

**MODELING ABANDONED OBJECT DETECTION AND
RECOGNITION IN REAL-TIME SURVEILLANCE**

W.W.W.W.T.W.K.M.R.N.D.B Weliwita

169341X

Faculty of Information Technology

University of Moratuwa

2020

MODELING ABANDONED OBJECT DETECTION AND RECOGNITION IN REAL -TIME SURVEILLANCE

W.W.W.W.T.W.K.M.R.N.D.B Weliwita

169341X

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements for Master of Science in Information Technology.

September 2020

Declaration

I declare that this thesis is our own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

.....

Name of the Student

.....

Signature of Student

Date

Supervised by

Mr S.C. Premaratne

.....

Name of the Supervisor

.....

Signature of the Supervisor

Date

Dedication

This research is dedicated to all the Military and Civil Staff in the Information Technology Section and rest of the Military and Non-Military staff of the Defence Services Command and Staff College (DSCSC) who supported me to obtain the requisite data repositories to conduct this research.

Acknowledgement

From the beginning of the Masters program Mr. Saminda Premaratne senior lecturer Faculty of Information Technology of University of Moratuwa was a guiding light to me every time I had problems and doubts along the way and was an asset on path of higher education. I am fortunate to have him supervise this thesis and my sincere gratitude goes to him for making my dream of obtaining a Master's Degree from the University of Moratuwa come true.

My sincere thanks goes to Mr Isuru Jayasooriya and Mr Sahan Wickramage for the tremendous support extended to me in making appropriate data sets that enabled me to continue with my research. They shares their knowledge with me without any reservation and were with me whenever I had doubts and I needed support. My batch mates, other friends and lab staff of the IT faculty also supported me in completing my thesis and during sessions at the University.

My wife, parents and twin girls were always encouraged me and helped me to concentrate on my studies. My wife took the burden of managing the family which was a really hard task for her and she did it on my behalf in an admirable manner to support me.

Abstract

This research mainly focuses on building models using the latest tools which are using to detect and recognize abandoned objects in real-time Surveillance. There are number of experiments are occurring up to date in different scenarios and there are lots of tools have developed to improve the capacity of detection and recognition of abandoned objects. Further, identifying of most reliable, practical and efficient tools to implement the said process in different circumstances will immensely benefited in various aspects of the requirement of any organization.

Instead of old manual outdated systems, use of latest accurate and efficient systems will save money, time and improve the safety. Therefore, research will mostly focus three tools identified by thorough reading of the capacity and capabilities. There the use of deep learning techniques and most reliable and efficient datasets to evaluate the experiment for better solutions and findings will directly involve during the future decision-making process. Then, the **Faster RCNN (Faster Region Convolutional Neural Network)**, **YOLO V3 (You Only Look Once Version 3)** and **Single Shot multibox Detector (SSD)** were tested. There researcher selected three video samples and checked the accuracy for each video clip separately by those models individually then final results illustrated that the maxim accuracy given by the YOLO-V, Second position taken by the SSD then third place taken by the Faster RCNN.

Further, after determining the accuracy by comparing the three models identified, the results can be used for further deep learning to make the process more efficient. This will immensely benefit number of fields and subjects which are interested in object detection and recognition during real time surveillance in the future.

Keywords: Deep Learning, Datasets, Faster Region Convolutional Neural Network (Faster RCNN), YOLO V3 (You Only Look Once Version 3), Single Shot multibox Detector (SSD)

Table of Contents

Declaration	ii
Dedication	iii
Acknowledgement	iv
Abstract	v
Introduction.....	1
1.1 Prolegomena.....	1
1.2 Back ground and Motivation.....	2
1.3 Problem Statement	2
1.3 Aims and Objectives	3
1.3.1 Aim	3
1.3.2 Objectives	3
1.5 Proposed Solution	4
1.6 Structure of the thesis	4
Chapter 2.....	6
Literature Review.....	6
2.1 Introduction	6
2.2 The existing Researchers about the Abandoned object detection and recognition to in real time surveillance.	6
2.3 Study of the Research Problem	11
2.3.1 The abandoned object.....	11
2.3.2 Detection and Recognition of abandon objects	11
2.3.3 Real time Surveillance.....	11
2.4 Summery	12
Chapter 3.....	13
Specific Technologies Used.....	13
3.1 Introduction	13
3.2 What is Image Processing	13
3.3 What is Video Processing	15
3.4 What is Object Recognition and Detection	16

3.5 Neural Net Work	17
3.6 The performances of Remote PC	17
3.7 Experimented Tools	18
3.7.1 Faster RCNN	18
3.7.2 You Only Look Once – Version 3 (YOLO – V3)	18
3.7.3 Single Shot Multibox Detector (SSD)	19
3.8 Summery	20
Chapter 4.....	21
Image and Video Processing approach to Comparison on an Abandoned object Detection and Recognition in real-time Surveillance.....	21
4.1 Introduction	21
4.2 Input	21
4.3 Output.....	22
4.4 Process.....	22
4.5 Users.....	22
4.6 Features	22
4.7 Regression	23
4.8 Classification.....	23
4.9 Summery	23
Chapter 5.....	24
The Research design for the comparison on abandoned object detection and recognition in real-time surveillance.....	24
5.1 Introduction	24
5.2 High level Architecture of Research Design.....	24
5.3 Summery	26
Chapter 6.....	27
Implementation	27
6.1 Introduction	27
6.2 Selection of Input Video clips.....	27
6.3 Pre-Processing of the input videos	28
6.4 Distance Calculation	29

6.5 Transformation	30
6.6 Object Detection.....	31
6.7 Decision Making Process (Abandoned / Non Abandoned)	33
6.8 Summery	34
Chapter 7.....	35
Evaluation	35
7.1 Introduction	35
7.2 Knowledge base distance calculation.....	35
7.3 The steps which occurred while input video clips going through the model	37
7.4 Steps taken to measure the output results.....	39
7.5 How to measure the Accuracy of the models.....	41
7.5.1 Step one	42
7.5.2 Step two	42
7.5.3 Step three	43
7.5.4 Step Four	43
7.6 Summery	44
Chapter 8.....	45
Conclusion and Future Works	45
8.1 Introduction	45
8.2 Limitations	46
8.3 Future works.....	46
8.4 Summery	47

List of Figures

Figure 3. 1 the Fundamentals of the Image Processing	14
Figure 3. 2 the component of the Image Processing.....	14
Figure 3. 3 Simple Block diagram of creation of Digital Video from Natural Scene.	15
Figure 3. 4 specifications of the remote PC and the GPU.	17
Figure 3. 5 Faster RCNN Function of an object detection and Recognition.	18
Figure 3. 6 the Function of object detection and recognition from YOLO-V3.	19
Figure 3. 7 the General architecture of the SSD network.	20
Figure 4. 1 General model of comparison on tools of object detection and recognition.	21
Figure 5. 1 an Activity Diagram	25
Figure 6. 1 related infrastructure.....	27
Figure 6. 2 Raw input video clip.....	28
Figure 6. 3 Pre-Processing	29
Figure 6. 4 Distance Calculation.....	30
Figure 6. 5 Transformation of the images and corner points	31
Figure 6. 6 Object Detection.....	32
Figure 6. 7 Class variables	32
Figure 6. 8 Decision making source code of abandoned/non abandoned.....	33
Figure 7. 1 The Equiladian equation: Calculation of the distance in between two points...36	
Figure 7. 2 the pixel distance limitation.....	37
Figure 7. 3 (a), (b) & (c) input and resultant output video clips	38
Figure 7. 4 results gathering by manual data system	41
Figure 7. 5 Confusion Matrix table.....	41
Figure 7. 6 the filtering of clear data set. Remove the unwanted 'No 'data values.....	43
Figure 7. 7 the calculated results of the three models.....	44
Appendix A: 1 Source code of Python 3.8. – Source code 1	50
Appendix A: 2 Source code of Python 3.8. – Source code 2.....	50
Appendix A: 3 Source code of Python 3.8. – Source code 3.....	51
Appendix A: 4Source code of Python 3.8. – Source code 4.....	51
Appendix A: 5 Source code of Python 3.8. – Source code 5.....	52
Appendix A: 6 Source code of Python 3.8. – Source code 6.....	52

Chapter 1

Introduction

1.1 Prolegomena

'Security' and 'Terrorism' are both very familiar words for all the citizen of the country during last three decades. Persons from all walks of life were affected. Not only that, war caused a huge impact on the Economic, Political development of the country. Even though the military forces of the country and the respective government of the day was able to end the war in 2009, again the fear of terrorism raised its ugly had with the Easter Sunday day attack at highly populated religious locations. It revealed that there is no limit to the level of terrorism. These, incidents raise the alarm on similar future calamities or challenges from different levels of terrorism. However, establishing the safety of each and every human being of a nation is not an easy task.

The use of science and technology to fight terrorism is one aspect of the solution. It is emphasized here that early detection of suspicious items or behaviors will help to avoid such disasters. Military establishments are increasingly engaged in similar experiments to find the best solutions. Focus on the use of Closed Circuit Television (CCTV) Cameras for the early recognition and detection of abandoned objects will significantly help to avoid such disasters caused by terrorists.

The Defence Services Command and Staff College (DSCSC) was selected to do this research because the said establishment already has 72 CCTV Cameras and monitoring systems. Selected models are compared to find out the model that gives the best results in accuracy by the readings of the camera footages. That will open several avenues for future security aspects.

The findings of the research will not only help the individual purpose of this research but the national level applications in the future.

1.2 Back ground and Motivation

A variety of CCTV Camera systems are functioning nowadays in different Organizations and establishments in the government and private sector, based on their requirements such as road tracking, monitoring stores, fences, road traffic...etc. Most of the places implement the system and operate manual monitoring only. Because of that, experts look to the use of robust automated systems to do those functions with less errors while maximizing accuracy. Therefore, number of Methods, Algorithms and Tools were developed in time and those products were used in different platforms to obtain successful results. This research emphasizing on the comparison of three selected models to obtain the most accurate, practical and efficient results out of CCTV Camera footages extracted from a military establishment Defense Services Command and Staff College (DSCSC) Batalanda Such comparison will help to improve the standard and concepts which exist DSCSC. The said results can be studied to implement in similar establishments such as Airports, Railway Stations, popular religious places etc.

Most of the time, the naked eye of the human being at operation rooms are used to monitor the CCTV network. Time has now come to an end those old and manual procedures. Therefore, the use of fully robust, automated and secure systems to obtain the maximum outcome of the latest science and technology have been invented today. However, still hundreds of thousands of experiments are being conducted on improving the systems every day, every hour and every second somewhere in the world. Sri Lanka, cannot keep using the old versions of science and technology and be in the dark on the new developments in such systems. This research is going to examine the most suitable models to capture abandoned objects and recognition early to avoid future physical disasters.

This research hopes that this is the best time to conduct a comparison of the different models, algorithms, methods to overcome future challenges on the topic.

1.3 Problem Statement

Undoubtedly the detection and recognition of abandoned objects in real-time is not an easy task because of the time taken to decide whether the object is abandoned or

not. It must be recognized very fast and need to have maximum accuracy due the risk of the item missing creating a dangerous situation is very high. Sooner the detection and recognition is completed is better as more physical steps such as alarming, attending to rescue the object or the evacuation, protection of the public and handling the public in a very systematic manner have to be considered thereafter.

Therefore, time is of essence. The research is going to analyze the best model to be used for detection out of three models and continuing testing of weak points and how do we overcome or finding the reasons to be avoided in the future which are the barriers to obtaining successful models which can ever use under all circumstances.

1.3 Aims and Objectives

1.3.1 Aim

- Is to develop a real time surveillance solution by investigating and integrating abandoned object detection and recognition models.

1.3.2 Objectives

- To enhance the knowledge of abandoned object detection and recognition models through a comprehensive literature survey.
- To identify the significant parameters on abandoned object detection and recognition.
- To obtain the relevant knowledge on algorithms, frameworks, libraries, neural network behaviours in the abandoned object detection and recognition environment.
- To determine accuracy performances in different models that provide most effective and efficient output results on abandoned object detection and recognition.

- To finalize the most accurate model after the proper evaluation cycle, and identify the weaknesses.

1.5 Proposed Solution

This research is going to be a comparative analysis of the most effective models that are being used to detect, identify or recognize abandoned objects nowadays. There we propose to use of background, foreground subtraction methods, Image processing and neural networking methods and the number of algorithms to obtain best accurate results, in order to obtain the best decision whether the particular object is abandoned or not.

Initially, the tools, algorithms and the methods which are related to the concept that we are going to use in live recognition and detection of objects is considered.

After that the required video clips that are going to test as the samples will be captured and test through the selected models separately. The research has to consider various factors and circumstances which cause the changes on final result there we need to check the reasons to those observations.

Finally, based on the results the finalized models will be used to implement on real time checking of the tested data set whether the accuracy and the practical efficiency on ground results through the CCTV footages.

1.6 Structure of the thesis

This thesis has grouped in to eight Chapters. Chapter one (1) present the problem and the motivation of this thesis. Chapter Two (2) include the Literature Review Where the piece of papers that already discussed about the analytics, neural network, machine learning, image/video processing. Chapter Three (3) contains the details and capacities of the specific technologies and their usability to obtain the desired objectives. Chapter Four (4) elaborated the proper approach to the research problem to follow the path towards goal to fulfill the desired objectives. Chapter Five (5) discussed about the top level design and

the analysis in order to provide an outline structure of the research. Chapter Six (6) illustrates and discussed the in detail explanation of the implementation of each steps of the previously elaborated design. Further it revealed the all algorithms, tools, models, libraries etc. Chapter Seven (7) followed the evaluation of the implemented system and its results. Chapter Eight (8) finally discussed the conclusion and the future works to be done in order to improve the findings and the observations.

Chapter 2

Literature Review

2.1 Introduction

Chapter 1 gave an overall description about the project. This chapter gives a critical review of the existing literature in relation to abandoned object detection and recognition to improve the real time surveillances.

2.2 The existing Researchers about the Abandoned object detection and recognition to in real time surveillance.

Number of researches have published regarding abandoned object detection under various scenarios are illustrated below. Most of the researches are based on the detection of objects which are untouched or unattached and away from the owner in real time.

In this research author propose early detection and recognition of an object as an abandoned and then try to compare and analyze the performance of three different models to provide fast and accurate detection results. Further, author used the most suitable tools, algorithms and theories to determine the best performance model, then carryout few experiments to evaluate the approaches and implement effectiveness.

Most of the researches have conducted their research using time to time discovered tools or of their updates by using different algorithms and methods to prove their hypothesis. Most of them use to follow the state-of-the-art frameworks to obtain more accurate and quick responses in real-time surveillances. Basically most of the researches are based on security surveillance detection platforms[1]. In early days, the identification of the abandoned object concepts were done by the background and foreground extraction techniques to recognize the static foreground regions as suspicious object candidates. These methods were very limited to identifying different types of objects. Therefore the results were not up to standard to what were expected. The evolution of computer vision, Convolutional Neural Network (CNN) got drastically advancement on image detection,

classification, semantic segmentation and pose estimation [2-6]. Therefore, several well advanced and efficient tools have developed in modern science and technology.

[7]The realtime tracking of the object has done by the use of blob tracking after modeling the background model. But there is a observation of weaknesses of the robustness. This research has adopted the Gussian distribution of the adaptive mixture model after that evaluated which are the most likely to result from a background process. The researches have used various environment of situations to apply such abandoned object detection in different circumstances and certain feactures that can help to conduct many functions such as back tracking verification for visual surveillance,[8] the researches have focused on the abandoned luggages in video surveillance. It explained that the combining of the short term and long term background models instread of extract the foreground models. Futher, it assisted by the temporal transition of code patterns. Then examine whether the candidate regions contained the abandoned objects from analyzing the back traced trajectories of luggage owners.there the researches have used the dataset of PETS2006 and AVSS2007. When, Security is a concern of any Organization and falls in to a very critical topic in a military cantonment or an establishment. The use of Closed Circuit Television (CCTV) facilities are common now a days everywhere but, has no value if it is just run with very low attention and care manually as opposed to being very advanced and critical. Actually, the use of most accurate and reliable automated or robust system to detect and recognize the suspicious or abandoned object cause large scale advantage rather than the manual system.

Therefore, the researchers have developed advance and high technical scientific Deep learning models to acquire all kinds of moving and static abandoned suspicious object detection and recognition by models such as Faster Region Convolutional Neural Network (Faster-RCNN), You Only Look Once and (YOLO) and Single Shot Multibox Detector (SSD) [9]. Most of the time every models will be evaluated during the result end. Because there must be any metric to evaluate the accuracy of any product otherwise there is no value of a particular product. Therefore, the use of most effective, efficient and accurate datasets [10] PETS2006, PETS2007 and [11]ABODA can evaluate any kind

of framework properly. Military is highly concerned about the security of the nation and it's public more than they are about themselves. But there may still be loop holes where terrorists can infiltrate easily, requiring the need to be prepared all the time. Number of experiments and studies have been conducted to strengthen the security situation via visual objects.

Wentong Liao and et al [1] elaborated the use of those concepts to find solutions. Through Security Event Recognition Dataset (SERD) from deep learning methods, the said researches mainly focused their effort to find abandoned objects and analyze their latter events in three different scenarios by identifying of the object's real owner, whether real owner will touch and take the suspected object back again, Someone else move it away to other place, otherwise steal it. There used fast RCNN for long- term and short-term abandoned object detections under state- of – the –art frameworks. Comparatively they could achieve the goals rather than they expected in the event they experienced. Further, they faced commonly certain issues like illumination changes, shadows and high density of moving objects under the study.[12] explained that critical study on the temporary static object detection and recognition in real-time in highly public areas with large number of camera operation, there researches paid maximum attention on the mechanisim on detection and calculation of the distance among the owner and the object to 100% accuracy to detect and recognition of the the object as an abandoned.Further, researches have emphasised of Pixel level background modeling there, focused of using the Gaussian Mixture Model (GMM), Modeling of Temporarily Static Objects and Region Level background Modeling.

The new angle of acquiring of the objects which consider to be fallen in to an abandoned category has introduced by [13] concept based on Three dimentional Image Information. Actually, initial abandoned object detection has been done by based on static foreground region segmentation algorithm on video surveillane obtained from the cameras. Secondly the proposed concept of Three-Dimentional (3D) object information reconstruction with images of the binocular camera.at last determine whether the detected

object is hazardous to normal road traffic, road plane equation and height of suspected abandoned object are calculated based on the 3D informations.

[14] explained that the robust foreground and abandonment analysis for large scale abandoned object detection in complex surveillance video such as more crowded locations. The theory behind the process is to develop foreground analysis which can effectively differentiate the foreground objects from the background under the challenging circumstances such as Lighting changes, low texture, low contrast and the cluttered background. Researcher has emphasized on minimizing the false positive output to obtain the maximum accurate abandonment decision. These researchers have used the confusion matrix to obtain the most accurate output. Again the use of Gaussian Mixture Model (GMM) for detecting the foreground has been used by [15] to model the background. The researchers have used three stages to implement the said concept. Firstly, it explained the use of Intermittent Updating Scheme (IUS) for updating the background model which will retain the static abandoned object in the foreground for abandoned object detection in the highway scene. Secondly, used to erase the dynamic foreground and environment noise. Finally, integrate and Mixture model based tracking method into the proposed framework. This could clearly and accurately detect the abandoned objects robustly. Further, experimental results were demonstrated and proved that findings caused a huge positive impact on real time surveillance in the highway.

The illumination changes badly impact on the abandoned object detection and recognition. This was discussed by [16] and it clearly revealed how to overcome the barriers from using of dual background model system. Not only that it was a solution even for the impact caused by the complex circumstances such as occlusion, long-term abandonment and owner re-attendance. Finally, even though the illumination changed during the process, the effect on the final output can precisely track the target object in order to decide the particular object becomes abandoned or not. Ugur Alganci [17] presented a very important comparison type research. The research selected the most important literature in the area and the deep learning algorithms as well as the high technology equipment capture Very High Resolution (VHR) Satellite images to analyze. This

research focused on latest advance deep learning models such as Faster RCNN, YOLO Version3 (YOLO-V3) and the SSD. The research went into detailed analysis of the very rare Satellite images. In the present research, it was very difficult to obtain similar images and do such a kind of research for lack of ability to obtain the required amount and quality of images to conduct an analysis. There, the team tested the performances of the models in different scenarios. At the end they found that the Faster RCNN comparatively performed fastest response than other two with the highest accuracy. YOLO-V3 took second place and SSD took the third place. SSD provided the lowest object detection performance but, it is best in object localization. Further, it was revealed that the larger and medium type plain images could be detected in higher accuracy.

There are number of researches or journals that had conducted research on detection and recognition of abandoned objects in real time. Therefore, from the beginning to date there were number of algorithms, methods, tools, techniques, and models conducted from different interest parties such as foreground object detection algorithms and background subtraction techniques, background segmentation models and Gaussian Mixture Model (GMM)...etc. There was a huge and drastic jump to finding deep learning concept from the identification of Region Convolutional Neural Network (RCNN). Then eventually steps took to obtain a real-time recognition or detection of objects within a very limited short time. It was a great achievement of the surveillance of real time detection. That has been developed up to not only having RCNN but expanded to having tools such as Faster RCNN, YOLO, and SSD ...etc.

The use of above findings continuously changed and were updated to cover the loop holes of those products to obtain a better outcome. But there are few researches that evaluate the efficiency of different tools in abandoned object detection and recognition on real time. Even the comparison of satellite images referred above which were taken through the high definition cameras was not on detection or recognition of abandoned objects in real-time. The value of the finding of this research has a big value in the context of internal security purposes. Therefore, the research plan is to conduct the evaluation on the Faster RCNN, YOLO-V3 and SSD under the real-time environment.

2.3 Study of the Research Problem

2.3.1 The abandoned object

There are different types of definitions to identify an object as 'abandoned'. Several types of metrics may cause to define any object (moving or static) as abandoned. Therefore, the time and distance are the main factors which have a bearing on the definition or identification of any kind of object[18].

2.3.2 Detection and Recognition of abandon objects

The use of technologies to identify or recognize the moving and static objects with the help of specifically created algorithms, methods and tools to acquire finalized accurate results for further action of process[18].

2.3.3 Real time Surveillance

Sri Lanka has the experience of witnessing terrorists causing large scale bomb blasts in public places and the whole nation gripped with fear. Therefore, there is use of modern video surveillance technology installed to cover such areas and those places are observed manually. However, the problem of late reactions against the suspicious incident, object or movement remains due to the time taken to take the precautions. Therefore, everyone is interested in an automated robust system instead of the manual system in order to save time and money.

Further, to make such system a success huge effort had to be taken by installing and adopting appropriate models, sophisticated electronic devices etc. In addition to that, while conducting a real-time surveillance process number of problems such as detecting moving characters between human and animals in different positions under quick movements, low light and shadows etc.[19].

2.4 Summery

Research exist on manual surveillance of abandoned object detection on real time. However, as highlighted above number of problems still exist. A comparison of these surveillance tools will lead to greater efficiency and help to avoid major disasters like what we have experienced in the past. Therefore, next chapter will explain the technologies that are required to address the problem.

Chapter 3

Specific Technologies Used

3.1 Introduction

The previous chapter focused on past literature on abandoned object detection and recognition to improve the real time surveillances. It gave a fair knowledge about the past and latest finding on above subject comprehensively. This chapter illustrates the latest technology and the capabilities in examining a comparison of abandoned object detection and recognition to improve real time surveillance.

3.2 What is Image Processing

An image simply can be explained as a kind of signal which contains information. It can be defined in a way containing mathematical function of X and Y spatial (Plane) [20]coordinates. Those coordinates give value of a frame call 'Pixel' in any prominent place of the image. The reflection of sunlight from any kind of physical object once go through the lens of a camera will become a 2D signal and result in image formation. This image will process through the digitizing techniques finally resulting in the 'digital image'. The 'Sampling' and 'Quantizing' are the two concepts of converting an analog image to digital image. Further, using analog images to work in a computer will require huge amount of storage capacity requiring to convert the analog image to digital image when we do other tasks through the computers.

There are number of fields used to smoothen work with the help of image processing technology such as Television field, Satellite images, Medical, Robotics and Law enforcement fields. This research will use image processing techniques for abandoned object detection. The handling of the image processing needs to skill in handling languages like Python, MATLAB, C++ or Java. The fundamentals of the image processing take important place when we consider the proper function of the said process.

Therefore, the best block diagram which indicates the figure 3.1 fundamental of the image processing as follows;

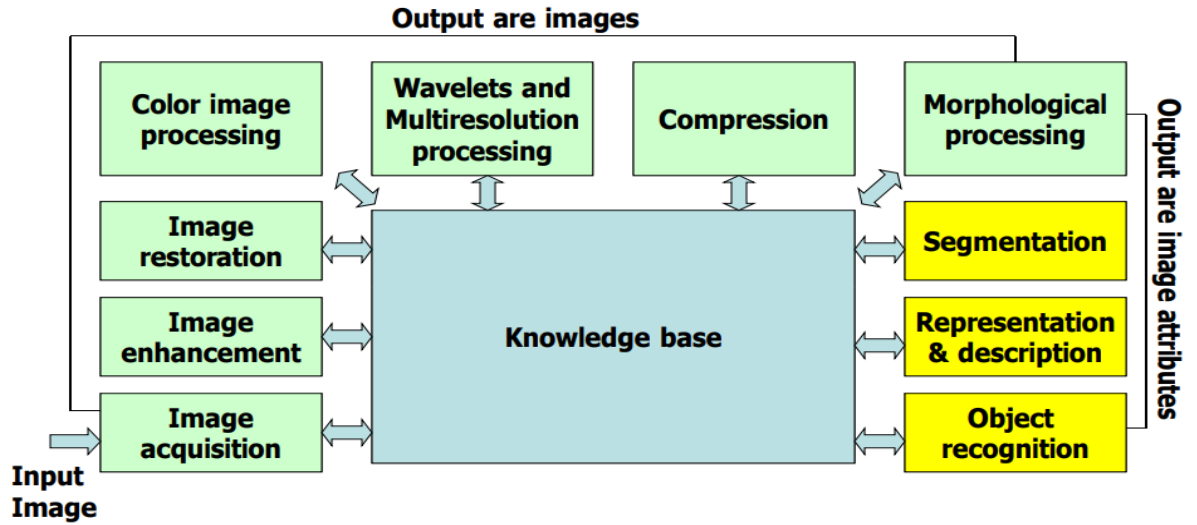


Figure 3. 1 the Fundamentals of the Image Processing

There are ten number of facts are directly deals with the Knowledge base system to obtain the best outcome of image processing. Then, there are another seven components which help image processing function as shown in the Figure 3.2.

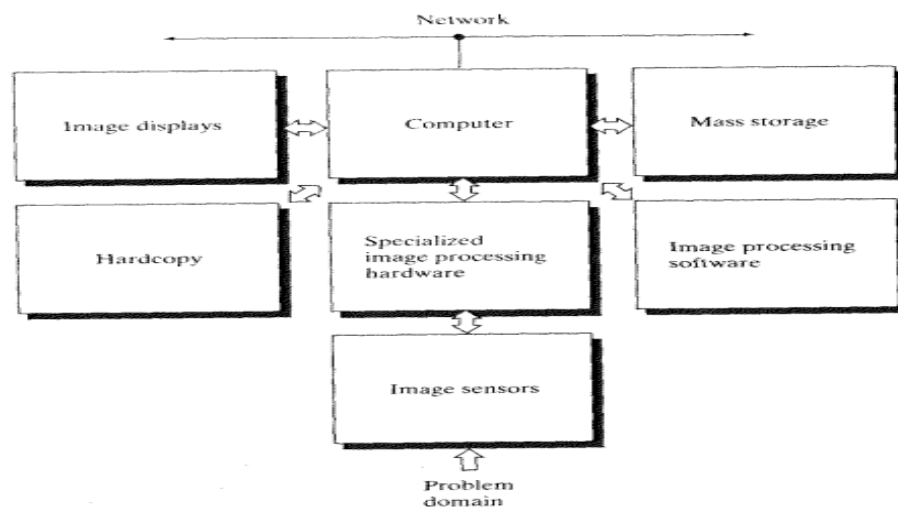


Figure 3. 2 the component of the Image Processing

3.3 What is Video Processing

In 'Video' there are two parts Analog and Digital. Simply put the 'Analog Videos', natural videos but this is contain very big capacity and also need to have big amount of storage capacity. Therefore, the modern Science and Technology was searching a good solution to overcome the use of big storage capacity when storing the analog video. There are number of an Analog video formats like NTSC, PAL and SECAM ...etc.

With the concept of Digitizing, the usage of analog has drastically gone down. Scientists were looking for a solution to the storage problem and lack of flexibility on handling and manipulating those sources and found digital video concept from the use of sampling techniques. There, quantizing of the spatial, temporal and after that the resultant pixel intensities are quantized. The figure 3.3 will explain very simply how a Digital Video is created.



Figure 3. 3 Simple Block diagram of creation of Digital Video from Natural Scene.

With the development of the digital video concept there are no of doors opened in several fields like video Teleconferencing, Multimedia authoring system and Education fields. Eventually, the process on using in multiple areas it was revolutionized in the world of multimedia then no of creations such as Digital Versatile Disk (DVD), Digital Satellite System (DSS), High Definition Television (HDTV) and Digital still and video Cameras are the few of new products came in to the field. Further, concept prevail towards the

techniques like video compression, video indexing. Video tracking and video segmentation ...etc.

3.4 What is Object Recognition and Detection

During the period of last recent years everywhere of the world automatically created a life threat among the human beings due to terrorist threat from mass destructions. This cause huge damage the asserts of the public and private not only that badly lost innocent humans lives too. Therefore expertise started to find solutions from installing and monitoring manually CCTV Camera systems regarding the suspicious activities, movements and keep owner-less kinds of boxes or luggage ...etc[21]. Occurring in the public areas. But this was became a very big waste due to use of big no of human power and even there are lot of mistakes due to human errors. Therefore, the interested parties searching some user friendly automated and efficient system to implement from obtaining those camera footages. Defining different kinds of theories and assumptions researches got use to analyze the digital images and video to identify the owner less items in the crowded areas. Those abandoned items got used to find through the techniques. There they first need to identify and confirm the particular item is an abandoned (Object recognition) then items need to be detected clearly what is that and what kind of category it belonging.

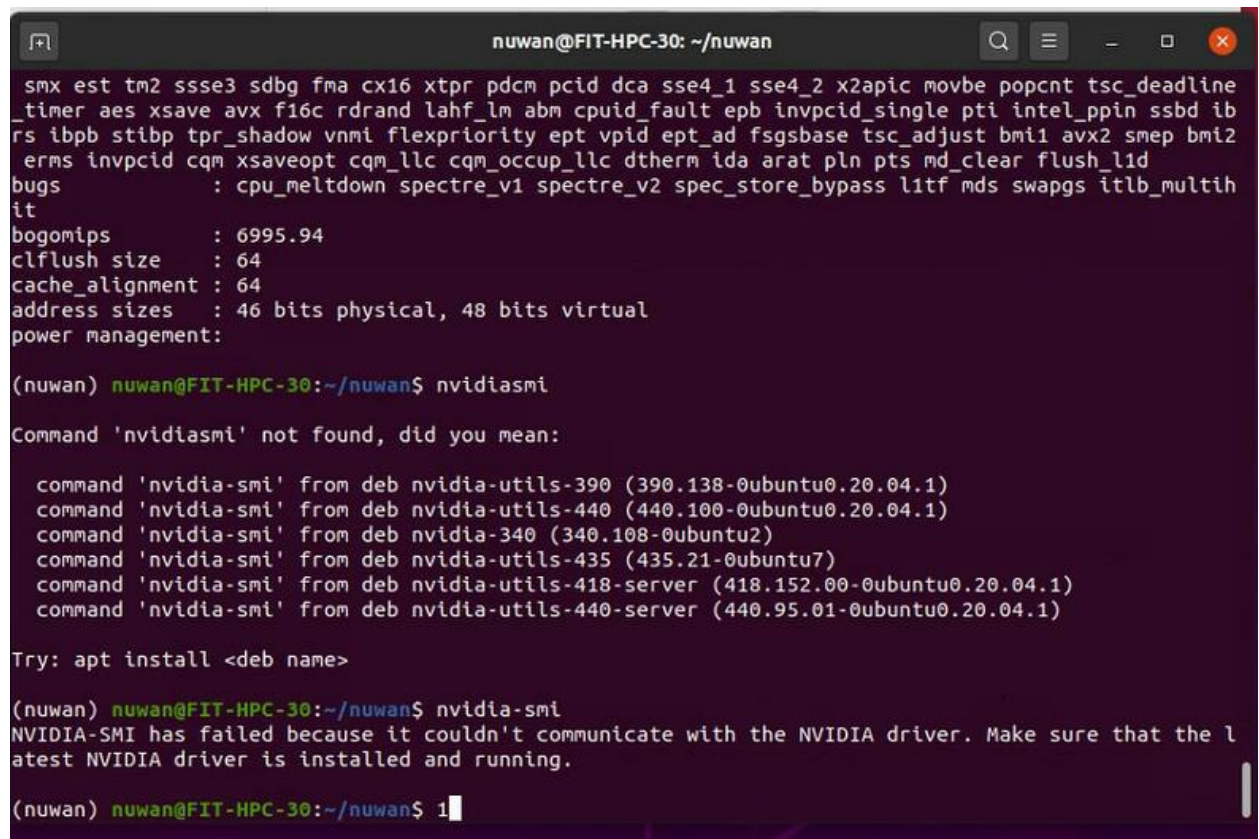
The concepts and technologies like foreground segmentation, abandonment validation, Stationary object detection, creating frame works and lots of efforts were done. Up to day expertise are still doing their experiments of real-time detection and recognitions to obtain as quickly as possible of correct and accurate decisions to be adopt. Further, they are finding solutions to get reduce the obstacles such as shadow, incidents like high density of moving the objects, illumination changes ...etc[22].

3.5 Neural Net Work

The Neural Network (NNW) is simply define is one of the best 'Deep learning' concept coming under machine learning which contains different algorithms and use to recognize the patterns. Mostly the input data will insert with labels as datasets to the NNW, then it will produce the correlation among the inputs and out puts. Further, the input can be an images, texts or any. NNW do the 'Classification' and 'Clustering' the input data for the deep learning to find the patterns or the correlation in between among them[23].

3.6 The performances of Remote PC

The remote PC used due to the high performances required for the deep learning process in the neural networking, the Operating System of the PC was Ubuntu and it contained the 8 no of processors. RAM was 16 GB. The following Figure 3.4 illustrates the performance details of the remote PC.



```
nuwan@FIT-HPC-30: ~/nuwan
smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline
_timer aes xsave avx f16c rdrand lahf_lm abm cpuid_fault epb invpcid_single pti intel_ppin ssbd ib
rs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2
erms invpcid cqm xsaveopt cqm_llc cqm_occup_llc dtherm ida arat pln pts md_clear flush_l1d
bugs          : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multih
it
bogomips     : 6995.94
clflush size : 64
cache_alignm : 64
address sizes : 46 bits physical, 48 bits virtual
power managem:

(nuwan) nuwan@FIT-HPC-30:~/nuwan$ nvidiasmi
Command 'nvidiasmi' not found, did you mean:

  command 'nvidia-smi' from deb nvidia-utils-390 (390.138-0ubuntu0.20.04.1)
  command 'nvidia-smi' from deb nvidia-utils-440 (440.100-0ubuntu0.20.04.1)
  command 'nvidia-smi' from deb nvidia-340 (340.108-0ubuntu2)
  command 'nvidia-smi' from deb nvidia-utils-435 (435.21-0ubuntu7)
  command 'nvidia-smi' from deb nvidia-utils-418-server (418.152.00-0ubuntu0.20.04.1)
  command 'nvidia-smi' from deb nvidia-utils-440-server (440.95.01-0ubuntu0.20.04.1)

Try: apt install <deb name>

(nuwan) nuwan@FIT-HPC-30:~/nuwan$ nvidia-smi
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make sure that the l
atest NVIDIA driver is installed and running.

(nuwan) nuwan@FIT-HPC-30:~/nuwan$ 1
```

Figure 3. 4 specifications of the remote PC and the GPU.

3.7 Experimented Tools

3.7.1 Faster RCNN

The Convolutional Neural Network (CNN) evolved speedily in deep learning since CNN till Faster RCNN through the CNN, RCNN, Fast RCNN then Faster RCNN. The ability and the efficiency of the fast and accurate detection and recognition of the abandoned object it could achieve the very good result. This was mainly depend upon the technique of the ‘Region Proposal Method’. Further, Use of Graphical Processing Unit (GPU) is most effective to obtain the out results more quickly. Figure 3.5 shows the block diagram of the Faster RCNN function.

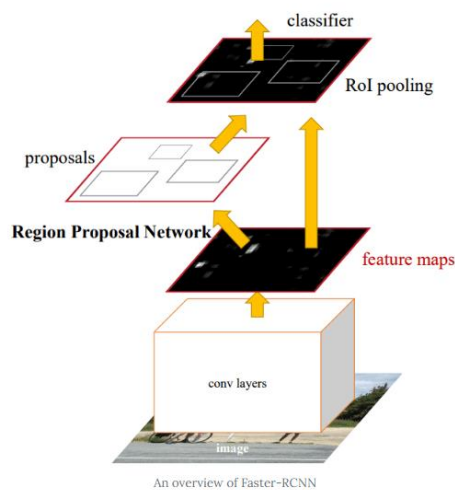


Figure 3. 5 Faster RCNN Function of an object detection and Recognition.

3.7.2 You Only Look Once – Version 3 (YOLO – V3)

This deep learning model presently has the most important feature of the detecting of the object in three different scales. This cause the output generated by 1 x 1 kernel on a feature map in three different sizes in three different places in the network.

Comparatively, it could detect small objects than the YOLO-V2. YOLO-V3 could create more bounding boxes for the particular image than the previous version. In brief the model is very fast in object detection comparatively other and previous tools in real-time detection environment without loss in too much of accuracy. Another important point is this model uses in the single neural network then it could be able to recognize not only the class labels but even the locations. Further, model could be able to divide an image in to different regions then each could predict bounding boxes probabilities for each region. The Figure 3.6 indicates the function of the YOLO- V3 to object detection and recognition.

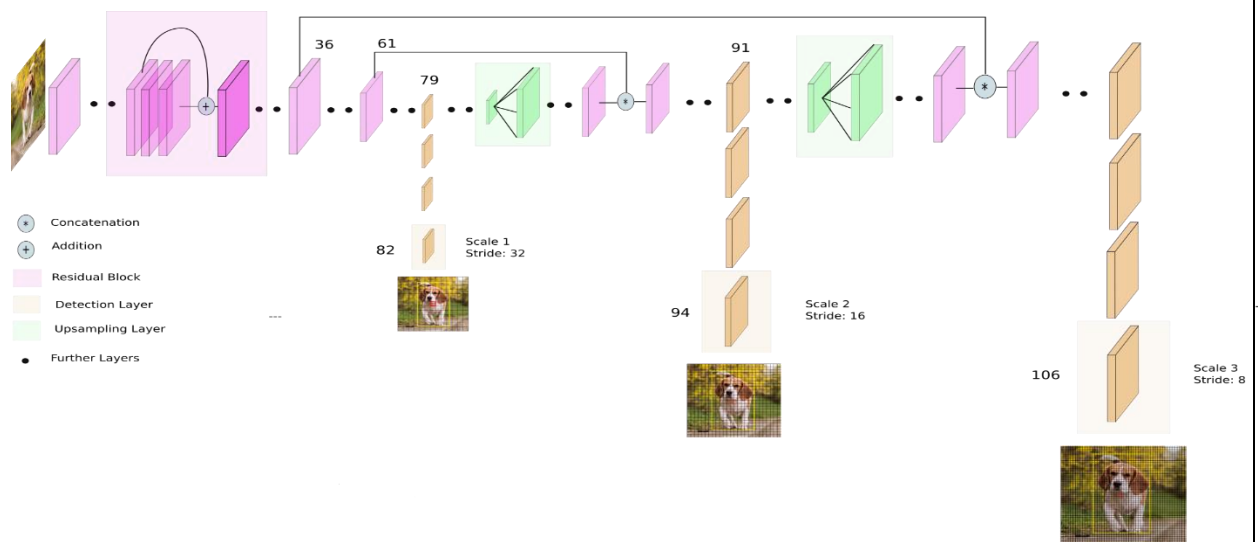


Figure 3. 6 the Function of object detection and recognition from YOLO-V3.

3.7.3 Single Shot Multibox Detector (SSD)

SSD was introduced to object detection since 2016 used the COCO or PascalVOC datasets. SD was good in localization and classification functions. Further the accuracy on an object detection was very high. SSD networks only use a single network to generate bounding boxes and simultaneously classification by regression-based method. SSD can achieve real-time processing on GPU. It breaks the input

image in to grids by detecting objects with SSD network, and each mesh predicts the trust or the confidence and point of two object boxes. Futher, the higher level of the network SSD worked with the default boxes of the feature maps. Then using the feature maps it can detect or recognize the different object types in different scales. Figure 3.7 illustrates the general architecture of the SSD network.

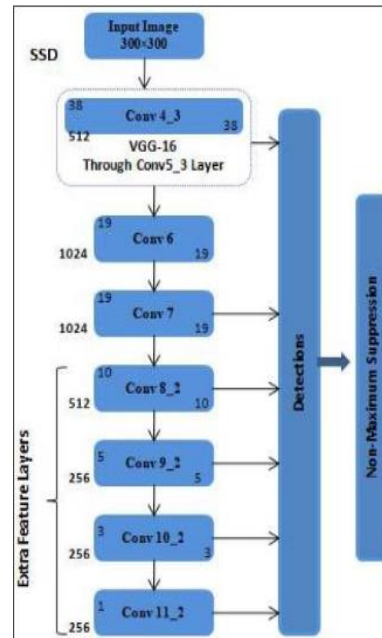


Figure 3. 7 the General architecture of the SSD network.

3.8 Summery

This chapter involved with the technologies that are going to be used during the research itself. Mainly the image and video processing are the main technology to compare the efficiency of the detection and recognition of the abandoned objects in real-time surveillance. In this sense the use of three deep learning tools of Faster RCNN, YOLO-V3 and the SSD and evaluate their performances, efficiencies will be calculated. The next chapter will discuss about the novel approach to obtain the desired results to finalize the most suitable tool (Out from faster RCNN, YOLO-V3 and the SSD) to use in future researches from above mentioned tools how far will it possible.

Chapter 4

Image and Video Processing approach to Comparison on an Abandoned object Detection and Recognition in real-time Surveillance.

4.1 Introduction

This chapter focuses on gathering of information and data, then categories data into two categories, like train data and test data. Those data will use to detect or recognize the abandoned objects by the use of Image and Video processing tools. After that research plan to do a comparison among three most advance and effective tools then will decide to find best resulting tools for the future works. The following Figure 4.1 illustrates the General model for the comparison on tools of object detection and recognition.

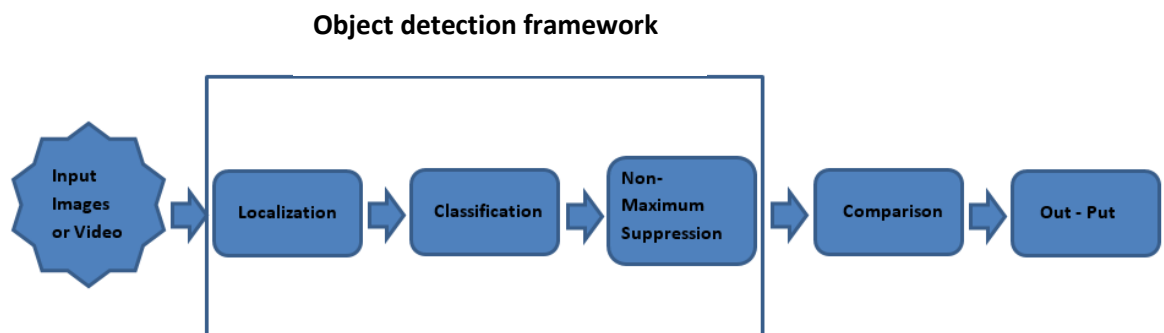


Figure 4. 1 General model of comparison on tools of object detection and recognition.

4.2 Input

Since this is a comparison type research, I could capture the required amount of the CCTV video and image footages under 75% of training data and 25% of testing data. The DSCSC camp itself I could obtain such amount of videos then I made the images by the

use tool call ANACONDA. After that the images validation needed for the future works. After that, the cleared data can be used for the future experiments.

4.3 Output

The main output of the research is to decipher the most efficient tool to detect and recognize of abandoned objects accurately through image processing concept. This will be an indicator for the future researches to select the most correct path to use the best tool for the relevant purpose as well.

4.4 Process

The process of image and video processing properly scrutinize each step one by one. The using of especial kinds of algorithms, methods and tools the object detection or recognition measured. But the comparison of the most efficient and effective tool out of three tools (Faster RCNN, YOLO-V4 and SSD) were measured through the final result of each tool. The accuracy percentage and less time consumption taken in to the account. Because those two factors directly considered to finalize the best tool to use for future experiments or the applications to quick and accurate response.

4.5 Users

The researches, manufacturers in different kinds of fields like medical, science, and engineering are vastly use this technology.

4.6 Features

The solution proposed from this research will immensely benefited to all researches in their future researches. Because, the findings of this research will use for future activities in order to obtain the maximum output. Further, these concepts can use in no of fields to under different circumstances. Most important point is using of correct algorithm or the tool to correct place or the correct problem to obtain the maximum or the best result could be finalize from the findings from this research.

4.7 Regression

In this research data images will be the Input to the different three object detection and recognition tools in the machine learning. Those are real time live and continuous images obtain directly from the CCTV camera and the tools will exhaust the output as the speed of input coming through the tool. The concept of the image processing will discuss under the supervised learning. Therefore, the identification of the abandoned objects more accurately will produce the minimum of errors. Then the tools which provide more accurate outputs with minimum of errors will be the best tool to use for further detection process.

4.8 Classification

The point of data classification under the supervised learning will support to identify the image whether directly right or wrong. That sense to algorithm to whether the input is a particular identified object or not. If it is 'yes' then the bounding box will appear as red color if it is 'No' it will not appear as such. This will be measured separately and individually from each three tools. Even though this is for the binary classification sometimes it may happen for multiple class classification. Then, in case of multiple classes tools or the models will performed for the multiple class object or the variables.

4.9 Summery

This chapter discussed the novel approach to find out the best tool out of all other tools at present. The research conducted from using most functional and effective three tools. The findings of the research would directly important for the future applications in different aspects of different platforms. The next chapter will illustrates the research design for the above mentioned novel approach.

Chapter 5

The Research design for the comparison on abandoned object detection and recognition in real-time surveillance.

5.1 Introduction

The last chapter explained the research with the required and prominent image and video processing approach to the abandoned object detection and recognition in real time with the minimum delay. During this chapter research focuses on best top level design for the abandoned object detection and recognition in real time surveillance. Further, there illustrates the comparison of three tools (Faster RCNN, YOLO-V4 and SSD) then obtain and select the best performance tool for future matters under the limitations.

5.2 High level Architecture of Research Design

The expansion of the science and the technology of the planet of human beings. They got used to create things to make the life comfortable in several no of fields like Administration, Daily affairs and living standards. Secondly they got use of machines to help their works and hoped to obtain massive out come in the sense of high efficiency. Because of that they had to face for 'competition'. This course them to think of the safety or the security in different angles and circumstances. Since the era of the rock all the way come to today there were lots of things happened, but today world become and entertain the digitized environment. Therefore, we need to concentrate of the safety of every matter. Therefore the security has become a vital role in everywhere.

The bad out result of terrorism came to the theatre with the life treat. Therefore, the use of every precautions really important. Then, States were spent billions of money to find most sophisticated solutions to live free and happiness. The avoidance of disasters by the

use of visual detectors and sensors really important than manual detection methods. Because it can minimize the life risk of those who are engaging with that duty or the treat. The introducing of the Computer Vision Technology and its application to the world there are lot of models, frameworks, theories, algorithms, environments...etc were innovated to the world. This caused to produce new Concepts, Products in order to solve the aforesaid quarries.

In this research author supposed to do a comparison on most effective three models which related to an object detection and recognition [17] in real-time under heterogeneous situation of video footages with most common and practical ground situation. First the data were obtain by the use of Static CCTV Camera footages and then those footages will be run through the latest software framework model to train and deploy for deep neural network in order to an obtain the quantitative results to analyze. Further, it obtains the support of the certain types of applications for data pre-processing and post processing. Then, the object detection processes by the three selected models. The analytical comparison considered by the accuracy of the three models separately. Further, there is a depth written program run through the python platform (latest Python 3.8 with anaconda) to indicate the results. Secondly, the same video footage tested through the three models simultaneously to compare the minimum time taken to indicate the results by keeping the rest of the other parameters stable. Following figure 5.1 Activity diagram illustrates the flow of the high level design clearly.

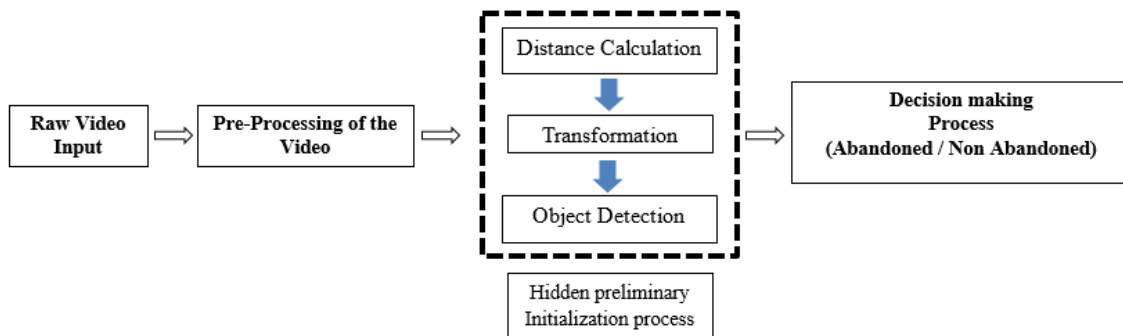


Figure 5. 1 an Activity Diagram

The input video were obtained from fixed static HD CCTV cameras which already installed in the DSCSC premises. Once the video clips obtained those clips were clearly and systematically resized, rectified the blur portions and transmission of the color pattern from the BGR to RGB of the selected clips. Thirdly, the padding adjustment, distance calculation then the initial object detection done before the next step. Finally, the decision making process of the recognized object will be an abandoned or not decide in the last step of the process.

5.3 Summery

This Chapter discussed about the high level architecture of the research design and the analysis. The activity diagram has explained the most important key steps and the algorithms, models, libraries and the based platform. It could give a fair understanding and forecast of the research. The next chapter do an in detail explanation of the proper Implementation of the each steps for the better understanding.

Implementation

6.1 Introduction

The previous chapter was discussed about the initial or the overall research design architecture. This chapter illustrates and discuss in detail description of each an individual step very clearly. Further, this chapter does a comprehensive clarification and explanation about all the methods, algorithms, models, libraries, frameworks and theories used to obtain the final desired results.

6.2 Selection of Input Video clips

The MXnet data framework developed by the Apache Software Foundation. It is open source deep learning framework that allow to train, define and deploy on deep neural network in wide array of platform. On top of the MXnet platform the OpenCV, GluonCV and the Common Object Context (COCO) is a large scale object detection, segmentation and captioning dataset. Further, MXnet framework comparatively accurate and fast both in GPU (Graphical Processing Unit) and CPU (Central Processing Unit) environments. Figure 6.1 related infrastructure indicated the required resources for the object detection. The OpenCV (Open Source Computer Vision Library) basically used to provide infrastructure of computer vision applications such as image processing.



```
2 import math
3 import os
4 from gluoncv import model_zoo, data
5 import mxnet as mx
6 import numpy as np
7 import cv2
8
9 COLOR_RED = (0, 0, 255)
10 COLOR_GREEN = (0, 255, 0)
11 COLOR_BLUE = (255, 0, 0)
12 BIG_CIRCLE = 60
13 SMALL_CIRCLE = 3
14
15 #####
16 #####
17 #####
18 nets = ['faster_rcnn_fpn_syncbn_resnet101_coco', 'yolo3_darknet53_coco',
19         'ssd_312_resnet50_v1_coco']
20 print("Available models are:")
21 for index, model_name in enumerate(nets):
22     print("{} [{}]-format(model_name, index)")
23 net_num = input("Enter the number of the net : ")
24 net = model_zoo.get_model(nets[int(net_num)], pretrained=True, ctx=mx.gpu(0))
25 net = model_zoo.get_model('yolo3_darknet53_coco', pretrained=True, ctx=mx.gpu(0))
26 net = model_zoo.get_model('ssd_312_resnet50_v1_coco', pretrained=True, ctx=mx.gpu(0))
27 object_ids = [40]
28 threshold = 0.4
29 size_frame = "720p"
30 distance_minInum = "400"
31
32 #####
33 #####
34 #####
35 # https://gist.github.com/AruntRC/7b3dadd004d04c00198557db5da4bda
36 class_name_mapping = (0: u'__background__', 1: u'person', 2: u'bicycle', 3: u'car', 4:
```

Figure 6. 1 related infrastructure

The raw input video clips initially obtain from the HD CCTV Outdoor and Indoor cams which already installed and operation in the DSCSC premises. Those cams are fixed and static continuously day and night capturing the real-time video for the manual surveillance. These video clips are taken in .AVI format for the research but it does not make any confusion even though using of formats like MP4, MPEG-4 or WMV so on. Here it is used the .AVI format throughout the research. Here Figures 6.2 indicated the example input video clip which was used for the research. The research basically done in the MXnet data framework.

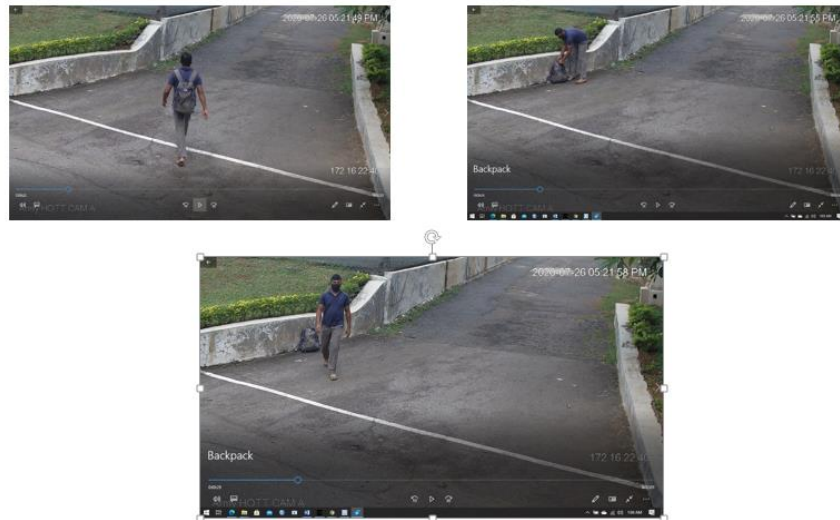


Figure 6. 2 Raw input video clip

6.3 Pre-Processing of the input videos

The resizing of the correctly absorbed live videos, transformation of the colors from BGR to RGB which is accepted pattern of the particular system, irradiating of unwanted formats or the data could be removed then the input sources were converted to a useful manner under the pre- processing stage under the figure 5.1 Activity diagram. The following Figure 6.3 pre-processing, source codes indicate how those functions were occurred as follows.

```
88
89 def get_points_from_box(box):
90     center_x = int(((box[1] + box[3]) / 2))
91     center_y = int(((box[0] + box[2]) / 2))
92     center_y_ground = center_y + ((box[2] - box[0]) / 2)
93     return (center_x, center_y), (center_x, int(center_y_ground))
94
95
96 def draw_rectangle(corner_points, frame):
97     cv2.line(frame, (corner_points[0][0], corner_points[0][1]), (corner_points[1][0],
98     corner_points[1][1]), COLOR_BLUE,
99     thickness=1)
100     cv2.line(frame, (corner_points[1][0], corner_points[1][1]), (corner_points[3][0],
101     corner_points[3][1]), COLOR_BLUE,
102     thickness=1)
103     cv2.line(frame, (corner_points[0][0], corner_points[0][1]), (corner_points[2][0],
104     corner_points[2][1]), COLOR_BLUE,
105     thickness=1)
106     cv2.line(frame, (corner_points[3][0], corner_points[3][1]), (corner_points[2][0],
107     corner_points[2][1]), COLOR_BLUE,
108     thickness=1)
109
110 def predict(frame):
111     bboxes_list = []
112     scores_list = []
113     box_ids_list = []
114     rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
115     xrgb = mx.nd.array(rgb).astype('uint8')
116     # x, orig_img = data.transforms.presets.rcnn.load_test(in_fname)
117     rgb_nd, xrgb = data.transforms.presets.rcnn.transform_test(xrgb, short=512, max_size=700)
118     box_ids, scores, bboxes = net(rgb_nd.as_in_context(mx.gpu(0)))
119     # img = gcv.utils.viz.cv_plot_bbox(frame, bboxes[0], scores[0], box_ids[0],
120     class_names=net.classes)
121     # gcv.utils.viz.cv_plot_image(img)
```

Figure 6. 3 Pre-Processing

6.4 Distance Calculation

The distance calculation in between the object and the owner who belongs to the item (Object) calculated in pixel values. The research has planned to do a comparison among three models by keeping other variables static. Therefore, first need to fix the Threshold value, minimum distance, and frame size properly to each video input, then could do the testing by using the three models individually separately. The said parameters have illustrated in Figure 6.4 Distance calculation how those figures have included for a particular video input.

```

4 from glob import glob
5 import mxnet as mx
6 import numpy as np
7 import cv2
8
9 COLOR_RED = (0, 0, 255)
10 COLOR_GREEN = (0, 255, 0)
11 COLOR_BLUE = (255, 0, 0)
12 BIG_CIRCLE = 60
13 SMALL_CIRCLE = 3
14
15 #####
16 # Load models and configs #
17 #####
18 nets = ['faster_rcnn_fpn_syncbn_resnet101_coco', 'yolo3_darknet53_coco',
19         'ssd_512_resnet50_v1_coco']
20 print("Available models are:")
21 for index, model_name in enumerate(nets):
22     print("{} {}".format(model_name, index))
23 net_num = input("Enter the number of the net : ")
24 net = model_zoo.get_model(nets[int(net_num)], pretrained=True, ctx=mx.gpu(0))
25     ctx=mx.gpu(0)
26 net = model_zoo.get_model('faster_rcnn_fpn_syncbn_resnet101_coco', pretrained=True,
27     ctx=mx.gpu(0))
28 net = model_zoo.get_model('yolo3_darknet53_coco', pretrained=True, ctx=mx.gpu(0))
29 net = model_zoo.get_model('ssd_512_resnet50_v1_coco', pretrained=True, ctx=mx.gpu(0))
30 object_ids = [40]
31 threshold = 0.4
32 size_frame = "700"
33 distance_mininum = "400"
34 #####
35 # Class id mapping #
36 #####
37 # https://gist.github.com/Arun111/7b3dadd04da04c00198557db5da4bda
38 class_name_mapping = {0: u'background', 1: u'person', 2: u'bicycle', 3: u'car', 4:
39     u'motorcycle', 5: u'airplane', 6: u'bus', 7: u'train', 8: u'truck', 9: u'boat', 10: u'traffic
40     light',
41     11: u'fire hydrant', 12: u'stop sign', 13: u'parking meter', 14: u'bench',

```

Figure 6. 4 Distance Calculation

The OpenCV library on MXnet framework has determined the distance calculation base on the following triangulation equation.

$$D = (W \times F) / P$$

D: Unknown Distance (Required distance to be calculated)

W: Known Distance

F: Focal Distance or Focal length

P: Per Width

6.5 Transformation

The transformation of the parameters such as corner_points, matrix, width, image, list_downloads were handled by the OpenCV. There the use of 'numpy' and 'cv2' packages used to do the transformation in aforesaid parameters. Further; two functions of 'transform.py' and 'cv2.getperspectivetransform' were used to do the proper transformation as shown in figure 6.5 of transformation of corner points, images. The numpy used to do numerical processing and th cv2 used OpenCV bindings. cv2.getperspectivetransform used to obtain a maxim accurate top down 'Bird eye view' of

the images. Further, there were two arguments were used one is 'rect' used to have the four (4) Region of Interest (ROI) and second argument was 'dst' to have the list of transform points. Finally, the 'cv2.warpperspective' provided the warped image as the return.

```

45
46 def compute_perspective_transform(corner_points, width, height, image):
47     corner_points_array = np.float32(corner_points)
48     img_params = np.float32([[0, 0], [width, 0], [0, height], [width, height]])
49     matrix = cv2.getPerspectiveTransform(corner_points_array, img_params)
50     img_transformed = cv2.warpPerspective(image, matrix, (width, height))
51     return matrix, img_transformed
52
53
54 def compute_point_perspective_transformation(matrix, list_downoids):
55     list_points_to_detect = np.float32(list_downoids).reshape(-1, 1, 2)
56     transformed_points = cv2.perspectiveTransform(list_points_to_detect, matrix)
57     transformed_points_list = list()
58     for i in range(0, transformed_points.shape[0]):
59         transformed_points_list.append([transformed_points[i][0][0], transformed_points[i][0][1]])
60     return transformed_points_list
61
62
63 def get_human_box_detection(boxes, scores, classes):
64     array_boxes = list()
65     for i in range(len(boxes)):
66         if int(classes[i]) == 0 and scores[i] > threshold:
67             array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]),
68 [3]))
69     return array_boxes
70
71 def get_object_box_detection(boxes, scores, classes, height, width, class_ids):
72     array_boxes = list()
73     for i in range(len(boxes)):
74         if scores[i] > threshold:
75             if (int(classes[i]) + 1) in class_ids:
76                 array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]),
77 int(boxes[i][3]))
78     return array_boxes

```

Packages: numpy and cv2
 Functions: transform.py & cv2.getperspectivetransform
 Arguments: rect & dst

Figure 6. 5 Transformation of the images and corner points

6.6 Object Detection

The packages of GluonCV on top of the MXnet framework used to detect the object. The object detection was following no of sequence to detect an object. First step uploaded the required libraries here Opencv and gluonCV, MXnet framework. Secondly, needed to read the image which already entered and preprocessed. Thirdly, the transformation occurred. There was one image created except the original picture. The secondly created picture used to plott the results and later it was released as the output video clip. Fourthly, got use of the pre-trained models from the gluonCV model zoo. There need to keep

remember that arguments in true position. During the detection position needed to use the MXnet framework array. Not only that the use of image classes index needed to when the objects were detected. Figure 6.6 and 6.7 shows the object detection coding and the image classes which already uploaded in the program.

```

63 def get_human_box_detection(boxes, scores, classes):
64     array_boxes = list()
65     for i in range(len(boxes)):
66         if int(classes[i]) == 0 and scores[i] > threshold:
67             array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]), int(boxes[i][3])))
68     return array_boxes
69
70
71 def get_object_box_detection(boxes, scores, classes, height, width, class_ids):
72     array_boxes = list()
73     for i in range(len(boxes)):
74         if scores[i] > threshold:
75             if (int(classes[i]) + 1) in class_ids:
76                 array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]),
77 int(boxes[i][3])))
78     return array_boxes
79
80
81 def get_centroids_and_groundpoints(array_boxes_detected):
82     array_centroids, array_groundpoints = [], []
83     for index, box in enumerate(array_boxes_detected):
84         centroid, ground_point = get_points_from_box(box)
85         array_centroids.append(centroid)
86         array_groundpoints.append(centroid)
87     return array_centroids, array_groundpoints
88
89 def get_points_from_box(box):
90     center_x = int(((box[1] + box[3]) / 2))
91     center_y = int(((box[0] + box[2]) / 2))
92     center_y_ground = center_y + ((box[2] - box[0]) / 2)
93     return (center_x, center_y), (center_x, int(center_y_ground))
94
95
96 def draw_rectangle(corner_points, frame):
97     cv2.line(frame, (corner_points[0][0], corner_points[0][1]), (corner_points[1][0],

```

Figure 6. 6 Object Detection

```

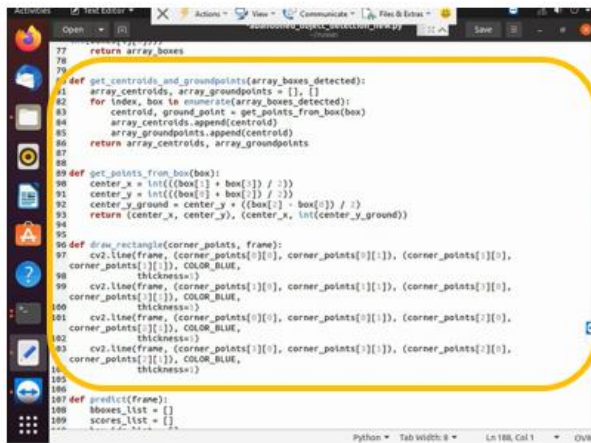
29 size_frame = 700
30 distance_minimum = "400"
31
32 ##### Class id mapping #####
33 # #####
34 #####
35 # #####
36 class_name_mapping = {0: u'background', 1: u'person', 2: u'bicycle', 3: u'car', 4:
u'motorcycle', 5: u'airplane', 6: u'bus', 7: u'train', 8: u'truck', 9: u'boat', 10: u'traffic
light',
37 11: u'fire hydrant', 12: u'stop sign', 13: u'parking meter', 14: u'bench',
15: u'bird', 16: u'cat', 17: u'dog', 18: u'horse', 19: u'sheep', 20: u'cow',
38 21: u'elephant', 22: u'bear', 23: u'zebra', 24: u'giraffe', 25: u'backpack',
26: u'umbrella', 27: u'handbag', 28: u'tie', 29: u'suitcase', 30: u'frisbee',
39 31: u'skis', 32: u'snowboard', 33: u'sports ball', 34: u'kite', 35:
u'baseball bat', 36: u'baseball glove', 37: u'skateboard', 38: u'surfboard', 39: u'tennis racket',
40 40: u'bottle', 41: u'wine glass', 42: u'cup', 43: u'fork', 44: u'knife', 45:
u'spoon', 46: u'bowl', 47: u'banana', 48: u'apple', 49: u'sandwich',
41 50: u'orange', 51: u'broccoli', 52: u'carrot', 53: u'hot dog', 54: u'pizza',
55: u'donut', 56: u'cake', 57: u'chair', 58: u'couch', 59: u'potted plant',
42 60: u'bed', 61: u'dining table', 62: u'toilet', 63: u'tv', 64: u'laptop', 65:
u'mouse', 66: u'remote', 67: u'keyboard', 68: u'cell phone', 69: u'microwave',
43 70: u'oven', 71: u'toaster', 72: u'sink', 73: u'refrigerator', 74: u'book',
75: u'clock', 76: u'vase', 77: u'scissors', 78: u'teddy bear', 79: u'hair drier', 80: u'toothbrush'}
44
45
46 def compute_perspective_transform(corner_points, width, height, image):
47     corner_points_array = np.float32(corner_points)
48     img_params = np.float32([[0, 0], [width, 0], [0, height], [width, height]])
49     matrix = cv2.getPerspectiveTransform(corner_points_array, img_params)
50     img_transformed = cv2.warpPerspective(image, matrix, (width, height))
51     return matrix, img_transformed
52
53
54 def compute_point_perspective_transformation(matrix, list_downoids):
55     list_points_to_detect = np.float32(list_downoids).reshape(-1, 1, 2)
56     transformed_points = cv2.perspectiveTransform(list_points_to_detect, matrix)
57     transformed_points_list = list()

```

Figure 6. 7 Class variables

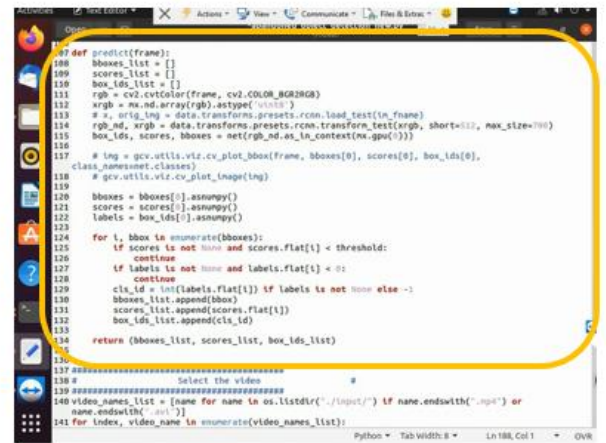
6.7 Decision Making Process (Abandoned / Non Abandoned)

Basically the object detection and the recognition has explained during the previous lectures. Once it comes to the decision making stage it is an important and critical position of the whole research concern. Because wrong decision cause big damages directly and indirectly in different kinds of environment. This research focused only on object detection and recognition and then it was going to decide whether particular object or the items is an abandoned or not. The research has mingled the MXnet with the GluonCV Toolkit to find the detected objects whether abandoned or not. If it is an abandoned then it was indicated in red color bounding box and when it is with the owner or having contact with the owner it indicates as non-abandoned indicated in Blue color. When the particular object become an abandoned like dis-attached from the owner then it indicated in Red color bounding box. The following figure 6.8 (a) & 6.8 (b) illustrated the particular source codes which used to do the correct decision making process of whether the detected or recognized object was an abandoned or non-abandoned.



```
77 return array_boxes
78
79
80 def get_centroids_and_groundpoints(array_boxes_detected):
81     array_centroids, array_groundpoints = [], []
82     for index, box in enumerate(array_boxes_detected):
83         centroid, ground_point = get_points_from_box(box)
84         array_centroids.append(centroid)
85         array_groundpoints.append(ground_point)
86     return array_centroids, array_groundpoints
87
88
89 def get_points_from_box(box):
90     center_x = int((box[0] + box[2]) / 2)
91     center_y = int((box[1] + box[3]) / 2)
92     center_x_ground = center_x + ((box[2] - box[0]) / 2)
93     return (center_x, center_y), (center_x, int(center_y_ground))
94
95
96 def draw_rectangle(corner_points, frame):
97     cv2.line(frame, (corner_points[0][0], corner_points[0][1]), (corner_points[1][0],
98     corner_points[1][1]), COLOR_BLUE,
99     thickness=1)
100     cv2.line(frame, (corner_points[1][0], corner_points[1][1]), (corner_points[2][0],
101     corner_points[2][1]), COLOR_BLUE,
102     thickness=1)
103     cv2.line(frame, (corner_points[2][0], corner_points[2][1]), (corner_points[3][0],
104     corner_points[3][1]), COLOR_BLUE,
105     thickness=1)
106
107 def predict(frame):
108     bboxes_list = []
109     scores_list = []
110     box_ids_list = []
111     rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
112     xrgb = mx.nd.array(rgb).astype('float32')
113     # x, orig_img = data.transforms.presets.rcnn.load_test(img=frame)
114     rgb_md, xrgb = data.transforms.presets.rcnn.transform_test(xrgb, short=112, max_size=100)
115     box_ids, scores, bboxes = net(rgb_md.as_inplace().get())
116
117     # img = gcv.utils.viz.cv_plot_bbox(frame, bboxes[0], scores[0], box_ids[0],
118     # gcv.utils.viz.cv_plot_image(img))
119
120     bboxes = bboxes[0].asnumpy()
121     scores = scores[0].asnumpy()
122     labels = box_ids[0].asnumpy()
123
124     for i, bbox in enumerate(bboxes):
125         if scores is not None and scores.flat[i] < threshold:
126             continue
127         if labels is not None and labels.flat[i] < 0:
128             continue
129         cls_id = int(labels.flat[i]) if labels is not None else -1
130         bboxes_list.append(bbox)
131         scores_list.append(scores.flat[i])
132         box_ids_list.append(cls_id)
133
134     return (bboxes_list, scores_list, box_ids_list)
```

(a)



```
107 def predict(frame):
108     bboxes_list = []
109     scores_list = []
110     box_ids_list = []
111     rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
112     xrgb = mx.nd.array(rgb).astype('float32')
113     # x, orig_img = data.transforms.presets.rcnn.load_test(img=frame)
114     rgb_md, xrgb = data.transforms.presets.rcnn.transform_test(xrgb, short=112, max_size=100)
115     box_ids, scores, bboxes = net(rgb_md.as_inplace().get())
116
117     # img = gcv.utils.viz.cv_plot_bbox(frame, bboxes[0], scores[0], box_ids[0],
118     # gcv.utils.viz.cv_plot_image(img))
119
120     bboxes = bboxes[0].asnumpy()
121     scores = scores[0].asnumpy()
122     labels = box_ids[0].asnumpy()
123
124     for i, bbox in enumerate(bboxes):
125         if scores is not None and scores.flat[i] < threshold:
126             continue
127         if labels is not None and labels.flat[i] < 0:
128             continue
129         cls_id = int(labels.flat[i]) if labels is not None else -1
130         bboxes_list.append(bbox)
131         scores_list.append(scores.flat[i])
132         box_ids_list.append(cls_id)
133
134     return (bboxes_list, scores_list, box_ids_list)
```

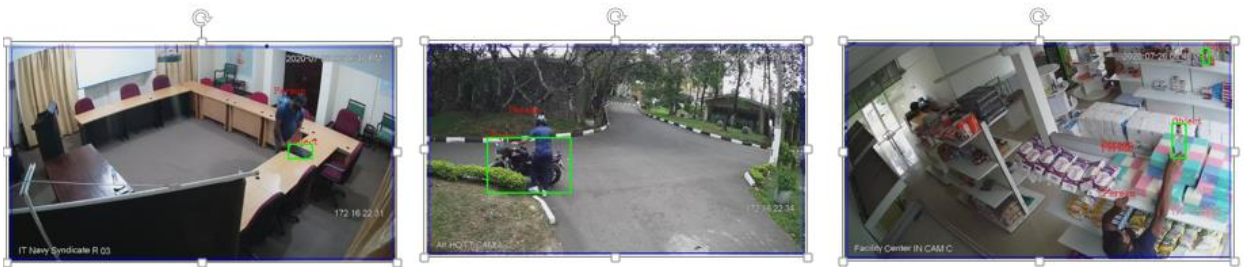
(b)

Figure 6. 8 Decision making source code of abandoned/non abandoned

Further, following Figure 6.9 (a) & (b) Abandoned and Non abandoned indicates the visual identification of a particular object become an abandoned or non abandoned once the above figure 6.8 source codes function correctly.



(a)



(b)

6.8 Summery

This Chapter explained in detail collaboration of step by step implementation from the high level architecture until the decision making process of particular detected object whether abandoned or non-abandoned. Further, time to time supported source codes were displayed for the easy understanding and do the proper investigation those were really helped. In next Chapter Seven (7) will illustrate the proper evaluation of the results of the previous implementation and the design.

Chapter 7

Evaluation

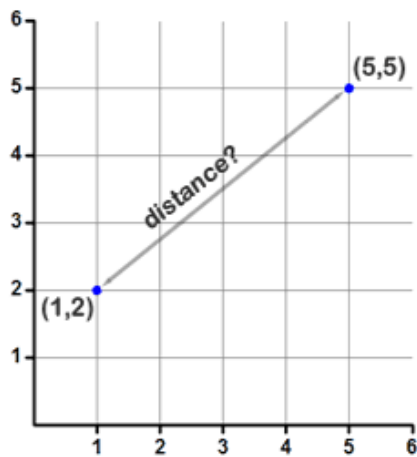
7.1 Introduction

The previous Chapter Six explained the proper implementation of the designed structure in practical and what were the expected outcomes. Further, realized about the practical issues when the implementation was carried out. This Chapter Seven (7) has gone to the evaluation of received out come as the results during the implementation of the research design. This may very much useful for the future researches and implementations to follow their studies.

7.2 Knowledge base distance calculation

The evaluation of this research is depend upon the outcome results. Those results used to do the comparison among the models. During the implementation whole process was ran on the MXnet platform and the detection and recognition was done by the GluonCV, then the OpenCV was used to do Image loading, pre-processing, Colour transformation, and bird view transformation. while, there were lot of other tools, methods, algorithms, calculations, were used to obtain the expected accurate results.

In this research there was an issue to calculate of the distance in two aspects one is distance between camera and the object, then second distance was to calculate the distance which unattached from the owner to identify whether particular object become an abandoned. The Triangular equation was used to calculate the distance for the first requirement under section of 6.4 para in implementation topic. The above second requirement was done by 'equiladian equation'. Actually three dimensional video will consider two dimensional plane then the calculation of the distance among the owner and object from the aforesaid equation. Following Figure 7.1 (a) (b) and (c) Equiladian distance calculation equation will illustrate the distance calculation as follows.



(a)

$$\text{distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

(b)

$$\text{distance} = \sqrt{(4)^2 + (3)^2}$$

$$\text{distance} = \sqrt{16 + 9}$$

$$\text{distance} = \sqrt{25}$$

$$\text{distance} = 5$$

(c)

Figure 7. 1 The Euclidean equation: Calculation of the distance in between two points

Basically, the whole program was written on Python 3.8 and it determines abandoned and non- abandoned dependency on distance calculation inbetween the owner and the object of particular video. Source code define to decide the said abandancy if the distance (in between object and the owner)is more than 400 pixels it detect and recognize as the abandoned (if the object is an abandoned, the bounding box indicates in Red colour) or if not it is non-abandoned (if the object is non-abandoned, the bounding box indicates in Green colour) following figure 7.2 the pixel distance limitation for the abandancy will illustrates the process of how to determine the distance calculation to decide the abandancy.

```

29
30     if threshold:
31         if y <= threshold * 1.01:
32             break
33         else:
34             print("threshold value is out of range!")
35
36 # get neural net name from user
37 net_names = ['faster_rcnn_fpn_symbcnn_resnet101_coco', 'yolo3_darknet53_coco', 'ssd_300_resnet101_v2_coco']
38 print('available models are:')
39 for index, model_name in enumerate(net_names):
40     print(' {} {}'.format(model_name, index))
41
42 net_number = None
43 while True:
44     try:
45         net_number = int(input("Enter neural network (net) number: "))
46     except ValueError:
47         print("Invalid input!")
48
49     if 0 <= net_number < len(net_names):
50         break
51     else:
52         print("neural network number is out of range!")
53
54 net_name = net_names[net_number]
55 net = model_zoo.get_model(net_name, pretrained=True, ctx=mx.gpu(0))
56 # net = model_zoo.get_model('faster_rcnn_fpn_symbcnn_resnet101_coco', pretrained=True, ctx=mx.gpu(0))
57 # net = model_zoo.get_model('yolo3_darknet53_coco', pretrained=True, ctx=mx.gpu(0))
58 # net = model_zoo.get_model('yolo3_darknet53_coco', pretrained=True, ctx=mx.gpu(0))
59 object_ids = []
60 # threshold = 0.4
61 size_frame = "720"
62 distance_minimun = "400"
63
64
65 #####
66 # Class Id Mapping
67 #####
68 # https://github.com/opencv/opencv/blob/master/modules/dnn/include/dnn.hpp
69 class_name_mapping = {0: 'w/ background', 1: 'w/ person', 2: 'w/ bicycle', 3: 'w/ car', 4: 'w/ motorcycle', 5: 'w/ airplane', 6: 'w/ bus', 7: 'w/ train', 8: 'w/ truck', 9: 'w/ boat', 10: 'w/ traffic light',
70 11: 'w/ fire hydrant', 12: 'w/ stop sign', 13: 'w/ parking meter', 14: 'w/ bench', 15: 'w/ stop sign', 16: 'w/ car', 17: 'w/ dog', 18: 'w/ horse', 19: 'w/ sheep', 20: 'w/ cow',
71 21: 'w/ elephant', 22: 'w/ bear', 23: 'w/ zebra', 24: 'w/ giraffe', 25: 'w/ backpack', 26: 'w/ umbrella', 27: 'w/ handbag', 28: 'w/ tie', 29: 'w/ suitcase', 30: 'w/ trailer',
72 31: 'w/ suitcase', 32: 'w/ handbag', 33: 'w/ sports ball', 34: 'w/ ball', 35: 'w/ baseball bat', 36: 'w/ baseball glove', 37: 'w/ baseball', 38: 'w/ tennis racket',
73 39: 'w/ bottle', 40: 'w/ wine glass', 41: 'w/ cup', 42: 'w/ fork', 43: 'w/ knife', 44: 'w/ spoon', 45: 'w/ bowl', 46: 'w/ banana', 47: 'w/ apple', 48: 'w/ sandwich',
74 49: 'w/ orange', 50: 'w/ broccoli', 51: 'w/ carrot', 52: 'w/ hot dog', 53: 'w/ pizza', 54: 'w/ pizza', 55: 'w/ donut', 56: 'w/ cake', 57: 'w/ chair', 58: 'w/ couch', 59: 'w/otted glass',
75 60: 'w/ bed', 61: 'w/ dining table', 62: 'w/ toilet', 63: 'w/ tv', 64: 'w/ laptop', 65: 'w/ mouse', 66: 'w/ remote', 67: 'w/ keyboard', 68: 'w/ cell phone', 69: 'w/ microphone',
76 70: 'w/ speaker', 71: 'w/ toaster', 72: 'w/ sink', 73: 'w/ refrigerator', 74: 'w/ oven', 75: 'w/ microwave', 76: 'w/ toaster', 77: 'w/ toaster', 78: 'w/ teddy bear', 79: 'w/ hair drier', 80: 'w/ toothbrush'}
77
78
79 def compute_perspective_transform(corner_points, width, height, image):
80     corner_points_array = np.float32(corner_points)
81     img_params = np.float32([[0, 0], [width, 0], [0, height], [width, height]])
82     matrix = cv2.getPerspectiveTransform(corner_points_array, img_params)
83     img_transformed = cv2.warpPerspective(image, matrix, (width, height))

```

Figure 7. 2 the pixel distance limitation

7.3 The steps which occurred while input video clips going through the model

This research used three input video clips for the research purpose. And each video clip's threshold value was kept static or unique for the three models (Faster RCNN, YOLO-V3 & SSD) during the testing. Like that there are three different threshold values used for the three video inputs to obtain maximum results. Initially, under the preparation stage first each input video clip had arranged the important video area so that model to acquire the required are of detect only to minimize the time waste while producing the real-time results as well as real time detections.

For the easy understanding of the object detection process of the videos, it explains as follows;

- (a) First those each input video while testing occurs model divide whole video clip in to one second time slots.
- (b) Then during the one second time period input video will cut in to five photo frames.

- (c) Mean time model use to send same video slot through the algorithm to mark the object whether abandoned or non-abandoned while it is creating same one second video clip freeze in to five photo frames with colored (Red color for the abandoned object and Green color for the non-abandoned object) bounding boxes.
- (d) Finally, python source code creates the folders for whole video input consisting with result and ground truth folders in each one second photo frame folder. Figure 7.3 indicates the ‘ground truth’ folder and the produced ‘result’ folders separately.

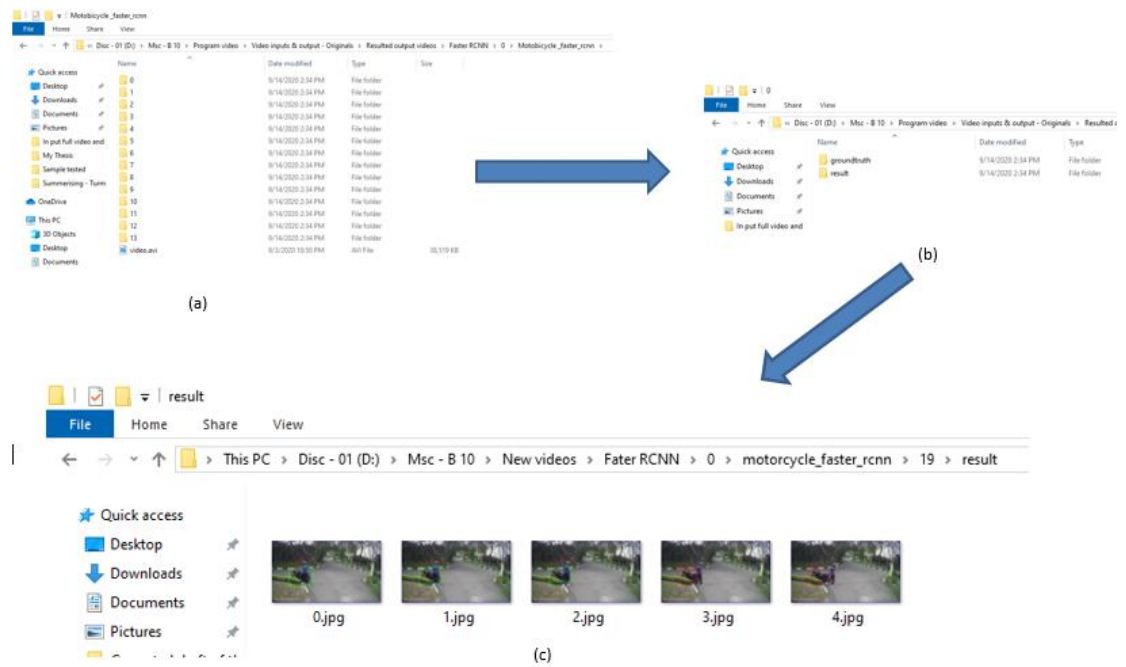


Figure 7. 3 (a), (b) & (c) input and resultant output video clips

Once the each three video input testing through the three models eventually obtained the output results according to the input videos. Those out puts obtain as produced video and its photo frame folders. This will create easy environment to check, decide and compare the ground truth and result output video more accurately. Then the system of deciding the

binary values for the inputs (ground truth binary value) and outputs (result binary value) indicates as follows;

- (a) If input/output video indicate 3 out of 5 same indication (if selected objects' bounding box indicates green colour majority then it consider as non-abandoned and if it gets Red then it consider as the abandoned) then that would be taken as a decision of particular one second time.
- (b) When the decision taken with regards to above (a), binary value = 1 (one) assign to indicate the object is an 'abandoned' value = 0 (zero) assign to indicate non abandoned.

7.4 Steps taken to measure the output results

The research has planned to use by three separate video inputs. The threshold value of each are different but that value were stable for three models for testing. The threshold values of each video input is as follows;

- (a) Bike video clip - threshold value is 0.004 for all three inputs.
- (b) Bottle video clip – threshold value is 0.06 for all three inputs.
- (c) Laptop video clip- threshold value is 0.39 for all three inputs.

Further, these kind of knowledge base activity need to be practiced very carefully. The procedure of deciding the values 1 or 0 to given input/ output have explained under the above para 7.3 First, one input video clip used to check through the three models separately then marked the input/output values accordingly as shown in the figure 7.4 of patterns of marking input/output values.

Ser No	Motor bike – Threshold value =(0.004)				Bottle – Threshold value = (0.06)				Laptop – Threshold value = 0.39			
	GT	Faster RCNN	YOLO	SSD	GT	Faster RCNN	YOLO	SSD	GT	Faster RCNN	YOLO	SSD
0	No	No	No	No	0	1	0	1	0	0	0	0
1	No	No	No	No	0	1	0	0	0	0	0	0
2	No	No	No	No	0	0	0	1	0	0	0	0
3	No	No	No	No	0	0	0	0	0	0	0	0
4	No	No	No	No	0	0	0	0	No	No	No	No
5	0	0	0	0	0	0	0	0	No	No	No	No
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0
8	0	1	0	0	1	0	0	0	0	1	0	0
9	0	0	0	0	1	0	0	0	0	1	0	0
10	0	0	0	0	1	1	1	1	0	0	0	0
11	0	0	0	0	1	1	1	1	0	0	0	0
12	0	0	0	0	1	1	1	1	0	0	0	0
13	0	0	0	0	1	1	1	1	0	0	0	0
14	0	0	0	0	1	1	1	1	0	0	0	0
15	0	0	0	0	1	1	1	1	0	1	0	0
16	0	0	0	0	1	1	1	1	0	1	0	0
17	0	0	0	0	1	1	1	1	0	0	0	1
18	0	0	0	0					0	0	1	1
19	1	0	0	0					1	1	1	1
20	1	1	0	1					1	1	1	1
21	1	1	1	1					1	1	1	1
22	1	1	1	1					1	1	1	1
23	1	1	1	1					1	1	1	1
24	1	1	1	1					1	1	1	1

- * No – uncountable raw values (required to remove when the filtering step)

Figure 7. 4 results gathering by manual data system

7.5 How to measure the Accuracy of the models

The machine learning use to measure the accuracy of binary values. This is call as binary classification matrix. The matrix call as ‘Confusion matrix’. There are basically four terms use in the Confusion matrix. True positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN). It can be demonstrate by figure 7.5 as follows;

	<i>Class 1 Predicted</i>	<i>Class 2 Predicted</i>
Class 1 Actual	TP	FN
Class 2 Actual	FP	TN

Figure 7. 5 Confusion Matrix table

The each model needs to test three input videos and needs to calculate the accuracy model for each video respectively. The following equation may use to calculate the accuracy of the model for particular input video.

$$\text{The Accuracy} = (TP + TN) / (TP+FP+FN+TN)$$

- TP: True Positive
- TN: True Negative
- FP: False Positive
- FN: False Negative

7.5.1 Step one

Insert binary raw values to the table as shown in the Figure 7.4 above for each input video against the ground truth value.

7.5.2 Step two

Filter the required range of input and output values. Remove of counting non response 'No' values cells by counting as inputs or outputs. Therefore, remove those values from whole dataset. Figure 7.6 shows the selecting of proper clear data set.

Serial No	Motor bike – Threshold value =(0.004)				Bottle – Threshold value = (0.06)				Laptop – Threshold value = 0.39			
	GT	Faster RCN	YOLO	SSD	GT	Faster RCN	YOLO	SSD	GT	Faster RCN	YOLO	SSD
0	0	0	0	0	0	1	0	1	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0	1	0	0
3	0	1	0	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0	0	0	1	0	0
10	0	0	0	0	1	1	1	1	0	1	0	0
11	0	0	0	0	1	1	1	1	0	0	0	1
12	0	0	0	0	1	1	1	1	0	0	1	1
13	0	0	0	0	1	1	1	1	1	1	1	1
14	1	0	0	0	1	1	1	1	1	1	1	1

15	1	1	0	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	1	1	1	1
18	1	1	1	1					1	1	1	1
19	1	1	1	1								

Figure 7. 6 the filtering of clear data set. Remove the unwanted 'No 'data values.

7.5.3 Step three

Use the confuse matrix to calculate the accuracy percentage for each model with respect to the input video clip. This values provides the quantitative measurement of the accuracy of each model as shown in above Figure 7.5 shows the clear data set to be used to calculate the accuracy.

7.5.4 Step Four

The calculation and results are as follows;

For **motorbike video input** – Table (a)

Faster RCNN,

Total readings 20 nos, TP=5, TN=12, FP=1 & FN=2

(Accuracy)- Faster RCNN = $[(TP + TN) / (TP+FP+FN+TN)] \times 100\%$

$$= [(5+12) / (5+1+1+13)] \times 100\%$$

$$= 85\%$$

Precision = $[TP / (TP+FP)] \times 100\% = [5 / (5+1)] \times 100\% = 83.33\%$

Recall = $[(TP) / (TP+FN)] \times 100\% = [5 / (5+2)] \times 100\% = 71.40\%$

The results of each video input against the model has shown in following figure 7.6 of calculated results of models.

Ser No	Video input Description	MODELS								
		Faster RCNN			YOLO – V3			SSD		
		P(%)	R(%)	A(%)	P(%)	R(%)	A(%)	P(%)	R(%)	A(%)
01	Motor bike	83.3	71.4	85	100	71.4	90	83.3	71.4	85
02	Bottle	61.5	80	61.1	100	80	88.8	72.7	80	72.2
03	Laptop	60	100	78.9	85.7	100	97.7	75	100	89.5

P – Precision R – Recall A - Accuracy

Figure 7. 7 the calculated results of the three models

According to the above figure 7.6 results chart of the models it indicates that the accuracy level changed not only the video input but with other number of reasons. According to the above readings accuracy comparison as follows;

- (a) Motor bike input – The best accuracy was performed by YOLO – V3(90%).
- (b) Bottle input – The best accuracy was performed by YOLO-V3(88.8%).
- (c) Laptop input – The best accuracy was performed by YOLO – V3(97.7%).

7.6 Summery

This chapter was examining of the best accurate model comparison for object detection and recognition. Above Figure 7.8 shown the final results of the models. It confirms that there are N number of reasons are affecting to the accuracy level of particular model. Therefore, the results will vary one to another in different video input even for same model. The nest chapter will do the conclusion and future works to be implemented for better function in future use.

Chapter 8

Conclusion and Future Works

8.1 Introduction

This research was mainly focusing on the modeling on abandoned object detection and recognition in real time surveillance. Actually, the background details were enriched the requirement of having such a effective model to in realtime surveillance rather than the manual monitoring methods. Then literature proved further what are the loop hols to be filled to fulfill the exsisting requirement. The use of specific technologies and their capabilities were discussed and finalized in order to beging with a proper approach to obtain the desired goals. Most of the time it is most important to have at least high level design or an architecture to go through till reach the end result. Therefore having a proper research design was really important and it is must. Though the design are drawn but must be implement in real world that was the biggest challenge to overcome. Each step of the design process needed to be expalained how do the research run through the each step and what are the difficulties and achievements and those out results were immensely used to obtain the next step success.

The implemented system needs to be checked whether it is correct or accurate then how far it is successful and what are the observed out results as failers. These were done during the process of the evalusation step. Actually from the beginning until the evaluation it was a continuos flow or determining what is the best performing model for the abandoned object detection and recognition during the real time surveillance environment this was deeply accurately calculated by the use of confution matrix algorithm among most popular three models. Among them statistically could be able to find the most accurate model for real time surveillance. During this chapter may elobarate the conclusion and future works to be done and what were the limitations had during the whole process. Finally, it noted that what are the future works to be done to uplift the more adaptobale, accurate and speed modeings to use in real time surveillance platform.

8.2 Limitations

The object detection and recognition real time is very deep machine learning concept. It is very new direction of study and very interested. There are many models, frameworks, libraries, algorithms... etc have been used and tested to obtain good output. Due to that reason lots of researches have done and still continuing their experiments to find the best system. because those results were specific for specific environment. This was a very big limitation. Because even capability of the cameras, their angles, different lighting effects ...etc were the barriers or the limitations. Secondly machine learning, and neural network fields associate with high end equipments such as VGA, OSs, RAMs, and PCs. therefore doing experiments or the researches on this field need to pay more financial weight rather than other IT related researches. Therefore high performing labs needed to be used for the experiments. This type of newly generated field of study it very must good and important it causes to improve the knowledge on particular subject or the field and it opens the doors to improve the skills. Therefore, lack of skill on several fields such as programming, neural networking, algorithms, frameworks, datasets, libraries, toolkits ...etc may cause huge advantage. But if it is low it would be a great limitation to carry on with the works.

8.3 Future works

This research evaluation was done by the use of basic mathematical equation call confusion matrix to find the accuracy of each model. again once taking ground truth data it was taken one second of video freeze in to five photo frames and decided the abandonment by naked eye. This needed to be improved in order to obtain most accurate ground truth input data.

Simply the research was focused on the determining of the accuracy of the three models. then identify the best performing model after the evaluation of the models. The finding of the speed among the models also very much important like finding the accuracy of the model. Therefore the most speedest and maximum accurate model among them in real-time surveillance also most important due to operations are running on real-time. Therefore, finding of most speedest model as such that needs to be done as the future

works. The testing of different models in different frameworks , and use of datasets,libraries, toolkits and algorithms will help to improve the further knowledge on the subject and related tools, algorithms to find better models for real-time surveillances. Further, deep study on background and foreground models and creating frams and bounding boxes and their definings needed to be done more precisely. Then the improvments of the out put results like accuracy would be able to enriched.

8.4 Summery

This chapter discussed the limitations had during the research and the barriers against the implementation. What are the points that had been concerned during the whole process. Then explained the future works to be done for the best output results and study on further deep through the subject and find new valuable findings for the future generation to implement and make interest of future generation to creat inventions to improve or uplift the present standards.

References

- [1]. Liao, W., et al., *Security event recognition for visual surveillance*. 2017. **4**(1W1): p. 19-26.
- [2]. Girshick, R., et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
- [3]. Girshick, R., et al., *Region-based convolutional networks for accurate object detection and segmentation*. 2015. **38**(1): p. 142-158.
- [4]. Krizhevsky, A., I. Sutskever, and G.E. Hinton. *Imagenet classification with deep convolutional neural networks*. in *Advances in neural information processing systems*. 2012.
- [5]. Shelhamer, E., et al., *Fully convolutional networks for semantic segmentation*. 2017. **39**(4): p. 640-651.
- [6]. Toshev, A. and C. Szegedy. *Deeppose: Human pose estimation via deep neural networks*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
- [7]. Stauffer, C. and W.E.L. Grimson. *Adaptive background mixture models for real-time tracking*. in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*. 1999. IEEE.
- [8]. Lin, K., et al., *Abandoned object detection via temporal consistency modeling and back-tracing verification for visual surveillance*. *IEEE Transactions on Information Forensics and Security*, 2015. **10**(7): p. 1359-1370.
- [9]. Rohith Sri Sai, M., S. Rella, and S. Veeravalli, *OBJECT DETECTION AND IDENTIFICATION A Project Report*. 2019.
- [10]. Tripathi, R.K., A.S. Jalal, and C. Bhatnagar. *A framework for abandoned object detection from video surveillance*. in *2013 Fourth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*. 2013. IEEE.
- [11]. Lin, K., et al., *Abandoned object detection via temporal consistency modeling and back-tracing verification for visual surveillance*. 2015. **10**(7): p. 1359-1370.
- [12]. Fan, Q. and S. Pankanti. *Modeling of temporarily static objects for robust abandoned object detection in urban surveillance*. in *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2011. IEEE.
- [13]. Zeng, Y., et al., *A novel abandoned object detection system based on three-dimensional image information*. *Sensors*, 2015. **15**(3): p. 6885-6904.
- [14]. Fan, Q. and S. Pankanti. *Robust foreground and abandonment analysis for large-scale abandoned object detection in complex surveillance videos*. in *2012 IEEE Ninth*

International Conference on Advanced Video and Signal-Based Surveillance. 2012. IEEE.

- [15]. Fu, H., et al. *Abandoned object detection in highway scene*. in *2011 6th International Conference on Pervasive Computing and Applications*. 2011. IEEE.
- [16]. Park, H., S. Park, and Y. Joo, *Robust detection of abandoned object for smart video surveillance in illumination changes*. *Sensors*, 2019. **19**(23): p. 5114.
- [17]. Alganci, U., M. Soydas, and E.J.R.S. Sertel, *Comparative research on deep learning approaches for airplane detection from very high-resolution satellite images*. 2020. **12**(3): p. 458.
- [18]. Porikli, F., Y. Ivanov, and T.J.E.J.o.A.i.S.P. Haga, *Robust abandoned object detection using dual foregrounds*. 2007. **2008**(1): p. 197875.
- [19]. Gupta, A., et al. *Real-Time Abandoned Object Detection Using Video Surveillance*. in *Proceedings of the International Conference on Recent Cognizance in Wireless Communication & Image Processing*. 2016. Springer.
- [20]. Gonzalez, R.C., R.E. Woods, and S.L. Eddins, *Digital image processing using MATLAB*. 2004: Pearson Education India.
- [21]. Filonenko, A. and K.-H.J.I.T.o.I.I. Jo, *Unattended object identification for intelligent surveillance systems using sequence of dual background difference*. 2016. **12**(6): p. 2247-2255.
- [22]. Luna, E., et al., *Abandoned Object Detection in Video-Surveillance: Survey and Comparison*. *Sensors (Basel, Switzerland)*, 2018. **18**(12): p. 4290.
- [23]. Hansen, L.K., P.J.I.t.o.p.a. Salamon, and m. intelligence, *Neural network ensembles*. 1990. **12**(10): p. 993-1001.

Appendix – A

Source code 1-

Source code of Python 3.8 for abandoned object detection and recognition.

```

1 import cv2
2 import math
3 import os
4 from glob import glob
5 import model_zoo as mz
6 import numpy as np
7 import cv2
8
9 COLOR_RED = (0, 0, 255)
10 COLOR_GREEN = (0, 255, 0)
11 COLOR_BLUE = (255, 0, 0)
12 RGB_WHITE = (255, 255, 255)
13 DNN_CIRCLE = 3
14
15 #####
16 # Load models and configs
17 #####
18 # get threshold value from the user
19 threshold = None
20 while True:
21     try:
22         threshold = input("Enter threshold value: ")
23         if threshold == "":
24             threshold = 0.4
25         break
26         threshold = float(threshold)
27     except ValueError:
28         print("Invalid input!")
29
30 if threshold:
31     if 0 <= threshold <= 1.0:
32         break
33     else:
34         print("Threshold value is out of range!")
35
36 # get neural net name from user
37 net_names = ['faster_rcnn_fpn_synchr_resnet101_coco', 'yolo3_darknet53_coco', 'ssd_312_resnet50_v1_coco']
38 print("Available models are:")
39 for index, model_name in enumerate(net_names):
40     print("{} [{}].format(model_name, index)
41
42 net_number = None
43 while True:
44     try:
45         net_number = int(input("Enter neural network (tool) number: "))
46     except ValueError:
47         print("Invalid input!")
48     break
49 if 0 <= net_number <= len(net_names):
50     break
51 else:
52     print("Neural network number is out of range!")
53
54 net_name = net_names[net_number]
55 net = model_zoo.get_model(net_name, pretrained=True, ctx=mx.gpu(0))

```

Appendix A: 1 Source code of Python 3.8. – Source code 1

Source code 2-

Source code of Python 3.8 for abandoned object detection and recognition.

```

56 # net = model_zoo.get_model('faster_rcnn_fpn_synchr_resnet101_coco', pretrained=True, ctx=mx.gpu(0))
57 # net = model_zoo.get_model('yolo3_darknet53_coco', pretrained=True, ctx=mx.gpu(0))
58 # net = model_zoo.get_model('ssd_312_resnet50_v1_coco', pretrained=True, ctx=mx.gpu(0))
59 object_ids = []
60
61 threshold = 0.4
62 size_frame = 700
63 distance_min_max = "500"
64
65 #####
66 # class id mapping
67 #####
68 # https://gist.github.com/ronintec/7b3d0d9d0d6c08f885570b5d0b7bde
69 class_name_mapping = ['0' u'background', 12 u'person', 13 u'bicycle', 14 u'car', 15 u'motorcycle', 16 u'airplane', 17 u'bus', 18 u'truck', 19 u'boat', 20 u'traffic light',
70 21 u'fire hydrant', 22 u'stop sign', 23 u'parking meter', 24 u'bench', 25 u'bird', 26 u'cat', 27 u'dog', 28 u'horse', 29 u'sheep', 30 u'cow',
71 31 u'sheep', 32 u'sheep', 33 u'cow', 34 u'giraffe', 35 u'backpack', 36 u'suitcase', 37 u'handbag', 38 u'bag', 39 u'frisbee',
72 40 u'kite', 41 u'baseball bat', 42 u'baseball glove', 43 u'skateboard', 44 u'surfboard', 45 u'tennis racket',
73 46 u'bottle', 47 u'wine glass', 48 u'cup', 49 u'fruit', 50 u'banana', 51 u'apple', 52 u'sandwich',
74 53 u'orange', 54 u'broccoli', 55 u'carrot', 56 u'hot dog', 57 u'pizza', 58 u'donut', 59 u'cake', 60 u'chair', 61 u'couch', 62 u'potted plant',
75 63 u'bed', 64 u'dining table', 65 u'table', 66 u'tv', 67 u'laptop', 68 u'mouse', 69 u'remote', 70 u'keyboard', 71 u'cell phone', 72 u'microwave',
76 73 u'oven', 74 u'toaster', 75 u'knife', 76 u'fork', 77 u'spoon', 78 u'teaspoon', 79 u'cupboard', 80 u'bench', 81 u'fire hydrant', 82 u'traffic light', 83 u'toothbrush']
77
78
79 def compute_perspective_transform(corner_points, width, height, image):
80     corner_points_array = np.float32(corner_points)
81     img_params = np.float32([width, height], [width, height])
82     matrix = cv2.getPerspectiveTransform(corner_points_array, img_params)
83     img_transformed = cv2.warpPerspective(image, matrix, (width, height))
84     return matrix, img_transformed
85
86
87 def compute_point_perspective_transformation(matrix, list_downloads):
88     list_points_to_detect = np.float32(list_downloads).reshape((-1, 2))
89     transformed_points = cv2.perspectiveTransform(list_points_to_detect, matrix)
90     transformed_points_list = list()
91     for i in range(transformed_points.shape[0]):
92         transformed_points_list.append((transformed_points[i][0], transformed_points[i][1]))
93     return transformed_points_list
94
95
96 def get_human_box_detection(boxes, scores, classes):
97     array_boxes = list()
98     for i in range(len(boxes)):
99         if int(classes[i]) == 0 and scores[i] > threshold:
100             array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]), int(boxes[i][3])))
101     return array_boxes
102
103
104 def get_object_box_detection(boxes, scores, classes, height, width, class_ids):
105     array_boxes = list()
106     for i in range(len(boxes)):
107         if scores[i] > threshold:
108             if (int(classes[i]) == 0) in class_ids:
109                 array_boxes.append((int(boxes[i][0]), int(boxes[i][1]), int(boxes[i][2]), int(boxes[i][3])))
110     return array_boxes

```

Appendix A: 2 Source code of Python 3.8. – Source code 2

Source code 3-

Source code of Python 3.8 for abandoned object detection and recognition.

```
111
112
113 def get_centroids_and_groundpoints(array_boxes_detected):
114     array_centroids, array_groundpoints = [], []
115     for index, box in enumerate(array_boxes_detected):
116         centroid, ground_point = get_points_from_box(box)
117         array_centroids.append(centroid)
118         array_groundpoints.append(centroid)
119     return array_centroids, array_groundpoints
120
121
122 def get_points_from_box(box):
123     center_x = int(((box[1] + box[3]) / 2))
124     center_y = int(((box[0] + box[2]) / 2))
125     center_y_ground = center_y + ((box[1] - box[0]) / 2)
126     return (center_x, center_y), (center_x, int(center_y_ground))
127
128
129 def draw_rectangle(corner_points, frame):
130     cv2.line(frame, (corner_points[0][0], corner_points[0][1]), (corner_points[1][0], corner_points[1][1]), COLOR_BLUE,
131              thickness=1)
132     cv2.line(frame, (corner_points[1][0], corner_points[1][1]), (corner_points[2][0], corner_points[2][1]), COLOR_BLUE,
133              thickness=1)
134     cv2.line(frame, (corner_points[2][0], corner_points[2][1]), (corner_points[3][0], corner_points[3][1]), COLOR_BLUE,
135              thickness=1)
136     cv2.line(frame, (corner_points[3][0], corner_points[3][1]), (corner_points[0][0], corner_points[0][1]), COLOR_BLUE,
137              thickness=1)
138
139
140 def predict(frame):
141     bboxes_list = []
142     scores_list = []
143     box_ids_list = []
144     rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
145     xrgb = np.ndarray(rgb).astype('uint8')
146     # x, orig_img = data.transforms.presets.rcnn.load_test((in_fname))
147     rgb_md, xrgb = data.transforms.presets.rcnn.transform_test(xrgb, short=10, max_size=700)
148     box_ids, scores, bboxes = net(rgb_md.as_inplace('cpu'))
149
150     # img = cv2.cvtColor(cv2.cvtColor(frame, bboxes[0], scores[0], box_ids[0], class_names=net.classes))
151     # cv2.imshow('img', img)
152
153     bboxes = bboxes[0].asnumpy()
154     scores = scores[0].asnumpy()
155     labels = box_ids[0].asnumpy()
156
157     for i, bbox in enumerate(bboxes):
158         if scores is not None and scores.flat[i] < threshold:
159             continue
160         if labels is not None and labels.flat[i] < 0:
161             continue
162         cls_id = int(labels.flat[i]) if labels is not None else -1
163         bboxes_list.append(bbox)
164         scores_list.append(scores.flat[i])
165         box_ids_list.append(cls_id)
166
```

Appendix A: 3 Source code of Python 3.8. – Source code 3

Source code 4-

Source code of Python 3.8 for abandoned object detection and recognition.

```
165     box_ids_list.append(cls_id)
166     return (bboxes_list, scores_list, box_ids_list)
167
168
169
170 ##### Select the video #####
171 #
172 #####
173 video_names_list = [name for name in os.listdir('./input/') if name.endswith('.mp4') or name.endswith('.avi')]
174 for index, video_name in enumerate(video_names_list):
175     print(' %d %s' % (index, video_name))
176 video_num = input("Enter the exact name of the video (including .mp4 or else) : ")
177 if video_num == "":
178     video_path = './input/PETS2009.avi'
179 else:
180     video_path = './input/' + video_names_list[int(video_num)]
181
182 #####
183 # Initialize video capture and corner #
184 #####
185 vs = cv2.VideoCapture(video_path)
186 (frame_exists, frame) = vs.read()
187 frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
188
189 pad = 5
190 corner_points = []
191 height_og, width_og, _ = frame.shape
192 corner_points.append([pad, height_og - pad])
193 corner_points.append([pad, pad])
194 corner_points.append([width_og - pad, height_og - pad])
195 corner_points.append([width_og - pad, pad])
196
197 ##### Compute transformation matrix #####
198 #
199 matrix, imgOutput = compute_perspective_transform(corner_points, width_og, height_og, frame)
200 height, width, _ = imgOutput.shape
201 blank_image = np.zeros((height, width, 3), np.uint8)
202 height = blank_image.shape[0]
203 width = blank_image.shape[1]
204 dstIn = (width, height)
205
206 #####
207 ##### START THE VIDEO STREAM #####
208 #####
209 #####
210 #####
211 #####
212 output_video = None
213 # counter = 0
214 counter = 0
215 while True:
216     (frame_exists, frame) = vs.read()
217     if not frame_exists:
218         break
219     else:

```

Appendix A: 4 Source code of Python 3.8. – Source code 4

Source code 5-

Source code of Python 3.8 for abandoned object detection and recognition.

```
217 if not frame_exists:
218     break
219 else:
220     frame = imutils.resize(frame, width=int(size_frame))
221     # clone or original frame to keep it separate
222     frame0 = frame.copy()
223     (boxes, scores, classes) = predict(frame)
224     array_boxes_detected = get_human_box_detection(boxes, scores, classes)
225     array_boxes_detected_object = get_object_box_detection(boxes, scores, classes, frame.shape[0],
226                                                         frame.shape[1], object_ids)
227     if len(array_boxes_detected_object) > 0:
228         array_centroids_objects, array_groundpoints_objects = get_centroids_and_groundpoints(
229             array_boxes_detected_object)
230         transformed_downloads_objects = compute_point_perspective_transformation(matrix,
231                                         array_groundpoints_objects)
232         if len(transformed_downloads_objects) > 0:
233             for index, download in enumerate(transformed_downloads_objects):
234                 xmin, ymin, xmax, ymax = [int(x) for x in array_boxes_detected_object[index]]
235                 cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), COLOR_RED,
236                               2)
237                 cv2.putText(frame, 'Object ',
238                             (xmin, ymin),
239                             cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 255), 1)
240                 if len(array_boxes_detected) > 0:
241                     array_centroids, array_groundpoints = get_centroids_and_groundpoints(array_boxes_detected)
242                     transformed_downloads = compute_point_perspective_transformation(matrix, array_groundpoints)
243                     for index_h, download_h in enumerate(transformed_downloads):
244                         if math.sqrt((download_h[0] - download_h[1])**2 + (download_h[1] - download_h[2])**2) < int(
245                             distance_minimum):
246                             xmin, ymin, xmax, ymax = [int(x) for x in array_boxes_detected_object[index]]
247                             cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), COLOR_GREEN,
248                                           2)
249                             cv2.putText(frame, 'Person ', (xmin_o, ymin_o),
250                                         cv2.FONT_HERSHEY_COMPLEX, 0.5, (0, 0, 255), 1)
251                             draw_rectangle(corner_points, frame)
252                             cv2.imshow('101168 picture', frame)
253                             key = cv2.waitKey() & 0xFF
254                             if output_video_1 is not None:
255                                 fourcc1 = cv2.VideoWriter_fourcc(*"H264")
256                                 if net_number == 0:
257                                     net_name_s = "faster_rcnn"
258                                 elif net_number == 1:
259                                     net_name_s = "yolo3"
260                                 elif net_number == 2:
261                                     net_name_s = "ssd"
262                                 output_folder_path = "%s/output/%s_%s" % (os.path.splitext(video_names_list[int(video_num)])[0], net_name_s)
263
264
265
266
267
268
269
270
271
```

Appendix A: 5 Source code of Python 3.8. – Source code 5

Source code 6-

Source code of Python 3.8 for abandoned object detection and recognition.

```
270 output_folder_path = "%s/output/%s_%s" % (os.path.splitext(video_names_list[int(video_num)])[0], net_name_s)
271
272 try:
273     os.mkdir(output_folder_path)
274 except FileExistsError:
275     pass
276
277 # folder to save images
278 img_folder_path = os.path.join(output_folder_path, str(counter // 10))
279 try:
280     os.mkdir(img_folder_path)
281 except FileExistsError:
282     pass
283
284 try:
285     os.mkdir(os.path.join(img_folder_path, "groundtruth"))
286 except FileExistsError:
287     pass
288 try:
289     os.mkdir(os.path.join(img_folder_path, "result"))
290 except FileExistsError:
291     pass
292
293 file_name = "%s.jpg" % ((counter // 10) // 0)
294 cv2.imwrite(os.path.join(os.path.join(img_folder_path, "groundtruth"), file_name), frame0)
295 cv2.imwrite(os.path.join(os.path.join(img_folder_path, "result"), file_name), frame)
296
297 output_video_1 = cv2.VideoWriter(os.path.join(output_folder_path, "video.avi"), fourcc1, 25, (frame.shape[1], frame.shape[0]), True)
298 elif output_video_1 is not None:
299     output_video_1.write(frame)
300     if counter % 6 == 0:
301         if counter // 10 == 0:
302             # folder to save images
303             img_folder_path = os.path.join(output_folder_path, str(counter // 10))
304             try:
305                 os.mkdir(img_folder_path)
306             except FileExistsError:
307                 pass
308             try:
309                 os.mkdir(os.path.join(img_folder_path, "groundtruth"))
310             except FileExistsError:
311                 pass
312             try:
313                 os.mkdir(os.path.join(img_folder_path, "result"))
314             except FileExistsError:
315                 pass
316             file_name = "%s.jpg" % ((counter // 10) // 0)
317             cv2.imwrite(os.path.join(os.path.join(img_folder_path, "groundtruth"), file_name), frame0)
318             cv2.imwrite(os.path.join(os.path.join(img_folder_path, "result"), file_name), frame)
319         if key == ord('q'):
320             break
321         # increment counter
322         counter += 1
323
```

Appendix A: 6 Source code of Python 3.8. – Source code 6