

# Deep Learning and Computer-Vision Based Algorithm for Screw Detection, Localization and Tracking in the Recycling Process.



&



Adedamola Ayodeji Sode

Department of Computer Science, Bioengineering, Robotics and Systems Engineering, University of  
Genoa

and

Control and Robotics Department, Advanced Robotics, Ecole Centrale de Nantes.

Supervisors: Professor Fulvio Mastrogiovanni, Dr. Alessandro Carfi, Mr. Michele Olivieri.

In partial fulfillment of the requirements for the degrees of

*Master of Science in Control and Robotics: Advanced Robotics, and Master of Science in Robotics Engineering*

August 31, 2022.

## **ACKNOWLEDGEMENTS**

I would like to express special thanks to my family, who have – through this long journey – empowered me with the means and encouragement to do more – So to my father, mother, brother, sister, and now late aunt, I say, E seun. I would also like to express my thanks to my long-time partner, M.R., who has been a source of great emotional support and encouragement. Beyond family, I would like to express my many thanks to the team at Hiro, especially my supervisor, Mr. Michele Oliveri for enlightening me, and relating positively with my work which gave me the strength to complete this thesis. I would also like to give thanks to my supervisors at UNIGE, Professor Fulvio Mastrogiovanni and Dr. Alessandro Carfi, for their guidance, and taking instances out of their precious time to check and aid my progress. Penultimately, I also give thanks to the University of Genoa, and Ecole Centrale de Nantes for giving me the opportunity to learn and develop in a field I cherish. Finally, I owe myself thanks. It has been 3 years, I have worked hard, or at least, hard enough.

## **DEDICATION**

There is a list of people this work could be dedicated to, but there is a minority group I would like to acknowledge for their tenacity and possibly highlight a little about their struggles - To the few would be robotics engineers with a background far from the requirements; To mechanical engineers wanting to be robotics engineers, I dedicate this to you. Unfamiliar with the tenets of programming; dumbfounded by object orientation, controlled by control systems - I promise you, it's not hard at all. Continuously strive to learn, develop daily, and prove yourself in the field, because you belong here too.

## **ABSTRACT**

Disassembling electronic waste is a reverse engineering process that relies on the creativity and intelligence that humans possess. Translating this skill into an automated process requires a step by step understanding of the overall process, and in this analysis, the removal of fasteners such as screws are the first hindrances to the separation of electronic waste into their various structural components. This thesis aimed to create a detection, localization, and tracking solution for screws in the industrial recycling process, taking into consideration system requirements. The process requires robust and speedy screw detection, which drove the project into an analysis of traditional and modern methods for object detection. Modern object detection techniques have been reviewed to provide more performance efficient solutions. Through training modern object detection models and evaluating them quantitatively on general computer vision metrics, and qualitatively on individual analysis of model performances and outputs, three object detection models: The SSD Resnet 50 FPN v1, YOLOv5s, and YOLOv6s were compared. The results through testing show comparable performance between models, with the YOLOv5s model being the most appropriate solution to the engineering problem presented.

## CONTENTS

ACKNOWLEDGEMENTS .....	i
DEDICATION .....	ii
ABSTRACT .....	iii
LIST OF FIGURES .....	vii
LIST OF TABLES .....	ix
CHAPTER ONE .....	1
1. INTRODUCTION .....	1
1.1. BACKGROUND OF STUDY .....	1
1.2. CHALLENGES FACED TOWARDS COUNTERING E-WASTE.....	2
1.2.1. The Volume of Electronic Waste.....	3
1.2.2. Material Complexity.....	3
1.2.3. Health Hazards .....	3
1.2.4. Lacking Recycling Standards .....	4
1.2.5. Policy Implementation .....	4
1.3. PROPOSED SOLUTIONS TO E-WASTE .....	4
1.3.1. Preventive Approaches to Minimize E-Waste Generation .....	5
1.3.2. Recycling E-Waste .....	5
1.4. RECYCLING FPD PANELS: HIRO-ROBOTICS CASE STUDY .....	5
1.4.1. Project Layout .....	6
1.4.2. Problem Statement .....	9
1.4.3. Research Aim and Objectives.....	10
1.4.3.1. Aim of Research .....	10
1.4.3.2. Objectives of Research.....	10
1.4.4. Justification of Research .....	11
1.4.5. Novelty .....	11
1.4.6. Benefits of Study .....	11
1.4.7. Scope and Limitations .....	11
CHAPTER TWO .....	13
2. LITERATURE REVIEW .....	13
2.1. OBJECT DETECTION .....	13
2.1.1. Traditional Machine Learning Methods.....	16
2.1.1.1. Template matching .....	16

2.1.1.2.	Viola-jones detectors .....	17
2.1.1.3.	Histogram of oriented gradient detectors.....	18
2.1.1.4.	Review of traditional object detectors in tasks of similar nature to screw detection	19
2.1.2.	Deep Learning Methods.....	20
2.1.2.1.	Two-stage detectors .....	22
2.1.2.1.1.	R-CNN family.....	23
2.1.2.1.2.	SPP-net.....	24
2.1.2.1.3.	FPN .....	25
2.1.2.2.	Single-stage detectors.....	26
2.1.2.2.1.	You only look once (YOLO).....	27
2.1.2.2.2.	Single shot detector (SSD).....	30
2.1.2.3.	Review of deep-learning object detectors for similar tasks .....	33
CHAPTER THREE	.....	34
3.	MATERIALS AND METHODS .....	34
3.1.	INTRODUCTION .....	34
3.2.	EQUIPMENT .....	34
3.2.1.	End-effector .....	35
3.2.1.1.	Camera.....	35
3.2.1.2.	Screwdriver .....	35
3.2.1.3.	Ring-Light .....	35
3.2.2.	Industrial Robot Arm.....	36
3.2.3.	Workstation .....	36
3.2.3.1.	Software tools.....	36
3.2.3.1.1.	Dataset preparation.....	36
3.2.3.1.2.	Model Development .....	37
3.3.	DATA.....	37
3.3.1.	Labelling .....	38
3.3.2.	Augmentation .....	39
3.3.3.	Splitting .....	41
3.4.	TRAINING .....	41
3.4.1.	SSD Resnet 50 FPN v1 .....	41
3.4.2.	YOLOv5s .....	43
3.4.3.	YOLOv6s .....	45

3.5.	EVALUATION .....	46
3.5.1.	Evaluation Metrics .....	47
3.5.1.1.	Intersection over union.....	47
3.5.1.2.	Precision and recall.....	48
3.5.1.3.	Precision-recall curve.....	49
3.5.1.4.	Average precision (AP) and mean average precision (mAP) .....	49
CHAPTER FOUR	.....	52
4.	RESULTS AND DISCUSSIONS.....	52
4.1.	INTRODUCTION.....	52
4.2.	SSD RESNET 50 FPN V1.....	53
4.2.1.	False Positives .....	55
4.2.2.	False Negatives .....	56
4.3.	YOLOv5.....	57
4.3.1.	YOLOv5 False Positives.....	60
4.3.2.	YOLOv5 False Negatives.....	62
4.3.3.	True Positive Extras.....	63
4.3.4.	TensorRT .....	64
4.4.	YOLOv6.....	64
4.4.1.	YOLOv6 False Positives.....	66
4.4.2.	YOLOv6 False Negatives.....	68
4.4.3.	YOLOv6 True Positive Extras .....	70
CHAPTER FIVE	.....	72
5.	CONCLUSION.....	72
5.1.	INTRODUCTION.....	72
5.2.	MILESTONES.....	72
5.3.	CONTRIBUTIONS .....	72
5.4.	OBSERVATIONS .....	73
5.5.	SUMMARY .....	73
REFERENCES	.....	74

## LIST OF FIGURES

Figure 1.1: Potential Revenue from E-Waste Streams [3].	2
Figure 1.2: CAD Render of FPD Recycling Process Work-Cells (Hiro-Robotics)	6
Figure 1.3: Dump of Old Monitors and FPD TVs	7
Figure 1.4: External Screws Holding the Panels Together	7
Figure 1.5: Internal Screws Holding Circuitry and Internals Together	8
Figure 1.6: Project Focused Recycling Process Flowchart	9
Figure 2.1: Traditional Object Detection Pipeline: (A) Informative Region Selection. (B) Feature Extraction. (C) Classification (i) Butterfly Classification (ii) Flower Classification.	14
Figure 2.2: Architecture of AlexNet [15].	15
Figure 2.3: Overview of HOG Feature Extraction and Object Detection Chain [23].	18
Figure 2.4: McCulloch-Pitts Neuron Model in an ANN [26].	21
Figure 2.5: Feed-Forward Network With 7 Inputs, and 5 Nodes in the Hidden Layer and 1 Output [26].	21
Figure 2.6: Convolutional Neural Network [27].	22
Figure 2.7: R-CNN Overview. (1) Taking Input Image, (2) Extract About 2000 Region Proposals, (3) Use CNN to Compute Features for Each Proposal, (4) Classification Using Linear SVMs. [30]	23
Figure 2.8: Fast R-CNN Architecture. A Single Image as Input with Multiple Regions of Interest (Rois) Sent into A Fully Convolutional Neural Network [31].	24
Figure 2.9: Faster RCNN, A Unified Network for Object Detection [32].	24
Figure 2.10: Network Structure with SPP Layer [33].	25
Figure 2.11: (a) Building A Feature Pyramid with An Image Pyramid. (b)Detection Systems Utilizing Single Scale Feature for Detection. (c) An Option to Reuse the Pyramidal feature Hierarchy Computed by CNNs like a Featurized Image Pyramid. (d)Proposed FPN by Lin T. et al. [34].	26
Figure 2.12: Original YOLO Architecture with 24 Conv. Layers, Followed by 2 Fully Connected Layers. The inclusion of 1x1 reduction filters lowers the feature space of the preceding layer, followed by a 3x3 convolutional filter [36].	27
Figure 2.13: The YOLO Model Solving Detection as A Regression Problem Dividing Each Image into an N x N Grid Where Each Grid Cell Predicts B Bounding Boxes [36].	28
Figure 2.14: Comparison of PP-YOLO with Other State of the Art Detectors [39].	29
Figure 2.15: Comparisons of Different Version of YOLOv5 with EfficientDet [40].	29
Figure 2.16: Comparisons of Different Object Detection Models with YOLOv6 [41].	30
Figure 2.16: A Comparison Between SSD and YOLO [43].	31
Figure 2.17: SSD Framework [43].	32
Figure 2.18: SSD Resnet FPN Architecture [43].	32
Figure 3.1: Robot Workcell.	34
Figure 3.2: IDOS UI-3070SE-C-HQ Camera.	35
Figure 3.3: Ur10e Robot Outfitted with Peripheral Devices for Unscrewing Task	36
Figure 3.4: Original Dataset Stats.	37
Figure 3.5: Image Label Heatmap.	38
Figure 3.6: Labelled Image.	38
Figure 3.7: Original Image Being Resized and Tiled.	39
Figure 3.8: Horizontal and Vertical Flipping of Images	40
Figure 3.9: From Right to Left, 1. Cropping by 25%. 2. Greyscaling. 3. Noise.	40
Figure 3.10: Images Generated from Combined Augmentation and Mosaiced.	41



Figure 3.11: Loss Metrics: 1. Classification Loss 2. Localization Loss 3. Regularization Loss.....	42
Figure 3.12: Total Loss. ....	43
Figure 3.13: Learning Rate. ....	43
Figure 3.14: Training Loss: 1. Localization Loss 2. Classification Loss.....	44
Figure 3.15: Validation Loss: 1. Localization Loss 2. Classification Loss.....	44
Figure 3.16: Test Dataset Stats. ....	46
Figure 3.17: Test Images Label Heatmap .....	46
Figure 3.18: Intersection Over Union [51]. ....	48
Figure 3.19: Precision and Recall [51]. ....	48
Figure 3.20: Different IoU Values [51]. ....	49
Figure 3.21: Average Precision [51]. ....	50
Figure 3.22: Mean Average Precision [51]. ....	50
Figure 3.23: Average Precision at Various IoUs and Scales[51]. ....	50
Figure 4.1: Precision@0.5:0.95 and Precision(Small) Respectively for the SSD Model.....	53
Figure 4.2: Precision@0.5 and Recall Respectively for the SSD Model .....	53
Figure 4.3: SSD Resnet Detections.....	54
Figure 4.4: SSD Resnet Detections .....	54
Figure 4.5: SSD Resnet False Positive Detections (Detections vs Ground Truths).....	56
Figure 4.6: SSD Resnet False Positive Detections (Detections vs Ground Truths).....	56
Figure 4.7: SSD Resnet False Negative Detections (Detections vs Ground Truths) .....	57
Figure 4.8: SSD Resnet False Negative Detections (Detections vs Ground Truths) .....	57
Figure 4.9: YOLOv5 Detections .....	58
Figure 4.10: YOLOv5 Detections .....	58
Figure 4.11: Precision and Recall Graphs.....	59
Figure 4.12: Precision-Recall Curve.....	60
Figure 4.13: YOLOv5 False Positive Detections (Detection VS Ground Truth).....	60
Figure 4.14: YOLOv5 Positive Detections (Detection VS Ground Truth).....	61
Figure 4.15: YOLOv5 Positive Detections (Detection VS Ground Truth).....	61
Figure 4.16: YOLOv5 False Negative Detections (Detection VS Ground Truth).....	62
Figure 4.17: YOLOv5 False Negative Detections (Detection VS Ground Truth).....	62
Figure 4.18: YOLOv5 False Negative Detections (Detection VS Ground Truth).....	63
Figure 4.19: YOLOv5 Extra Detections (Detection VS Ground Truth).....	64
Figure 4.20: YOLOv5 Extra Detections (Detection VS Ground Truth).....	64
Figure 4.21: YOLOv6 Detections .....	65
Figure 4.22: YOLOv6 Detections .....	65
Figure 4.23: YOLOv6 False Positives (Detection VS Ground Truth). ....	67
Figure 4.24: YOLOv6 False Positives (Detection VS Ground Truth). ....	67
Figure 4.25: YOLOv6 False Positives (Detection VS Ground Truth). ....	68
Figure 4.26: YOLOv6 False Negatives (Detection VS Ground Truth).....	68
Figure 4.27: YOLOv6 False Negative (Detection VS Ground Truth). ....	69
Figure 4.28: YOLOv6 False Negatives (Detection VS Ground Truth).....	69
Figure 4.29: YOLOv6 True Extras (Detection VS Ground Truth). ....	70
Figure 4.30: YOLOv6 True Positive Extras (Detection VS Ground Truth).....	70

## LIST OF TABLES

Table 3.1: Pipeline Configuration for SSD Resnet 50 FPN v1 .....	41
Table 3.2: SSD Resnet 50 FPN v1 Model Parameters.....	42
Table 3.3: YOLOv5s Model Parameters. ....	43
Table 3.4: YOLOv5s Custom Model Training Parameters. ....	45
Table 3.5: YOLOv6s Model Parameters. ....	45
Table 4.1: Table of Primary Properties for SSD Detections .....	55
Table 4.2: Results for SSD Detection.....	55
Table 4.3: Table of Primary Properties for YOLOv5 Detections.....	59
Table 4.4: Results for YOLOv5 Detection .....	59
Table 4.5: Table of Primary Properties for YOLOv6 Detections.....	65
Table 4.6: Results for YOLOv6 Detection .....	66

## CHAPTER ONE

### 1. INTRODUCTION

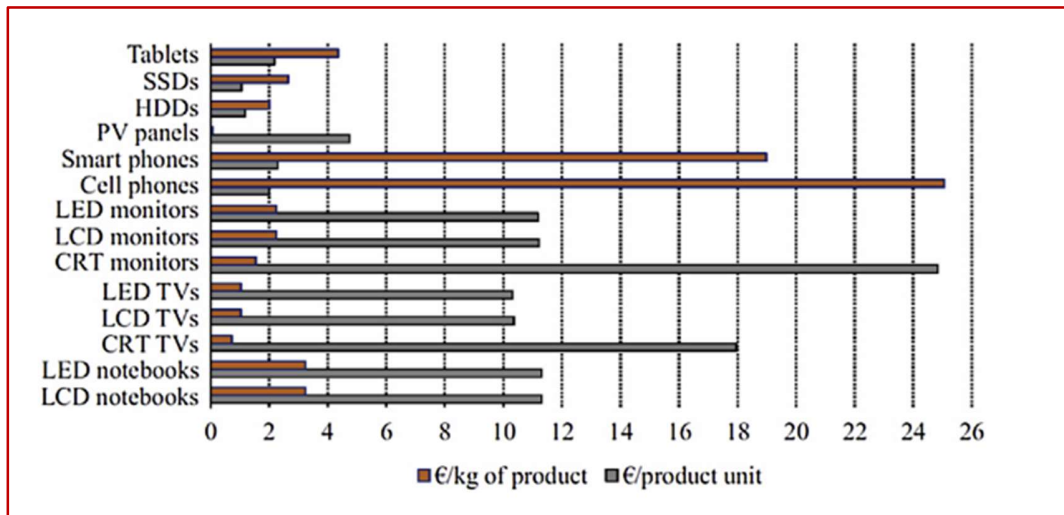
#### 1.1. BACKGROUND OF STUDY

Electronics, majorly, consumer electronics are ubiquitous objects in the daily lives of the average human. From mobile phones, laptops, tablet devices and televisions, there are a numerous amount of consumer devices that go through a product lifecycle from production to consumer's possession and then to end-product. Consumers themselves go through a vastly different lifespan as compared to the devices they end up acquiring, which as a result concludes with most individuals using multiple of versions of the same devices through their lifetime. These various iterations of devices begin their cycle through production, to purchase by consumers, and end their cycle at disposal from these same hands – This begets a new cycle of the next generation of these devices, arriving where the previous generation was abandoned. Thinking further on the lifecycle of these devices, we follow a product lifetime study by Thiébaud E. et al. in 2018 on, “The service lifetime, storage time and disposal pathways of electronic equipment: A Swiss case study”, The average service lifetime for mobile phones is within 4.3 years at the disposal of the consumer; for a headset, the average service lifetime is within 3.3 years and for an FPD TV is within 5-8 years of use [1]. This paints a picture detailing the average consumer lifecycle certain electronic devices undergo. With the Italian life expectancy equaling 83.20 years, the average Italian is subject to use 19.35 phones in his/her/their lifetime, but a basic reality is that consumers are changing their devices more regularly subject to technological advancements, functionality and contemporary trends, and this reality can easily overshadow today's projections. This leads to an accumulation of electronic waste as the former device is discarded, and in historical likeness, an exponential increase of electronic waste as seen where electronic waste was doubled in Europe between 1998 and 2010 [2]. In conscious effort to avert the waste crisis and build a cleaner world, countries have taken direct in fighting waste generation by issuing directives such as Europe's Waste from Electrical and Electronic Equipment Directive (WEEE) which requires manufacturers to take returned devices at no cost towards recycling at least 65% of their average weight, and the efforts of the United States' Environmental Protection Agency in coordination with the International E-waste Management Network to exchange best practices on waste management and support the country's government's national strategy for electronics stewardship. These policies and directives take a broad scope towards the management of the E-waste crisis from the top down, by generally:

- Incentivizing recyclable and green friendly design of electronics
- Increasing recycling

- Incentivizing collective responsibility and representative leadership

The relative high costs at the early stage of the formal e-waste recycling industry put off promising industrial growth, and consequently moved the recycling industry through informal and/or illegal industries that generate second markets for the retrieved resources. It is quite telling that a major incentive towards undergoing monumental industrial investments must follow strong economic incentives to the participating enterprises. Through technological advances, growing research, and political incentives (subsidies and tax-cuts), the industry is burgeoning especially concerned with growing climate change concerns.



*Figure 1.1: Potential Revenue from E-Waste Streams [3].*

There has been a global demand for metals such as copper tin and silver in the growing electronic and technological industries as applications experience growth in an era of booming industries, and the digital age. The estimated value of e-waste in Figure 1.1 experiencing demand from the booming technological industry was estimated at 48 billion euros as of 2015[3].

## 1.2. CHALLENGES FACED TOWARDS COUNTERING E-WASTE

Efforts to counter e-waste are still short from optimal as the complicated nature of recycling electronic waste is multifaceted, and the complexity of the solution is a response to the component nature of the waste and the poor early implementations towards recycling standards. The nature of this endeavor is met with varying degrees of

challenges that impede its quick development. A brief overview of challenges mitigating the efforts for better e-waste management include:

### **1.2.1. The Volume of Electronic Waste**

Estimations put the mass of electronic waste generated at 40 million tons per year on a global scale [4]. In 2019 alone, the global mass of generated e-waste rested at 53.6 million tons, with a growth rate of 9.3 million tons per year since 2014, and a projected size of 74.7 million tons by 2030 [5]. Comparatively, the formal documentation places the amount recycled by legal government and commercial enterprises at 17.4% of the total generated electronic waste in 2019. These sizes are fueled by the short life cycles of electronic devices, coupled with high consumer adoption and few pre-end cycle recycling options such as refurbishment and repair.

### **1.2.2. Material Complexity**

From the perspective of material design, the elemental composition of electronic waste spreads to up to 69 elements in the periodic table, with no incentive towards the product end cycle, many components such as precious metals like iridium or critical raw materials like germanium and indium pose challenges for extraction from electronic waste. The separation and sorting, towards extraction of these important elements can prove difficult based on the design parameters of the producers [5].

### **1.2.3. Health Hazards**

Various studies highlight the dangers towards human health when exposed to toxins such as lead which are prominent in electronic waste. E-waste workers commonly report various health complications such as headaches, nausea, shortness of breath, chest pain, weakness, and stress [5]. While limited data is available towards the common exposure of individuals to e-waste, it should be considered that the immensity of the waste, gives a natural consequence of a considerable population being exposed to the toxins of the waste. Varying adverse effects on exposure to the toxic waste might include adverse birth outcomes, altered neurodevelopment, adverse learning outcomes, adverse cardiovascular effects, adverse respiratory effects, adverse immune system effects, hearing loss, skin effects and even leading up to DNA damage [5]. These adverse effects are not the exclusive reserve of E-waste workers, as studies show that during the process of electronic waste disposal, harmful toxins such as heavy metals, polychlorinated dibenzo-p-dioxins and dibenzofurans diffuse into the ambient environment when disposed primitively [4].

#### **1.2.4. Lacking Recycling Standards**

Recycling the immense volume, and complex device structure of electronic waste is far from a simple task. With technological advances allowing devices to cram more components into a smaller form factor, and the profit incentives pushing production companies into creating less modular and repair friendly electronics creates a market of devices with complex disassembly and sorting processes. Formal e-recycling processes generally commence from the point of sorting; testing to find faulty devices; refurbishing and repairing those devices if possible; then disassembling, occasionally shredding and consequent sorting through a combined automated and human labor process [6]. With the late growth of the E-recycling industry, it faces the challenges of any young field, finding itself in a dearth of research in realization of a more robust formal sector, leading to underperformance. This dearth of research impedes the possible engineering infrastructure and optimal techniques that could be used to mitigate the greater issue and leaves the brunt of the recycling tasks to informal sectors to shoulder, where dangerous and unprofessional recovery techniques are frequently utilized [7].

#### **1.2.5. Policy Implementation**

Another consequence of the lack of quantitative and qualitative research finds itself in policy making, where sufficient and reliable data is lacking, and this impedes policy makers designing and proposing e-waste management strategies to the government and associated enterprises wishing to make rational investment decisions [8].

These challenges impede the hasty management, recycling, or disposal of e-waste. These challenges are derivatives of the nature/characteristics of e-waste as a complex electronic by-product, combined with the result of poor early implementations of waste management solutions due to governmental foresight and careless ambition.

### **1.3. PROPOSED SOLUTIONS TO E-WASTE**

E-waste cannot be completely eradicated, but alleviated, as it is an inevitable end of all devices – and all things. Therefore, approaches to solving the e-waste crisis could be considered from two perspectives based on the product life stage of e-waste generation to mitigate its continually increasing volume [7].

### **1.3.1. Preventive Approaches to Minimize E-Waste Generation**

Implementing policies and programs offering technical support to local manufacturers; also implementing standards as required to hazardous materials that should be prohibited and supporting the use of eco-friendly materials wherever possible; training and education of those also involved in the exchange cycle of electronic devices - from consumer to enterprise – whereby all actions culminate into mitigating the likelihood of non-reusable, toxic, and nonrecyclable products being designed.

### **1.3.2. Recycling E-Waste**

Incentivizing recycling from a policy and global-social standpoint; giving enterprises a possible reward towards investing in the formal sector of the e-recycling industry, thereby, eliminating the dearth of research and hence building a foundation for better industrial and engineering standards towards the implementation of recycling processes.

Facing this monumental issue, this project hopes to contribute towards the solutions alleviating our growing e-waste problem, taking a focus on an engineering case study by implementing a technological solution towards the recycling of a particular type of e-waste. Through solving one specific problem, we aim to create the foundation to solve the later ones, rightfully playing into a proposed solution to the problem.

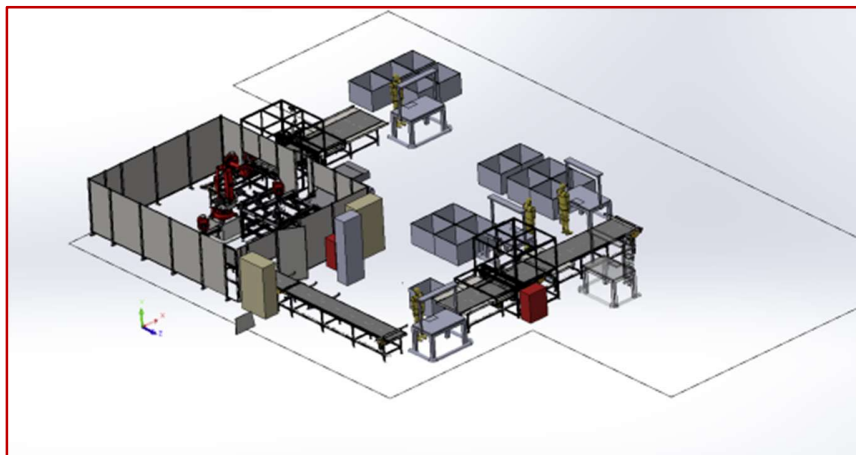
## **1.4. RECYCLING FPD PANELS: HIRO-ROBOTICS CASE STUDY**

As stated in the background of study, it is not unusual for FPD TVs to have an average life service of 5-8 years of use, which by the average Italian life expectancy, brings the average number of FPDs of individual use in a contemporary lifetime to about 13. Envisioning the problem as process in the formal recycling sector, optimization is paramount to garner sufficient returns on investment to the contracted company.

This present engineering task has been taken up by Hiro-Robotics, with the aim to propose an innovative system that integrates collaborative robotics, artificial vision and intelligence algorithms dedicated to plants specialized in the recycling of electronic waste and of flat screen TVs and monitors. A background into the company - Hiro-Robotics is a robotics startup cofounded in 2018 by Davide Labolani, Jacopo Lottero, Michele Olivieri and Tomasco Manca, who were all formerly robotics master's students in the European Master's in Advanced Robotics (EMARO+) program. This innovative startup aims at achieving and implementing innovative

processes and research projects, developing systems that have been imagined but have yet to be created. The company specializes in robotics, computer vision and machine learning and is based in Genoa, Liguria, Italy. The primary focus of Hiro-Robotics at this moment centers on the waste management and recycling sector, and in determination to successfully implement this major recycling project have collaborated with the Iren Group S.p.a., to realize an automated and robotized recycling line.

To accomplish this task, the company foresaw the exploitation of the following technology as an aspect of the total recycling process: Computer vision and intelligence for vision guidance of a robot in the dismantling process of flat panel displays like televisions and monitors, with an underlying aspect of this problem being the detection, localization and tracking of screws during the detection process before dismantling, the task that has been given to the target of this thesis.



*Figure 1.2: CAD Render of FPD Recycling Process Work-Cells (Hiro-Robotics)*

#### **1.4.1. Project Layout**

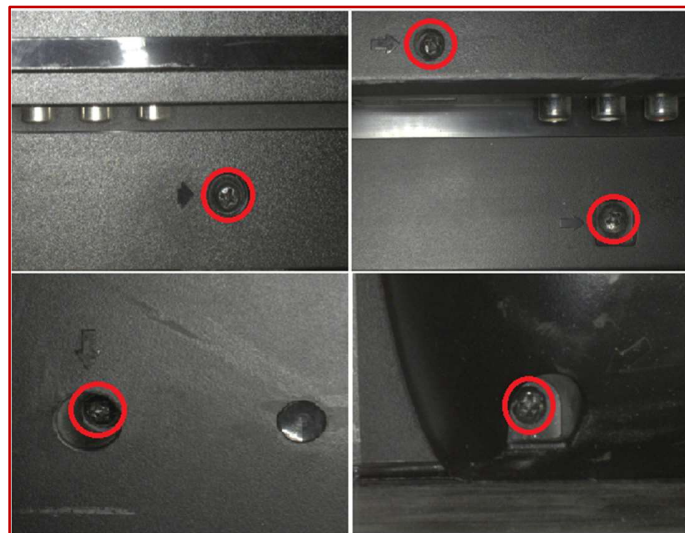
To facilitate the dismantling of FPD panels, a structured process dedicated to the general design of these monitors must be established. First an analysis of the general structure of FPD waste.





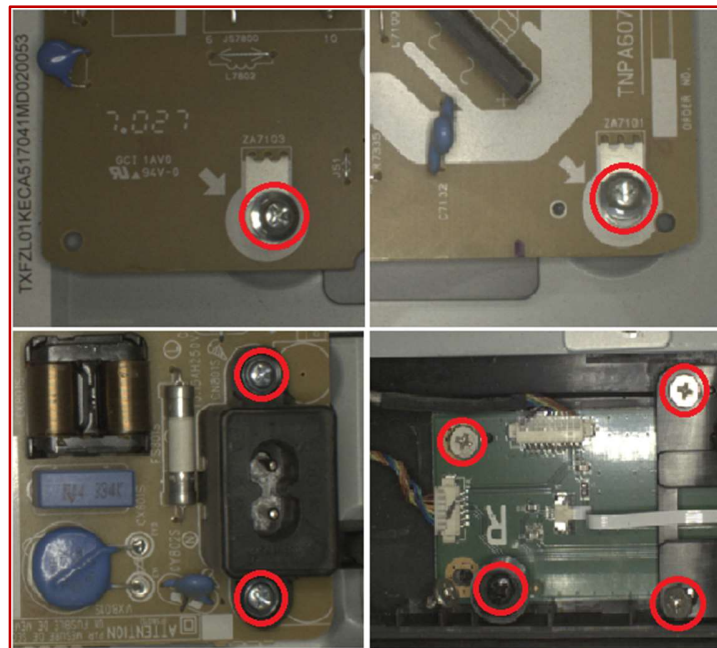
*Figure 1.3: Dump of Old Monitors and FPD TVs*

Figure 1.3 displays a dump of old flat panel display monitors discarded in a landfill, the image serves to display a common theme of the problem, and on closer inspection, highlights the first problem encountered on their dismantling. The internals of each panel are generally encased in a plastic shell held together by screws. Humans have no issues disassembling these devices given the right set of tools and an understanding of how the structural building-blocks of these devices are arranged, but for autonomous systems, the process of dismantling is more a logical and step by step process, hence, to tackle the problem screw detection is has been generally a hardcoded task.



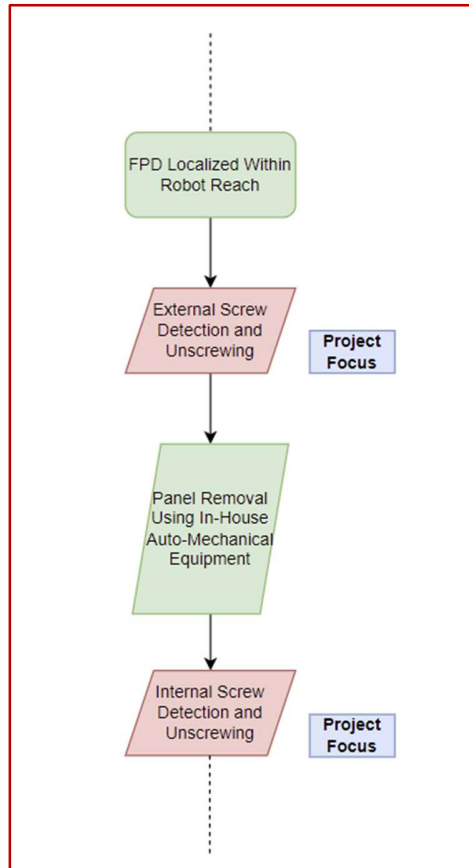
*Figure 1.4: External Screws Holding the Panels Together*

To accomplish the given task of recycling the FPD monitors, it is obvious that screws need to be removed to make way into the internals component, but even considering the internals, screws are also used to fasten various parts of the circuitry in place, and hence, to have a flat-out exploded view of components through disassembling, priority is attached to dismantling all screws automatically.



***Figure 1.5: Internal Screws Holding Circuitry and Internals Together***

Given the project and the first problem featured towards its implementation, we realize that we must teach the robot processing the dismantling and unscrewing with a model or models (possibly based off the different environments: Internal and external) to discern screws with high enough accuracy to differentiate similar objects in the camera periphery. Also, due to the nature of the recycling process, the robot operates at high speeds, for high volume processing, as a strong economic factor commonly incentivizing enterprises, therefore, the real time application and speed of detection is of great importance when selecting/developing a model for screw detection.



**Figure 1.6: Project Focused Recycling Process Flowchart**

The project focuses on the subprocesses of detection, localization and tracking of screws within the comparatively macro-recycling process.

#### **1.4.2. Problem Statement**

To properly start the project, the problem needs to be outlined explicitly, and it is defined as follows:

- Electronic waste management is a field of high importance for our global health, and there is a dearth of research and implementation assigned to the problem compared to its scale. Due to these conditions, we are currently undergoing an underutilized industry with a recycling efficiency of 43% in Europe, which is one of the better documented and implemented systems [7]. Still generating more waste than we have the capability to process, it is imperative that we boost the sector technologically and devise a means to improve throughput.

- In the microsystem of electronic waste management - A common task in the disassembly process during is the autonomous perception of screws before action can be taken upon them. Due to the nature of the intelligence field, training models with a contextual approach will give obvious improvements, therefore, creating the required dataset for the specific tasks of interior and exterior FPD panel screw detections will be of benefit to performance and model variety, as opposed to using already trained models with generic contexts or backgrounds [9].
- Solving this problem paves a clearly defined path towards engineering the recycle process for FPD TV panel dismantling for Hiro-Robotics and the Iren Group. It also provides a basis of research for other dismantling processes in the formal electronic recycling industry of a similar nature. Also factoring into the dearth of research current in the formal recycling industry

### **1.4.3. Research Aim and Objectives**

The task of identifying screws in an image is posed like any class identification task of objects in an image, this area falls under the field of machine learning and computer vision, specifically object detection. Object detection is a field posed to answer possibly and sufficiently, one of the age old and notable computer vision problems, *what objects are where in an image?*[10]. The aims and objectives are given based off the possible solution.

#### **1.4.3.1. Aim of Research**

To implement a deep learning and computer-vision based solution for screw detection, localization and tracking in the FPD TV recycling process.

#### **1.4.3.2. Objectives of Research**

The objectives of this research are:

- Select an appropriate object detection model(s) for the given task based on project requirements, in relation to state of the art and available literature.
- Create a dataset to train the preferred object detection model/models.
- Successfully train an object detection model to identify screws during the recycling process.

- Optimizing trained model for improved performance and throughput.
- Complete Implementation of an object detection pipeline for FPD monitor recycling process.

#### **1.4.4. Justification of Research**

“Electronic waste, or e-waste is an emerging problem as well as a business opportunity of increasing significance, given the volumes of e-waste being generated and the content of both toxic and valuable materials in them” – Rolf Widmer *et al.* [8].

Given the task at hand, the quote explicitly exemplifies the problem to build the foundations of a growing industry – the formal e-recycling industry. The aphorism “one man’s trash is another man’s treasure”, places itself nicely within this context and is added for clarity. Contributing to the research in this area provides definite financial benefits to the companies involved, and in moving forward, economic betterment of the global adopters of the state of the art in the formal e-recycling industry, leading to a betterment of the living standards of hundreds to possibly millions.

#### **1.4.5. Novelty**

- Object detection model(s) tailored for FPD TV recycling screw detection.
- Examination of developed screw detection model for recycling process.

#### **1.4.6. Benefits of Study**

- Improvement of volume throughput and respective economic metrics of the recycling process.
- Contribution to various sustainable development goals leading to a better earth.
- Resource efficiency is increased because of the reuse of resources, reinforcing market resources and stalling resource scarcity

#### **1.4.7. Scope and Limitations**

Due to the project focus, the given tasks are prioritized in the given descending order:

- Developing an object detection model for detecting screws.

- Comparing and evaluating various models based on performance
- Developing object tracking from object detection.
- Developing pipeline for deployment on UR10e collaborative robot.

The project is limited by the following constraints:

- Time and Specificity: Decisions made on the project direction take time for experimentation and development, even if these developments do not show up in the final project.
- Compute: The training process is limited by the capabilities of the available workstation, where training can take hours, and the trained model might not necessarily be a useful model.
- Technological Limitations: Performance is limited by the available technology in the areas of computer vision and artificial intelligence.
- Dataset: The dataset is self-produced and therefore limited in size. Developing a larger dataset will ensure better results.

## CHAPTER TWO

### 2. LITERATURE REVIEW

The disassembly process is a very important step to obtain the disjointed non-homogenous pieces of material that structure FPD TVs, and generally all electronic recyclable waste, which can make first level sorting an easier endeavor. Disassembly is a reverse engineering process and is thereby a complex process that humans dominate consistently, as various other automated processes fall short on the quality and quantity of these returned end-of-life products [11]. To enable these advanced technologies the ability to disassemble FPD TVs successfully, a logical process of the disassembly must be considered, and generally, for humans, the first step is an analysis of the general structure of TVs, for example, an LCD, a type of FPD TV, generally composes of six main structural parts and four connectors based of selective disassembly:

- Structural Components: Back Cover, PCD Cover, PCBs, Carrier, LCD module and Front Cover.
- Fasteners: Screws, Snap-Fits, Electric Cables, and Plastic Rivets [11].

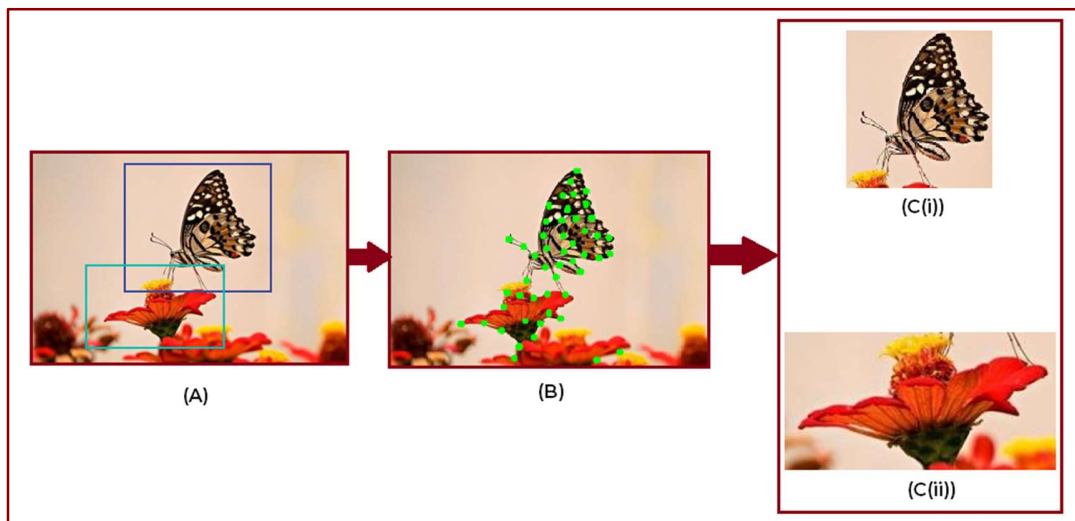
After detaching associated fasteners, the related structural components can then be removed. The initial task is therefore to detect the connectors. We are already aware that crosshead screws are a favorite associated with many FPD monitors [12], therefore, we are to propose a method to detect this specific type of screw and instruct the robot to use this algorithm to detect multiple screws during the recycling process and to consequently unscrew the detected screws using an attached screw-driver end-effector. The search for the optimum means for screw detection will drive our research and derive the overall outcome of this project. The detection of screws on the surfaces of the panels and PCBs will be treated in likeness with the general inquisition to describe the position and nature of objects in an image. This aspect of computer vision intersects the subfields of classification and localization and culminates into object detection. Object detection is an important computer vision task that deals with recognizing instances of visual objects in a certain class, such as cars, humans, and animals, in each digital image [10].

#### 2.1. OBJECT DETECTION

Object detection a subset of machine learning's deep learning techniques, which as opposed to image classification does not solely define what an object is in each image, but also its location in that image. The aim of object detection is to derive answers to some of the aged questions plaguing computer vision: *what objects are*

where? [10]. The question thus derives itself from the perspective of classification and localization, where the former defines the “what” and the latter defines the “where”. Historically, the development of object detection commenced traditionally with high dependence on feature engineering (detection of important features of objects in consideration in the image), which was usually artificially defined, and as a factor of this dependence on hard defined algorithms and architectures have fallen short when faced with anomalies or unreferenced situations, and as a consequence, in speed and generality [10][13][14] as compared to modern object detection offerings. Based off the understanding of the problem of object detection, traditional object detection methods refer to a general pipeline towards solving the problem, which goes as follows:

1. Informative region selection,
2. feature extraction,
3. and classification[14].

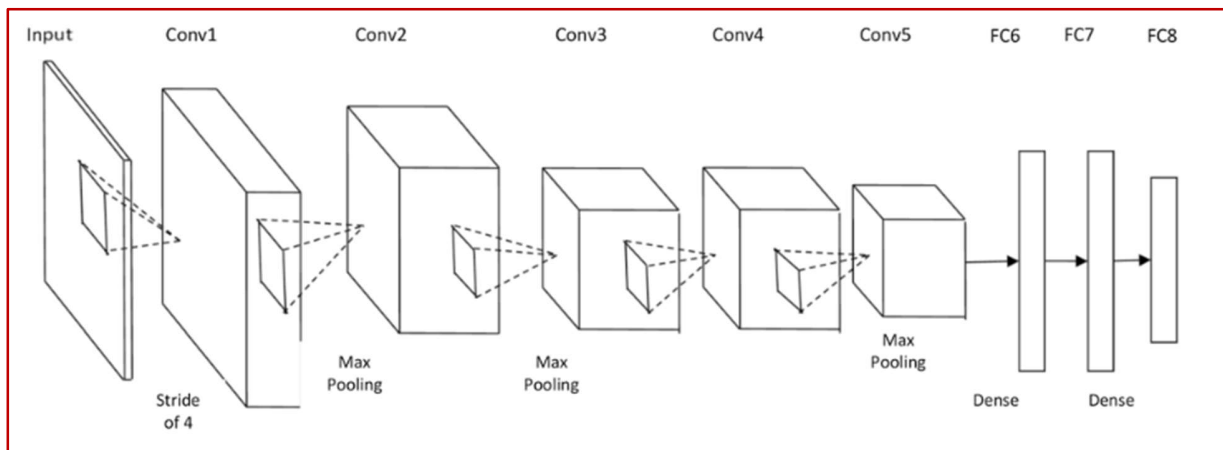


**Figure 2.1: Traditional Object Detection Pipeline: (A) Informative Region Selection. (B) Feature Extraction. (C) Classification (i) Butterfly Classification (ii) Flower Classification.**

From Figure 2.1, the pipeline involved in object detection necessitates this order as the latter can only be derived if the former has been achieved. The informative region selection stage is a requirement to exclude non-important regions of the image from the computational task of feature extraction. Different objects come in different sizes, orientations, and positions in an image, this proves a problem as choosing an important region as we also want to avoid the problem of unobserved important regions of the image. It comes naturally as a choice to select the sliding window approach which scans through the whole image with a multiscale sliding window, but this is an exhaustive



strategy that makes the entire process computationally expensive, slow and increases redundancy as repeated detections are bound to occur [14]. To identify the uniqueness of objects in images, features of certain objects that can provide semantic and robust representations of the object are extracted, examples of those used are the histogram of oriented gradients (HOG), scale-invariant feature transform, Haar-like features [14], Viola Jones detectors and DPM [10]. Although some of these detectors like the HOG can be used to detect different object classes, they were primarily created to solve the problem of pedestrian detection, and as so, has been a foundation of many consequently developed object detectors that have existed through the years. A breakthrough in the development of object detection and image classification occurred in 2012 with the introduction of convolutional neural networks which were computing architectures based off a surface level description of the working of the human brain, imbued with convolutional layers holding filters able to learn robust, high-level feature representations of images [10].



*Figure 2.2: Architecture of AlexNet [15]*

Convolutional neural networks were introduced by Fukushima in 1998 and possess a wide range of uses from text recognition to object detection. They are made of three main layers: Input layer, hidden layer, and output layer. The hidden layer is where most of the work is done. The hidden layer is a combination of multiple series of convolutional, pooling, fully connected layers, and normalization layers [15]. The layers work as follows:

- The convolutional layer applies a convolution operation, merging multiple sets of information from the base image and multiple convolutional filters. This extracts robust sets

of features from the image and gives a reaction that imitates individual neurons' feedback to visual stimuli.

- The pooling layer reduces the dimensionality of the convolutional layer output by operations such as extracting the maximum/average value in a region of the feature map (Max Pooling and Average Pooling respectively).
- The fully connected layer connects the various neurons to output classes.

In CNNs, the feature maps contain pixels which act like neurons, and each of these, 'neurons', in a convolutional layer is connected to the feature map of the previous layer using a set of weights, which represents the filter. Therefore, the input image is continuously convoluted, and with each layer – increasing depth – the receptive field increases [16]. Figure 2.2 displays the AlexNet architecture which is a deep neural network that pioneered in the field. It utilized augmentations comprising of image translation, patch extractions and horizontal reflection, then implements dropout layers which drop certain neurons in the deep learning architecture to reduce overfitting [15]. It was a novel and clear implementation, and a basis for network architectures moving forward, as it contains 5 convolutional layers each followed by a pooling layer, and then 3 fully connected networks to pool the aggregate data to a one-dimensional value, which then outputs a class. This review aims to explore the span of object detection in both traditional and deep learning methodologies, with hopes of understanding which model and architecture best fits as a solution to the screw detection problem.

### **2.1.1. Traditional Machine Learning Methods**

Through the field of computer vision, it has been established that objects are most easily located by exploiting notable features, features such as circles, straight lines arcs, holes, and corners. Corners are specifically of importance as they contain information on the orientation and location of objects and a means to observe the measures of their physical dimensions [17], [18], they are also a form of stable key points [18]. From this observation initializes multiple techniques that will be explored, with their possible scale with the aim of the project reviewed.

#### **2.1.1.1. *Template matching***

Template matching by normalized correlations is a widely used computer vision technique for detection of shapes and their locations in images. It is capable of accurately estimating the pose of a new object given limited data. It

simply requires using a given template, or base image, and finding the areas within the image of interest that match the given template. A significant problem with template matching is the rigidity of the model, as prior knowledge of the object of interest is required to find a suitable template, therefore searching for dynamic features will prove difficult to achieve. Also, template matching is time consuming. Matching a template to regions in an image and matching for various orientations and positions can prove time consuming and computationally expensive.

In the paper, “Fast and High-Performance Template Matching Method”, by Alexander Sibiryakov in 2011, a template matching method was proposed, which was robust enough for outliers and fast enough for near real time operation. The experiment was performed on a desktop computer with the following specifications: CPU Q6700 @2.66GHz and 3 GB of ram. This template matching technique while significantly improved lacks the robustness to observe the diversity of objects with various differing features, as well as comes short in terms of rotation and scale invariance [19]. The performance is also an improvement on traditional template matching but falls short by a small margin for real time applications.

In “Multi-task Template Matching for Object Detection, Segmentation and Pose Estimation Using Depth Images”, by Kiru P. *et al.* (2019), a novel framework was proposed for 6D pose estimation and segmentation of objects using a template set of depth images. This method proposed redirects the adamant machine learning need to rely on large datasets and curbs the reliance on object textures which traditionally makes it difficult to differentiate objects with similar geometry but dissimilar textures. The MTTM is limited by its lack of use of color information which renders it impossible to produce local correspondences and another problem is the length of the computation time, with evaluation taking 20 seconds and recognition taking 1-2 seconds per image[20].

#### **2.1.1.2. *Viola-jones detectors***

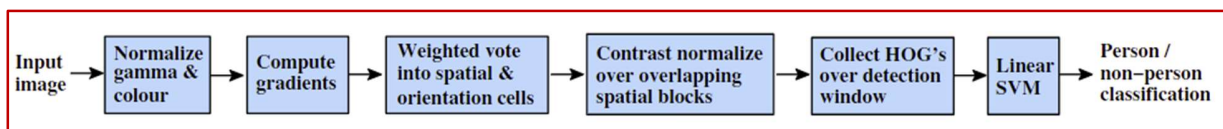
Twenty one years ago, for the first time in real time, the detection of human faces had been achieved by P. Viola and M. Jones avoiding constrains such as skin pigments, This has been made possible by three significant contributions towards creating the Viola Jones Detectors: At the time, a novel image representation called “Integral Image” which allowed for faster computations with the designed detector; the AdaBoost learning algorithm which stood in as a classifier for selecting the most prominent features from a vast population; finally, a technique for combining classifiers in a ‘cascade’ which allowed for concentration of face like features [21]. Whilst just running on a 700Mhz Pentium, the detector could still be up to a hundred times faster than other

algorithms of the comparable period, matching or surpassing their accuracy [10]. The Viola-Jones detector uses the most straight forward approach towards obtaining sample regions - Sliding windows. This exhaustive and computationally expensive method is of reproach towards high-speed intensive tasks as we will see moving forward where regional proposal methods are created as a direct counter algorithm to mitigate the computational losses created by this approach[10].

Where the Viola-Jones detectors found success in many aspects of facial feature detection and tracking, it often lacked in other areas of research and object detection. In the paper, “An Enhanced Viola-Jones Vehicle Detection Method from Unmanned Aerial Vehicles Imagery”, by Xu Y. *et al.* (2017), the Viola-Jones detector is cited as sensitive of object orientations and only works when object orientations are known prior to detection. The proposed solution to this weakness of the Viola-Jones detector was a rotation of the image before detection [22]. This proves a problem in terms of robustness as changes in orientations are significant enough to throw the detector off its tracks, where in the case of screws, or even general-real-time object detection, objects continuously on the move and in different poses that would make it hard for the detector to perform generally.

### 2.1.1.3. *Histogram of oriented gradient detectors*

The HOG descriptors were originally proposed by N. Dalal and B. Triggs in 2005, in their “Histogram of oriented gradients for human detection” paper, which was an outcome of the study of feature sets for robust object recognition. The outcome of this search was the histogram of oriented gradient, a novel feature set for human and general object detection, outperforming the other existing feature sets at the time [23]. The HOG can be described as a significant upgrade to the scale-invariant feature transform (SIFT), edge orientation histograms and shape contexts. The idea behind the HOG is basically that local object appearance and shape can be accurately characterized by the distribution of local intensity gradients or edge directions without a precise knowledge of their locations. The HOG can be used to detect several different classes, but it was brought about mainly to solve the problem of pedestrian detection.



*Figure 2.3: Overview of HOG Feature Extraction and Object Detection Chain [23].*

#### **2.1.1.4. Review of traditional object detectors in tasks of similar nature to screw detection**

Various traditional approaches to object detection exist and have been deployed for use in industrial applications concerning the detection of outliers, machine parts and the general deployment of perception systems for robust automated applications. This section tries to review the implementation of these traditional algorithms in various pipelines to foster industrial effectiveness, and their corresponding outcomes and possible implications for screw detection using deep learning moving further.

In the paper, “Autonomous Disassembly of Electric Vehicle Motors Based on Robot Cognition”, by Bdiwi M. *et al.* (2016), a robot workstation is proposed for the automatic disassembly of electric vehicle motors, and therefore, a novel object detection algorithm for the detection of motor screws is developed. The workstation proposed is composed of an industrial KUKA robot (KR 240-2), assisted by two Microsoft, X-box Kinect cameras each with 2 640x480 pixel cameras at 30Hz, for RGB and IR depth-finding cameras respectively, coupled with a force/torque sensor and a controlled environment for no variations in lighting conditions [24]. The project further proposes a novel idea for scale, rotation and translation invariant detection of screws and bolts, using the characteristics of these mechanical parts with respect to their greyscale, depth and HSV values, using the Harris corner detector algorithm to generate distinct features from the screws. The vision process goes as follows: A region of interest is considered, which is the center of the image frame. The features are detected using the Harris feature detector, which was chosen for its higher true positive values at the time and for feature stability. Then optimization is done through the removal of noisy features at holes using the HSV color space by eliminating the range for holes. The features left after the first optimization are used to generate regions yet again and undergo further optimization to determine important parameters like size and shape for the further filtering of false positive regions. Another optimization stage is to optimize for the location of the screw. The computing time for Bdiwi M. *et al.*'s proposed algorithm is dependent on the number of feature inputs consequently obtained from Harris detection. A single frame is processed in 71ms at 60 key points on an intel i7 CPU @3.40 GHz in controlled contrast ratio.

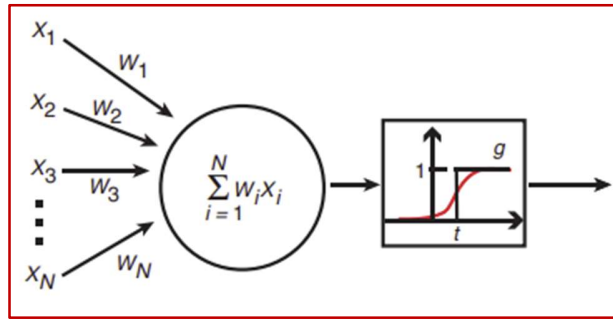
From the publication of the 2017 IEEE international conference on advanced intelligent mechatronics, a paper on a novel visual detecting and positioning method for screw holes was published which introduced an improved Hough transform algorithm to detect the circle screw holes, and then further classify the circle detections into holes using template matching based on the characteristics of gradation histogram [25]. It is noted that clearly,

pure use of the Hough transform would not obtain accurate detection results. The results of the given algorithm put the average positioning error within 0.01mm in the horizontal position and 0.02mm in the 45-degree obliqueness position. No description was made on the inference speed, but the peculiarity of the vision setup and its specifics show a physical constraint of the system: The robot arm, camera and light source were separated physically; An oblique lighting method based on double red annular LEDs was proposed to differentiate black screw holes from dark backgrounds; Mounting parameters of the proposed LEDs. This proposed screw-hole detection method required accounting for more variables and a complex system with many environmental influences, hence, not very general.

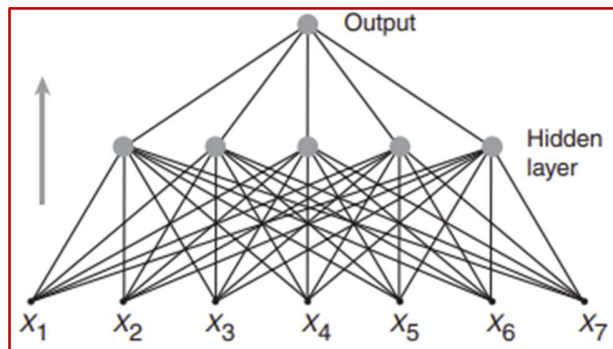
In the paper, “High-speed object matching and localization using gradient orientation features”, by Xu X. *et al.* (2014), A new object matching method was proposed which proved more efficient over traditional vision methods at the time, using the method of decomposition of orientation and position estimation into two cascade steps [18]. The process goes as follows: Initial orientation and position is determined with HOG, reducing orientation search from 2D template matching to 1D correlation matching; for the second stage, the obtained orientation and position values are refined using Dominant Orientation Template (DOT) for matching. The combination of these two vision algorithms into a cascade resulted into – at-the-time – high-speed and robust object matching. The accuracy and speed of the system was tested on an image dataset for industrial inspection applications containing various parts, and in particular, screws, using a SHARP factory inspection camera. The performance on an intel atom notebook gave an inference speed of 501.0ms for screw detections.

### **2.1.2. Deep Learning Methods**

As discussed previously, the object detection field had a significant revelation with the advent of deep convolutional neural networks which were able to learn robust and high-level feature representations of images. The CNNs are a class of artificial neural networks generally used for analyzing visual images, which fundamentally is inspired by biological neural networks and how neurons are connected and are activate base off certain stimulus.

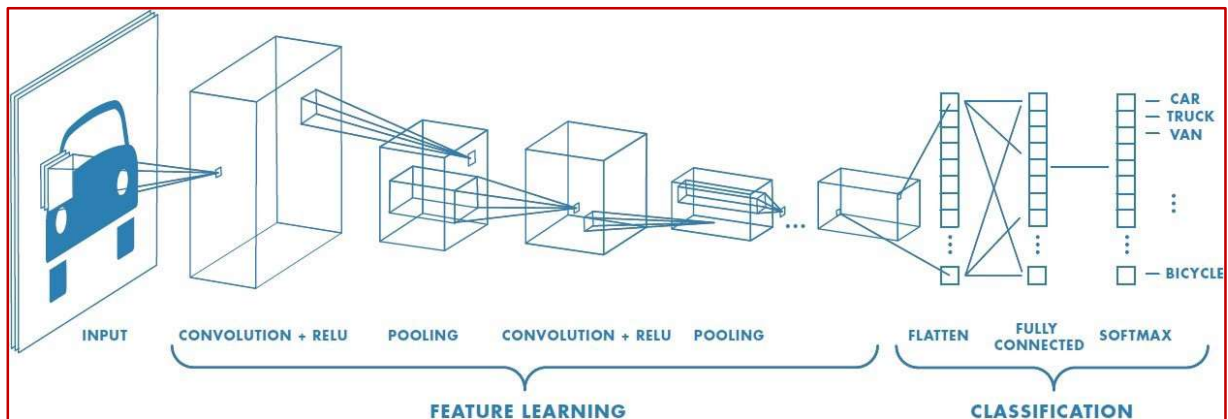


*Figure 2.4: McCulloch-Pitts Neuron Model in an ANN [26].*



*Figure 2.5: Feed-Forward Network With 7 Inputs, and 5 Nodes in the Hidden Layer and 1 Output [26].*

The ANNs consist of nodes connected as a network, therefore, each node having an input, which is scaled by a weight ( $w$ ), summed in a node and a bias added to the value. This value is then passed through a function which determines if it fulfils the constraints to output a significant value (functions such as the sigmoid, or ReLU function). The weights are trained using a loss function dependent on the required output, and this works in entirety similarly to linear regression. The changes made by convolutional neural networks is the change of the input, which is usually a visual image, and the imposition of convolution operations for feature extraction.



*Figure 2.6: Convolutional Neural Network [27].*

It is seen as in figure 2.6 that the convolutional neural network showcases similar architecture to the ANN in figure 2.5, where they are both feed-forward neural networks as in the cases showcased currently. The CNNs have been discussed, but solely on their reliance for the work of image classification, but what we need in this case is object detection. How then is a classification algorithm adapted for use in object detection? There exist two distinct groups of object detectors from deep learning techniques, the single stage detectors and the two stage detectors, where the former frames the detection as “complete in one step”, and the latter describes it as “coarse-to-fine” [10].

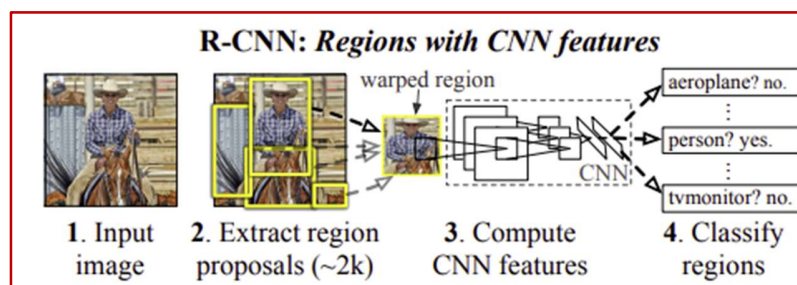
### **2.1.2.1. Two-stage detectors**

As the name suggests, two-stage detectors split the object detection tasks into two significant stages: (1) Region proposal; (2) Classification of proposed regions. In the initial phase, the detector tries to optimize which areas in the image are most relevant, i.e., Identify regions most likely to contain objects of interest. This is to highly influence the recall capabilities of the object detector. The second stage requires a deep learning model to classify the proposed regions into one of the described regions or background. The entire process is then optimized for localization of the original region proposals [28]. Object detectors in this category include the Region-based Convolutional Neural Network (R-CNN) family, Spatial Pyramid Pooling Network (SPP-net) and Feature Pyramid Network (FPN).

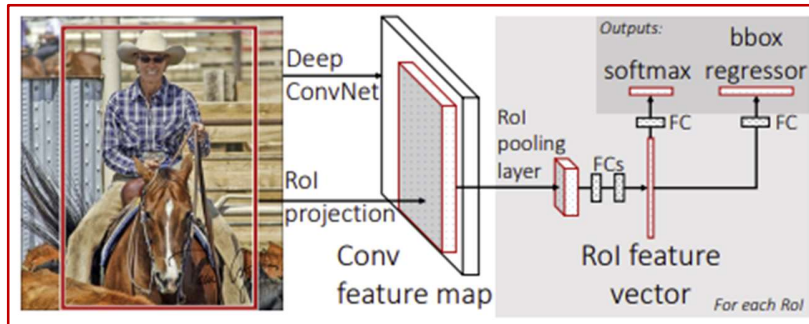


### 2.1.2.1.1. R-CNN family

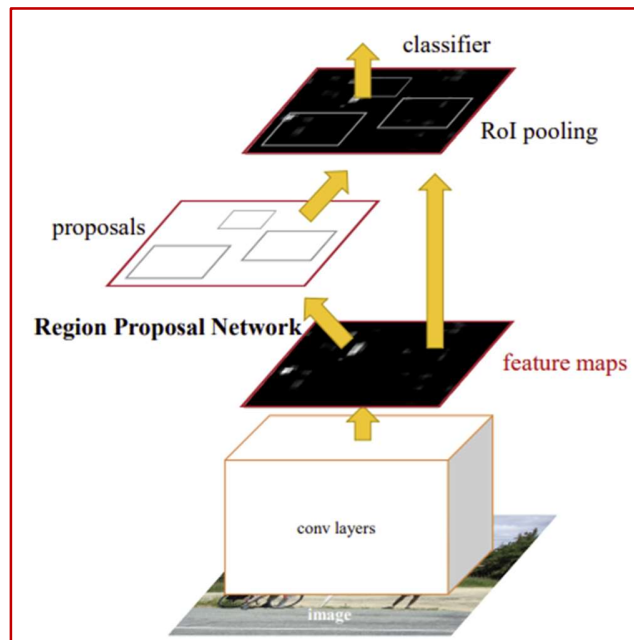
The R-CNN, also known as region-based CNN is a pioneering object detector in the field of deep learning developed by R. Girshick *et al.* (2014), with the first generation object detector working on a simple principle of the extraction of object proposals – object candidate boxes – by selective search, which is a region proposal algorithm designed to be fast with very high recall, and is based on computing the hierarchical grouping of similar regions based on image features such as color, shape, texture and size. Then each proposal is rescaled and sent into a CNN model pretrained on ImageNet to extract features. After extraction, a linear SVM classifier is utilized to classify the objects [10]. The mean average precision (mAP) of this model on COCO evaluations (COCO is a set of object detection evaluation rules where the mAP is calculated by averaging the average precision over 80 defined object categories and 10 intersection over union thresholds from 0.5 to 0.95) at intersection of union = 0.5 is 58.5%. Consequent updates to the R-CNN include the Fast R-CNN (2015), Faster R-CNN (2015), and Mask R-CNN (2017). Fast R-CNN is faster than the original R-CNN by Girshick who improved the training procedure by unifying the independent models for each regional proposal into a jointly trained framework. The speed was improved, but not significantly, consequently, Faster R-CNN was proposed by S. Ren *et al.*, as an end-to-end implementation with near real time detection at VOC07 mAP=73.2% and COCO [mAP@.5](#) = 43.7%. The most significant contribution made by the Faster R-CNN is the implementation of a Regional Proposal Network (RPN) that allows for low-cost region proposals. Mask R-CNN extends Faster R-CNN to pixel-level image segmentation by adding a third branch for predicting the object masks [29].



**Figure 2.7: R-CNN Overview. (1) Taking Input Image, (2) Extract About 2000 Region Proposals, (3) Use CNN to Compute Features for Each Proposal, (4) Classification Using Linear SVMs. [30]**



**Figure 2.8: Fast R-CNN Architecture. A Single Image as Input with Multiple Regions of Interest (RoIs) Sent into A Fully Convolutional Neural Network [31].**

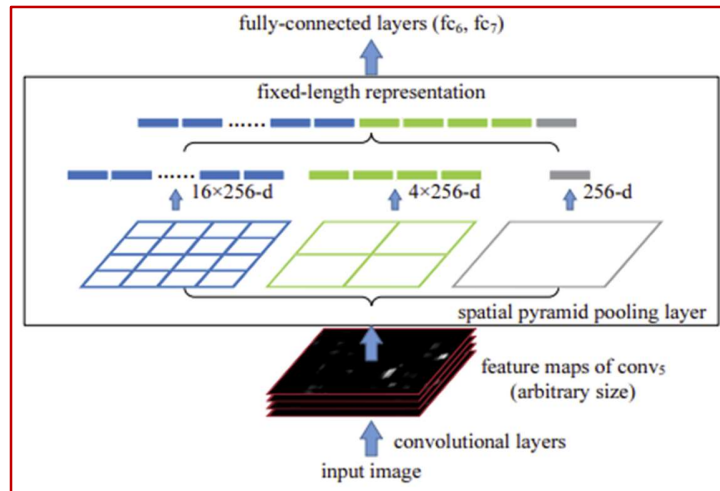


**Figure 2.9: Faster RCNN, A Unified Network for Object Detection [32].**

#### 2.1.2.1.2. SPP-net

He *et al.* were inspired by the idea of spatial pyramid matching and proposed SPP-net, standing for Spatial Pyramid Pooling Network, which introduced the spatial pyramid pooling layer, allowing for CNNs to generate fixed length representation of the images regardless of their initial size, or descriptions of the regions of interests without scaling. The SPP divides the feature map into an  $N \times N$  grid, for various values of  $N$ , making the model size invariant. Pooling is then performed on each cell of the grid to give feature vectors, which are then concatenated and fed into region SVM classifiers and bounding box regressors [28]. The SPP-net is up to and

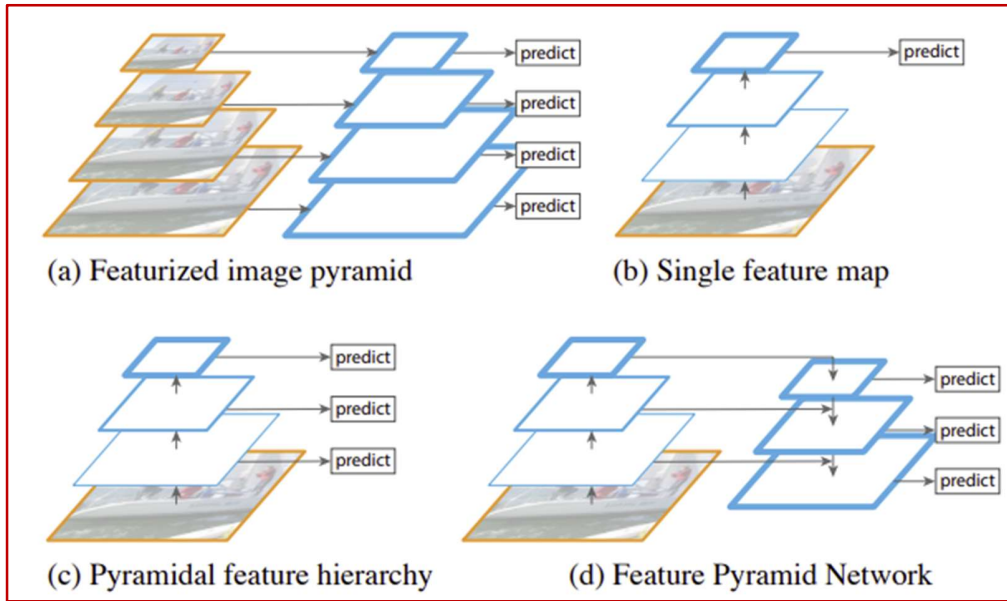
surpassing 20 times the speed of the original R-CNN, whilst maintaining similar accuracy at VOC07 mAP=59.2% [10].



**Figure 2.10: Network Structure with SPP Layer** [33].

### 2.1.2.1.3. FPN

FPNs, fully known as Feature Pyramid Networks are built upon image pyramids, with the purpose of creating a size invariant deep learning object detector. Lin T. *et al.* developed feature pyramids with marginal costs by exploiting the inherent multi-scale pyramidal hierarchy of deep CNNs [34]. Preceding the existence of FPNs, most deep learning object detectors ran detection solely on the network's top layer, due to the lack of localization benefits from the deeper layers of a CNN, therefore, a top-down architecture with lateral connections was developed in FPNs for building high-level semantics at all scales. A Faster R-CNN paired with an FPN can achieve [COCOMAP@.5=59.1%](#).



**Figure 2.11:** (a) *Building A Feature Pyramid with An Image Pyramid.* (b) *Detection Systems Utilizing Single Scale Feature for Detection.* (c) *An Option to Reuse the Pyramidal feature Hierarchy Computed by CNNs like a Featurized Image Pyramid.* (d) *Proposed FPN by Lin T. et al. [34].*

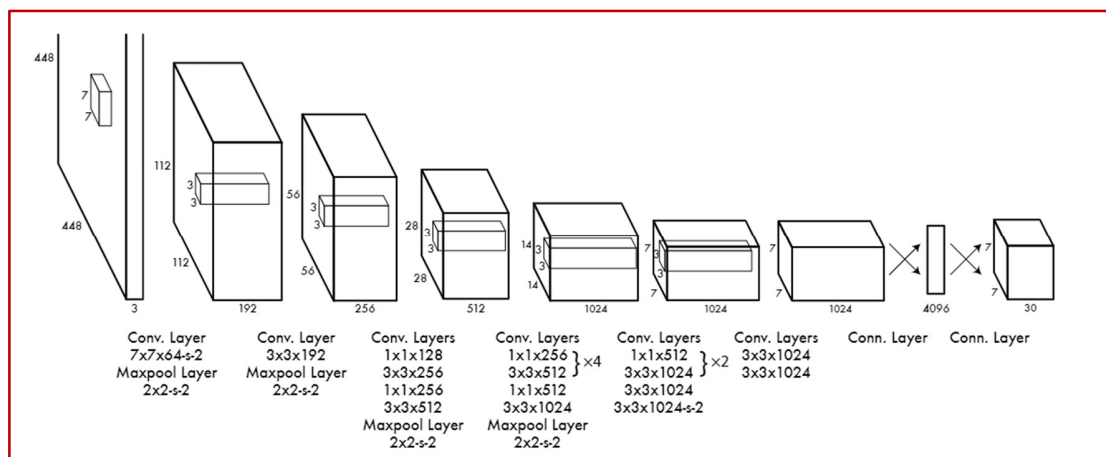
### 2.1.2.2. Single-stage detectors

Besides the top-of-the-line accuracy of regional proposal-based object detectors, their complexity and model sizes give them a significant detriment: They are computationally slow, or rather, are not able to achieve high enough framerates for use in computationally restricted devices. This poses a problem for real time applications at higher frame rates. Single-stage object detectors solve that problem, they are unified pipelines that are architected to directly predict class probabilities and bounding box offsets from images with a single feed-forward CNN that skips on regional proposal generation and feature resampling [16]. They generally consider all positions on the image as possible object positions [28]. Notable single stage detectors include OverFeat by Sermanet *et al.*, which was one of the first successful and early one-stage detector which performed object detection by using a DCNN classifier into an end to end fully convolutional object detector [28]. Later in 2015, YOLO (You Only Look Once) was another innovative proposal for object detection in the front of speed and utility. It was a unified, real-time framework which used a single convolutional network for the prediction of object classes and their locations, hence, simplifying the entire detection process into a regression problem and as an advantage compared to a two-stage detector like R-CNN, incorporates global information from the consideration of the whole image as compared to regions which increases invariability to errors introduced by backgrounds [35]. The YOLO family has spanned 5 generations with some various branch builds, which include significant changes that have served

to improve both accuracy and speed through the years. The original YOLO, albeit fast and revolutionary stuttered on efforts at detecting small objects grouped together, this was due to the dimensionally spatial constraints imposed on bounding box predictions, which also created a difficulty in generalizing for data subject to various spatial dimensions, such as new aspect ratios/configurations [14]. With a focus on these issues, the Single-Shot Detector (SSD) was proposed by W. Liu *et al.* introducing a multi-scale architecture which greatly improved the accuracy of single shot object detectors, while retaining their notable speed advantage [10].

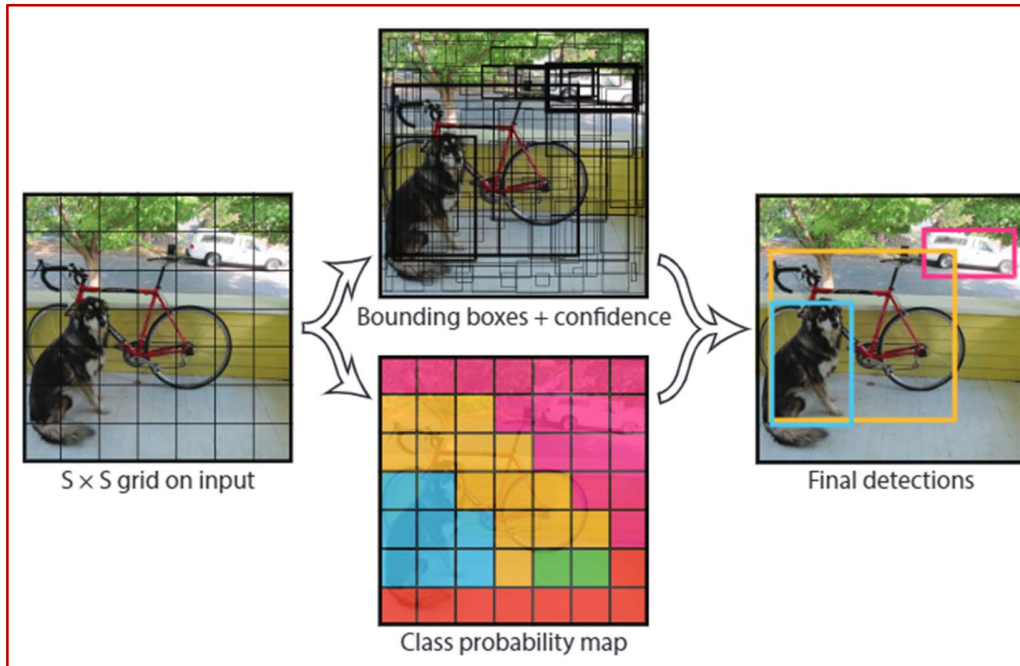
### 2.1.2.2.1. *You only look once (YOLO)*

The YOLO object detection model was introduced in May of 2016, by Joseph R. *et al.* with the aim of creating a unified architecture for optimized performance. The logic behind the algorithm framed object detection as a regression problem assigning spatially separated bounding boxes to their associated class probabilities. The acronym, Y.O.L.O, gives a significant meaning to the execution of its concept requiring the model to only see the image once and then proceed for predictions. The original YOLO's network architecture was inspired by GoogLeNet and designed to push the boundaries for fast object detection.



**Figure 2.12: Original YOLO Architecture with 24 Conv. Layers, Followed by 2 Fully Connected Layers. The inclusion of 1x1 reduction filters lowers the feature space of the preceding layer, followed by a 3x3 convolutional filter [36].**

YOLO is designed to take in an input image and split into a grid of NxN dimensions where each grid is responsible for the classification of one object, dependent on if the object centroid is within the grid. Each grid cell predicts bounding boxes with a respective confidence score.



**Figure 2.13: The YOLO Model Solving Detection as A Regression Problem Dividing Each Image into an  $N \times N$  Grid Where Each Grid Cell Predicts  $B$  Bounding Boxes [36].**

The original YOLO was vastly improved by works of its successors - YOLOv2 and YOLO 9000, by increasing recall and localization while maintaining classification accuracy (Comparing YOLO to R-CNN). This was done by focusing on novel concepts that serve influence it's improvement such as: Batch normalization, high resolution classifiers, convolution with anchor boxes, dimension clustering, multi-scale training and the replacement of the base network with darknet-19 for faster classification [37]. YOLOv3 is released in 2018 by Joseph R. *et al.* with little upgrades to make for a better experience and modality, and a new darknet backbone, the darknet-53 which performs better than the darknet-19 and more efficient than the corresponding Resnets at the time (50, 101, 252) [38]. There have existed other branch versions from the model family such as PP-YOLO, YOLOX, PP-YOLOE, YOLO-Z and so on, the two other main branch versions include YOLOv4 and YOLOv5 which does not have an equivalent peer-reviewed publication but has been released since May of this year, 2022.

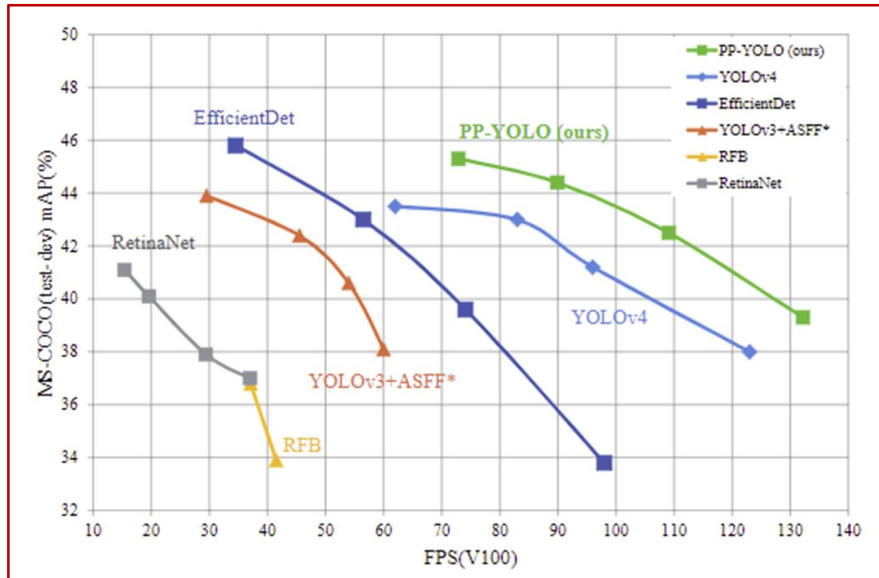


Figure 2.14: Comparison of PP-YOLO with Other State of the Art Detectors [39].

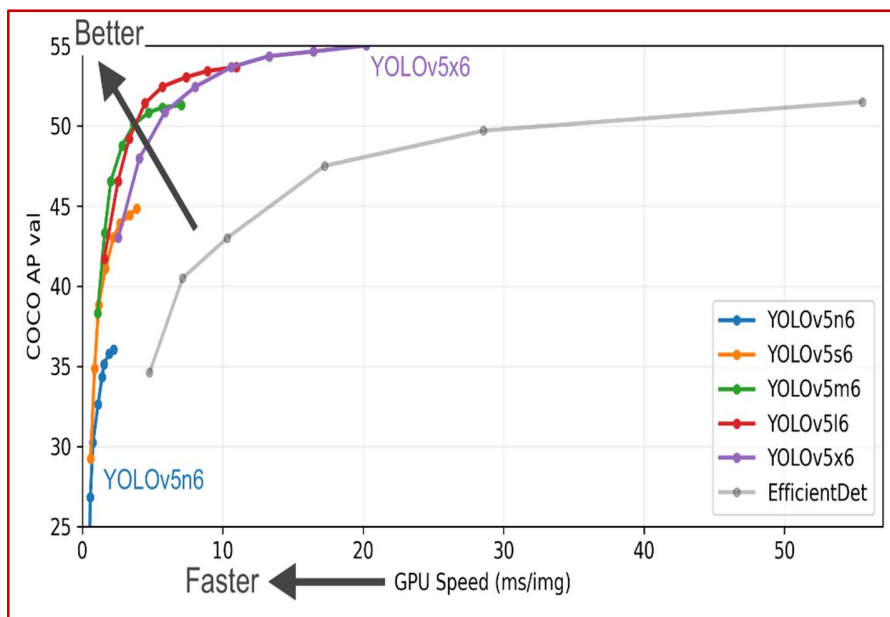


Figure 2.15: Comparisons of Different Version of YOLOv5 with EfficientDet [40].

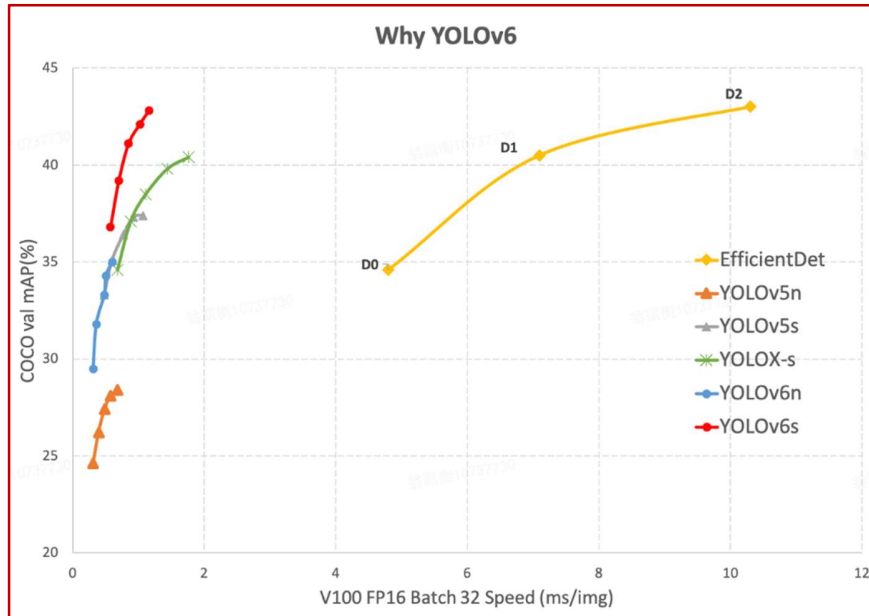


Figure 2.16: Comparisons of Different Object Detection Models with YOLOv6 [41].

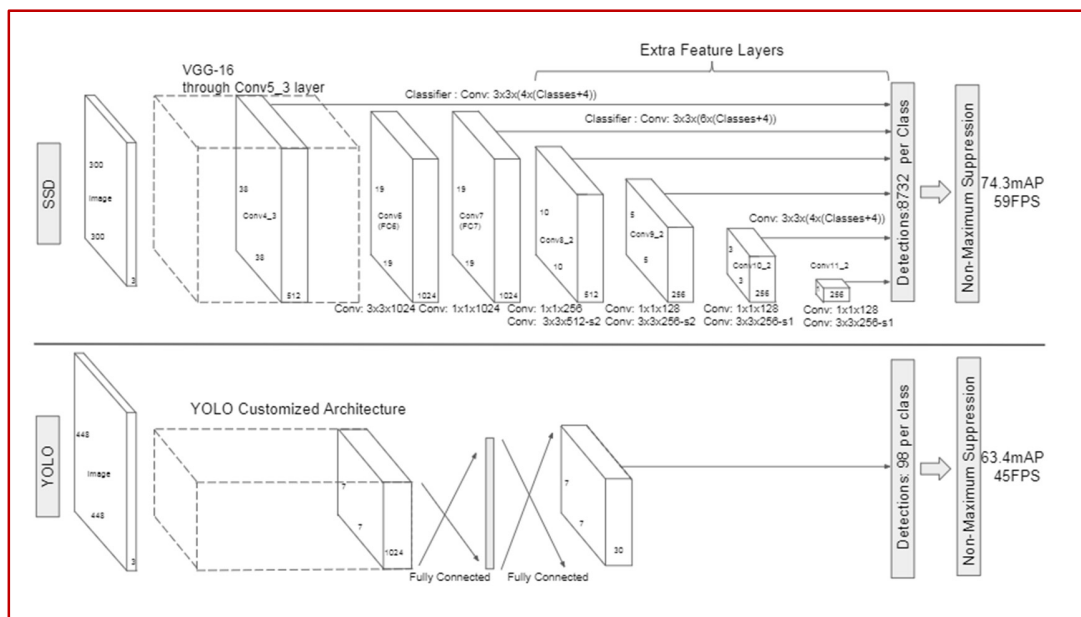
As seen in the graphs showcased in figures 2.14 and 2.15 respectively, we can compare the state of the relatively recent version of YOLO with EfficientDet and extrapolate some inferred knowledge. It is seen in the former image that YOLOv4 and PP-YOLO outperform YOLOv3 and EfficientDet in terms of speed, with very little compromise in accuracy when compared to EfficientDet once again. With YOLOv4 attaining peak mean average precision of >43 at just above 60 frames per second. Comparably, EfficientDet attains around 46 mAP with the 35 FPS range. In the latter figure, it is evident that various versions of YOLOv5 display greater mAP values at even higher speeds compared to EfficientDet. The massive improvement in the generations of the YOLO model object detectors is factored in the evolution of the field, as many improved methods culminate in achieving state-of-the-art results. There also exists YOLOv6 [42], preliminarily released in a couple days as I write this, which I would like to explore and potentially research base on the project use case. The figure 3.16 also showcases a comparison between the newly released YOLOv6, the previous generation and other comparable models.

#### 2.1.2.2.2. Single shot detector (SSD)

The SSD, standing for single shot multibox detector is a single stage method that discretizes the output space of bounding boxes into a set of default boxes with varied aspect ratios and scales dependent of the located feature maps. Proposed by Wei *et al.* (2016), the single shot multibox detector works in various ways to account for

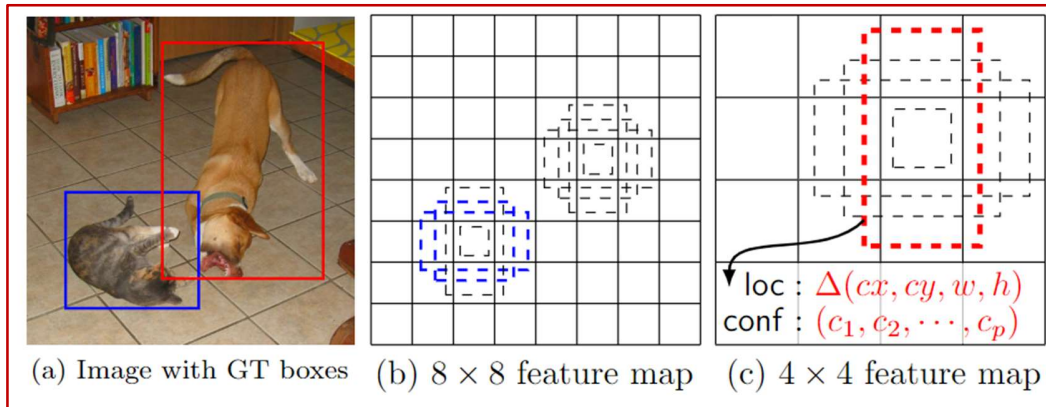


detection of objects with varied sizing, by automatically adjusting the default boxes to better match the object space and combining predictions from multiple feature maps developed from convolutional layers at different image resolutions [43]. The contribution of this object detector was a state-of-the-art detector faster and significantly more accurate at the time – YOLO; High detection accuracy for scaled detections and balancing better the speed vs accuracy trade off.



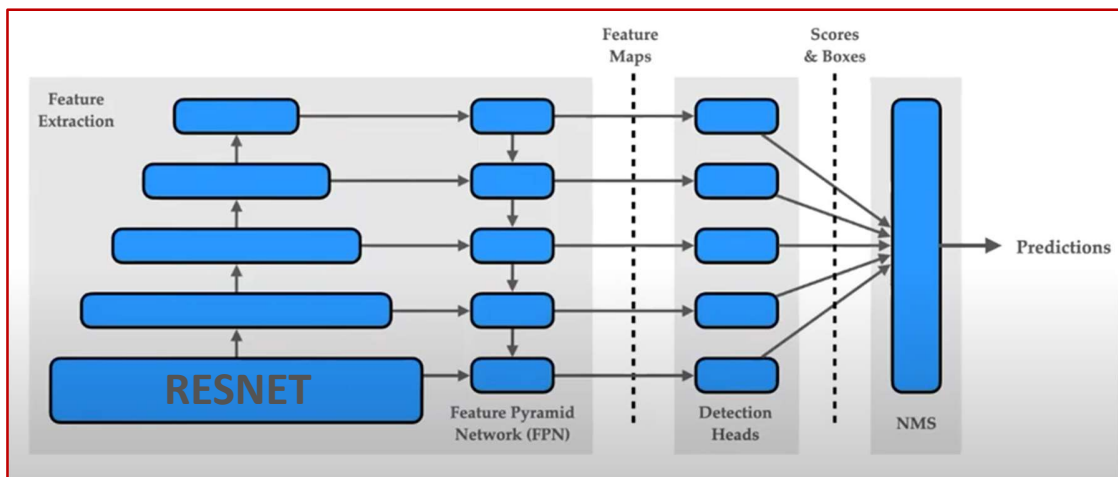
**Figure 2.16: A Comparison Between SSD and YOLO [43].**

The Figure 2.16 compares the general architectures of the SSD and YOLO, and almost immediately we notice a significant difference in detections per class, which is a result of the SSD structure with multiple aspect ratios for detection boxes, and classifiers from every hierarchy of convolutional layer, outputting a feature map. The SSD architecture includes multiple convolutional feature layers loaded on top of a truncated base network like VGG, RESNET or Inception, which decrease in size progressively to allow for detections at differing scales [43]. In the original paper. “SSD: Single Shot MultiBox Detector”, Liu W. *et al.* utilized the VGG-16 as a base network for detection.



**Figure 2.17: SSD Framework** [43].

The SSD object detectors accept an image that passes through the base CNN classifier, where a small set of default boxes at various aspect ratios pass through each location at different scales from the different scaled convolutional layers, where detections at various scales are computed. The SSD serves as a framework for a base classifier and possibly other architectural configurations and can be modified to enhance the detection process. Major implementations of the SSD range from adaptation with VGG, Resnet, Mobilenet and Inception classifiers. Other modifications include the attachment of a Feature Pyramid Network (FPN) to the feature map outputs to serve as a feedback mechanism to share significant features at different scales with each other. An example of an implementation of this technique is with the SSD Resnet50 FPN V1.



**Figure 2.18: SSD Resnet FPN Architecture** [43].

Figure 2.18 displays the SSD method implemented with a ResNet backbone, and a feature pyramid network attached from the output feature maps to distribute detected features as a top-down architecture. With the ResNet architecture implemented, which solved the problem of network degradation from the formerly more popular VGG classifier, the detectors were able to be built with deeper structures that improves classification results [44].

### **2.1.2.3. Review of deep-learning object detectors in tasks of similar nature to screw detection**

In the paper, “Screw detection for disassembly of an electronic waste using reasoning and retraining of a deep learning model”, by Gwendolyn F. *et al.* (2021). The authors described using the Faster RCNN with Inception v2 pretrained on the COCO dataset and taught to learn the features of crosshead screws through transfer learning. This method takes them through multiple augmentations and implementation of an ontology that manages the knowledge representation between the relationship and the physical structure of the LCD TV the screws are fastened to [12]. 109 labels of screws were obtained from 24 images, with training done for 4.5 hours, with testing done on 16 images with 67 new instances. The precision and recall attained were at 91.8% and 83.8% percent respectively, attaining an experimental accuracy of 78% in the last, most suitable scenario (6 scenarios were tested). The detector was able to detect screws at a variation of orientations slightly different from orthogonal to the axis of the camera. This implementation performed favorably, but falls short due to the dataset size limitation, and the variation of model selected, which performs positively with respect to accuracy, but falls short of real time speeds.

A paper by Erenus Y. *et al.* (2019) explores detection for automated disassembly processes by implementing a DCNN coupled with a proposal generation algorithm – the Hough transform for candidate detection. The detection process is done by an integrated image classifier, which was coupled using the Inception V3 and Xception models to form a singular image classification model, utilizing both their strengths. 300 images of hard drives were collected with over 10000 samples as the dataset, with the images partitioned 2:1 for training to test set respectively. The training was done on an intel i7-4770 CPU @ 3.40GHz, with 16GB of RAM and a GTX Titan X GPU. The system was able to achieve high accuracy and real time detection, with an AP of 80.23, and when compared to YOLOv3 detector, with an AP of 66 [45].

## CHAPTER THREE

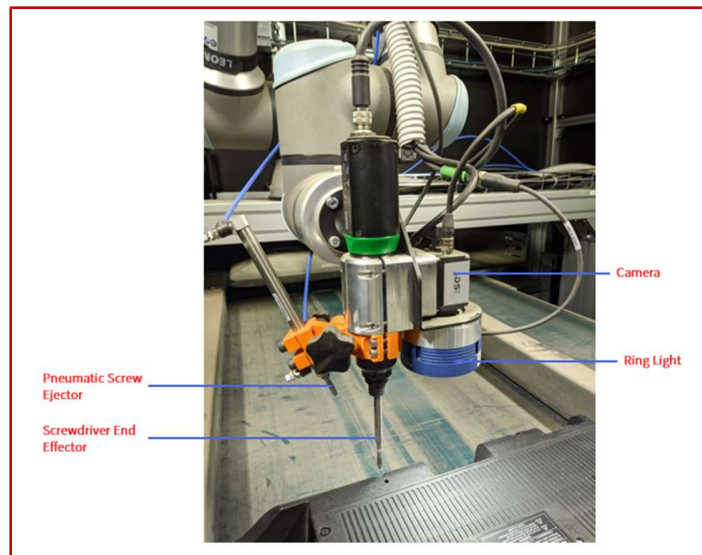
### 3. MATERIALS AND METHODS

#### 3.1. INTRODUCTION

Accomplishing the task of detection, tracking, and unscrewing of screws in the automated recycling process of electronic waste requires an application of the theory and literature analyzed by selecting tools which best fit the requirements to achieve the project needs. The various stages of the project, such as data gathering, data augmentation, data labelling, model environment preparation, model training, model evaluation, object tracking, and consequent unscrewing all require their respective equipment and methods. In this chapter, an exhaustive overview into the equipment and methods will be analyzed to give an insight into the whole engineering and scientific approach, to provide a means of reproducibility and verification of the data gathered thereof.

#### 3.2. EQUIPMENT

The workspace required for the processes of model training, evaluation and possible deployment will be highlighted. For model training and evaluation, a convenient workstation outfitted with enough memory and compute will be needed, preferable graphics processing compute. For possible deployment, the model will need to be deployed on a working robotics system. This section gives information on the respective work-cell.



*Figure 3.1: Robot Workcell.*

### **3.2.1. End-effector**

The end effector of the robot is composed of a camera and screwdriver unit connected, with the tool tip at the end of the screwdriver.

#### **3.2.1.1. Camera**

The camera used for the perception step of the project is the IDOS UI-3070SE-C-HQ, which is based on a CMOS sensor with a global shutter system. It is a 3MP camera outputting at a resolution of 2056 x 1542 pixels, with an exposure time of 0.018ms – 999ms, and maximum frame rate of 134 frames per second. The camera is connected by USB Type-C for data transfer and power-in. This camera is suitable due to its high use in industrial robotics applications, providing with it a wide range of software and drivers for frameworks like ROS for robotics applications and VISP for visual servoing applications.



*Figure 3.2: IDOS UI-3070SE-C-HQ Camera.*

#### **3.2.1.2. Screwdriver**

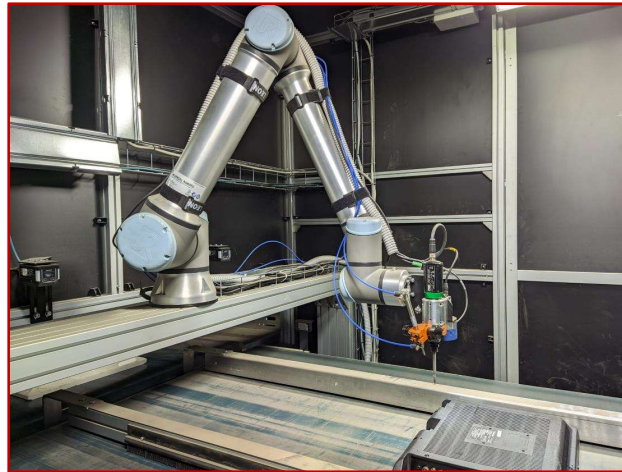
The robot setup includes a screwdriver that is set to unscrew screws given detection and placement of the robot in the proper position. The screwdriver is accompanied with a pneumatic screw remover which blows off the screws attached to the screwdriver to allow the robot to unscrew other screws on the FPD TV.

#### **3.2.1.3. Ring-Light**

An industrial robot ring-light is connected to the end-effector contraption to illuminate the objects in view during the detection process. The ring-light is strobed during the detection to output greater luminosity whilst conserving energy and preventing overheating during the robot movement.

### 3.2.2. Industrial Robot Arm

The Ur10e collaborative robot is an industrial robot developed and distributed by universal robots with a load capacity of 12.5kg, and a range of 1300mm making it a suitable candidate for a variety of industrial applications. It is also proposed as an OEM robotics system with a 3-position enabling programming terminal.



*Figure 3.3: Ur10e Robot Outfitted with Peripheral Devices for Unscrewing Task*

### 3.2.3. Workstation

The computing setup is based off a Linux workstation with the given hardware:

- CPU: Intel i7-HK
- GPU: Nvidia Quadro RTX 4000 with 8gbs of GDDR6
- RAM: 16gbs of DDR4

#### 3.2.3.1. Software tools

Various developer tools were used to bring this project into fruition, and these tools cover most significant aspects of the projects: Dataset preparation and model development.

##### 3.2.3.1.1. Dataset preparation

To prepare the dataset, tools such as LabelImg, Roboflow API, and libraries like OpenCV, NumPy, and Pyautogui were used to label, augment, and manipulate the data as needed, whilst preparing the required

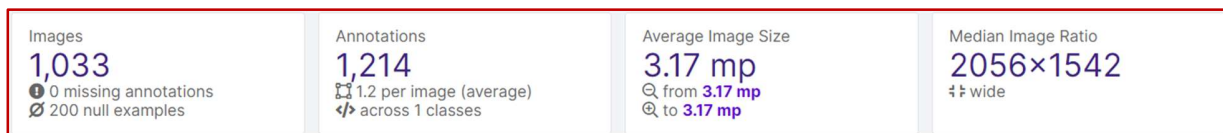
annotations files for each model format: Pascal VOC for SSD, and the YOLO txt format for the YOLO model family.

### 3.2.3.1.2. *Model Development*

The training for SSD was implemented using the object detection API for Tensorflow 2, with the recommended Nvidia CUDA and CUDnn libraries. The YOLO models were trained and deployed using the Pytorch library.

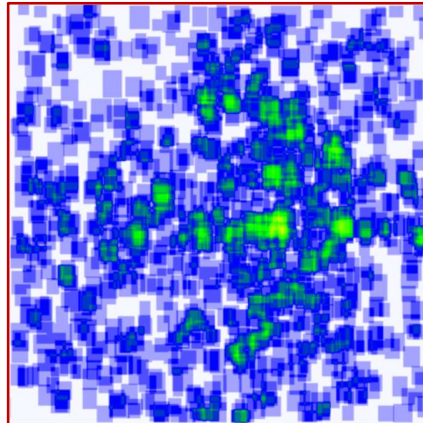
## 3.3. DATA

Data in form of images is required to train CNN based detectors, where the requirement for the dataset is set for generality and robustness of the to-be-trained model. This gives a requirement of fulfilling contextual obligations of the task – Where Object detection of screws is done on the internal and external of FPD panels, and in an industrial setting, the images acquired are specific still shots of internal and external components of FPD panels, at the given resolution of the IDS camera (2056 x 1542). The data is gathered from the workcell in figure 3.3, where each image is stored as a conveyor moves multiple FPD TVs under the camera attached to the ur10e robot. The original images have the given analytics:



**Figure 3.4: Original Dataset Stats.**

The figure 3.4 shows the original images being 1033 in number, with a combination of 1214 screw labels across all images. The dataset comes with 200 null examples, which will give the model a means to negative mine background and false positive detections.

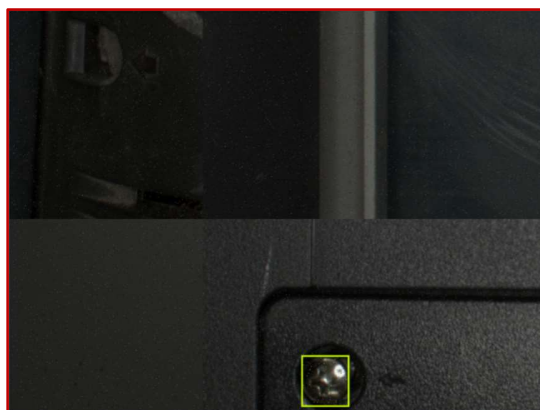


*Figure 3.5: Image Label Heatmap.*

The figure 3.5 shows the heatmap of the single label – screws – in an increasing intensity of white, to blue to green. The heatmap displays that most labels in the image fall into the center of the image with a skewed distribution towards the right of the image.

### **3.3.1. Labelling**

The Image data was gathered using a camera of the same type indicated in section 1.2.1.1. CAMERA. The images were labelled, producing corresponding annotation files in Pascal VOC format for the SSD Resnet Implementation and YOLO format for the YOLO implementations of the object detection task. During the course of the project, the LabelImg [46] annotation tool and the inbuilt roboflow ([Screw Detection - v21 YoloV6 \(roboflow.com\)](https://roboflow.com)) annotation tools were used to label multiple images of display panels.



*Figure 3.6: Labelled Image.*



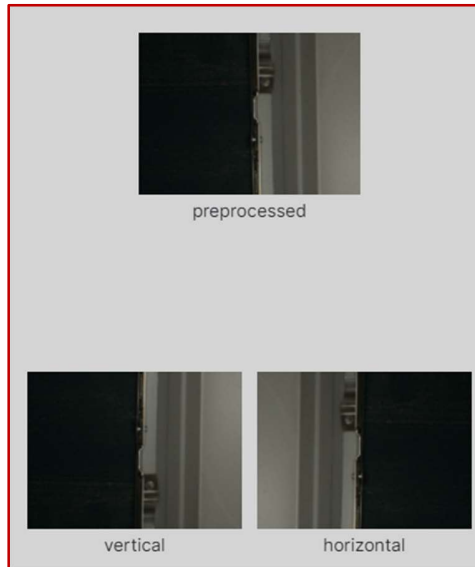
### 3.3.2. Augmentation

The images were augmented to multiply the dataset and give generality to the trained models. First the images were resized to a resolution of 416 x 416 pixels, then tiled into 2 x 2 grids to help for the detection of small objects such as screws as researched in the small object detection guide by roboflow [47]. Various iterations of augmentations were tested to determine what characteristic of dataset was optimal for model training. The best augmentations were complex enough (noise, mosaic) to prove challenging to the algorithm, hence learning important features deeply, but not so complex to be unlearnable. Simple data augmentations (image flipping, reorientation) constantly underperformed, reaching performance saturation earlier. Morphing image details slightly gave good performance.

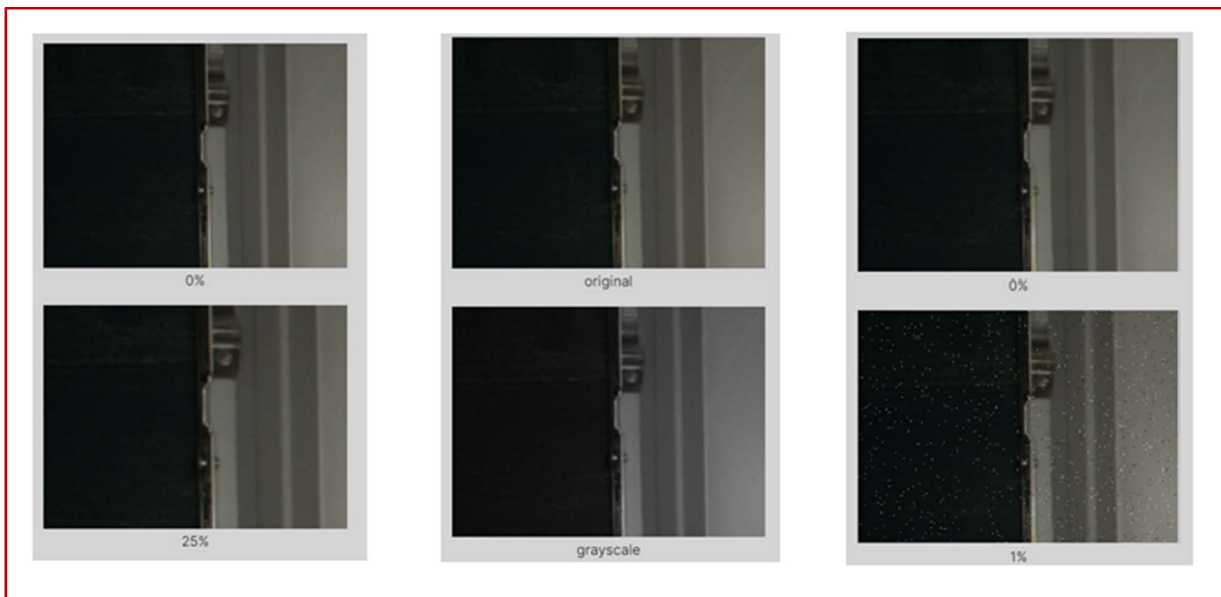


*Figure 3.7: Original Image Being Resized and Tiled.*

The input image is resized and tiled for compute and model inference reasons, given the limited compute ability of the available setup, having high resolution images passed through the model makes for large computations, which do not necessarily improve the results significantly. The tiled images then go through numerous other augmentations arbitrarily to foster a larger dataset. The given augmentations are noise, cropping, horizontal and vertical flipping, greyscaling and mosaicking.



**Figure 3.8: Horizontal and Vertical Flipping of Images**



**Figure 3.9: From Right to Left, 1. Cropping by 25%. 2. Greyscaling. 3. Noise.**



*Figure 3.10: Images Generated from Combined Augmentation and Mosaiced.*

### 3.3.3. Splitting

The generated dataset consisted of 10,858 images which was split into 3 sets of training, validation, and testing set. The data was split according with respect

## 3.4. TRAINING

The dataset is trained on 6 distinct models: The SSD Mobilenet 320x320, SSD Resnet 50 FPN v1 640x640, YOLOv5s, YOLOv5l, YOLOv6s and YOLOv6s. The training was done with an average of 200 epochs, with some models reaching saturation of accuracy earlier. The models were trained under various hyper parameters which do not largely affect performance, but training time, and mainly adjusted due to computational restrictions. 3 models were shortlisted for evaluation and further testing based off their performance and the project needs: The SSD Resnet 50 FPN v1, YOLOv5s and YOLOv6s.

### 3.4.1. SSD Resnet 50 FPN v1

The SSD Resnet 50 FPN v1 [48] is trained using the TensorFlow object detection API [49], with the pipeline configuration file configured as follows:

*Table 3.1: Pipeline Configuration for SSD Resnet 50 FPN v1*

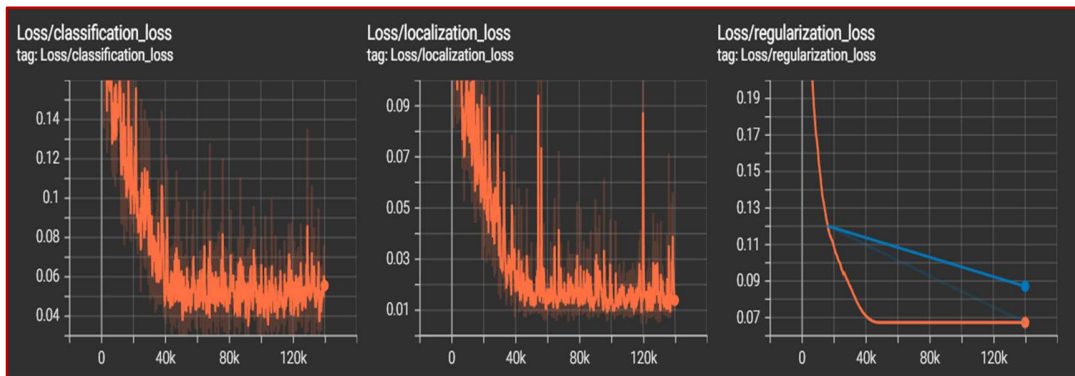
Setting	Configuration
Number of Classes	1
Fixed Shape Resizer	640x640
Batch Size	16

Step Size	140,000
Score Converter	Sigmoid

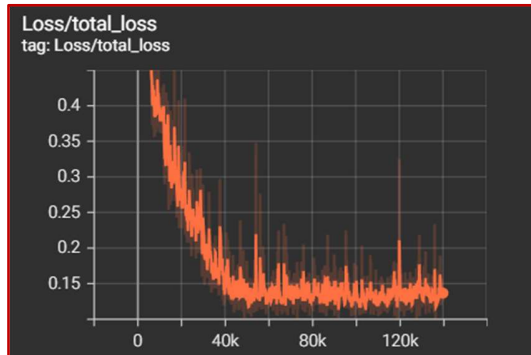
**Table 3.2: SSD Resnet 50 FPN v1 Model Parameters.**

Parameter	Configuration
Size	640
mAP val (0.5:0.95)	34.3
Speed(ms)	39
Params (M)	56.9326
FLOPS (B)	1178.7

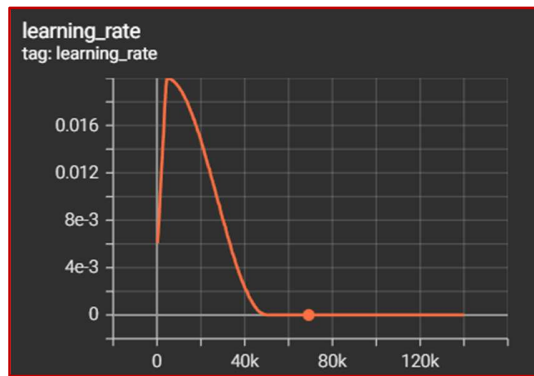
The training overwent 140000 steps, with the requirement of having to go through 679 steps to complete one epoch on account of the batch size of 16. The equivalent number of epochs trained at is 206 epochs. The training data gathered, considering loss metrics is as follow in the images of figures 3.11 – 3.13.



**Figure 3.11: Loss Metrics: 1. Classification Loss 2. Localization Loss 3. Regularization Loss.**



*Figure 3.12: Total Loss.*



*Figure 3.13: Learning Rate.*

The figures 3.11 to 3.13 show the training process of the SSD model, and from the images, it can be inferred the loss reached its minimum at 50,000 steps, where the model was unable to pick up any more features to learn. The associated evaluation loss could not be fully determined as it is regularly assessed during training but limited by the computing already allotted to training.

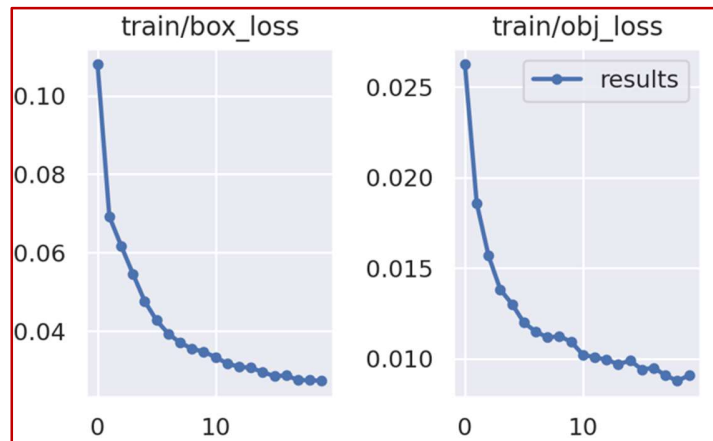
### 3.4.2. YOLOv5s

The YOLOv5s [40] is a convenient model that was selected due to its size and possible speed benefits, whilst retaining great metrics compared to models of the same period. The table 3.2 shows the model parameters.

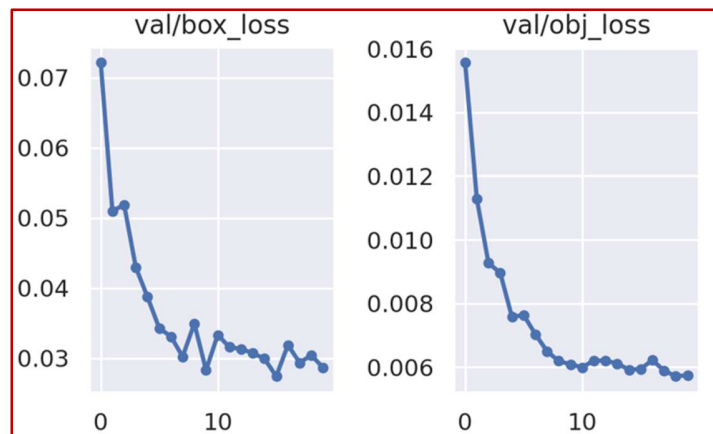
*Table 3.3: YOLOv5s Model Parameters.*

Parameter	Configuration
Size	640

mAP val (0.5:0.95)	37.4
mAP val (0.5)	56.8
Params (M)	7.2
FLOPS (B)	16.5



**Figure 3.14: Training Loss: 1. Localization Loss 2. Classification Loss.**



**Figure 3.15: Validation Loss: 1. Localization Loss 2. Classification Loss.**

The table 3.2 showcases the parameters of the YOLOv5s which lands on the lower end of the spectrum in terms of predictive performance in the YOLOv5 family but does sufficiently well in the computational speed criteria. A focus which determined the selection of this model due to the high-speed intense robot movement predicted

which would have the end effector moving at 2cms per second, needing at least 60fps for the model to be applicable to the given situation.

The figures 3.14 and 3.15 showcase the training saturation of the model parameters at the 20<sup>th</sup> epoch, where training beyond did not help improve the performance. The model was also trained on different augmented datasets, at different epochs, and against the larger models of the same family, but this trained model outperformed all other instances.

***Table 3.4: YOLOv5s Custom Model Training Parameters.***

<b>Parameter</b>	<b>Configuration</b>
Image Size	640
Batch Size	16
Workers	8
Weights	YOLOv5 s
Epochs	20

The table 3.3. displays the tunable parameters selected as the outcome of the preferred yolov5 s model trained.

### **3.4.3. YOLOv6s**

The YOLOv6 [42] was shortly released during the project and has not been explored extensively by the computer vision community, but is open source and available for use, which is grounds for inquisition.

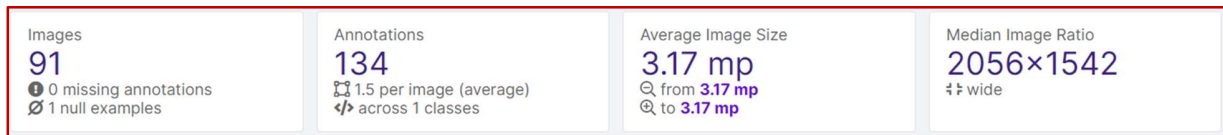
***Table 3.5: YOLOv6s Model Parameters.***

<b>Parameter</b>	<b>Configuration</b>
Size	640
mAP val (0.5:0.95)	43.1
Params (M)	17.2
FLOPS (B)	44.2

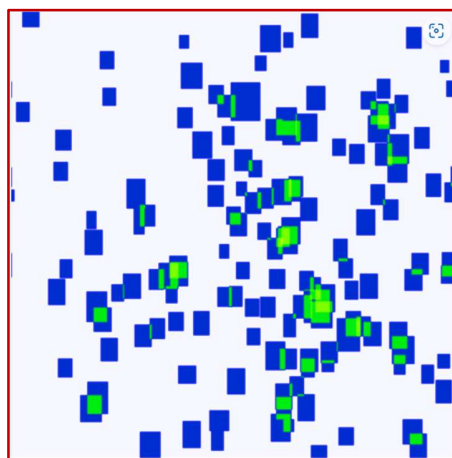
Image Size	1000
Batch Size	32
Epochs	300

### 3.5. EVALUATION

After training, the models are first validated and tested on the initial validation and test splits during the training, and then for fair evaluation, tested on a separately selected set sourced from the same recycling plant. This new test set consists of 91 images all at the original resolution.



*Figure 3.16: Test Dataset Stats.*



*Figure 3.17: Test Images Label Heatmap.*

From figure 3.16, it is known that the 103 images in the test dataset contain 134 annotations of screws. Giving a ground truth of 134 label boxes to detect for the single screw class. The models will be valuated using generalized criteria in the computer vision field, such as mAP.



### 3.5.1. Evaluation Metrics

Exploring a plethora of options at our current disposal to evaluate the performance of the derived models to streamline decision making, this project settles on the use of the Average Precision (AP) metrics which are a canonical metric in the field. This metric evaluates the accuracy of object detection models by approximating the area under a curve with respect to the relationship between precision and recall at various confidences [50]. To explore the use of the average precision metric, various terms and expressions need to be established to move forward and cement the judicial criteria.

At the most basic level, evaluating the performance of an object detection model is based off the correctness of the model, i.e., measuring how many times the model correctly pinpoints and classifies an object in an image. Important terms to be considered when defining correctness:

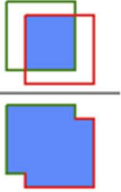
- *True Positive (TP)*: Correct available detection made by the model.
- *False Positive (FP)*: Erroneous detection made by the model.
- *False Negative (FN)*: Incorrect non-detection made by the model.
- *True Negative (TN)*: Correct unavailable detection made by the model.

The terms above require the definition of correctness and incorrectness as related to object detection, and a common way to do this is with the intersection over union (IoU) which is a measurement based off the Jaccard index, a coefficient of the similarity between two sets of data [50]. The Intersection over union is yet another term important towards our exploration of the performance of our developed model but requires more nuance towards its description.

#### 3.5.1.1. Intersection over union

The intersection over union is a metric which evaluates the degree of overlap between the ground truth box and the predicted box, where the ground truth box is the real position of the object in the image, and the predicted box is the output position of the image given by the model. This metric can be mathematically expressed by the area of intersection between the predicted bounding box and the ground truth box, and the total area occupied by both boxes [51].

$$IoU = \frac{\text{area}(gt \cap pd)}{\text{area}(gt \cup pd)}$$

$$IoU = \frac{\text{area of overlap}}{\text{area of union}}$$


**Figure 3.18: Intersection Over Union** [51].

The intersection over union is within the ranges of 0 and 1, and can therefore be evaluated in given these conditions:

- TP is a detection for which  $IoU > \alpha$  (thresholding value that determines if the detection is correct or not).
- FN and FP are respectively non-detections and detections where  $IoU < \alpha$ .

### 3.5.1.2. Precision and recall

After defining the means of gauging correctness of our model, we consider a more empirical approach by comparing the correct detections to all detections, i.e., Precision; Then comparing all correct detections to all ground truths, i.e., Recall. Based off this, Precision is the degree of exactness of a model to evaluate only relevant detections, while recall is the ability of a model to detect all ground truths present. Where:

$$P = \frac{TP}{TP + FP} = \frac{TP}{\text{all detections}}$$

$$R = \frac{TP}{TP + FN} = \frac{TP}{\text{all ground-truths}}$$

**Figure 3.19: Precision and Recall** [51].

A good model will without a doubt possess both high precision and recall. A perfect model will possess perfect recall and precision, i.e., Precision = 1, and recall = 1, but this is obviously not possible now.

### 3.5.1.3. Precision-recall curve

The precision-recall curve is a plot of precision and recall at varying values of confidence, but what is confidence? Well, confidence relies on threshold values just as the IoU described previously. The threshold, determines which IoU is regarded as a positive detection or not, therefore, raising the IoU will restrict the positive detections within a small confinement of higher IoU values, and remove detections which do not achieve that threshold. This will be of great influence on the precision of the model, but negatively impact the recall of the model by missing out on close detections that barely meet the cut-off score set by the threshold. A good model with steady performance will undoubtedly retain high precision and recall when the confidence is varied.

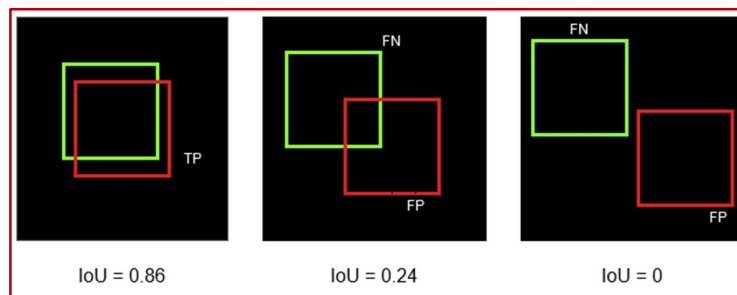


Figure 3.20: Different IoU Values [51].

As discussed, and observable from the figure 3.20, with different IoU values, the threshold chosen will determine what detection is chosen as a positive detection or not. For example, at a threshold of 0.5, the first detection of Figure 3.20 is regarded as a positive detection, while the remaining are regarded as negative. At a threshold of 0.9, even the first image is discarded as a true negative, but at a threshold of 0.2, the first two objects are detected as positive detections.

### 3.5.1.4. Average precision (AP) and mean average precision (mAP)

The average precision at a given threshold is the area under the precision recall curve at that given threshold.

$$AP@{\alpha} = \int_0^1 p(r) dr$$

**Figure 3.21: Average Precision** [51].

The average precision is measured at various threshold, but some specified thresholds are standards of the field. The AP@50 and the AP@90 mean the average precision calculated from the area under the precision recall curve at the confidence scores of 50% and 90% respectively. The mean average precision is an average calculated from the combination of all average precisions based on the model's ability to detect multiple classes, the mathematical description is described in figure 3.22.

$$mAP@{\alpha} = \frac{1}{n} \sum_{i=1}^n AP_i \quad \text{for } n \text{ classes.}$$

**Figure 3.22: Mean Average Precision** [51].

As this project focuses solely on detection one class of object in various images – Screws, the mean average precision and average precision are therefore interchangeable metrics. Canonically, the average precision can be calculated based off multiple thresholds with the [AP@.5](#) and [AP@.5:.9](#) being the most common metrics. Focusing further on the specifics of the project, the average precision metrics can also be specified based off the accuracy of detecting objects of certain sizes within a given image.

```

Average Precision (AP) :
AP                % AP at IoU=.50:.05:.95 (primary challenge metric)
APIoU=.50         % AP at IoU=.50 (PASCAL VOC metric)
APIoU=.75         % AP at IoU=.75 (strict metric)
AP Across Scales:
APsmall           % AP for small objects: area < 322
APmedium          % AP for medium objects: 322 < area < 962
APlarge           % AP for large objects: area > 962

```

**Figure 3.23: Average Precision at Various IoUs and Scales**[51].

Figure 3.23 shows the average precision at various IoUs, where the primary challenge metric is usually the AP at  $\text{IoU}=0.50:0.05:0.95$ , and for this project, due to the requirements for small object detection, a focus on the ability for the model to detect AP small objects at an area of less than 32 pixels square.

## CHAPTER FOUR

### 4. RESULTS AND DISCUSSIONS

#### 4.1. INTRODUCTION

Screw detection is a peculiar task with its own intrinsic characters that make creating/training/finetuning an object detection model a detailed task. To achieve a high level of accuracy with the model, many considerations need to be made with respect to the project's context. The context of the object in question considers:

- Object Presence: Where objects have their typical environments like cars in garages or ships on the sea, the screw object is contextualized by its appearance in this industrial environment by its placement on PCB boards.
- Object Appearance: The screw object appearing on PCBs, whether internally or externally, showcases different appearances in luster, color – sometimes black, silver, or brown from rust – reflectivity and so on, that need to be considered when the dataset is created for ample training.
- Object Location: Localizing the likelihood of screws as by methods like visualizing heatmaps shown in figures 3.5 and 3.14, where the screws skew towards the center, with a shift towards the upper right corner. Also, notable, the screws are placed at certain areas on the PCBs, like possibly next to arrows indicated for easy assembly/disassembly, and generally away from potential circuit lanes to avoid short circuiting, or on plastics joints.
- Object Size: A prime aspect contextualizing screws and the importance of designing a proper dataset, as the screw sizes are usually in the range of AP small detections. The most obvious management of this property is priming the data for small object detection by tiling, resizing, cropping and model definition by adjustment of anchor box parameters.

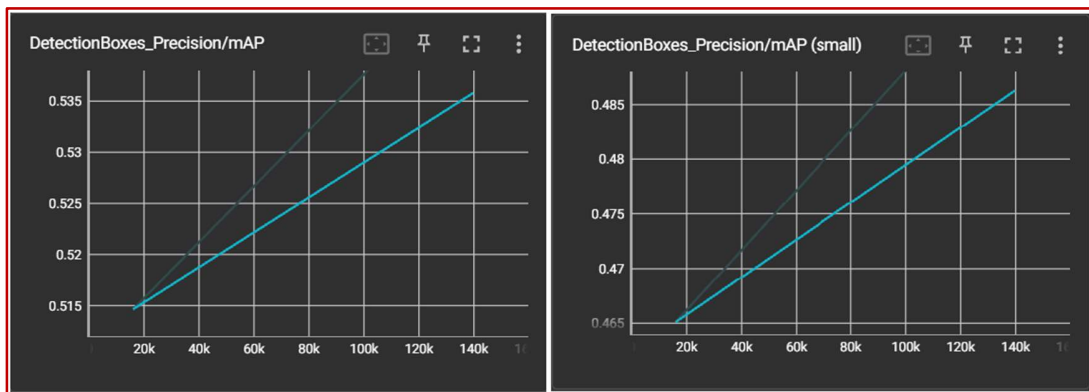
These are a few of the properties taken into consideration to well define the context of the object detection project, as there is wide agreement on the importance of the role of context during object detection in the computer vision circles [52]. As this determines what the model defines as a screw, we also must set the criteria for the terms of a successful screw detection:

- Ground truths are screws detected with the full cross heads perpendicular to the axis of the camera.
- Partial detections are positive detections for the model, but not a primary concern

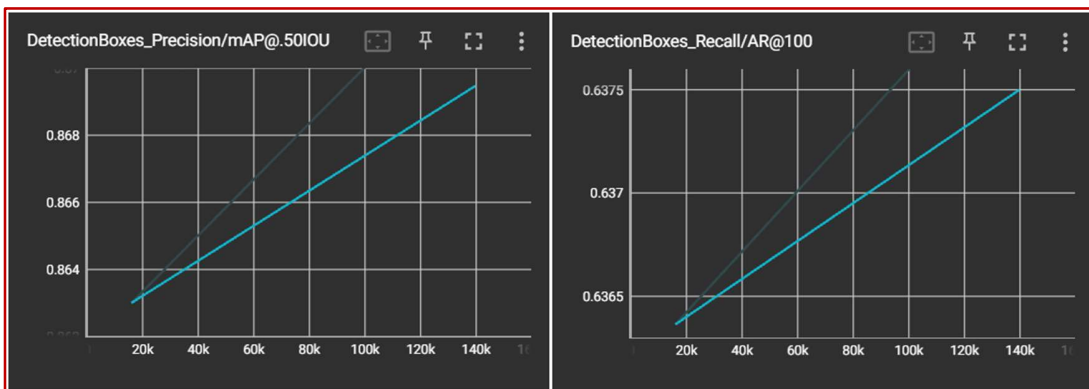
- Screws obstructed from view are positive detections, but not a primary concern

#### 4.2. SSD RESNET 50 FPN V1

The SSD Resnet's performance was highly outdated and severely overshadowed by the performances of the newer generations of single stage object detection models. The age of the model was apparent when thrust in the more general testing area, and with little to show in terms of speed and accuracy.

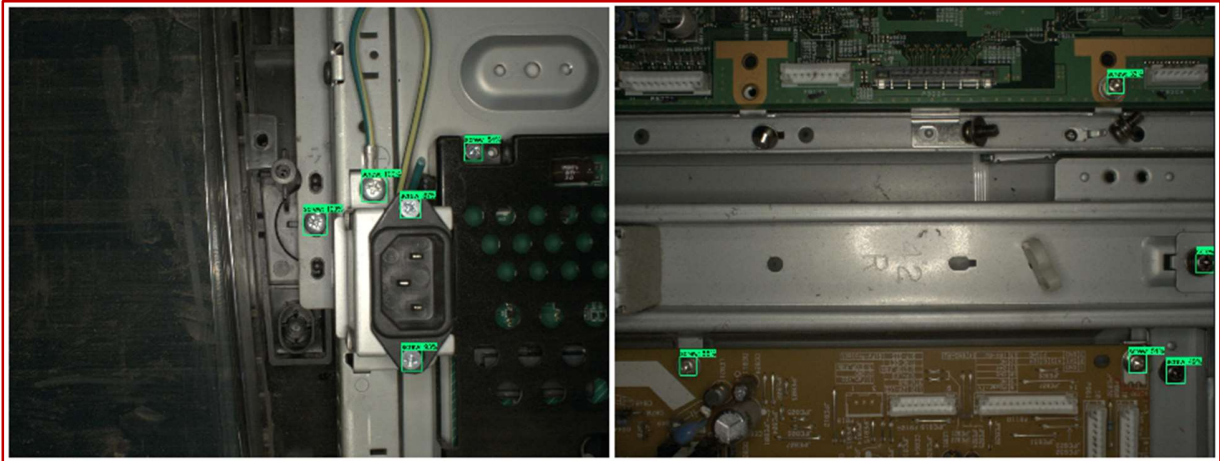


*Figure 4.1: Precision@0.5:0.95 and Precision(Small) Respectively for the SSD Model*

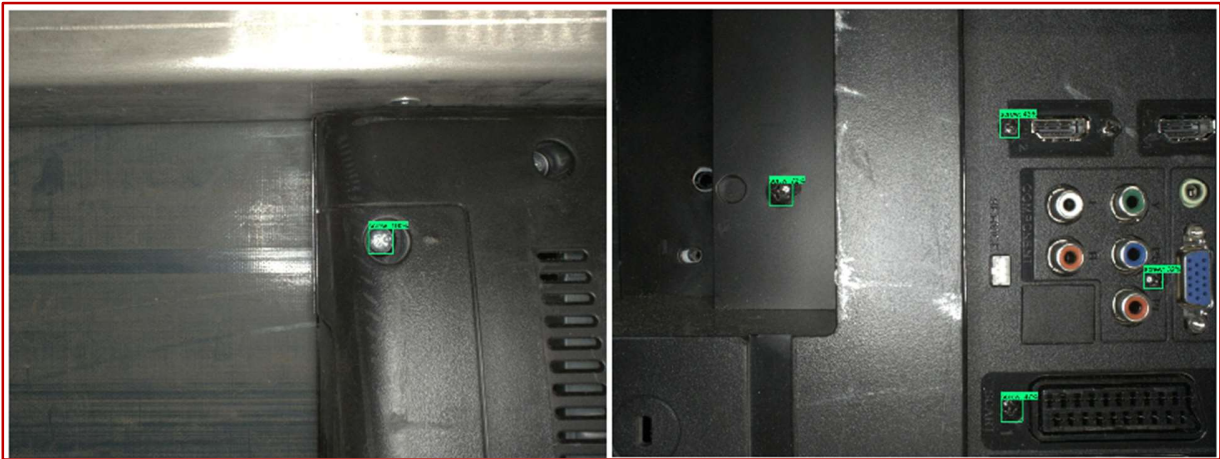


*Figure 4.2: Precision@0.5 and Recall Respectively for the SSD Model*

Figures 4.1 and 4.2 show the model performance on the accompanied validation images during training. The SSD model is more dependent on image size and as a result, very limited in terms of performance, as large image sizes limit batch size to 16.



*Figure 4.3: SSD Resnet Detections*



*Figure 4.4: SSD Resnet Detections*

From the images, it is observable that the SSD Resnet performs well with screws directly perpendicular, and badly with screws on the externals of the FPDs or flushed with the surrounding material in texture and color.



*Table 4.1: Table of Primary Properties for SSD Detections*

<b>Properties</b>	<b>Values</b>
True Positives	110
False Positives	7
False Negatives	20

The table 4.2 shows a good performance with respect to the lack of false positives, but a low recall with ground truths being completely missed. This should be improved using more training data. The SSD Resnet is more size variant as compared to the other more advanced detection models, and retains high performance on increase image size, but crippling performance to mere single digit frames per second.

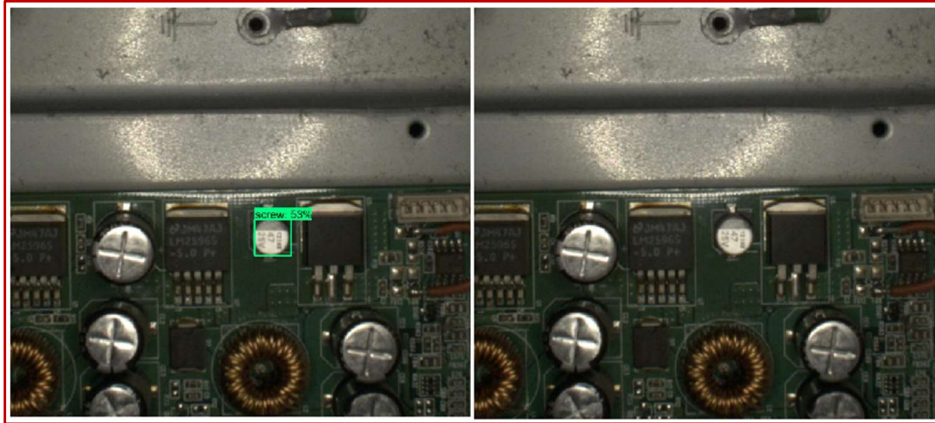
*Table 4.2: Results for SSD Detection*

<b>Properties</b>	<b>Values</b>
Precision	0.750
Recall	0.541
mAP@.5	0.750
mAP@.5:.95	0.419
Preprocess Time (ms)	>5
Inference Time (ms)	>10

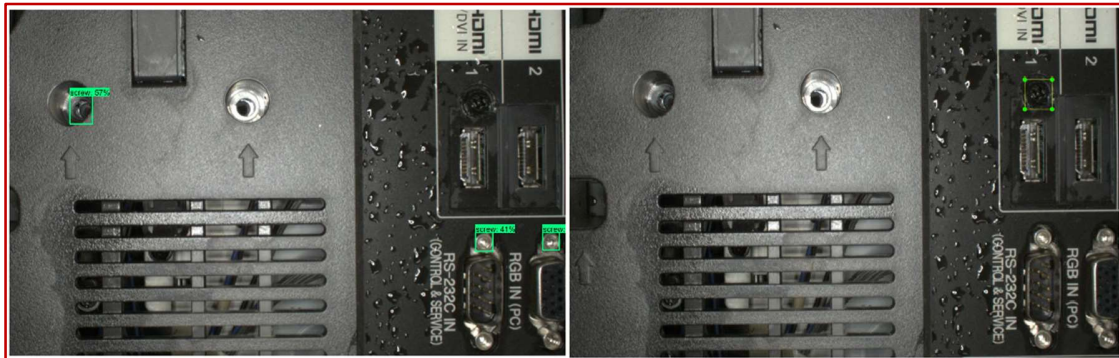
The SSD model performs well but is not sufficiently fast for deployment on the screw detection work cell.

#### **4.2.1. False Positives**

The SSD model gives low false positives, but still manages to miss the mark most times with a general low confidence on even a considerable amount of ground truth screws detected. This possess a problem when trying to filter by gauging the confidence value because even ground truths will be lost, halving, or sometimes crippling the performance of the model.



*Figure 4.5: SSD Resnet False Positive Detections (Detections vs Ground Truths)*

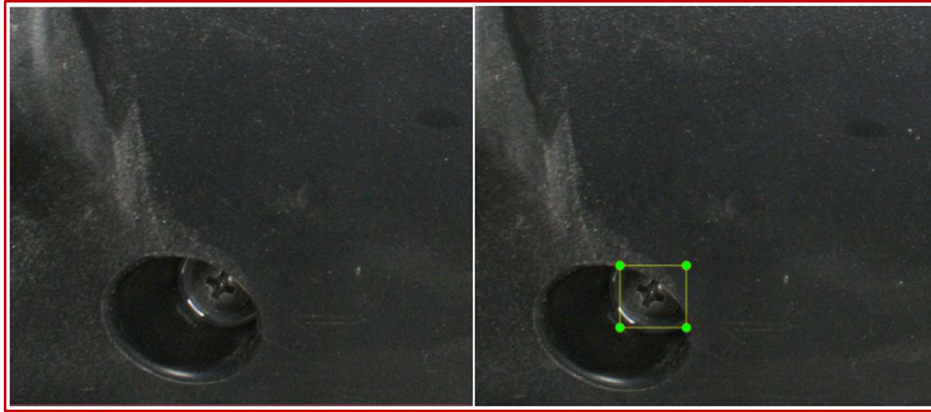


*Figure 4.6: SSD Resnet False Positive Detections (Detections vs Ground Truths)*

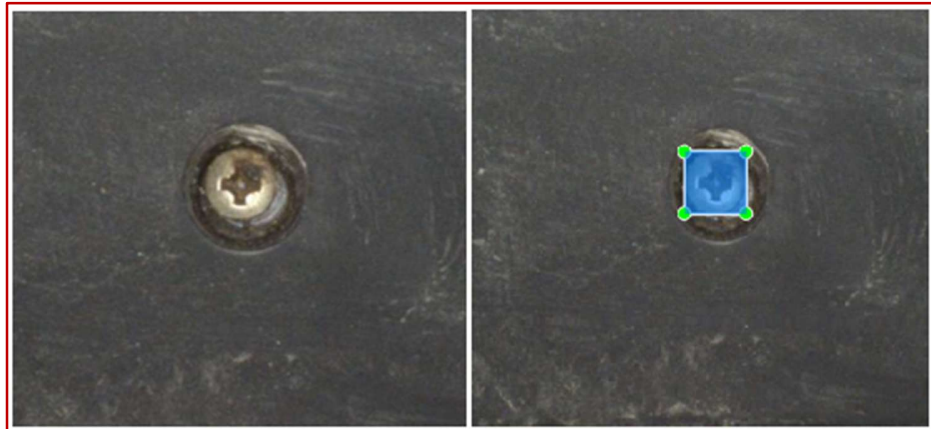
Figures 4.5 and 4.6 display the difficulties the model is having a hard time with differentiating holes, capacitors, and various objects with screw like features from screws.

#### 4.2.2. False Negatives

The model skips on a lot of possible ground truths, mostly in the images of the external FPD components, and obscured screws in holes. This is a big disadvantage due to the nature of FPD structures with slant edges and holes that will be occluded when lit in certain perspectives.



*Figure 4.7: SSD Resnet False Negative Detections (Detections vs Ground Truths)*



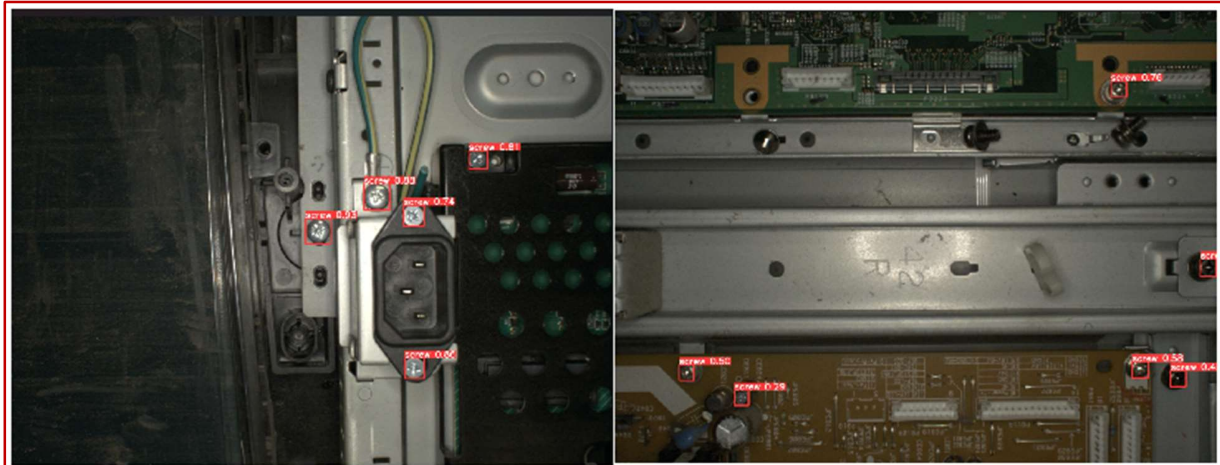
*Figure 4.8: SSD Resnet False Negative Detections (Detections vs Ground Truths)*

The figures 4.7 and 4.8 show the lack of detections on parts of the images where the latter models succeed effortlessly. Detection on the exteriors seem to be a problem for the SSD Resnet model. For the latter image, the screw is located exactly at the center of the image, and yet, the SSD model cannot detect it. It is an obvious display of the lack of generality the other models can achieve, probably owing to the advances in architecture that the model lacks.

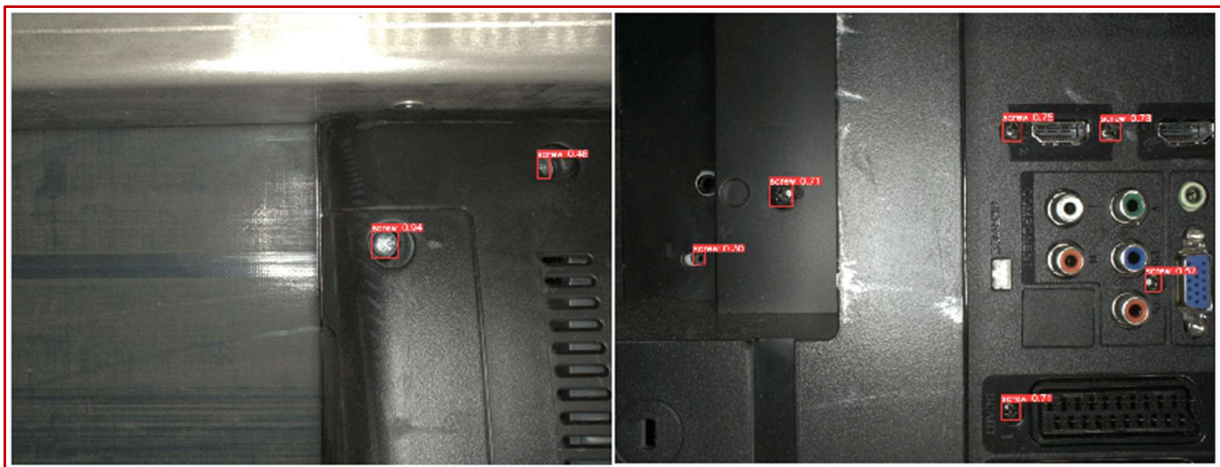
### 4.3. YOLOv5

The performance of the 5<sup>th</sup> generation of the YOLO object detection model family does extremely well in detection tasks, particularly with the peculiar nature of screw detection as a difficult detection task. The training

time whilst comparably smaller, achieves comparably, and sometimes better inferences than other models, as well as performs the detections at real time at high framerates.



*Figure 4.9: YOLOv5 Detections*



*Figure 4.10: YOLOv5 Detections*

The YOLOv5 model was able to detect screws based off different contexts associated to the task, with infrequent misclassifications. Based off the final evaluation test set, the model gave the following values of true positives, false negative and false positives – Note, true negatives do not conceptualize themselves in object detection due to the nature of the detection proves. A true negative would be a correct non-detection, but if truly the detector correctly detects nothing, or isn't able to define with certainty that the object does not exist in the image is debatable.

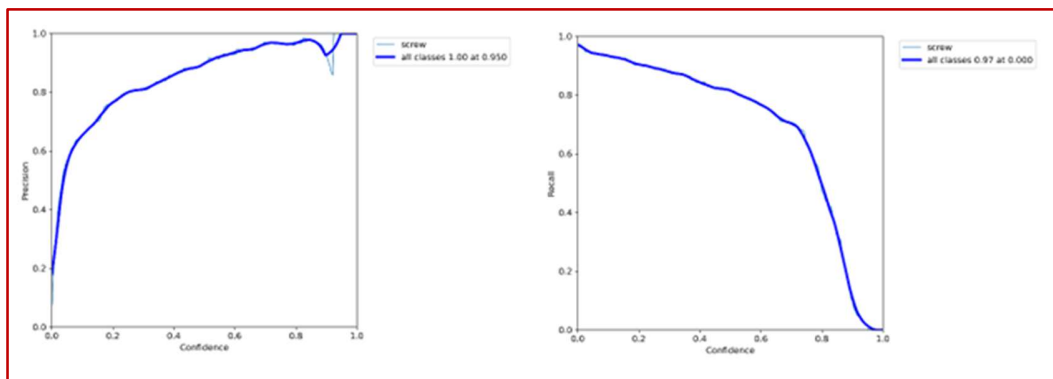
**Table 4.3: Table of Primary Properties for YOLOv5 Detections**

Properties	Values
True Positives	123
False Positives	26
False Negatives	11

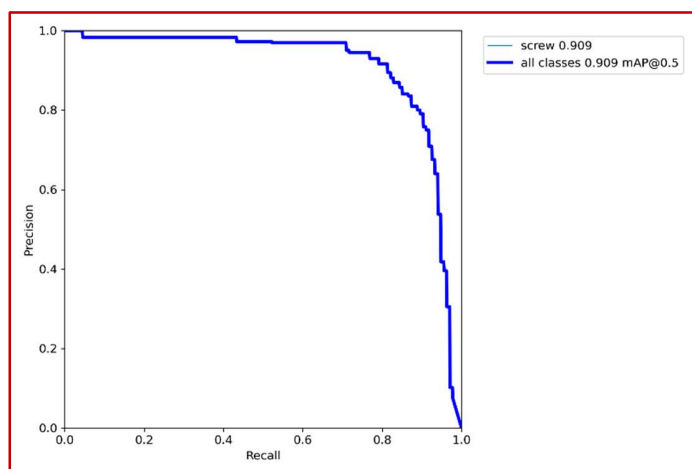
The table displays the number of true positives, false positives, and false negatives associated with the current detection task. Although the total number of screw labels adds up to 134 instances, the model detects screws at the fringes of the selection criteria for this project.

**Table 4.4: Results for YOLOv5 Detection**

Properties	Values
Precision	0.915
Recall	0.805
mAP@.5	0.909
mAP@.5:.95	0.536
Preprocess Time (ms)	0.3
Inference Time (ms)	3.6
Non-Maximum Suppression Time (ms)	1.1



**Figure 4.11: Precision and Recall Graphs**

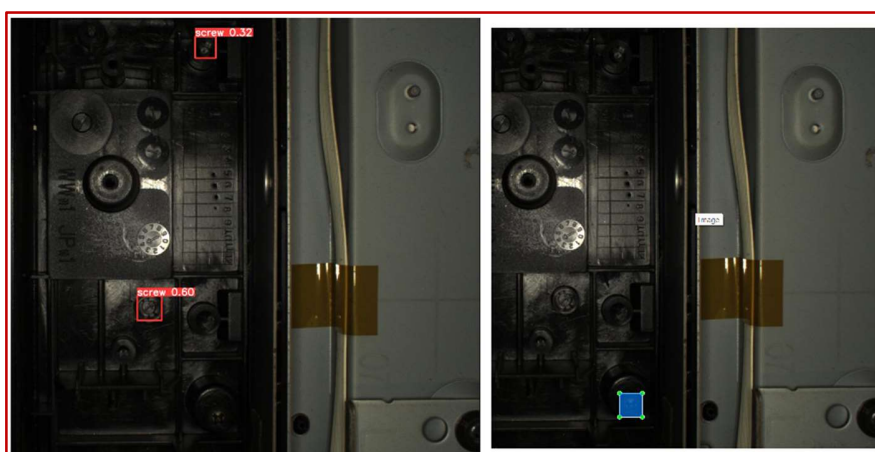


**Figure 4.12: Precision-Recall Curve**

The object detection model works very well, but we need to explore where the failings of the model are and explore the options dedicated towards improving the current performance.

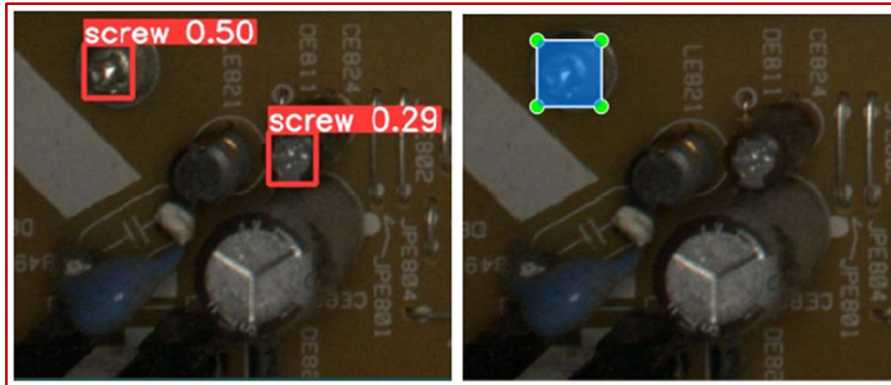
### 4.3.1. YOLOv5 False Positives

The YOLOv5, shown by table 4.3 tends to display False Positive detections. Although most of these detections can be alleviated using an IoU of a determined empirical value, it is still notable to explore which detections it makes and understand why these detections are made to improve the results.



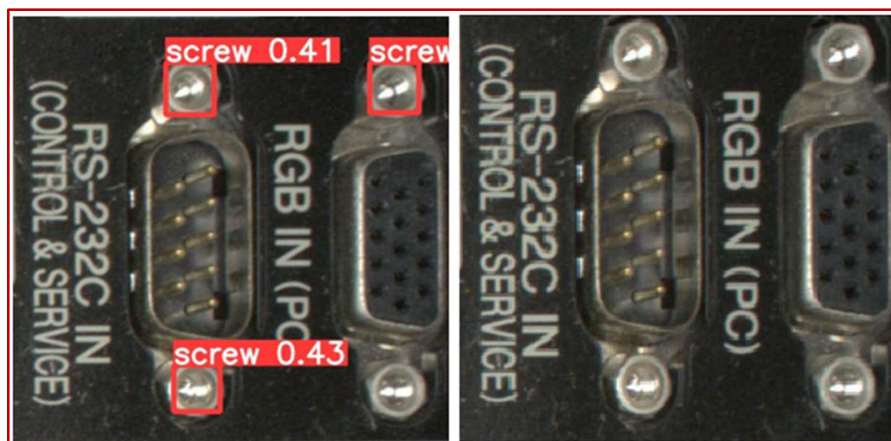
**Figure 4.13: YOLOv5 False Positive Detections (Detection VS Ground Truth)**

The Figure 4.13 shows two false detections that could be easily curbed by increasing the IoU threshold significantly, but the higher values false detection would mean the loss of possible true detections under the threshold of 0.6.



*Figure 4.14: YOLOv5 Positive Detections (Detection VS Ground Truth)*

Figure 4.14 shows a detection easily removed by a threshold over 0.3, which signifies a capacitor and not a screw.

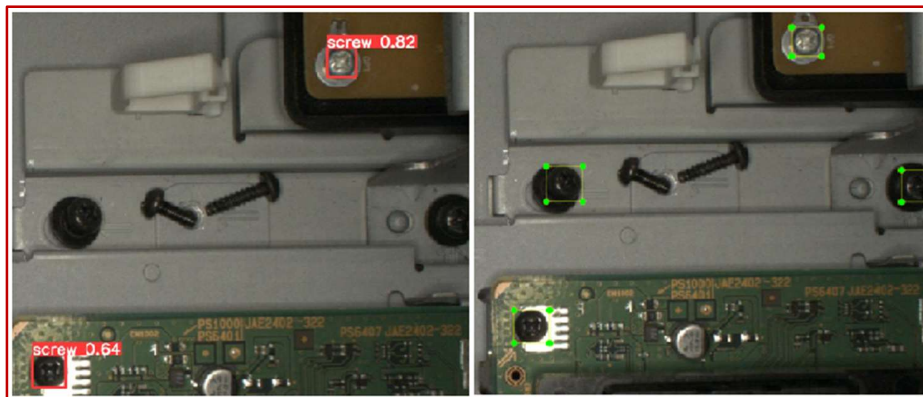


*Figure 4.15: YOLOv5 Positive Detections (Detection VS Ground Truth)*

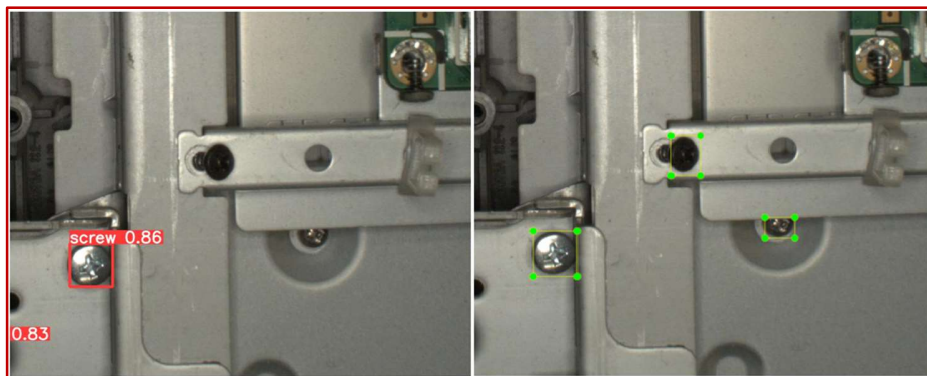
Figure 4.15 displays false positives just over under the threshold of .5, which could potentially lose some true positive detections if the threshold were raised any higher. The balance will be struck as described previously between precision and recall, as some losses will be made in quantitative detections as opposed to the quality of the detections available.

### 4.3.2. YOLOv5 False Negatives

The YOLOv5 possesses very high recall and hardly misses screws during detection. The screws missed are often of a rare color as opposed to the size of the dataset, or weirdly positioned on the image surface. The detector easily detects screws with the cross directly perpendicular to the camera axis with no problem and does better than the YOLOv6 at detecting screws skewed by part unscrewing, or screws huddled together around a simple region.



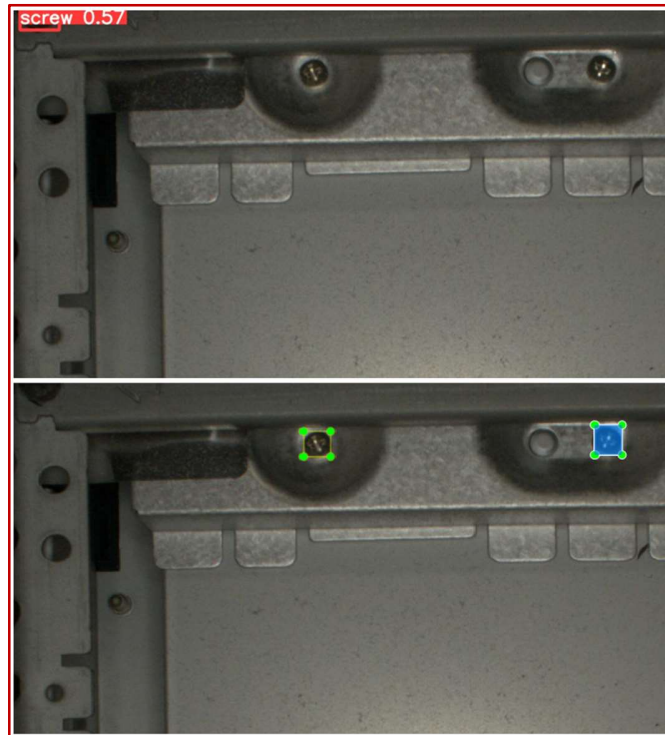
*Figure 4.16: YOLOv5 False Negative Detections (Detection VS Ground Truth)*



*Figure 4.17: YOLOv5 False Negative Detections (Detection VS Ground Truth)*

From the figures 4.16 and 17, these are some obvious detections completely overlooked by the model. They come frequently in cases of screws of black coloring, or screws hidden underneath a PCB part, and a deficiency is detections of a particularly small size.



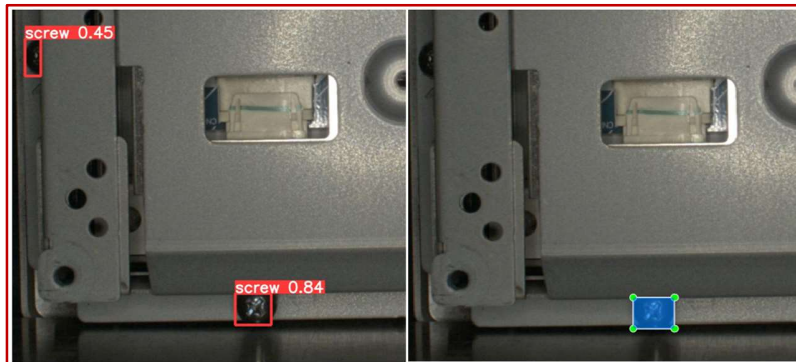


*Figure 4.18: YOLOv5 False Negative Detections (Detection VS Ground Truth)*

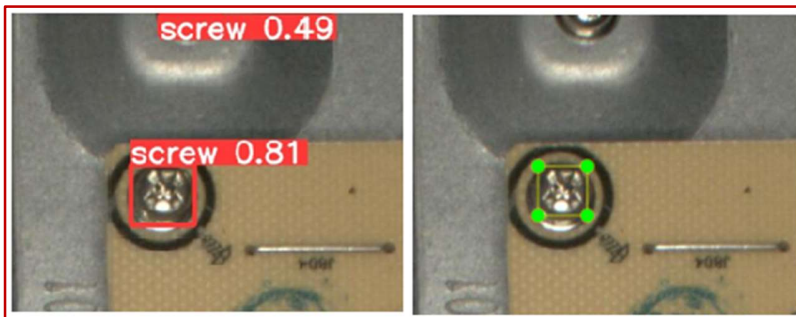
The figure 4.18 displays an overlook by the model in detecting screws obviously placed in the image, albeit, small and blended into the environment. This is a notable miss that can be ideally solved by centering the screws by moving the camera.

#### **4.3.3. True Positive Extras**

The models sometimes display a tendency to detect screws not classified as a ground detection in the test dataset, which can be ruled as a positive point in favor of the model showcasing its generality.



*Figure 4.19: YOLOv5 Extra Detections (Detection VS Ground Truth)*



*Figure 4.20: YOLOv5 Extra Detections (Detection VS Ground Truth)*

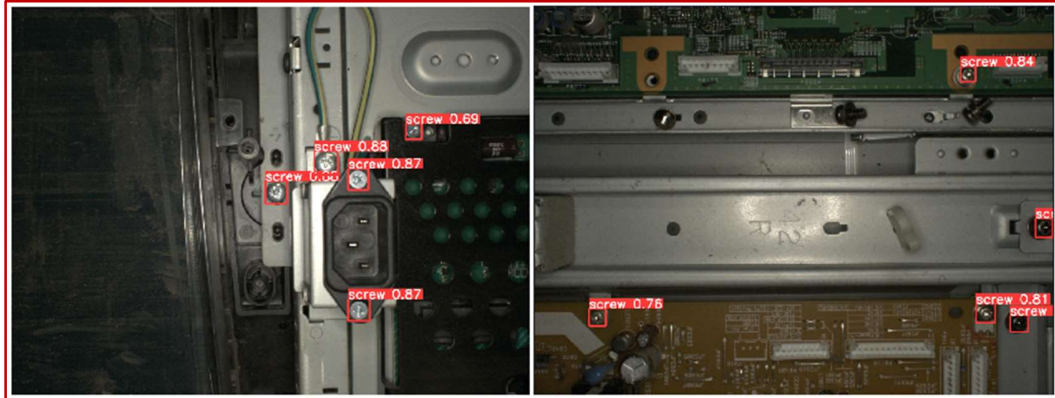
As displayed in the figures 4.19 and 4.20, the model’s fondness of detecting partially displayed screws, although, with much lower confidence. This feature can be alleviated by increasing the threshold values.

#### 4.3.4. TensorRT

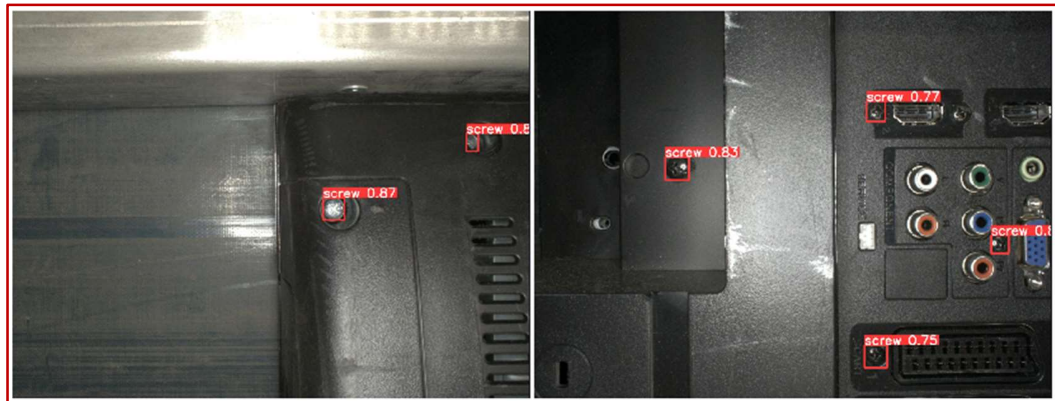
YOLOv5 is a high-performance object detector which can be lightweight and still very effective. To deploy the YOLOv5 object detector for detection purposes on the current Hiro-Robotics architecture for screw detection and removal, the model needed to be optimized into a “.engine” file, deployable using Nvidia’s TensorRT library for their GPUS. This process optimizes the model architecture for higher performance gains in aspects of inference and real time detection speeds.

#### 4.4. YOLOv6

The YOLOv6 is yet another outstanding contender for modern day object detection projects. It is high performing and efficient and performs well on the screw detection dataset.



*Figure 4.21: YOLOv6 Detections*



*Figure 4.22: YOLOv6 Detections*

The YOLOv6 model was able to detect screws successfully, with contrast in primary properties and results attained from tables 4.5 and 4.6 respectively. This owes to the fact that the YOLOv6 model has more reserved detections and succeeds in detecting edge cases not labelled primarily in this test dataset. This discrepancy for the model to identify screws just out of the range of choice gives it bad results in comparison.

*Table 4.5: Table of Primary Properties for YOLOv6 Detections*

Properties	Values
True Positives	119
False Positives	14
False Negatives	11

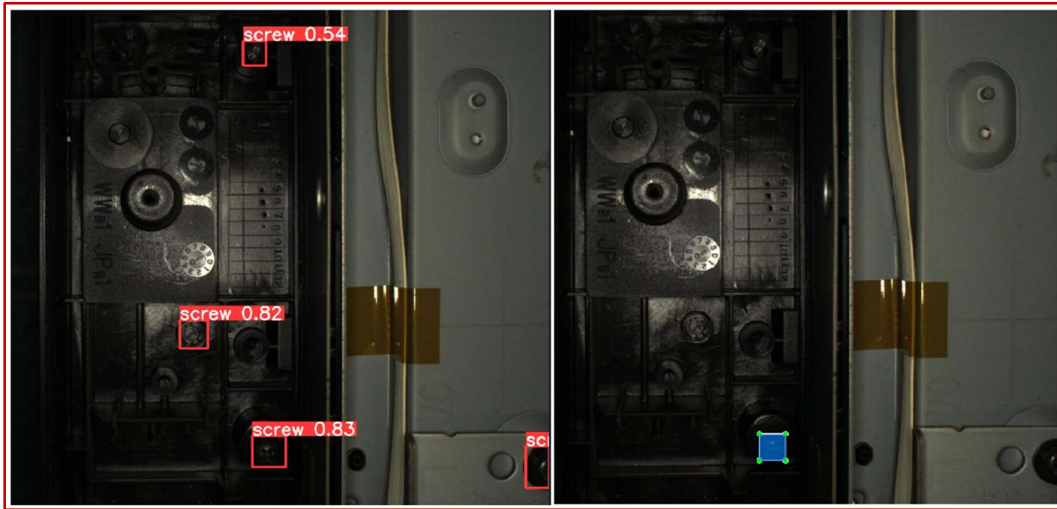
The table 4.5 displays the number of true positives, false positives, and false negatives associated with the current detection task. The model detects 133 objects, 119 of which are positively reinforced by the test dataset and 14 being false positives. On exploration of the detection by checking each image individually, it is good to note that some false positives are at the fringe of the detection criteria and are indeed screws, while some non-detections can be excused by lighting scenario and other peculiar properties of the image. This will be well featured in the true positive, false positive and false negative sections of this subsection.

***Table 4.6: Results for YOLOv6 Detection***

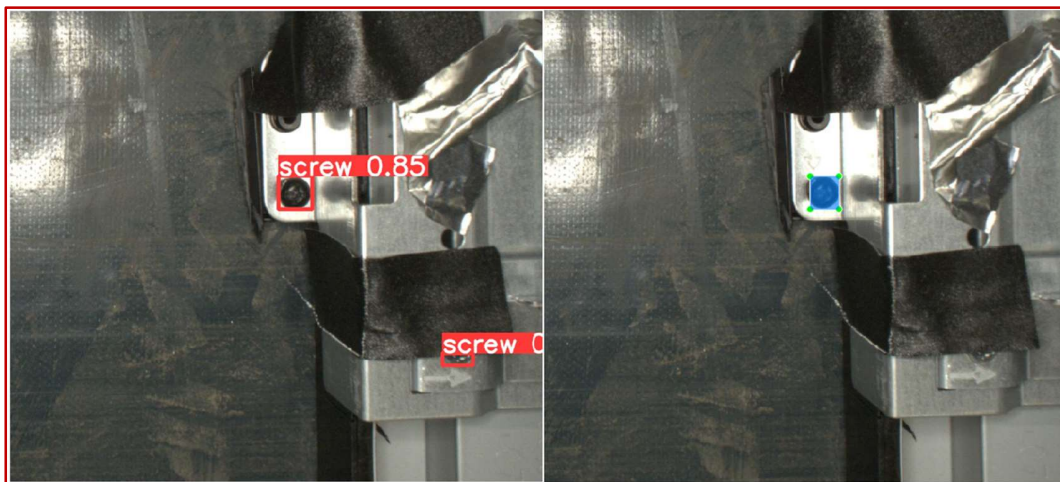
<b>Properties</b>	<b>Values</b>
Precision	0.881
Recall	0.631
mAP@.5	0.871
mAP@.5:.95	0.546
Preprocess Time (ms)	0.32
Inference Time (ms)	4.51
Non-Maximum Suppression Time (ms)	0.98

#### **4.4.1. YOLOv6 False Positives**

The YOLOv6 holds less false positive values owing to the lower recall values. The False positives also tend to be screws detected in obstructions, which reflect negatively on the empirical evaluation, negatively impacting the scores of the model.

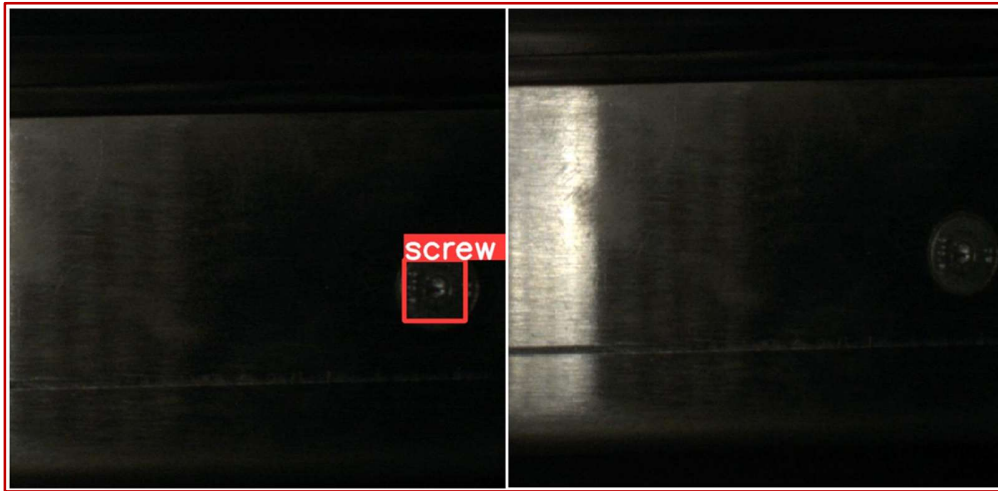


*Figure 4.23: YOLOv6 False Positives (Detection VS Ground Truth).*



*Figure 4.24: YOLOv6 False Positives (Detection VS Ground Truth).*

The false positive detection in the figure 4.24 is a screw obstructed by another FPD part. This however proves difficult based on how much of the features of the screw is being displayed by the image, and whether the company deems it import enough to try to observe screws partially obstructed by view, but reachable by end effector, I.e., screws behind wires, or bent screws.

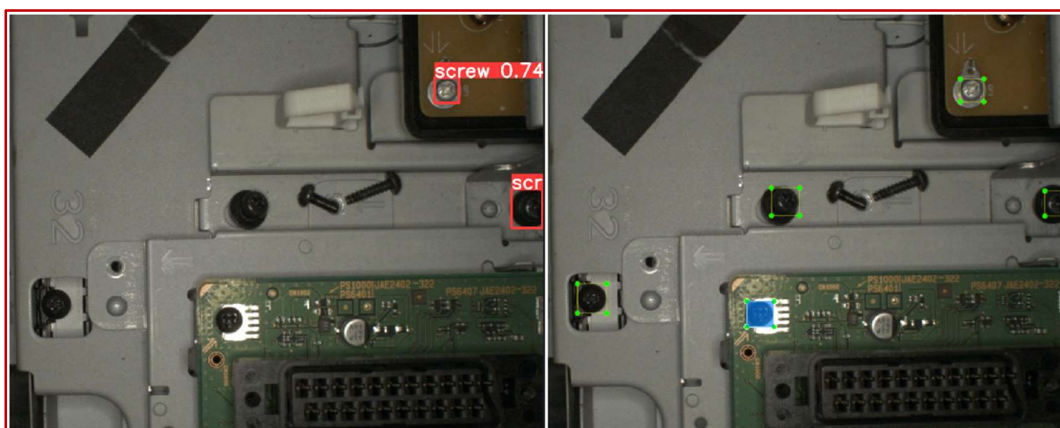


*Figure 4.25: YOLOv6 False Positives (Detection VS Ground Truth).*

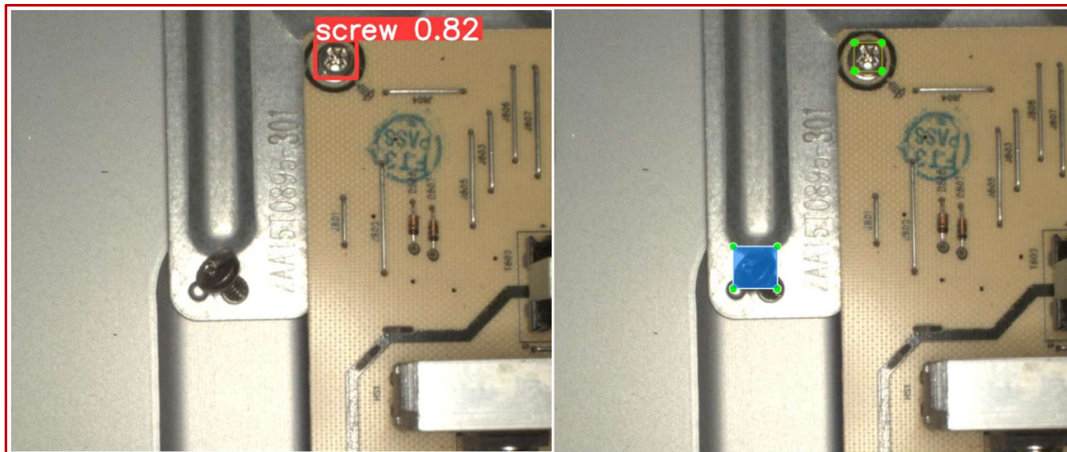
From figure 4.23 and figure 4.25 displayed, it is notable that whilst the YOLOv6 model can cope better with worse light conditions to detect ground positive screws, it is also susceptible to overcompensate by detecting screw-looking details in the dark as false positives as well. Alleviating these dark conditions should improve the model scores drastically

#### 4.4.2. YOLOv6 False Negatives

The lower recall of the model gives it a tendency to overlook ground truth screws. This can only be improved for this case by changing the augmentation properties of the dataset and enlarging the dataset for better training.



*Figure 4.26: YOLOv6 False Negatives (Detection VS Ground Truth).*



*Figure 4.27: YOLOv6 False Negative (Detection VS Ground Truth).*

The figures 4.26 and 4.27 show ground truth detections missed by the model. Where the former image shows more glaring missed by the model, the latter displays a miss with respect to the orientation of the screw. It is notable that with these two images, the YOLOv5 performs better in observing all the ground truths thanks to the benefits of its higher recall values.

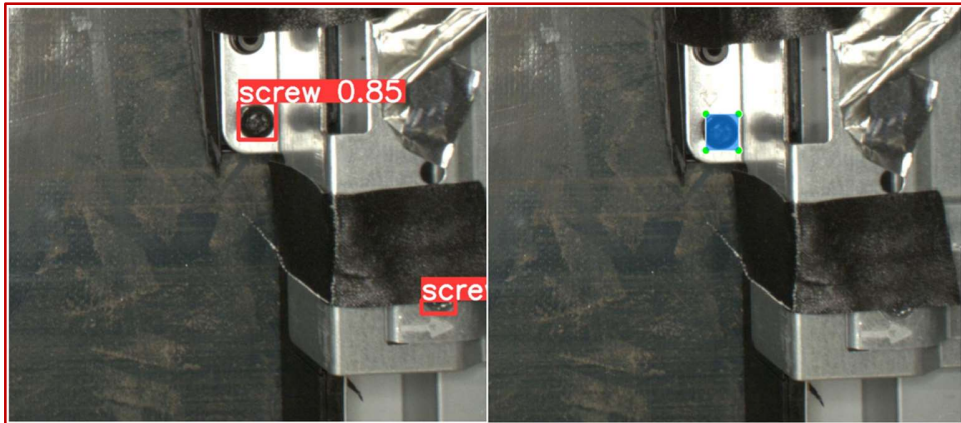


*Figure 4.28: YOLOv6 False Negatives (Detection VS Ground Truth).*

The figure 4.28 also displays a lacking in the model to detect a black screw in an ordinary position, which is less likely for the YOVOv5 model to do.

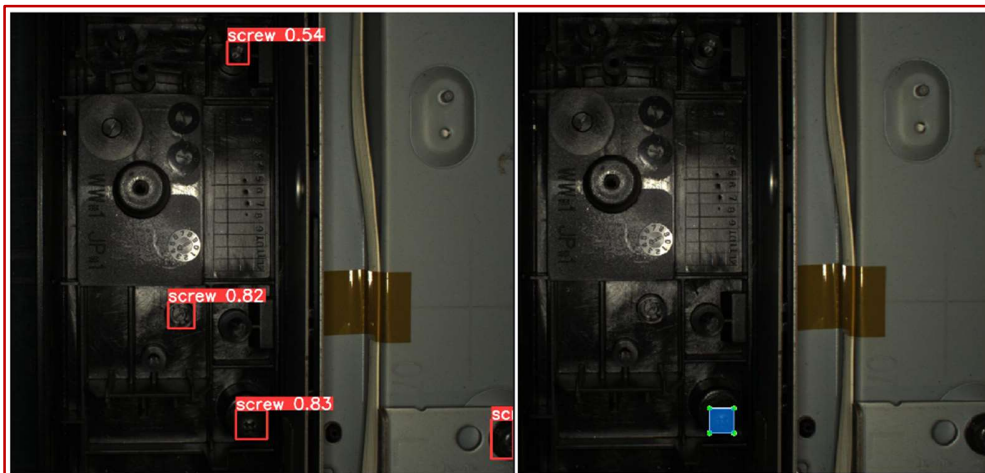
#### 4.4.3. YOLOv6 True Positive Extras

The YOLOv6 model just like the previous generation is capable of detection outliers at the edge of the image or obstructed by parts of the part of interest.



*Figure 4.29: YOLOv6 True Extras (Detection VS Ground Truth).*

The figure 4.29 and 4.24 are the same image, with both false positives giving true extra detection, leading to a loss in score of the model, but outlining the ability of the model to learn intrinsic features of the screws, giving it a generality to detect beyond obstructions and edges.



*Figure 4.30: YOLOv6 True Positive Extras (Detection VS Ground Truth).*



The figure 4.30 displays the model being able to identify outliers as displayed in the image. This image also shows a shortcoming where objects which are not screws are detected with high confidences in the image, leading to indecision whilst selecting model confidence thresholds to work on. This phenomenon has not been mass observed, but noted as this is a dark background, which makes it hard for the model to work accurately.

## CHAPTER FIVE

### 5. CONCLUSION

#### 5.1. INTRODUCTION

Object detection is a significant field of endeavor in the areas of computer vision and artificial intelligence for perception – intelligent perception. Realistically, to build a robot, a device, which is reprogrammable intelligent and autonomous, it is of great importance to build aspects of its perception and sensory self to observe the world better to make intelligent decisions autonomously. Being able to identify objects in each view and set them apart by their class opens a world of possibilities to the structure of knowledge a robot can have, and hence, act upon.

Object detection has given machines a means to observe the world and act based on a classed system, with this important computer vision technology, we have developed a solution that provides autonomous aid to the world, in terms of industrial output and green living. Building an object detection model for screw detection in one of the manifolds of entrances towards computer vision for industrial and recycling applications.

#### 5.2. MILESTONES

A lot of work has been done towards the completion of the project, and they will be summarized as follows:

- Learning of traditional and modern object detection techniques
- Dataset preparation
- Dataset labelling
- Learning and using TensorFlow and Pytorch for the object detection API, and YOLO training.
- Training multiple models such as the SSD mobilenet, SSD Resnet, YOLOv5s, YOLOv5L, YOLOv6s, YOLOv6m, with various augment combinations for the dataset.
- Developing complementary tracking algorithm for deployment with Ur10e robot.

#### 5.3. CONTRIBUTIONS

This project has contributed to the body of knowledge of robotics in by:

- Creating screw dataset for object detection
- Trained models for screw detection to be deployed for various Tensorflow and Pytorch Platforms
- Compared model performances for screw detection

- Developed tracking implementation for YOLOv5 model

#### 5.4. OBSERVATIONS

The objective of the project was to develop a model to achieve near human detection by an object detection model to automate the process of screw detection and removal during recycling. Even humans require a bit of attention when detecting screws in structures they are not completely familiar with, with reviewing the dataset, it was particularly difficult to ascertain screws from time to time. We have been able to train various object detection models to do just that.

With respect to which model excels best for the purpose, we finally compare the strengths and weaknesses of the models' side by side:

- *Performance:* The YOLOv5 model excels in performance, achieving higher inference speeds than both the YOLOv6 and the SSD Resnet. This is due to the optimized nature of the YOLOv5's framework, whereas, the YOLOv6 was just newly released, and the SSD Resnet was too computationally heavy from the density of its parameters.
- *Accuracy:* The YOLOv5 model produces excellent results, with higher recall, and with a tuned IoU of 0.5 and upwards is subject to perform much better, missing some possible true positives. The recall gives it a better performance with engaging the higher number of screws but falls short with false detections. The YOLOv6 performs comparably to the YOLOv5, but with lower recall, hence, does not observe all screws accordingly, and the SSD Resnet is a good performer, but lags with respect to efficiency. Its accuracy could be easily raised but comes at a computational cost unlike the other two models.

#### 5.5. SUMMARY

The YOLOv5 model is the model with the best advantages now given faster inference speeds with the TensorRT implementation, and higher recall with a tendency to observe more screws but also give false positives.

## REFERENCES

- [1] E. Thiébaud (-Müller), L. M. Hilty, M. Schluep, R. Widmer, and M. Faulstich, "Service Lifetime, Storage Time, and Disposal Pathways of Electronic Equipment: A Swiss Case Study," *J Ind Ecol*, vol. 22, no. 1, pp. 196–208, Feb. 2018, doi: 10.1111/jiec.12551.
- [2] A. Appelbaum, "Europe cracks down on E-waste," *IEEE Spectr*, vol. 39, no. 5, pp. 46–51, 2002, doi: 10.1109/6.999794.
- [3] A. Kumar, M. Holuszko, and D. C. R. Espinosa, "E-waste: An overview on generation, collection, legislation and recycling practices," *Resources, Conservation and Recycling*, vol. 122. Elsevier B.V., pp. 32–42, 2017. doi: 10.1016/j.resconrec.2017.01.018.
- [4] K. Zhang, J. L. Schnoor, and E. Y. Zeng, "E-waste recycling: Where does it go from here?," *Environ Sci Technol*, vol. 46, no. 20, pp. 10861–10867, Oct. 2012, doi: 10.1021/es303166s.
- [5] S. Adrian *et al.*, "Quantities, flows, and the circular economy potential The Global E-waste Monitor 2020."
- [6] D. M. Ceballos and Z. Dong, "The formal electronic recycling industry: Challenges and opportunities in occupational and environmental health research," *Environment International*, vol. 95. Elsevier Ltd, pp. 157–166, Oct. 01, 2016. doi: 10.1016/j.envint.2016.07.010.
- [7] E. R. Rene *et al.*, "Electronic waste generation, recycling and resource recovery: Technological perspectives and trends," *J Hazard Mater*, vol. 416, Aug. 2021, doi: 10.1016/j.jhazmat.2021.125664.
- [8] R. Widmer, H. Oswald-Krapf, D. Sinha-Khetriwal, M. Schnellmann, and H. Böni, "Global perspectives on e-waste," *Environ Impact Assess Rev*, vol. 25, no. 5 SPEC. ISS., pp. 436–458, 2005, doi: 10.1016/j.eiar.2005.04.001.
- [9] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert, "An empirical study of context in object detection," Mar. 2010, pp. 1271–1278. doi: 10.1109/cvpr.2009.5206532.
- [10] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," May 2019, [Online]. Available: <http://arxiv.org/abs/1905.05055>
- [11] S. Vongbunyong, S. Kara, and M. Pagnucco, "Application of cognitive robotics in disassembly of products," *CIRP Ann Manuf Technol*, vol. 62, no. 1, pp. 31–34, 2013, doi: 10.1016/j.cirp.2013.03.037.
- [12] G. Foo, S. Kara, and M. Pagnucco, "Screw detection for disassembly of electronic waste using reasoning and re-training of a deep learning model," in *Procedia CIRP*, 2021, vol. 98, pp. 666–671. doi: 10.1016/j.procir.2021.01.172.
- [13] X. Zou, "A Review of object detection techniques," in *Proceedings - 2019 International Conference on Smart Grid and Electrical Automation, ICSGEA 2019*, Aug. 2019, pp. 251–254. doi: 10.1109/ICSGEA.2019.00065.

- [14] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11. Institute of Electrical and Electronics Engineers Inc., pp. 3212–3232, Nov. 01, 2019. doi: 10.1109/TNNLS.2018.2876865.
- [15] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2. Springer, pp. 85–112, Jun. 01, 2020. doi: 10.1007/s13748-019-00203-0.
- [16] L. Liu *et al.*, "Deep Learning for Generic Object Detection: A Survey," *Int J Comput Vis*, vol. 128, no. 2, pp. 261–318, Feb. 2020, doi: 10.1007/s11263-019-01247-4.
- [17] E. R. Davies, *Computer Vision: Principles, Algorithms, Applications*. 2018.
- [18] X. Xu, P. van Beek, and X. Feng, "High-speed object matching and localization using gradient orientation features," in *Intelligent Robots and Computer Vision XXXI: Algorithms and Techniques*, Feb. 2014, vol. 9025, p. 902507. doi: 10.1117/12.2038026.
- [19] A. Sibiryakov, "Fast and high-performance template matching method," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1417–1424. doi: 10.1109/CVPR.2011.5995391.
- [20] K. Park, T. Patten, J. Prankl, and M. Vincze, "Multi-task Template Matching for Object Detection, Segmentation and Pose Estimation Using Depth Images," 2019.
- [21] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," 2004.
- [22] Y. Xu, G. Yu, X. Wu, Y. Wang, and Y. Ma, "An Enhanced Viola-Jones Vehicle Detection Method from Unmanned Aerial Vehicles Imagery," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1845–1856, Jul. 2017, doi: 10.1109/TITS.2016.2617202.
- [23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, 2005, vol. I, pp. 886–893. doi: 10.1109/CVPR.2005.177.
- [24] M. Bdiwi, A. Rashid, and M. Putz, "Autonomous disassembly of electric vehicle motors based on robot cognition," in *Proceedings - IEEE International Conference on Robotics and Automation*, Jun. 2016, vol. 2016-June, pp. 2500–2505. doi: 10.1109/ICRA.2016.7487404.
- [25] Y. Wang, G. Liang, S. Huang, C. Wang, and X. Wu, "A Novel Visual Detecting and Positioning Method for Screw Holes," in *Computer Vision Systems*, 2017, pp. 564–575.
- [26] A. Krogh, "What are artificial neural networks?," 2008. [Online]. Available: <http://www.r-project.org/>
- [27] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way," *Medium*, Dec. 15, 2018. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> (accessed Jun. 06, 2022).
- [28] X. Wu, D. Sahoo, and S. C. H. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing*, vol. 396, pp. 39–64, Jul. 2020, doi: 10.1016/j.neucom.2020.01.085.

- [29] L. Weng, "Object detection for dummies part 3: R-CNN family," *lilianweng.github.io*, Dec. 2017. <https://lilianweng.github.io/posts/2017-12-31-object-recognition-part-3/> (accessed Jun. 06, 2022).
- [30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Nov. 2013, [Online]. Available: <http://arxiv.org/abs/1311.2524>
- [31] R. Girshick, "Fast R-CNN," Apr. 2015, [Online]. Available: <http://arxiv.org/abs/1504.08083>
- [32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," Jun. 2015, [Online]. Available: <http://arxiv.org/abs/1506.01497>
- [33] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," Jun. 2014, doi: 10.1007/978-3-319-10578-9\_23.
- [34] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," Dec. 2016, [Online]. Available: <http://arxiv.org/abs/1612.03144>
- [35] X. Zou, "A Review of object detection techniques," in *Proceedings - 2019 International Conference on Smart Grid and Electrical Automation, ICSGEA 2019*, Aug. 2019, pp. 251–254. doi: 10.1109/ICSGEA.2019.00065.
- [36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Jun. 2015, [Online]. Available: <http://arxiv.org/abs/1506.02640>
- [37] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," Dec. 2016, [Online]. Available: <http://arxiv.org/abs/1612.08242>
- [38] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, [Online]. Available: <http://arxiv.org/abs/1804.02767>
- [39] X. Long *et al.*, "PP-YOLO: An Effective and Efficient Implementation of Object Detector," Jul. 2020, [Online]. Available: <http://arxiv.org/abs/2007.12099>
- [40] G. Jocher, A. Stoken, J. Borovec, A. Chaurasia, and L. Changyu, "ultralytics/yolov5," *GitHub Repository*, YOLOv5, 2020.
- [41] Nir Barazida, "YOLOv6: next generation object detection - review and comparison," <https://dagshub.com/blog/yolov6/>, Jun. 28, 2022.
- [42] meituan, "YOLOv6," <https://github.com/meituan/YOLOv6>, Jun. 23, 2022.
- [43] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," Dec. 2015, doi: 10.1007/978-3-319-46448-0\_2.
- [44] L. Xin, K. Xin, N. Shun, and Fuji Ren, "Object detection based on SSD-ResNet," *2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, vol. IEE, pp. 89–92, 2019, doi: 10.1109/CCIS48116.2019.9073753.

- [45] E. Yildiz and F. Worgotter, "DCNN-Based screw detection for automated disassembly processes," in *Proceedings - 15th International Conference on Signal Image Technology and Internet Based Systems, SISITS 2019*, Nov. 2019, pp. 187–192. doi: 10.1109/SITIS.2019.00040.
- [46] heartexlabs, "labellmg," <https://github.com/heartexlabs/labellmg>, Dec. 03, 2018.
- [47] Jacob Solawetz, "Small Object Detection Guide," <https://blog.roboflow.com/detect-small-objects/>, Aug. 19, 2020.
- [48] Tensorflow, "Tensorflow 2 Detection Model Zoo," [https://github.com/tensorflow/models/blob/master/research/object\\_detection/g3doc/tf2\\_detection\\_zoo.md](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md), Jul. 10, 2020.
- [49] Lyudmil Vladimirov, "TensorFlow 2 Object Detection API," <https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/#>, Jul. 26, 2020.
- [50] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in *2020 international conference on systems, signals and image processing (IWSSIP)*, 2020, pp. 237–242.
- [51] Kiprono Elijah Koech, "Object Detection Metrics With Worked Example," <https://towardsdatascience.com/>, Aug. 26, 2020.
- [52] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert, "An empirical study of context in object detection," Mar. 2010, pp. 1271–1278. doi: 10.1109/cvpr.2009.5206532.