. The point is he's doing object detection first then he's crossing the road.

Same thing we'll train our machine to do but it can't recognize directly the way humans do. We have to train the model first for that we're doing image segmentation. This step is typically used to recognize the objects or other relevant information.

The role of image segmentation is very important in image processing. The splitting an image in multiple chunks make it easy for further process thus after completing the operations image will be re-joined. Segmentation increases the accuracy of recognizing the object in an image and reduce the loss. Semantic segmentation and instance segmentation are the available types of image segmentation based on the problem we use image segmentation:

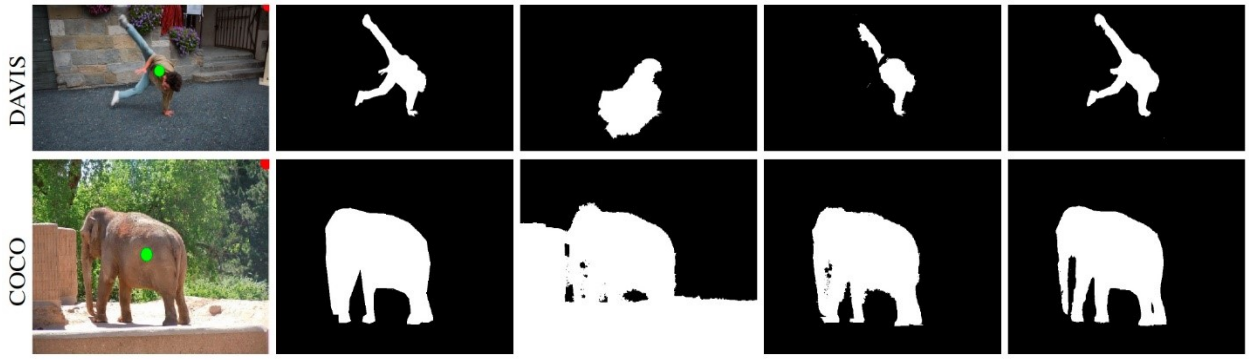

Figure 1.1: Image Segmentation Using Multiple Approaches [1]

1.2 PROBLEM STATEMENT

The object detection task focuses on finding an object of interest in the scene, with no prior information available about the target's initial position and orientation. In the case of 6-DoF tracking, where the initial position and orientation of the object is generally known in the scene, the objective of the method is to record the movement of the object continuously during a sequence of frames of the scene. In order to track and detect objects in 6-DoF in real time, the algorithms seek to understand the characteristics of the scene, extracting information about the object of interest at runtime, and to perform its correlation with information previously acquired from the object to estimate the pose of the object, either through manually selected information (techniques commonly called Geometric or Classic Computer Vision) or using information selected through machine learning techniques Most of the methods currently proposed deal with information from RGB sensors (cameras) or RGBD (depth sensor cameras, the latter being able to obtain better accuracy between state of the art works due to extra depth information available The big problem with RGBD methods is the cost of obtaining the sensor, which is generally more expensive than RGB cameras. RGBD cameras also have limitations such as the influence of sunlight that can affect the accuracy of the sensor, and the distance from the image obtained. Recent methods seek to apply RGB images together with algorithms based on machine learning methods Some approaches manage to obtain results similar to RGBD sensors using only the color information, causing in recent years, a strong resumption in the investigation of methods for RGB cameras. The great limitation of these methods is related to the performance of the algorithm. Generally, the learning algorithms that deal with RGB images have high execution time, or need refinement processes that hinder the execution time of the algorithm. More recent techniques based on deep neural networks (DNN, Deep Neural Networks) use models capable of maintaining high performance in the execution time and obtaining good results of precision of the estimation of pose possibly without the need for post-processing, in models called single shot. However, the limitation of these methods lies in the high amount of data necessary for training the models and the need to obtain high computational power devices for their execution. Obtaining and labeling data is a process that requires a lot of time and effort. Some databases can be found publicly with images and notes for validation of the developed techniques but for applications that require specific objects and information, obtaining data can be a major problem in development. Another problem is that generally the available data is aimed at testing applications of known and / or

controlled scenarios, making it difficult to evaluate the performance of the model for several challenges that may occur in real applications due to changing the sensor, environment, lighting, etc. To overcome this problem, authors are looking for ways to synthetically generate training data. Generating images from 3D models of the objects of interest), using data adaptation techniques to automatically obtain a set of information similar to a small set of real images known or generating a large amount of variation in the data in order to induce greater generalization of the learning models 16 used

1.3 MOTIVATION

Detection of objects using semantic segmentation is a task widely studied in the area of Computer Vision. The use of detection and tracking has applications in several areas, such as augmented reality among others. In augmented reality, the real environment is part of the context of the application, and the objective is to increase the user's perception of the real world using the virtual information entered through tracking, enabling a natural interaction for the human-machine relationship. Because of this, these applications have a high requirement of necessity and must occur in real time, allowing a fluid interaction between the user and an application. In robotics, the movements and actions of autonomous robots are carried out by tracking information inserted in the environment, enabling a robot with objects of interest, manipulation of objects and avoiding collisions during movement. It is possible to apply detection and / or tracking to perform various tasks that involve the recognition and understanding of the environment, enabling an interaction with real objects and scenarios. Some examples are the location and movement of autonomous cars and robots, facial recognition applications, identification and manipulation of real objects, rendering of virtual models in environments, scene recognition for virtual reconstruction of the environment, among others.

1.4 CONTRIBUTION

This work presents two main contributions: presentation of a new method for automatic detection of motorcyclists without a helmet; and a new hybrid descriptor based on geometric characteristics, border and texture. The aim of the study is to develop an automatic detection system for motorcyclists without a helmet, creating a strategy divided into two steps: motorcycle detection and detection of helmet use.

1.5 THESIS ORGANIZATION

The thesis is structured as follows: Section 2 is where we review some of the previous work and implementation on smart grids. Section 3 is where we give a background about all the components. Section 4 is where we explain our model in details and implementation of that model in details, Section 5 is where we simulate our model and record the results obtained. Section 6 is where we conclude our work and put a future scope into perspective.

2. LITERATURE REVIEW

Image segmentation is a most important part in the image processing, it is used almost everywhere to process the images so our model should be able to recognize what's inside the image. The segmentation splits the image into many sections or objects. The level to which the splitting the image is being carried rely on the problem is which has been solved. When the object of an image has been segregated, segmentation should stop on that time. For example, we have an image so our aim is to recognize the objects into the image, for that we segment the image the details should be visible so our model work if there is a presence of an outliers or the paths aren't clear or broken. It's of no use to segment the image it may not be able to identify those elements.

The base of an image segmentation is having two basic properties. Either it will be discontinuity or likewise. In this, the first division is the approach of an image partition is based on sudden changes. We can take an example of edges in an image. While in another category it is segmenting an image based on region similarities according to predefined criteria. Like thresholding, whether the region is growing or splitting or merging. There are several methods to detect object in an image such as points, lines and edges etc. These methods can be implemented using several threshold techniques. These thresholds can be fixed based on the region oriented segmentation approaches The author Vijay Badrinarayanan has proposed Semantic pixel-wise is currently the famous topic in the research field. The deep fully convolution neural network architecture for semantic segmentation called as SegNet. The learning capability of segmentation is having an encoding network, to the equivalent decoder for the network come after by the pixel-wise classification. The role of decoder is to map the low resolution encoded features maps for pixel-wise classification. The SegNet lies in this way in which it has to put input feature and the decoder uses un-samples its lower resolution. For performing non-linear sampling, the decoder usually uses pooling technique [1].

The author Liang-Chieh Chen has proposed the network which can explore the incoming features by using pooling techniques or filters that has the ability to encode information which can be multi-scale. The productive area of view at several rates by slowly spatial information has increased while the networks are able to capture the sharp boundaries. Here the author extends DeepLabv3, in this to clarify the segmentation results along with the object boundaries used to add an effective decoder module. For further process, the author is exploring the xception model and then it's

applying the depth wise separable convolution of both Atrous spatial pyramid pooling and decoder module [2].

The authorZongwei Zhou is presenting UNet++ for medical image segmentation. It is the most powerful architecture in this sector. It is a kind of supervised network where it is connected with encoder and decoder. It's using dense pathways connected with a series of nested structure. The aim is to reduce the semantic breach what's happening in between encoder-decoder sub network by redesigning the skip pathways. The author has to evaluate UNed++ using datasets of four medical imaging that covers cell nuclei segmentation, liver segmentation, colon segmentation, lung nodule segmentation. The experiment has been recoded that it has successfully IoU gained 3.9 and 3.4 over the U-Net and the wide U-Net by using the U-Net++. [3].

The author Shir Gur has presented unsupervised segmentation of blood vessels has been presented by author by using a deep learning. The method has been inspired by active contours. In this, the author has introduced a new term which has no edges optimization method (ACWE) with morphological engage which is representing the boundaries.

The novel pooling layers are playing the role of morphological operator that is absorbed by the network architecture. It provides the art of solution based on novel losses and new types of layers also turning classical and powerful computer vision case study for deep learning methods [4].

3. MATERIALS AND METHODS

3.1 INTRODUCTION

In this chapter, each of the necessary steps to implement the R-CNN architecture will be presented, applied to the vehicle detection problem. Thus, the chapter is divided between each of these steps, which are:

a. data pre-processing.
b. CNN training.
c. refinement of the classified regions.

The area of image processing has long been an object of growing interest as it allows the feasibility of a large number of applications, mainly with regard to the improvement of information for human interpretation and automatic computer analysis of information extracted from a scene. However, the big boost in the area of Image Processing came with the advent of the first large digital computers and the beginning of the North American space program, which used computational techniques for image enhancement at the Jet Propulsion Laboratory, when images of the moon were transmitted were processed by computer, using enhancement and restoration techniques in order to correct distortion of the TV camera attached to the probe (FILHO; NETO, 1999). Currently, the image processing area has been very promising, showing great growth and its applications permeate almost all branches of human activity. In medicine, the use of images in medical diagnosis is routine and has allowed the development of new equipment, as well as greater ease in interpreting images produced by the equipment. In Biology, the automatic processing of images from microscopes, facilitates the execution of laboratory tasks that require high precision and repeatability. Geography areas have benefited from the processing and automatic interpretation of satellite images in remote sensing, meteorology and reprocessing. Automatic image restoration techniques help archaeologists recover blurred photos of rare, often destroyed artifacts.

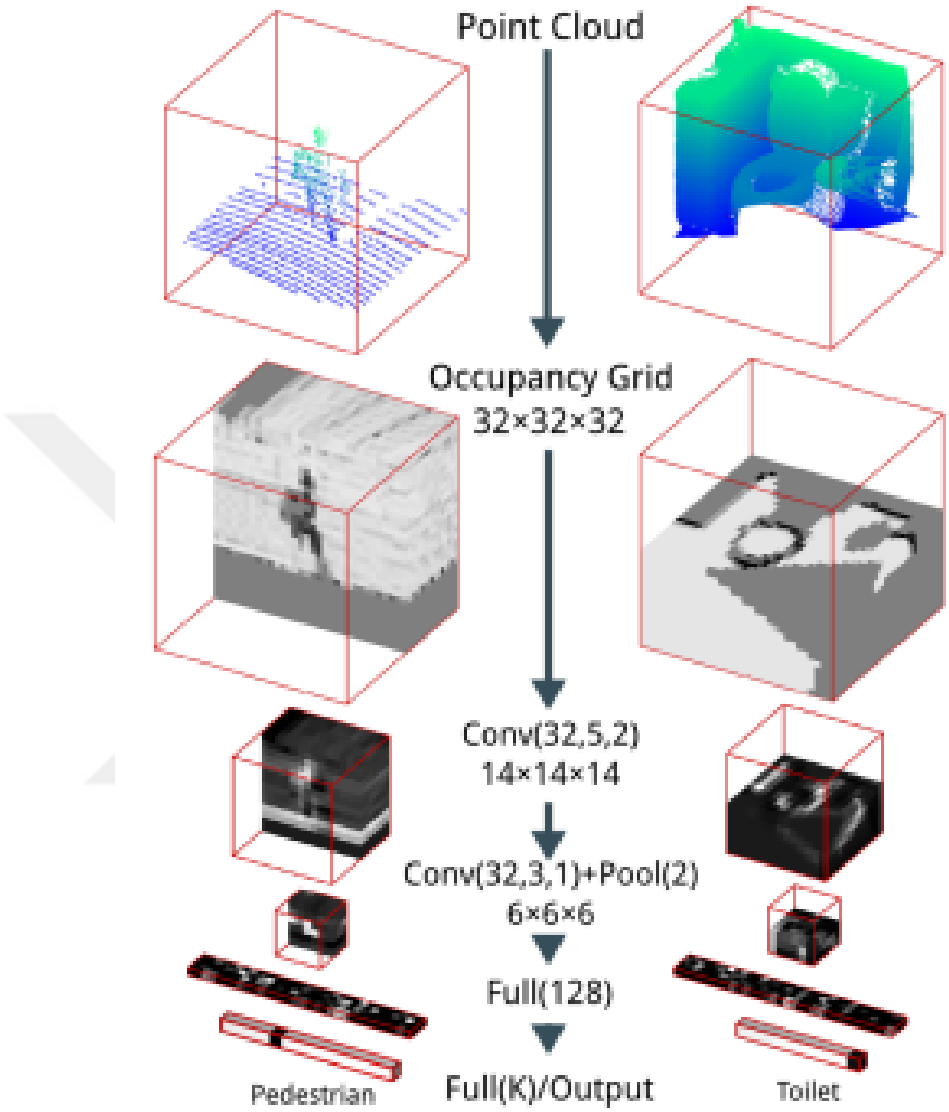

Figure 3.1: Point Cloud 2D Processing

Segmentation has the function of dividing the image into regions and distinguishing these regions as independent objects from each other and from the background. 57 It is usually the critical stage of the standard PADI sequence, as it is through it that objects of interest are recognized and identified, on which the next stage of extracting image characteristics will be carried out.

Extraction of Characteristics from the Image The stage of characteristics extraction, aims to extract characteristics from the images resulting from the segmentation using descriptors that allow to accurately characterize each digit and that present good discrimination between similar digits, such

as the '6' and the '5'. These descriptors are represented by a data structure suitable for the recognition algorithm, in this step the input is an image, but the output is a set of data corresponding to that image. There are two types of measurements in the feature extraction stage, field measurements and region measurements.

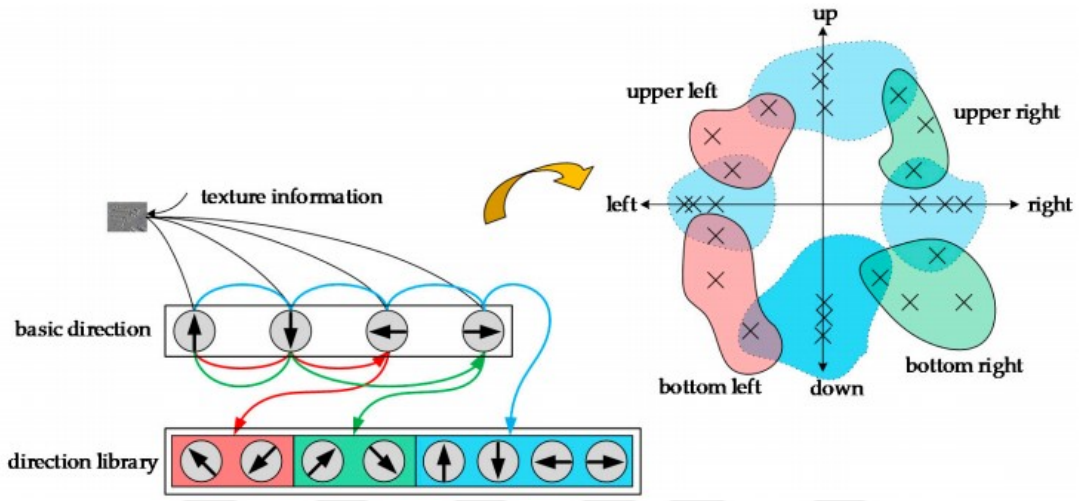

Figure 3.2: Texture extraction

Field measurements are made on the image, in order to fully characterize it, and can also be done in subfields of the image, characterizing them individually as different images. Field measurements are generally divided into measures of:

a. Texture.

b. Counting of objects.

c. Intersections.

d. Area.

e. Intensity.

The measurements of the region are carried out on the objects segmented in the image, in order to characterize them individually, they are generally divided into measurements from:

f. Size.

g. Texture.

h. Shape.

i. Intensity.

j. Position.

Image Recognition and Interpretation The image recognition and interpretation step is the final step of the standard PADI sequence. In this step, recognition is the process of assigning a label to an object based on its characteristics. The task of interpretation, on the other hand, consists in assigning meaning to a set of recognized objects. The techniques of recognition and interpretation, allow the treatment of the quantitative data obtained in the stage of extraction of characteristics of the image, interpreting them, in order to provide a result of a higher level. Thus, this stage constitutes a transformation of information into knowledge. Pattern recognition aims to build a simpler representation of a data set, by means of its most relevant characteristics, which allows its division into classes. In PADI, pattern recognition and interpretation techniques can be used to classify objects in an image, as can be seen in Figure 3.8 and Figure 3.9 The use of robots is also frequent, through artificial vision in tasks such as quality control in production lines, agribusiness and numerous other areas such as Law, Astronomy, Advertising, Security, among others. In all the areas highlighted above, machine learning and CNN can be used. When using images as input to a neural network for classification or pattern detection, some steps are common in all applications. The images need to be labeled, that is, for each image used, information is needed to classify it according to the characteristics that are desired to be learned by the neural network. 49 The images must be obtained in conditions that allow sufficiently general knowledge, for example, if you want the network to work in different light conditions, the images must be collected in different illuminations.

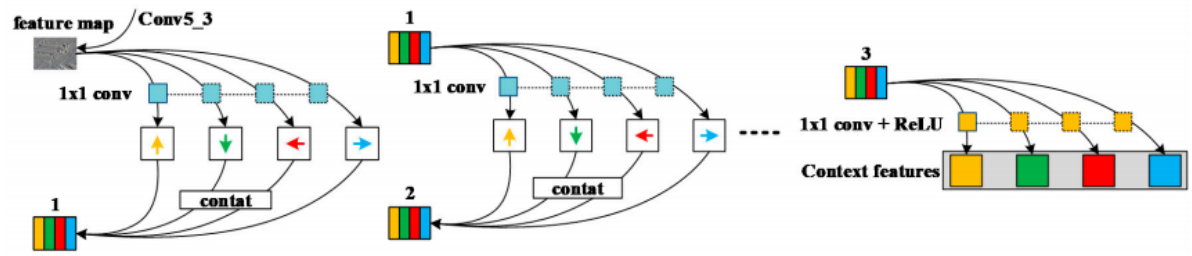

Figure 3.3: Convent CNN Network

The learning by deep neural networks that is intended in this work depends on that the images contain sufficient variations of elements that are not relevant for the classification (generalization) and sufficient examples of what is to be registered as a classification (specialization). The balance between these conditions is more easily achieved with large volumes of images and long periods of training as discussed in Chapter 6. In Figure 3.1 it is possible to observe a comparison between the human visual system and an artificial vision system According to Acharya et al. (2006), for good image processing, understanding the sampling and quantization processes is of paramount importance. For an image to be computationally processed, that image needs to be digitized, so it is necessary to perform quantization and sampling of the analog image. According to Gonzalez et al. (2002), Woods et al. (2008), sampling is discretization in space, whereas quantization refers to discretization in amplitude. 51 For ASSUMPÇÃO (2013), the sampling process occurs with the conversion of the analog image into an image matrix with A x B dots (pixels), which defines the image resolution. The most frequent sampling is called uniformly spaced and each sample is taken at equal intervals, however there are other sampling techniques that use spacing of different sizes, being less common. Filho and Neto (1999), state that the higher the values of A and B, the higher the image resolution, which directly interferes with the image quality, which also depends on the requirements of a particular application. However, higher resolution results in greater computational complexity, which is necessary to perform image treatments and also a higher storage cost.

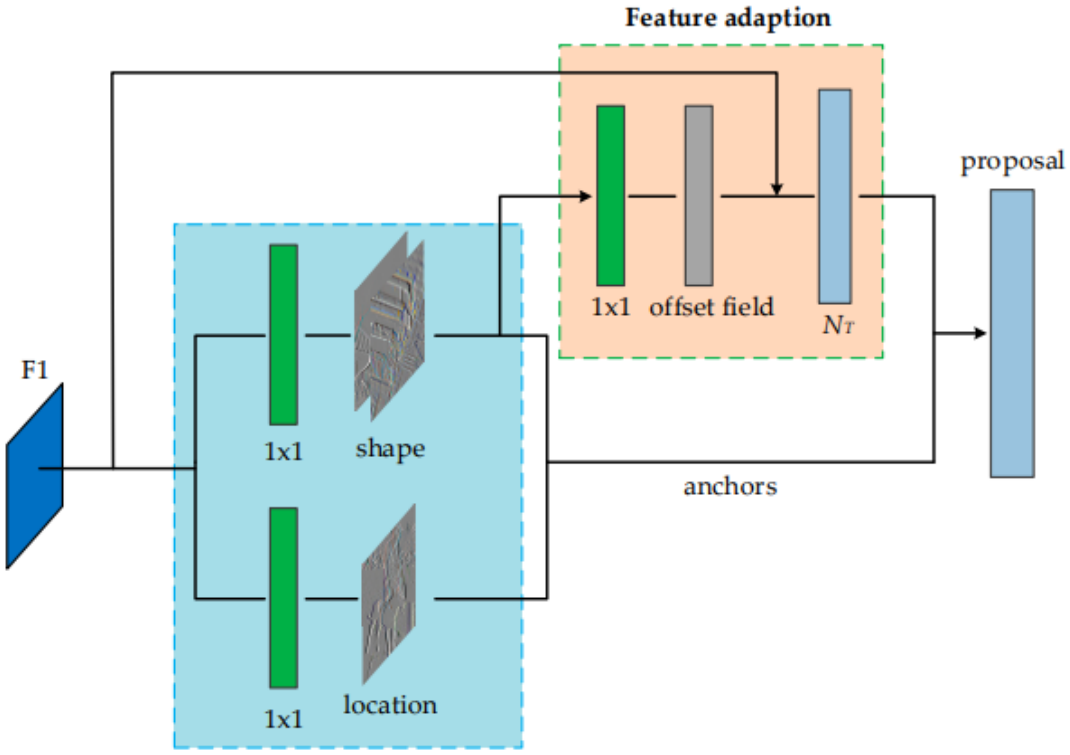

Figure 3.4: Feature Adaptation in CNN

It is possible to compare the same image with different quantization's and thus illustrate the effects of reducing the number of gray levels on the image quality as can be seen in Figure 3.2, the lower the number of gray levels, the more noticeable the appearance of an imperfection in the image, known as a false outline (FILHO; NETO, 1999). During quantization, each pixel value starts to assume an integer value, varying from 0 to 2n-1, with n being the number of shades of gray. The resulting image will then have 2n possible shades of gray, with n being the number of bits needed to represent each pixel. 52 3.2 The RGB Color Space According to Gonzalez et al. (2002) in the RGB color model, each component is represented in its primary form red, green, blue or Red, Green, Blue, in this model based on a Cartesian coordinate system, all other colors are composed from these three colors primary. The colors in this model are represented as points within the cube as can be seen in Figure 3.3.

3.2 DATA PRE-PROCESSING

Analyzing the R-CNN pipeline, it is clear that, to solve the object detection task, CNN must be able to classify regions of the image, obtained from a Region Proposal Algorithm, in this case, Selective Search. Thus, CNN training must be carried out from regions of the images, whether or not they contain vehicles. Considering that, initially, only the original images are available, a pre-processing step is necessary in order to have a coherent database for training the network. In this way, the database that will be used for training the network will consist of regions containing vehicles divided into the following classes: 'car' (cars), 'bus' (buses),' van '(vans) and' others' (mainly trucks, but, are vehicles not belonging to the other classes). However, a class has been assigned to regions that do not contain any vehicles, defined as ‘background’. Figure 10 illustrates some examples of the classes. The generation of the regions that will compose the training set is carried out from the application of Selective Search on the images resized to 224 × 224 of the database. Otherwise, the time required to perform Selective Search on all images in their original size, approximately 20 days, would make the work very costly. Regions proposed by Selective Search that have an IOU greater than 0, 5 with a Ground Truth present in the image will be considered valid samples for the associated Ground Truth class. However, a sample can only be considered negative, called background, if the region presents a maximum IOU of 0, 3 with any Ground Truth present in the image. In addition, for this specific problem, other criteria were used to define the background samples that will be presented in the next.

$$\|X\|_p = \left((x_1)^p + (x_2)^p + \cdots + (x_n)^p\right)^{\frac{1}{p}}. \tag{7}$$

The loss function of L2 regularization is

$$L(\omega) = L_D(\omega) + \frac{\lambda}{2n}\sum_{i=1}^{n}\omega_i^2. \tag{8}$$

The gradient of the loss function $L(\omega)$ is calculated by

$$\frac{\partial L(\omega)}{\partial \omega} = \frac{\partial L_D(\omega)}{\partial \omega} + \lambda\omega. \tag{9}$$

The parameter ω is updated to become

$$\omega' = \omega - \eta\frac{\partial L(\omega)}{\partial \omega} = \omega\left(1 - \frac{\eta\lambda}{n}\right) - \frac{\partial L_D(\omega)}{\partial \omega} \tag{10}$$

The first step in the process is the acquisition of images that aims to obtain a digital image, which is a file basically composed of a header, with various information and a matrix of numbers, which identifies the color or intensity of the pixel of corresponding position in the image. Thus, this matrix is constituted in a map that reproduces the image pixel by pixel. A sensor and a digitizer are required. The function of the sensor is to convert the optical information into an electrical signal and the function of the digitizer to transform the analog image into a digital image. Other important aspects include the choice of sensor type, the set of lenses used, the lighting conditions, the speed of acquisition, the resolution, among others. This step produces a digitized image at the output. Therefore, the acquisition step aims to convert an image into a numerical representation suitable for digital processing that can be represented by bits 0 and 1. The pixel, whose abbreviation is picture element, is the basic unit of digital imaging. 56 It is a spatial resolution that consists of the size, the real image that a pixel of the digital image represents, so the resolution is the maximum discrimination capacity of two points in the image. Image pre-processing The image resulting from the acquisition may present imperfections, such as the presence of noisy pixels, inadequate contrast and / or brightness, interrupted data. Thus, the function of the digital image pre-processing step is to improve the image quality for the following steps. The operations carried out in this step work directly with the pixel intensity values, the image resulting from this step is a digitized image of better quality than the original. Thus, pre-processing is the step that aims to improve the image, correcting defects generated during its acquisition, being ready for the

segmentation step. Image Segmentation The segmentation step digitally reproduces the task of recognizing regions of an image as objects, the basic task of the step being to divide an image into its significant units, that is, in the objects of interest that compose it, is an extremely sophisticated process realized by human vision. Figure 3.7 shows an example of image identification.

Table 3.1: IRNN Feature Map

mAP (%)	Time (sec)	Number of IRNNs				
		1	2	3	4	5
70.5	0.58	√				
72.2	0.71	√	√			
76.7	0.79	√	√	√		
76.9	0.84	√	√	√	√	
77.1	1.63	√	√	√	√	√

3.3 CNN TRAINING

After the data pre-processing step, the database necessary for CNN training to be carried out is obtained. However, training a deep CNN takes time. Networks known as ResNet, VGG or Inception (HE et al., 2016) (SIMONYAN; ZISSERMAN, 2014) (SZEGEDY et al., 2015) report training time of weeks. To solve this problem, the Transfer Learning technique was used (YOSINSKI et al., 2014). The objective of this technique is to take advantage of the weights of networks that were previously trained for a problem similar to the problem to be solved, in this case, vehicle classification. The weights and initial layers of a pre-trained network are used, with the last layers being discarded and new layers being trained so that the new problem is solved. The premise used is that the initial layers of the network tend to learn abstract characteristics of the images such as shape, contours and textures, while deeper layers will be more specialized (YOSINSKI et al., 2014). In this way, it is possible to take advantage of initial layers and retrain the final layers to solve the new problem. The network used to carry out Transfer Learning was VGG16 (SIMONYAN; ZISSERMAN, 2014). VGG weights and convolutional layers were used, pre-trained for the classification problem proposed by ImageNet (DENG et al., 2009), and the last

layers of VGG, the Fully Connected Layers, were discarded to insert new layers that will be trained for the current problem. Figure 11 shows the architecture of the VGG.

3.4 REFINEMENT OF CLASSIFIED REGIONS

After the classification of the images proposed by Selective Search, the result of object detection is obtained. However, it is possible that there are redundant BBs, that is, several BBs can be associated with a single object, as shown in Figure 12. Based on this premise, a stage of refinement of the classified regions is necessary. The Non-Maximum Suppression algorithm is applied to remove BBs that have a high IOU between them. In addition to the use of Non-Maximum Suppression, the problem in question has particular challenges that require further steps to refine the classification that will be detailed in the next chapter. At the end, the proposed regions are obtained, the classification of the regions via CNN and the refinement of the classified regions. In this way, the R-CNN architecture pipeline is complete.

Table 3.2: mAPs In Time Training Methods

Method	Train	mAP (%)	Time (sec)
FR	07 + 12	70.8	2.2
RS-FR	07 + 12	73.2	1.1
RC-FR	07 + 12	71.7	0.6
SC-FR*	07 + 12	75.8	2.9
SC-FR	07 + 12	77.6	0.8

3.5 SUMMARY

This chapter focuses on explaining the steps required to develop the proposed solution to the problem under study. The next chapter will present details of the implementation of these steps as well as the results obtained

(1) Comparison of detection accuracy:

- When detecting small objects:

$$SC - FR > SC - FR* > RC - FR > RS - FR > FR$$

- When part of the object is occluded:

$$SC - FR > SC - FR* > RS - FR > RC - FR > FR$$

(2) Comparison of detection efficiency:

$$FR > RC - FR > RS - FR > SC - FR > SC - FR*$$

Figure 3.5: SC-RF Object Recognition in MRCNN

4. PROPOSED METHOD

4.1 INTRODCTION

In this chapter, the results obtained after the implementation of the proposed architecture are presented. The chapter begins by describing the database used, the pre-processing step and the training. Then, metrics are presented to evaluate the performance of the architecture as well as the result of the experiments. Finally, a comparison is made between the results of this work with other architectures used applied to the same database.

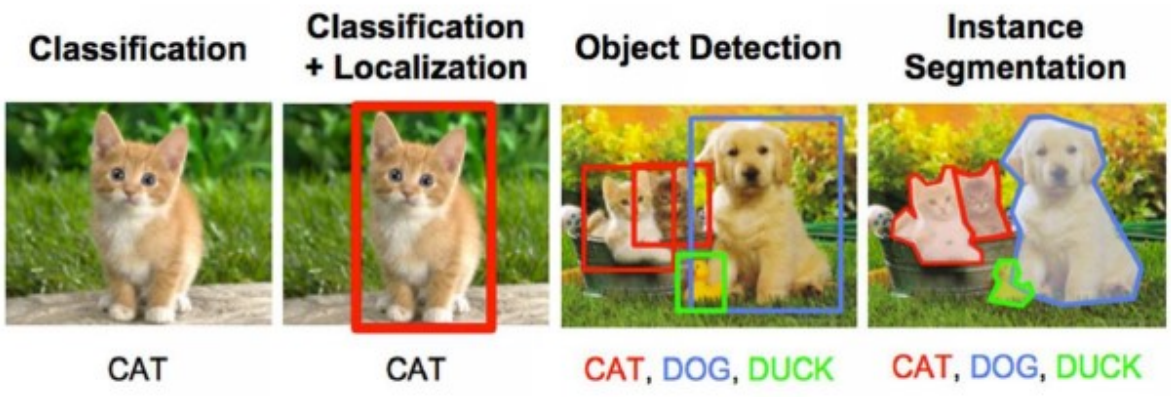

Figure 4.1: Different Objects Recognized in an Image

4.2 OMPUTATIONAL RESOURCES

Software Resources The training and testing stages of the proposed architecture were carried out using Keras with backend based on Tensorflow, open source software developed by Google brain Team1, intended for numerical programming using programming based on data flow in graphs (ABADI et al., 2016). Despite being used for other purposes, due to the wide support for the use of GPUs, Tensorflow is often used in ML and DL problems to accelerate the training of neural networks. Hardware Resources The hardware necessary for the development of the work is restricted to, at least, a computer with sufficient processing capacity for the training of the CNN architecture that has been implemented. Thus, the work was carried out, mainly, on the author's personal computer. In order to speed up the training and testing process of the implemented networks, in addition to the author's personal computer, computers available in the VIROS and

CISNE laboratory at UFES were used. These computers have the following configuration: (i) Linux operating system, Ubuntu Server 16.04 distribution; (ii) Intel Core i7-7700 processor, 3.60GHz with 4 physical cores; (iii) 16 GB RAM memory; (iv) 1TB storage unit (hard disk); (v) Nvidia Geforce GTX 1080 video card, with 8 GB of dedicated memory. UA-DETRAC database The database used in the work, for training and validation of the object detector implemented based on CNN, was the 1 DETRAC database, presented by (WEN et al., 2015). DETRAC consists of 10 hours of video recorded by traffic monitoring cameras located in 24 different points in the cities of Beijing and Tanjin, China. The videos were recorded at a rate of 25 FPS. Thus, the database has more than 140,000 frames with a resolution of 960 × 540 pixels, totaling 1.21 million BB of labeled objects and 12 Gb of data. presents images that make up the database

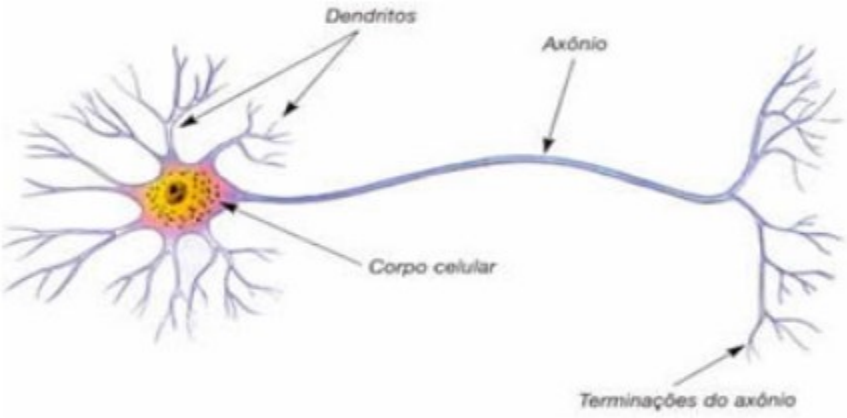

Figure 4.2: Neuron Structure

4.3 IMPLEMENTATION

This section is dedicated to providing more detailed explanations of the implementation of the architecture applied to the specific vehicle detection problem. The subsections are divided in order to highlight the particularities found in the implementation steps related to:

a. intrinsic characteristics of the database used,

b. specific implementation problems for training a CNN from a large database,

c. techniques and criteria used in the refinement stage of the classified regions.

4.3.1 Particularities of the Database

4.3.1.1 Scale

Figure 14 presents two examples of images present in the database, highlighting the difference in the images obtained due to the positioning of the cameras in relation to the highways being different. The existence of a difference in scale is noticeable in the images generated by cameras parallel to the highway. Vehicles close to the camera are represented by larger BBs and vehicles distant from the camera are represented by smaller BBs. Within the same vehicle class, there may be positive examples with different scales. The images generated by perpendicular cameras further accentuate the problem of scale difference, since they produce positive examples with BB larger than those of a parallel camera and change the relationship between width and height of the BB observed for images generated by parallel cameras. In short, the scale problem directly influences two points:

a. definition of the parameters to be used by the Selective Search algorithm.

b. filtering of proposed regions.

The filtering of the proposed regions provided by Selective Search is performed based on the intrinsic characteristics of the images available in the database. In this way, regions with a height and width ratio that exceed the pattern observed between the vehicles are discarded, since the objective is to find only the vehicles present in the image. In addition, considering the pattern observed in the images generated by cameras parallel to the highway, the maximum size that a valid proposed region can have is proportional to its position in the image, that is, regions located at the top of the image have a lower acceptable maximum size. to regions located at the bottom of the image. Figure 15 illustrates the influence of this filtering. The use of Selective Search requires three parameters to be previously defined: scale, sigma and minimum size of the proposed regions. Different combinations of these parameters will result in different sets of proposed regions. All three parameters are described in (FELZENSZWALB; HUTTENLOCHER, 2004). The minimum size defines the smallest area value, in pixels, of a region so that it is a valid proposed region. The sigma parameter corresponds to the width of the Gaussian filter used by the segmentation technique. Finally, the scale parameter defines the maximum possible size of the pixel groups for the proposed regions, that is, the larger this parameter, the larger these groups will be. The

definition of these parameters is arbitrary, therefore, each one of them was isolated in several images so that the values were defined so that vehicles of different scales could be covered by the largest number of proposed regions. Figure 16 illustrates the variation made in each parameter.

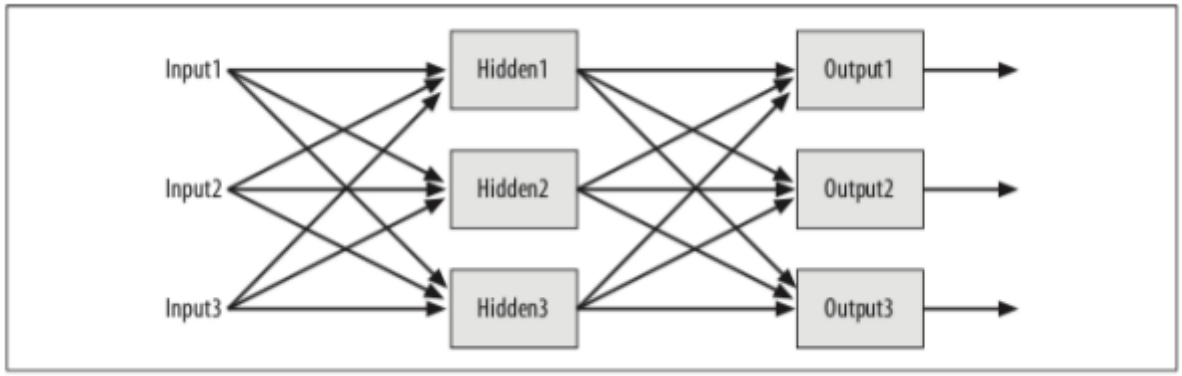

Figure 4.3: ANN Structure Similar to Neuron

4.3.1.2 Ignored Regions

The database defines some zones in each image that are not considered, containing parked cars or cars in small dimensions in the image. Thus, using regions that include parts of the ignored zones as negative vehicle samples could make CNN's training unfeasible, since those zones may contain vehicles. Figure 17 shows an example of zones ignored by DETRAC. The ignored zones directly impact the pre-processing step, since regions proposed by Selective Search that include such zones cannot be used as background samples, since there may be vehicles belonging to the ignored regions (see Figure 17). For this reason, in addition to the IOU criterion, an overlap criterion was added, that is, proposed regions cannot have more than 80% of their area contained in an ignored zone nor can an ignored zone have more than 80% of their area contained in a proposed region. The overlap criterion is necessary to solve particular cases where one of the regions, proposed or ignored, is much larger than the other. Allowing one of the two to be fully inserted in the other, making only the IOU measure not enough to avoid proposed regions contained in ignored areas. Figures 18 and 19 illustrate the particular case in which the use of the IOU alone would not be enough, the first presents a situation in a generic way while the second presents the problem in an image of the database used.

4.3.2 Particularities of CNN Training

DETRAC, as presented in section 4.2, has more than 140,000 images. Thus, important points must be observed when using a database of this dimension. The following subsections detail particularities found in the use of a large database for CNN training.

4.3.2.1 Hyper parameters

The size of the database directly influences the choice of hyper parameters. The estimated training time using batches of 32 images exceeded 24 hours per season. For this reason, and in order to reduce training time, hyper parameter values based on the original RCNN architecture article were used (GIRSHICK et al., 2014). In this way, batches of 256 images were used, which provide estimates of training time for a 3-hour period. In addition, the training was carried out for 20 seasons, reaching the level where the cost function for the validation set stops reducing.

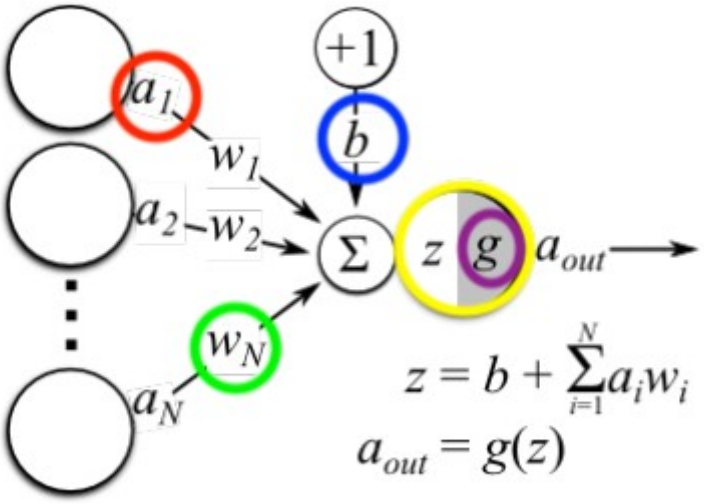

Figure 4.4: ANN Dataflow

4.3.2.2 Generators

The large number of images present in the database makes it impossible to load all of them in RAM memory to carry out CNN training. This problem is solved by the use of generators provided by the keras module (CHOLLET et al., 2015). The generators' goal is to allow CNN to be trained directly by the database image files, that is, trained directly by files saved on the hard disk, solving

the RAM overflow problem for large databases (CHOLLET et al., 2015). Based on a defined folder structure, in which each folder must be named with the name of the image class it contains, keras is responsible for reading and using the image data on the fly, that is, simultaneously with the process CNN training course. Figure 20 shows the directory structure necessary for the correct use of generators. Considering that keras himself is in charge of reading image files, when making predictions with the trained model, it is necessary to use data encoded in the same way as used in training, otherwise the predictions will not be consistent. For example, keras uses the PIL module, which encodes images read in RGB format by default, while another very common module, OpenCV, uses BGR encoding by default, that is, insert an image read by OpenCV without paying attention to coding difference would provide wrong predictions for the image.

4.3.3 Post-Processing Particularities

At the end of the classification stage of the regions proposed by Selective Search, a very wide set of regions detected as a vehicle is obtained. However, not all of these regions have been correctly classified and, even when they have been, several regions can point to the same vehicle. In this way, Non-Maximum Suppression is used to exclude redundant BBs, however, it is necessary a criterion that defines which BBs should be excluded and which BBs should be kept. In addition, it is necessary that the number of BBs classified as vehicles incorrectly be minimized. The following subsections explain the processes adopted to solve these problems.

4.3.3.1 Criteria for the exclusion of redundant BBs

The criterion used is based on the probability attributed to each region by CNN. For each region, CNN distributes among the vehicle classes the probability that that region has to belong to each of the classes, totaling 100%. Thus, when executing Non-Maximum Suppression, the algorithm prioritizes regions that have a higher probability of belonging to a class and discards regions with lower probabilities, that is, only the region with the highest probability among the overlapping regions will remain at the end. Figure 21 illustrates the proposed regions before running Non-Maximum Suppression and after running.

4.3.3.2 Thresholds of probability and size of regions

The problem of regions classified as vehicles incorrectly (false positives) is minimized by the use of thresholds that will validate or not the positive classification of certain regions.

The first threshold used is the probability threshold. Regions positively classified as a vehicle, that is, not being a background, will only be presented as a true detection if the probability associated with the vehicle class attributed to the region is higher than the pre-established threshold value. In this way, regions that CNN attributed a low probability are ruled out. In addition to the probability threshold, a maximum and minimum size threshold was used for the proposed regions depending on their position in the image. Due to the positioning of the cameras when generating the images, it is noticed that vehicles at the top of the image tend. Thus, proposed regions located at the top of the image that have an area greater than the maximum allowed area are discarded. Likewise, proposed regions located at the bottom of the image that have an area below the minimum allowed area are also discarded.

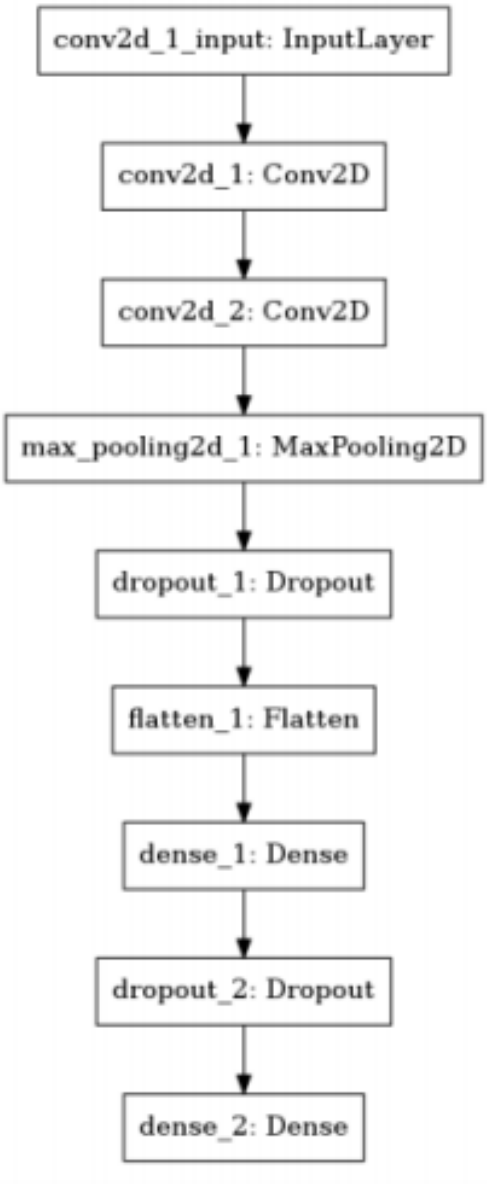

Figure 4.5: CNN Layer Structure

4.4 PERFORMANCE EVALUATION

The performance evaluation of the vehicle detector will be carried out in two stages: (i) analysis of the performance of CNN in the classification of regions and (ii) analysis of the performance of the vehicle detector considering the BBs expected and obtained at the end of the year. detection process. The metrics used for the assessment of detection will be presented simultaneously with the results obtained.

4.4.1 CNN

CNN training was carried out from the set of images generated after the data pre-processing step. The training lasted 42 hours and was carried out via the Docker container (NICKOLOFF, 2016). Table 1 shows the division of training and validation sets, showing the number of samples present in each class in each set.

Figure 4.6: Training and Validation Time in CNN

Analyzing the evolution of the indicators presented during the training of CNN, it is observed that from the third season onwards, the loss function for the validation set stops decreasing and starts to grow. In addition, the accuracy for the validation set starts to fluctuate while the accuracy for the training set continues to grow, however, at a rate lower than the rate previously observed. In this way, it is possible to perceive the beginning of the overfitting process, in which the network tends to become specialized in the training set and loses the necessary generality for a good performance in the validation set. In order to avoid the use of a model influenced by the overfitting process, the weights of the model trained until the third season are used, with an accuracy of 86% for the validation set. The predictions in the images present in the validation set are performed and the confusion matrix shown in Figure 24 is obtained. The confusion matrix allows a deeper analysis of CNN's performance to be made than the final accuracy value. CNN is more efficient in identifying between 'background' or vehicle, since 94, 92% of the 'background' regions have been correctly classified. However, among existing classes of vehicles, CNN was less efficient. Looking at the ‘car’ and ‘van’ classes, it is noticeable that they are classified correctly in more than 70% of cases and that in virtually all the rest of cases of incorrect classification, CNN tends to confuse these two classes with each other. This phenomenon can be influenced by the similarity of these two types of vehicles when evaluating the front and rear regions of both.

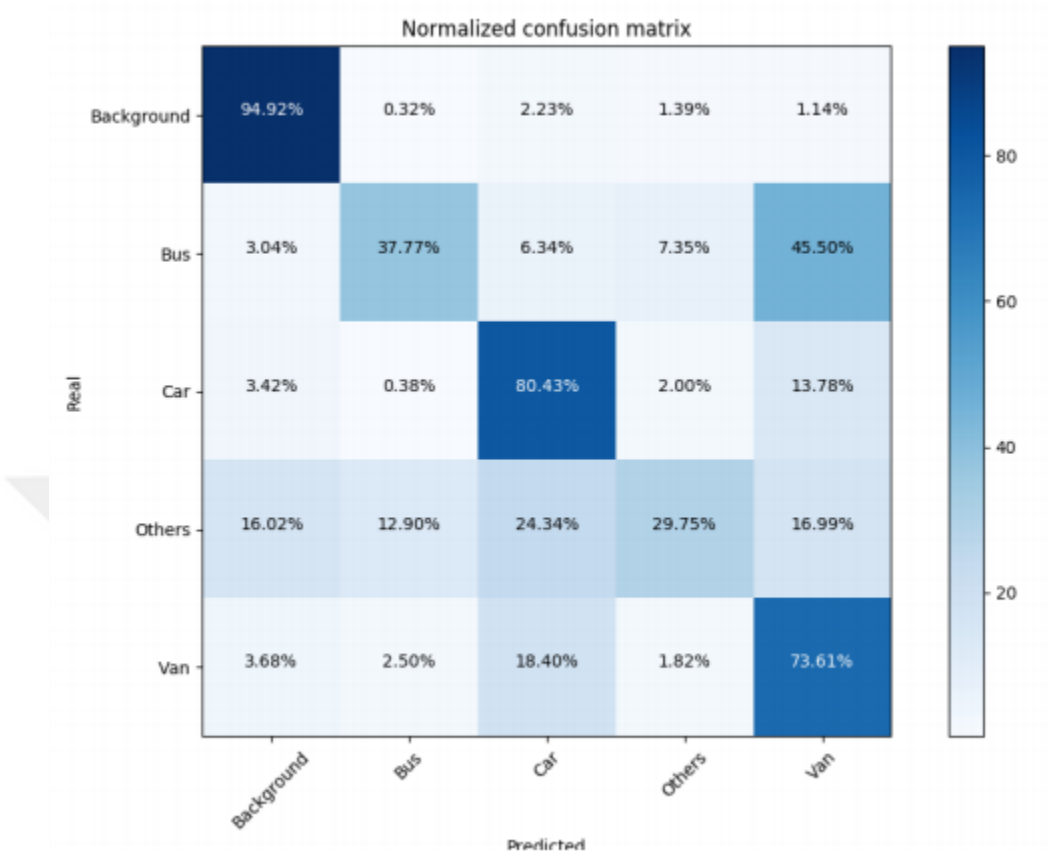

Figure 4.7: Confusion Matrix of CNN Results

obtained for the images in the database. Figure 28 presents some of the images generated after the execution of the vehicle detection. BBs classified as 'car', 'van', 'bus' and 'others' are represented by the colors red, blue, yellow and green, respectively. Analyzing the images, it is possible to observe that the detector is able to locate and classify vehicles present in the image. However, it is also possible to observe the existence of errors in the detections. Next, an analysis of the errors presented by the detector is made. Observing the images (c), (e), (g) and (l), the classification error for the classes 'bus' and 'others' is noticeable (observe the confusion matrix values for these classes in Figure 24), which causes not only the incorrect classification of the proposed region, but also the failure to detect other vehicles. It is worth mentioning that classification errors also occur for the other classes, as can be seen in the images (b), (i) and (k). In addition, images (a), (d), (e) and (f) show a division of a single vehicle into several BBs, causing multiple detections to be generated for the same vehicle, even though the Non-Maximum Suppression has been applied previously. This situation is possibly linked to the size of the vehicles, since the parameters chosen for

Selective Search tend to propose regions of intermediate sizes. Another type of error observed is location errors. There are BBs that represent correct detection, that is, they indicate a vehicle correctly. However, these BBs can only partially encompass the associated vehicle (see image (j)). Finally, existing vehicles are observed in the images that are not detected. This situation is possibly related to two factors: (i) no region proposed via Selective Search encompasses these vehicles and (ii) the regions proposed for these vehicles are discarded by the size filter, proportional to the location of the BB in the image. For example, images (d) and (h), generated by cameras not parallel to the highway, have larger BBs at the top of the image than expected, since the vehicles are closer to the camera. Thus, even if there are proposed regions for these vehicles, they will be discarded. In addition to the analysis of the detections, the evaluation of the performance of the detector can be performed from the values of Precision and Recall. The values are obtained from the variation of the overlapping threshold between the detected BBs and the Ground Truths present in the image, generating different values for each threshold used. Precision and Recall values are calculated from equations 4.1 and 4.2, respectively.

$$P = VPVP + FP \quad (4.1)$$

$$R = VPVP + FN \quad (4.2)$$

In the above equations, the following parameters are used: (i) VP are the true positives (number of objects correctly detected), (ii) FP are the false positives (number of objects detected incorrectly) and (iii) FN are false negatives (number of objects not detected). Table 2 presents the results obtained.

5. CONCLUSION

The main contributions of this dissertation are summarized in the list below: • Evaluation of the 6-DoF detection method with deep real-time learning proposed by Tekin et al. Proposition and evaluation of variations of the technique with synthetic datasets to minimize problems such as overfitting, capable of handling different cameras and real scenarios. Generation of synthetic data for training based on with a focus on assessing the influence of the synthetic data set on the model's result, use of transfer-learning and combination of synthetic and real information. Generation and evaluation of synthetic images with adaptation of the real domain, simulating use cases and evaluating the (in) dependence of training with real data Generation of synthetic data with domain randomization for 6-DoF detection with assessment of the influence of variations in characteristics, usage scenarios and reduction of over-adjustment in training. Investigation of the variation of camera parameters in data generation for generic training of deep learning architectures in 6-DoF detection, adapting use for use in synthetic data generation techniques. For future work, it will be important to evaluate the data generation procedure, investigating ways to simplify synthetic data generation and improve the generation process in order to reduce the number of failures. New ways of obtaining the rendering parameters will also be investigated, using learning architectures to generate data automatically. Through the obtained results it is possible to notice that the use of synthetic data in training can help in reducing the dependence on real data, optimizing the cost of developing deep learning models and improving the result for different use cases. In order to understand how to improve this synthetic data generation process, future investigations will be carried out seeking to find and evaluate the most relevant characteristics in the rendering process, removing the need to insert a set of real images in the training to transfer the information between the real domain and synthetic, as shown by Zakharov et al. (ZAKHAROV et al., 2018) using synthetically generated depth images to estimate the pose of objects in real scenarios through a generator based on adversarial architecture. Regarding domain randomization, other forms of distraction and variation of the information used will be tested, seeking to understand the model's behavior and the representation of the characteristics learned in different scenarios. An in-depth study should be carried out regarding the learning of the parameters involved in the training and how the model represents these variations, in order to optimize the learning process by generalizing the information to several scenarios avoiding the over-adjustment of the actual information used.

To perform the temporal tracking, ways of adapting the developed architecture will be studied, with the objective of improving the stability of the model, reducing Jitter in the response. Jitter is a big problem in applications that use time tracking information, such as augmented reality applications that try to deliver the best interaction experience, leading the user to the perception that virtual objects are perceived as natural parts of the environment. Approaches that seek to use deep learning for tracking can be investigated, such as the use of CNN 3D (SU et al., 2015), recurring architectures (NING et al., 2017), and two methods previously described (GARON; LALONDE, 2017; TAN; NAVAB; TOMBARI, 2017). We will evaluate how to adapt these models to use 6-DoF tracking using only RGB images and maintaining the real-time performance required for augmented reality applications.

REFERENCES

[1] R.H. Lasseter and P. Paigi, "Microgrid: a conceptual solution," in IEEE 35th Annual Power Electronics Specialists Conference, 2004, pp. 4285 - 4290.

[2] R.H Lasseter, "Microgrids," Power Engineering Society Winter Meeting, vol. 1, pp. 305- 308, 2002.

[3] F. Katiraei and M.R. Iravani, "Power Management Strategies for a Microgrid with Multiple Distributed Generation Units," IEEE Transactions on Power Systems, vol. 21, no. 4, pp. 1821 - 1831, November 2006.

[4] Carla, Garrido-Echevería,Natalia,Jiménez-Estévez,Guillermo Alvial-Palavicino, Lorenzo Reyes, and Rodrigo Palma-Behnke, "A methodology for community engagement in the introduction of renewable based smart microgrid," Energy for sustainable development, vol. 15, no. 3, pp. 314-323, Septiembre 2011.

[5] Fernando Lanas, "Desarrollo y validación de un modelo de optimizacion energetica para una micro-red," in Memoria para optar al título de ingeniero civil electricista., 2011.

[6] R Palma-Behnke, D. Ortiz, L. Reyes, G. Jimenez-Estevez, and N. Garrido, "A social SCADA approach for a renewable based microgrid — The Huatacondo project," in IEEE Power and Energy Society General Meeting, San Diego, CA, 2011, pp. 1-7.

[7] Z Bashir and M El-Hawary, Applying Wavelets to Short-Term Load Forecasting Using PSO-Based Neural Networks. Canada: IEEE, 2009.

[8] Goran Strbac, Demand Side Management: Benefits and Challenges. Londres: Elsevier Ltd., 2008.

[9] Adeel Abbas Zaidi and Friederich Kupzog, Microgrid Automation - A Self-Configuring Approach. Vienna: IEEE, 2008.

[11] M. Afshin and A. Sadeghian, "PCA-based Least Squares Support Vector Machines in Week-Ahead Load Forecasting," IEEE Industrial & Commercial Power Systems Technical Conference, pp. 1-6, May 2007.

[12] A. Setiawan, I. Koprinska, and V. Agelidis, Very Short-Term Electricity Load Demand Forecasting Using Support Vector Regression.: IEEE Internation Joint Conference on Neural Network, pp 288-294, June 2009.

[13] E.A Feinber and D. Genethliou, "Load Forecasting," in Applied Mathematics for Restructured Electric Power System, J.H. Chow, F.F. Wu, and J. Momoh, Eds.: Springer, 2005, ch. 12, pp. 269-285.

[14] Lee:W.J. and Ch.Y. Lee, Comparative Study on Load Forecasting Technologies for Different Geographical distributed Loads.: IEEE Power and Energy Society General 91 Meeting, pp 1-8, July 2011.

[15] T. Haida and S Muto, "Regression based peak load forecasting using a transformation technique," IEEE Transaction on Power Systems, vol. 9, no. 4, pp. 1788-1794, November 1994.

[16] R.F. Engle, C. Mustafa, and J. Rice, "Modelling peak electricity demand," Journal of forecasting, vol. 11, no. 3, pp. 241-251, April 1992.

[17] J. Lu and X. Zhang, "A real-time adaptive forecasting algorithm for electric power load," IEEE/PES Transmission and Distribution Conference and Exhibition, pp. 1-5, 2005.

[18] M.Y. Cho, J.C Hwang, and C.S. Chen, "Customer short term load forecasting by using ARIMA transfer function model," International Conference on Energy Management and Power Delivery, vol. 1, pp. 317-322, November 1995.

[19] A Jain and E Srinivas, "Short term load forecasting using fuzzy adaptive inference and similarity," World Congress in Nature & Biologically Inspired Computing, pp. 1743-1748, December 2009.

[20] W. Dai and P. Wanh, "Application of pattern recognition and artificial neural network to load forecasting in electric power system," IEEE International Conference on Natural Computation, 2007.

[21] Z.A. Bashir and M.E. El-Hawary, Applying Wavelets to Short-Term Load Forecasting Using PSO-Based Neural Networks.: IEEE Transaction on power systems,Vol. 24, No. 1, February 2009.

[22] Ch. Hsu and Ch. Chen, "Regional load forecasting in Taiwan: Applications of artificial neural networks," Energy Conversion and Management, vol. 44, no. 12, pp. 1941-1949, July 2003.

[23] Y. Lu, "Short-term load forecasting method based on structural neural network," IEEE World Congress on Intelligent Control and Automation, pp. 4434-4438, June 2008.

[24] K.H. Osman, M.L Awad, and T.K. Mahmoud, "Neural network based approach for shortterm load forecasting," IEEE Power System Conference and Exposition, pp. 1-8, March 2009.

[25] N. Kandil, R. Wamkeue, M. Saad, and S. Georges, "An efficient approach for shorterm load forecasting using artificial neural networks," IEEE ISIE, pp. 1928-1932, July 2006.

[26] H. Ho et al., "Short term load forecasting of Taiwan power system using a knowledge based expert system," IEEE Transactions on Power Systems, vol. 5, pp. 1214-1221, 1990.

[27] S. Rahman and O. Hazmin, "Load forecasting for multiple sites: Develpment of an expert system-based technique," Electric power systems research, vol. 39, pp. 161-169, 1996.

[28] A. Pratondo, Fuzzy Rule Base for Analytical Demand Forecasting Enhancement.: IEEE Second International Conference on Advances in Computing, Control and Telecomunication Technologies, 2010.

[29] V.H Hinojosa and A Hoese, Short-Term Load Forecasting Using Fuzzy Inductive Reasoning and Evolutionary Algorithms.: IEEE Transactions on Power Systems, Vol. 25, No. 1, February 2010.

[30] S. Sachdeva and Ch. Verma, "Load forecasting using fuzzy methods," Power System 92 Technology and IEEE Power INdia Conference, pp. 1-4, October 2008.

[31] H. Xu, J. Wang, and S. Zheng, "Online daily load forecasting based on support vector machines," IEEE International Conference on Machine Learning and Cybernetics, vol. 7, pp. 3985-3990, August 2005.

[32] S. Fan, K. Methaprayoom, and W.J. Lee, "Multi-area load forecasting for system with large geographical area," IEEE/IAS Industrial and Commercial Power System Technical Conference, pp. 1-8, May 2008.

[33] T. Senjyu, H. Takara, K. Uezato, and T Funabashi, "One-hour ahead load forecasting using neural network," IEEE Transactions on power systems, vol. 17, no. 1, pp. 113-118, February 2002.

[34] Ch. Jaipradidtham, Next Day Load Demand Forecasting of Future in Electrical Power Generation on Distribution Networks using Adaptative Neuro-Fuzzy Inference.: First International Power Energy Conference, November 2006.

[35] K. Liu et al., Comparison of Very Short-term Load Forecasting Techniques.: IEEE Transactions on power systems,Vol. 11, No. 2, May 1996.

[36] P. Chan, W. Chen, W.Y.NG Wing, and D. Yeung, Multiple Classifier System for Short Term Load Forecast of Microgrid.: International Conference on Machine Learning and Cybernetics, July 2011.

[37] N. Amjady, F. Keynia, and H. Zareipour, Short-Term Load Forecast of Microgrid by a New Bilevel Prediction Strategy.: IEEE Transactions on Smart Grid,Vol. 1, No. 3, December 2010.

[38] N Amjady and F Keyna, "Electricity Market Price Spike Analysis by a Hybrid Data Model and Featured Selection Technique," Electr. Power Syst. Res., vol. 80, no. 3, pp. 318-327, March 2010.

[39] N. Amjady and F. Keyna, Day-ahead price forecasting of electricity markets by mutual information technique and cascaded neuro-evolutionary algorithm.: IEEE Trans. Power Syst., Vol. 24, No 1, pp. 306-318, Feb 2009.

[40] R. Storn and K. Price, Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces.: Journal of Global Optimization, Vol. 11, No. 4, pp. 341-359, Dec 1997.

[41] P. Chan, W.-C. Chen, Wing W. Y. NG, and D. Yeung, Multiple Classifier System for Short Term Load Forecast of Microgrid.: Internation Conference on Machine Learning and Cybernetics, July 2011.

[42] Huibini Sui, Ying Sun, and Wie-Jen Lee, A Demand Side Management Model Based on Advanced Metering Infrastructure.: IEEE, 2011.

[43] Hans Nilsson, The many faces of demand-side management. Stockholm, 1994. [44] Chen Xiang-ting, Zhou Yu-hui, Duan Wei, Tang Jie-bin, and Guo Yu-xiao, Desing of Intelligent Demand Side Management System Respond to Varieties of Factors. Beijing, China, 2010.

[45] Michael Angelo Pedrasa, Ted Spooner, and Iain MacGill, Scheduling of Demand Side Resources Using Binary Particle Swarm Optimization.: IEEE, 2009. 93

[46] A Molina, J.A Gabaldón, and C Álvarez, Implementation and assessment of physically based electrical load models: application to direct load control residential programmes. España: IEEE, 2003.

[47] Vandad Hamidi, Furong Li, Yao Liangzhong, and Masoud Bazargan, Domestic Demand Side Management for Incresing the Value of Wind. Bath, UK, 2008.

[48] Molina A., Roldán C., Fuentes A.,Gómez E., Ramírez-Rosado I.J., Lara P., Domínguez J.A., García-Garrido E., Tarancón E. Gabaldón A., Assessment and Simulation of DemandSide Management Potential in Urban Power Distribution Networks. Cartagena, España: IEEE.

[49] J Han and M.A Piette, Solutions for Summer Electric Power Shortages: Demand Response and its Applications in Air Conditioning and Refrigerating Systemas., 2008.

[50] M Aldabi and E Saadany, "A summary of demand response in electricity markets," Electric power systems research, vol. 78, no. 11, pp. 1989–1996, November 2008.

[51] Daniel O'Neil, Marco Levorato, Andrea Goldsmith, and Urbashi Mitra, "Residential demand response using reinforcement learning," in First IEEE International Conference in Smart Grid Communications, 2010.

[52] S Pourmousavi and M Nehrir, "Demand response for smart microgrid: Initial results," , January 2011, pp. 1- 6.

[53] P Khajavi, H Abniki, and A.B Arani, The Role of Incentive Based Demand Response Programs in Smart Grid.: IEEE, 2011.

[54] Kumar Nunna and Suryanarayana Doolla, "Demand response in smart microgrids," in Innovative smart grid technologies, December 2011, pp. 131-136.

[55] Weihao Hu, Zhe Chen, and Birgitte Bak-Jensen, Optimal Load Response to Time-of-Use Power Price for Demand Side Management in Denmark. Denmark: IEEE, 2010.

[56] Joseph Tate and Jana Sebestik, Interactive Lessons Addressing Wind Integration and Timeof Use Pricing. Toronto: IEEE, 2011.

[57] Alexander Thornton and Carlos Rodríguez, Distributed Power Generation in the United States. Madrid: Elsevier Ltd., 2011.

[58] Dirk Westermann and Andreas John, Demand Matching Wind Power Generation with Wide-Area Measurement and Demand Side Management. Ilmenau, Alemania: IEEE, 2007.

[59] Paolo Piagi and Robert Lasseter, Autonomous Control of Microgrids. Montreal: IEEE, 2006.

[60] Robert Lasseter et al., The CERTS MicroGrid Concept. California, 2002.

[61] THE SMART GRID: AN INTRODUCTION. [Online].
http://energy.gov/oe/downloads/smart-grid-introduction-0

[62] Fabrice Saffre and Richard Gedge, Demand-Side Management for the Smart Grid. Abu Dhabi: IEEE, 2010. [63] Friederich Kupzog, Tehseen Zia, and Abbas Zaidi, Automatic Electric Load Identification in Self-Configuring Microgrids. Vienna: IEEE, 2009.

[64] Adeel Abbas Zaidi and Friederich Kupzog, Microgrid Automation - A Self-Configuring 94 Approach. Vienna: IEEE, 2008.

[65] Adeel Abbas Zaidi, Friederich Kupzog, and Peter Palensky, Load Recognition for Automated Demand Response in Microgrids. Vienna: IEEE, 2010.

[66] Luigi Martirano, Serena Fornari, Alessandro Di Giorgio, and Francesco Liberati, "A case study of a commercial/residential microgrid integrating cogeneration and electrical local users," in 12th International Conference on Environment and Electrical Engineering (EEEIC), Wroclaw, Poland, 2013, pp. 363 - 368.

[67] E. Barklund, M. Prodanovic, and C. Hernandez-Aramburo, "Energy Management in Autonomous Microgrid Using Stability-Constrained Droop Control of Inverters," IEEE Transactions on Power Electronics, vol. 23, no. 5, pp. 2346 - 2352, September 2008.

[68] V. Hamidi and F., Robinson, F. Li, Responsive Demand in Network with High Penetration of Wind Power. Bath, UK.: IEEE, 2008.

[69] Chenye Wu, Mohsenian-Rad Hamed, Jianwei Huang, and Amy Yuexuan Wang, Demand Side Management for Wind Power Integration in Microgrid Using Dynamic Potencial Game Theory. China: IEEE, 2011.

[70] Bingnan Jiang and Yunsi Fei, "Dynamic residential demand response and distributed generation management in smart microgrid with hiearchical agents," Energy Procedia, vol. 12, pp. 76-90, September 2011.

[71] R. Palma-Behnke, C. Benavides, E. Aranda, J. Llanos, and D. Sáez, Energy Management System for a Renewable based Microgrid with a Demand Side Management Mechanism. Santiago: IEEE, 2011.

[72] E Matallanas et al., Neural network controller for Active Demand-Side Management with PV energy in the residential sector. Madrid: Elsevier Ltda, 2011.

[73] I.C. Paschalidis, Binbin Li, and M.C. Caramanis, "Demand-Side Management for Regulation Service Provisioning Through Internal Pricing," IEEE Transactions on Power Systems, vol. 27, no. 3, pp. 1531 - 1539, August 2012.

[74] T. Logenthiran and Tan Zong Shun, "Multi-Agent System for Demand Side Management in smart grid," in IEEE Ninth International Conference on Power Electronics and Drive Systems (PEDS), Singapore, 2011 , pp. 424 - 429.

[75] C. Ibars and L. Giupponi, "Distributed Demand Management in Smart Grid with a Congestion Game," in First IEEE International Conference on Smart Grid Communications (SmartGridComm), Gaithersburg, MD, 2010, pp. 495 - 500.

[76] R. Babuska, Fuzzy Systems, Modeling and Identification.

[77] T. Takagi and M. Sugeno, Fuzzy Identification of Systems and its Applications to Modeling and Control. Tokyo: IEEE, 1985.

[78] M. Jamshidi, André Titli, Lotfi Z., and S. Boverie, Applications of Fuzzy Lugic. New Jersey: Prencite Hall, 1997.

APPENDIX OF ALL THE CODES USED IN THIS THESIS

```
function varargout = User_Interfacefig(varargin)

% USER_INTERFACEFIG MATLAB code for User_Interfacefig.fig

%           USER_INTERFACEFIG, by itself, creates a new
USER_INTERFACEFIG or raises the existing

%      singleton*.

%

%       H = USER_INTERFACEFIG returns the handle to a new
USER_INTERFACEFIG or the handle to

%      the existing singleton*.

%

%
USER_INTERFACEFIG('CALLBACK',hObject,eventData,handles,...)
calls the local

%      function named CALLBACK in USER_INTERFACEFIG.M with
the given input arguments.

%

%      USER_INTERFACEFIG('Property','Value',...) creates a
new USER_INTERFACEFIG or raises the

%     existing singleton*.  Starting from the left, property
value pairs are

%     applied to the GUI before User_Interfacefig_OpeningFcn
gets called.  An
```

```
%       unrecognized property name or invalid value makes
property application
%                stop.     All   inputs   are   passed   to
User_Interfacefig_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES


% Edit the above text to modify the response to help
User_Interfacefig


% Last Modified by GUIDE v2.5 17-Mar-2015 14:04:23


% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@User_Interfacefig_OpeningFcn, ...
```

```
                        'gui_OutputFcn',
@User_Interfacefig_OutputFcn, ...
                        'gui_LayoutFcn',  [] , ...
                        'gui_Callback',   []);
    if nargin && ischar(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end


    if nargout
        [varargout{1:nargout}]     =     gui_mainfcn(gui_State,
varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end
    % End initialization code - DO NOT EDIT




    % --- Executes just before User_Interfacefig is made
visible.
    function User_Interfacefig_OpeningFcn(hObject, eventdata,
handles, varargin)
    % This function has no output args, see OutputFcn.
```

```
    % hObject    handle to figure
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    % varargin   command line arguments to User_Interfacefig (see VARARGIN)


    % Choose default command line output for User_Interfacefig
    handles.output = hObject;


    % Update handles structure
    guidata(hObject, handles);


    % UIWAIT makes User_Interfacefig wait for user response (see UIRESUME)
    % uiwait(handles.figure1);



    % --- Outputs from this function are returned to the command line.
    function varargout = User_Interfacefig_OutputFcn(hObject, eventdata, handles)
```

```
    % varargout   cell array for returning output args (see
VARARGOUT);

    % hObject    handle to figure

    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles     structure with handles and user data (see
GUIDATA)


    % Get default command line output from handles structure

    varargout{1} = handles.output;


    % --- Executes during object creation, after setting all
properties.

    function V_COST1_CreateFcn(hObject, eventdata, handles)

    % hObject    handle to V_COST1 (see GCBO)

    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles    empty - handles not created until after all
CreateFcns called


    % --- Executes during object creation, after setting all
properties.

    function V_COST2_CreateFcn(hObject, eventdata, handles)
```

```
    % hObject    handle to V_COST2 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    empty - handles not created until after all
CreateFcns called


    % load small data
    % --- Executes on button press in pushbutton1.
    function pushbutton1_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton1 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    structure with handles and user data (see
GUIDAT)
    %%%%%%------------------------------------------------
---------
    %% Import data from spreadsheet
    % Script for importing data from the following spreadsheet:
    %
    %    Workbook: C:\Stuff\Downloads\MS BSAN\Quarter 2\OPR
    %    620\Project\test\Data.xlsx Worksheet: Small
    %
```

```
    % To extend the code for use with different selected data
or a different
    % spreadsheet, generate a function instead of a script.

    % Auto-generated by MATLAB on 2015/03/15 18:38:32

    %% Import the data
    [~, ~, raw] = xlsread('C:\Stuff\Downloads\MS BSAN\Quarter
2\OPR 620\Project\test\Data.xlsx','Small','A2:H4');
    raw(cellfun(@(x) ~isempty(x) && isnumeric(x) &&
isnan(x),raw)) = {''};
    cellVectors = raw(:,1);
    raw = raw(:,[2,3,4,5,6,7,8]);

    %% Create output variable
    data = reshape([raw{:}],size(raw));

    %% Allocate imported array to column variable names
    Appliance = cellVectors(:,1);
    Job = data(:,1);
    RequestHour = data(:,2);
    Duration = data(:,3);
```

```
    StartHour = data(:,4);

    DelayCost = data(:,5);

    PowerCost = data(:,6);

    TotalCost = data(:,7);



    %% Clear temporary variables

    clearvars data raw cellVectors

    %%%%%%%5555555555555555555555555555555555555555555555555555
5555555555

    Schedule = [Appliance    num2cell([Job  RequestHour
    Duration  StartHour DelayCost PowerCost TotalCost])];

    set(handles.uitable1,'data',Schedule);

    set(handles.pushbutton4,'Visible','On');



    % solve button

    % --- Executes on button press in pushbutton2.

    function pushbutton2_Callback(hObject, eventdata, handles)

    % hObject    handle to pushbutton2 (see GCBO)

    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles     structure with handles and user data (see
GUIDATA)
```

```
    % without scheduling
    set(handles.edit3,'String','100');
    %with scheduling
    set(handles.edit4,'String','200');
    %savings
    set(handles.edit5,'String','100');


    % display graph button
    % --- Executes on button press in pushbutton3.
    function pushbutton3_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton3 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    structure with handles and user data (see
GUIDATA)
    active = get(hObject,'Value');
    if active
        set(handles.axes2,'Visible','On');
        set(hObject,'String','Hide Graph');
    else
        set(handles.axes2,'Visible','Off');
```

```
        set(hObject,'String','Display Graph');
    end


    % reset button
    % --- Executes on button press in pushbutton4.
    function pushbutton4_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton4 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    %reset values
    set(handles.uitable1,'data',[]);
    set(handles.axes2,'Visible','Off');
    set(handles.pushbutton4,'Visible','Off');
    set(handles.pushbutton3,'String','Display Graph');
    % without scheduling
    set(handles.edit3,'String','');
    %with scheduling
    set(handles.edit4,'String','');
    %savings
```

```
    set(handles.edit5,'String','');
    %total power
    set(handles.edit6,'String','');
    %power utilized
    set(handles.edit7,'String','');
    %set slider vallue to zero
    set(handles.slider1,'Value',0);


    function edit3_Callback(hObject, eventdata, handles)
    % hObject    handle to edit3 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)


    % Hints: get(hObject,'String') returns contents of edit3 as text
    %        str2double(get(hObject,'String')) returns contents of edit3 as a double


    % --- Executes during object creation, after setting all properties.
```

```
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit4 as text
```

```
%        str2double(get(hObject,'String')) returns contents
of edit4 as a double

% --- Executes during object creation, after setting all
properties.

function edit4_CreateFcn(hObject, eventdata, handles)

% hObject    handle to edit4 (see GCBO)

% eventdata  reserved - to be defined in a future version
of MATLAB

% handles    empty - handles not created until after all
CreateFcns called



% Hint: edit controls usually have a white background on
Windows.

%       See ISPC and COMPUTER.

if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end

function edit5_Callback(hObject, eventdata, handles)

% hObject    handle to edit5 (see GCBO)

% eventdata  reserved - to be defined in a future version
of MATLAB
```

```
    % handles    structure with handles and user data (see
GUIDATA)


    % Hints: get(hObject,'String') returns contents of edit5 as
text
    %        str2double(get(hObject,'String')) returns contents
of edit5 as a double
    % --- Executes during object creation, after setting all
properties.
    function edit5_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to edit5 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    empty - handles not created until after all
CreateFcns called


    % Hint: edit controls usually have a white background on
Windows.
    %       See ISPC and COMPUTER.
    if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end
```

```
    % load large data
    % --- Executes on button press in pushbutton5.
    function pushbutton5_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton5 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    structure with handles and user data (see
GUIDATA)
    %% Import data from spreadsheet
    % Script for importing data from the following spreadsheet:
    %
    %    Workbook: C:\Stuff\Downloads\MS BSAN\Quarter 2\OPR
    %    620\Project\test\Data.xlsx Worksheet: Large
    %
    % To extend the code for use with different selected data
or a different
    % spreadsheet, generate a function instead of a script.
    % Auto-generated by MATLAB on 2015/03/17 00:55:59


    %% Import the data
```

```
    [~, ~, raw] = xlsread('C:\Stuff\Downloads\MS BSAN\Quarter
2\OPR 620\Project\test\Data.xlsx','Large','A2:H13');

    raw(cellfun(@(x) ~isempty(x) && isnumeric(x) &&
isnan(x),raw)) = {''};

    cellVectors = raw(:,1);

    raw = raw(:,[2,3,4,5,6,7,8]);


    %% Create output variable

    data = reshape([raw{:}],size(raw));


    %% Allocate imported array to column variable names

    Appliance = cellVectors(:,1);

    Job = data(:,1);

    RequestHour = data(:,2);

    Duration = data(:,3);

    StartHour = data(:,4);

    DelayCost = data(:,5);

    PowerCost = data(:,6);

    TotalCost = data(:,7);


    %% Clear temporary variables
```

```
    clearvars data raw cellVectors;


    Schedule = [Appliance    num2cell([Job  RequestHour
    Duration  StartHour DelayCost PowerCost TotalCost])];
    set(handles.uitable1,'data',Schedule);
    set(handles.pushbutton4,'Visible','On');
    function edit6_Callback(hObject, eventdata, handles)
    % hObject    handle to edit6 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles     structure with handles and user data (see
GUIDATA)


    % Hints: get(hObject,'String') returns contents of edit6 as
text
    %        str2double(get(hObject,'String')) returns contents
of edit6 as a double


    % --- Executes during object creation, after setting all
properties.
    function edit6_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to edit6 (see GCBO)
```

```
    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles    empty - handles not created until after all
CreateFcns called



    % Hint: edit controls usually have a white background on
Windows.

    %       See ISPC and COMPUTER.

    if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

        set(hObject,'BackgroundColor','white');

    end

    function edit7_Callback(hObject, eventdata, handles)

    % hObject    handle to edit7 (see GCBO)

    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles     structure with handles and user data (see
GUIDATA)



    % Hints: get(hObject,'String') returns contents of edit7 as
text

    %        str2double(get(hObject,'String')) returns contents
of edit7 as a double
```

```
    % --- Executes during object creation, after setting all
properties.
    function edit7_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to edit7 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
    % handles    empty - handles not created until after all
CreateFcns called


    % Hint: edit controls usually have a white background on
Windows.
    %       See ISPC and COMPUTER.
    if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end
    % --- Executes on slider movement.
    function slider1_Callback(hObject, eventdata, handles)
    % hObject    handle to slider1 (see GCBO)
    % eventdata  reserved - to be defined in a future version
of MATLAB
```

```
    % handles    structure with handles and user data (see
GUIDATA)

    % Hints: get(hObject,'Value') returns position of slider

    %           get(hObject,'Min') and get(hObject,'Max') to
determine range of slider

    power=get(hObject,'Value');

    power=round(power);

    set(handles.edit6,'String',power);

    % --- Executes during object creation, after setting all
properties.

    function slider1_CreateFcn(hObject, eventdata, handles)

    % hObject    handle to slider1 (see GCBO)

    % eventdata  reserved - to be defined in a future version
of MATLAB

    % handles    empty - handles not created until after all
CreateFcns called

    % Hint: slider controls usually have a light gray
background.

    if                isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

        set(hObject,'BackgroundColor',[.9 .9 .9]);

    end
```