



# ACADEMIA DA FORÇA AÉREA



## **Application of Artificial Intelligence to the Detection of Foreign Object Debris at Aerodromes' Movement Area**

**João Miguel Brito de Almeida**

*ASPAL/PILAV 140665-F*

Dissertation to obtain the Master of Science Degree in  
**Military and Aeronautical Sciences – Aviator Pilot**

### **Examination Committee**

Chairperson: BGEN/ENGINF 099828-B Ana Cristina Domingos de Oliveira Telha

Supervisor: CAP/ENGEL 133004-H Gonçalo Charters Santos Cruz

Co-Supervisor: MAJ/ENGEL 132274-F Tiago Miguel Monteiro de Oliveira

Member of the Committee: Doctor Ricardo Adriano Ribeiro

**Sintra, July 2022**

“An expert is a person who has made all the mistakes that  
can be made in a very narrow field.”

Niels Bohr

The use of trademarks or names of manufacturers in this dissertation is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by neither the Portuguese Air Force Academy nor the author.

*intentionally left blank*

## **Acknowledgements/ Agradecimentos**

A concretização da presente dissertação foi um caminho longo e repleto de obstáculos que, sem o apoio direto e indireto daqueles que estiveram perto de mim, não se teria concretizado. Deste modo, gostaria de deixar um agradecimento a todas essas pessoas que, de uma maneira ou de outra, contribuíram para este trabalho e, em particular, àqueles que menciono a seguir.

Ao Capitão Engenheiro Eletrotécnico Gonçalo Cruz, o meu orientador, que foi muito além do seu papel nesta dissertação e, sem quem, a realização deste trabalho não teria sido possível. As infindáveis horas passadas a explicar conceitos completamente novos para mim, a disponibilidade demonstrada, a ajuda permanente nos testes de campo, o seguimento contínuo do trabalho desenvolvido e a paciência, são alguns demonstradores das suas qualidades enquanto ser humano e militar.

Ao Major Engenheiro Eletrotécnico Tiago Oliveira, o meu coorientador, que, apesar de não ser da área abordada, foi essencial no apoio à escrita e estruturação do documento e demonstrou sempre interesse em manter-se a par do trabalho.

Ao Engenheiro Eletrotécnico Elói Pereira que, apesar da curta passagem pela dissertação desenvolvida, ajudou a traçar aquele que veio a ser o principal caminho seguido neste trabalho.

Ao Capitão Engenheiro Eletrotécnico Diogo Silva que se mostrou sempre disponível para ajudar na resolução dos infinitos problemas com os computadores.

Aos Major Virgílio, Major Andrade, Capitão Rodrigues e Capitão Almeida, pela disponibilidade demonstrada ao aceitarem responder às minhas questões em entrevista.

A todos os membros do CIAFA que sempre se disponibilizaram para ajudar na concretização deste trabalho. Em especial, ao Major Engenheiro Aeronáutico Luís Félix pela ajuda na impressão 3D e ao Sargento Ajudante Paulo Mendes pela ajuda na construção e instalação do suporte para as câmaras.

A todos os meus camaradas KAISERS que me acompanham desde o dia zero nesta Academia, com quem cresci e aprendi a testar os meus limites todos os dias. O vosso apoio foi e sempre será indispensável. É um enorme prazer poder partilhar o dia-a-dia e inúmeras experiências convosco. Em especial, quero agradecer às minhas ‘maria’ Dinis Salgado e José Moura com quem nunca tive receio de confidenciar as maiores dificuldades. Admiro muito a

vossa personalidade, resiliência e boa disposição incessantes. Um obrigado especial ainda ao João Alves pela entajuda, sobretudo académica, durante toda a Academia e a realização da presente dissertação.

À Marta pelo apoio incondicional em todos os momentos e por mostrar sempre o lado positivo das situações. Obrigado por estares sempre do meu lado e por me fazeres crescer um bocado todos os dias.

Por último, quero agradecer à minha família. Mãe, Pai, Maria e Avó, muito obrigado pela vossa presença e apoio incondicionais. Sem vocês, não estaria a cumprir o meu sonho nem seria a pessoa que sou. Mãe e Pai, obrigado por fazerem de mim a pessoa que sou hoje; são o derradeiro exemplo a seguir. Maria, obrigado pelo teu carinho infinito que eu espero um dia conseguir retribuir. Avó, obrigado pelos bons momentos passados juntos e por mostrares que a idade é, de facto, apenas um número.

## **Abstract**

The present dissertation aims to develop a preliminary low-cost and passive system that detects Foreign Object Debris (FODs) at aerodromes based on computer vision with neural networks. FODs are a twofold problem involving safety risks and high associated costs. Although some systems already exist to detect FODs, these are based on radars, making them expensive.

We build a dataset of images to test the viability of this solution, which other authors have already attempted, but the datasets are not publicly available. Moreover, we build a simplified system architecture to capture the images. In parallel, we develop a software pipeline that starts with image capturing scripts and ends in evaluating the models of neural networks we selected.

The datasets created result from three different electro-optical sensors: visible, near-infrared and long-wave infrared. From the first, resulted a dataset of 9,260 images, 5,672 from the second and 10,388 from the third.

Our approach to this problem is based on supervised learning with image classification and object detection, and we train the models in subsets of the datasets. We choose Xception as the neural network for image classification, achieving a 98.86% accuracy. In the case of object detection, we opt for a single-stage detector – YOLOv3 –, achieving an AP of 91.08%.

Finally, we test the same models on new examples and verify a decrease in their performance to 77.92% accuracy for the classifier and 37.49% AP for the detector.

**Keywords: Foreign Object Debris; Computer Vision; Dataset; Image Classification; Object Detection.**

*intentionally left blank*

## ***Resumo***

O objetivo desta dissertação é desenvolver um sistema preliminar de baixo custo e passivo que detete Objetos Estranhos (OEs) em aeródromos baseados em visão computacional com redes neurais.

Os FOD apresentam dois problemas que se prendem com riscos de segurança e os elevados custos associados. Embora já existam alguns sistemas para detetar FOD, estes são baseados em radares, tornando-os caros e exigindo uma série de permissões.

Para testar a viabilidade desta solução, criamos um banco de imagens, o que já foi tentado por outros autores, mas os respetivos bancos não estão disponíveis ao público. Além disso, construímos uma arquitetura simplificada do sistema para capturar as imagens. Em paralelo, desenvolvemos um *pipeline* de *software* que começa pela captura de imagens e termina na avaliação dos modelos de redes neurais que selecionamos.

O banco de dados criados são o resultado de três sensores eletro-óticos diferentes: visível, infravermelho próximo e infravermelho de onda longa. Do primeiro, resultou um conjunto de dados de 9,260 imagens, do segundo 5,672 e do terceiro 10,388.

A nossa abordagem a este problema baseia-se na aprendizagem supervisionada com classificação de imagens e deteção de objetos e treinamos os modelos em subconjuntos do banco de imagens. Para a classificação, escolhemos a Xception como rede neural, obtendo uma exatidão de 98.86%. No caso da deteção de objetos, optamos por um detetor *single-stage* – YOLOv3 –, atingindo uma AP de 91.08%.

Finalmente, testamos os mesmos modelos em novos exemplos e verificamos uma diminuição do seu desempenho para 77.92% de exatidão para o classificador e 37.49% de AP para o detetor.

**Palavras-chave: Objetos Estranhos; Visão Computacional; Banco de Imagens; Classificação; Deteção de Objetos.**

# Table of Contents

List of Figures .....	xiii
List of Tables.....	xv
List of Abbreviations.....	xvii
List of Symbols .....	xx
<b>CHAPTER 1 .....</b>	<b>1</b>
1. Introduction.....	2
1.1. Background and Motivation .....	2
1.2. Scope .....	4
1.3. Objectives .....	4
1.4. Method.....	4
1.5. Outline of the document .....	5
<b>CHAPTER 2 .....</b>	<b>7</b>
2. State-of-the-Art.....	8
2.1. FOD Problem.....	8
2.1.1. FOD Characterisation .....	8
2.1.2. FOD Sources and Location.....	9
2.1.3. Damage Resulting from FODs.....	10
2.1.4. FOD in Military Aviation .....	11
2.1.5. Portuguese Air Force .....	11
2.2. Current FOD Detection Systems .....	13
2.3. Computer Vision and Machine Learning Background.....	14
2.3.1. Learning Methods (supervised, semi-supervised, reinforcement, unsupervised) .....	15
2.3.2. Conventional Methods .....	18
2.3.3. Deep Learning Methods.....	19
2.4. FOD Detection with Computer Vision.....	25

2.4.1.	FOD Detection with Region Proposal and Spatial Transformer Network..	26
2.4.2.	FOD Detection with Region Proposal and Focal Loss .....	27
2.4.3.	FOD Detection with YOLOv3.....	27
<b>CHAPTER 3</b>	.....	<b>29</b>
3.	System Architecture.....	30
3.1.	Image Acquisition Platform.....	31
3.1.1.	Vehicle .....	31
3.1.2.	Cameras.....	32
3.1.3.	Setup .....	33
3.2.	Software Pipeline.....	35
3.3.	Neural Networks.....	37
3.3.1.	Classification Network.....	37
3.3.2.	Detection Network.....	41
<b>CHAPTER 4</b>	.....	<b>49</b>
4.	FOD Dataset .....	50
4.1.	Existing Datasets .....	50
4.2.	Objects Selected for the Dataset.....	51
4.3.	Data Acquisition .....	52
4.3.1.	First Acquisition.....	53
4.3.2.	Second Acquisition .....	54
4.3.3.	Third Acquisition .....	56
<b>CHAPTER 5</b>	.....	<b>61</b>
5.	System Training and Testing .....	62
5.1.	First Steps .....	62
5.2.	Image Preparation.....	63
5.3.	Metrics for Evaluation .....	65
5.3.1.	Xception Implementation and Results .....	67

5.3.2.	YOLOv3 Implementation and Results.....	70
5.4.	Evaluation with New Objects .....	73
5.4.1.	Results on the new Dataset with Image Classification .....	74
5.4.2.	Results on the new Dataset with Object Detection.....	76
<b>CHAPTER 6</b>	.....	<b>79</b>
6.	Conclusions and Future Work .....	80
6.1.	Conclusion .....	80
6.2.	Future Work.....	82
Bibliography	.....	84
Annexes	.....	1

## List of Figures

Figure 1 – FOD distribution by size and weight (adapted from: McCreary, 2010, p. 111). .....	9
Figure 2 – Traditional methods' image understanding pipeline. ....	15
Figure 3 – As the network goes deeper, its spatial resolution decreases, but the semantic content increases from low-level to high-level features (adapted from Lakshmanan, 2021, p. 139). ..	20
Figure 4 – Normal convolution (Bendersky, 2018). ....	21
Figure 5 – FOD Detection process with YOLOv3 (Li & Li, 2020, p. 7098).....	28
Figure 6 – System training and deployment methodology. After labelling, training and evaluating a model, the latter should be uploaded to the system for its deployment. When the vehicle passes through an FOD in the field of view of the sensor, it should warn the driver of its presence recurring to a visual and/or audible warning. ....	31
Figure 7 – Mobile platform used in our project to capture images while moving with the cameras mounted on its top. ....	32
Figure 8 – Simplified setup architecture. The setup is divided into two main areas: interior and exterior. In the interior, we placed computer screens to monitor the image capturing process and help control the cameras. In the exterior, the computational boards and cameras are fixed to a support structure and connected to a power bus.....	33
Figure 9 – Raspberry Pi with 3D printed protective case and Visible+NIR sensor mounted..	34
Figure 10 – NVIDIA® Jetson TX2 with 3D printed protective case.....	34
Figure 11 – Field of view of each sensor: 01 (in blue), 02 (in red) and 03 (in green). Image to scale.....	34
Figure 12 – Label Studio image labelling example.....	36
Figure 13 – Software pipeline representation.....	36
Figure 14 – Depthwise convolution. The image and filters are divided into three different channels to be convoluted separately and regrouped afterwards (Bendersky, 2018). ....	39
Figure 15 – Xception architecture (Chollet, 2017, p. 5).....	40
Figure 16 – Visual representation of the inference time versus MS COCO AP for YOLOv3, RetinaNet and SSD (Redmond, J. and Farhadi, A., 2018, p.1).....	42
Figure 17 – YOLOv3 network architecture. When employed for detection, the Average Pooling, Connected, and SoftMax layers are removed (Valdez, 2020).....	46
Figure 18 – Frames from Munyer et al.'s (2022) 'MetalPart21'. The frames are saved from a video where the camera moves around the object, providing different perspectives of it. ....	50

Figure 19 – Locations of image capturing during the first acquisition (Map data ©2022, CNES/Airbus, IGP/DGRF, Maxar Technologies). .....	53
Figure 20 – Image from runway sequence with 10 FODs (sensor 01). The background is complex when compared to our target – the runway –, leading to a higher difficulty in observing the objects due to the high dynamic range. ....	54
Figure 21 – Structure used to secure the cameras to the mobile platform. ....	55
Figure 22 – Location of image capturing during the second acquisition. (Map data ©2022, CNES/Airbus, IGP/DGRF, Maxar Technologies). ....	56
Figure 23 – Object occurrences by class and sensor for the third acquisition. ....	57
Figure 24 – Box plot of pixels dimensions from the third acquisition objects. ....	58
Figure 25 – Confusion matrix for binary classification problems.....	65
Figure 26 – Training plot of the curves of loss and accuracy for training and validation. ....	70
Figure 27 – Plot of the AP versus IoU on the images of the third acquisition.....	72
Figure 28 – Correct detections of FODs by the classifier. It can detect small objects on the edges with a high degree of certainty, as seen in the middle and right images (sensors 02 and 01)..	72
Figure 29 – Example of a false positive detection on the left image (sensor 01) and false negative detection on the right image (sensor 02). ....	73
Figure 30 – Correct classification of the plastic tube (sensor 02). ....	76
Figure 31 – False positive classification of plants as FOD (in green) and the true positive classification (in orange). Image from sensor 01. ....	76
Figure 32 – Plot of the AP versus IoU on the images of the third and fourth acquisitions.....	77
Figure 33 – True positive detections of different objects previously unseen by the model. Top images (sensor 02), bottom images (sensor 01). ....	78
Figure 34 – Depiction of incorrect detections. False negative (left) and false positive (right). ....	78

## List of Tables

Table 1 – False alarm rate, screw and stone recall rates, and mean average precision of the tested methods (adapted from Cao et al. 2018, pp. 10-11).....	26
Table 2 – Recall rates for several methods (adapted from Liu et al., 2018, pp. 552-553).....	27
Table 3 – mAP, average RR and FPS results from SSD300, faster R-CNN and YOLOv3 (adapted from Liu & Liu, 2020, p. 7099).....	28
Table 4 – mAP scores from three different object detectors and datasets. Adapted from Liu et al. (2021). .....	43
Table 5 – mAP scores of single and double stage object detectors with different backbones on MS COCO+SUN small objects. Adapted from (Nguyen, 2020). .....	44
Table 6 – mAP scores of single and double stage object detectors with different backbones on the subsets VOC_MRA_0.58 and VOC_MRA_10. Adapted from (Nguyen, 2020). .....	44
Table 7 – Performance of single-stage and double stage methods in terms of inference time, Test RAM and Train RAM on MS COCO+SUN small object dataset and PASCAL VOC 2007 subsets. Adapted from (Nguyen et al., 2020; Pham et al., 2017). .....	45
Table 8 – Object description by class and dimensions.....	52
Table 9 – Number of FODs in each sequence per capturing location.....	54
Table 10 – Capture sequence and objects’ characteristics of the third acquisition.....	58
Table 11 – Area of the objects compared to the image size from the third acquisition. ....	59
Table 12 – Comparison of the median relative area of the objects for the original images and their cropped versions for image classification. ....	64
Table 13 – Comparison of the median relative area of the objects for the original images and their cropped versions for object detection. ....	65
Table 14 – Number of images per label used in each dataset for classification.....	68
Table 15 – Summary of the results obtained during training and testing for each model.....	69
Table 16 – Number of tiles employed in each set for object detection. ....	71
Table 17 – Summary of the model's performance on the test set.....	71
Table 18 – Model's fps versus minimum required system's fps. ....	72
Table 19 – Average precision for each IoU on the images of the third acquisition. ....	72
Table 20 – Object description by class and dimensions of the fourth acquisition. ....	73
Table 21 – Capture sequence and objects’ characteristics of the fourth acquisition.....	74
Table 22 – Area of the objects compared to the image size from the fourth acquisition.....	74

Table 23 – Comparison of the median relative area of the objects for classification on the third and fourth acquisitions. ....	75
Table 24 – Comparison of the performance of the classifier in the test subset of the third and fourth ....	75
Table 25 – Comparison of the median relative area of the objects for detection on the third and fourth acquisitions. ....	77
Table 26 – Average precision for each IoU on the images of the third and fourth acquisitions. ....	77

## List of Abbreviations

<b>AI</b>	Artificial Intelligence
<b>AC</b>	Advisory Circular
<b>AP</b>	Average Precision
<b>ATSB</b>	Australian Transportation Safety Bureau
<b>BA5</b>	Airbase no. 5
<b>CAA</b>	Civil Aviation Authority
<b>CNN</b>	Convolutional Neural Network
<b>MS COCO</b>	Microsoft Common Objects in Context
<b>CPU</b>	Central Processing Unit
<b>DOTA</b>	Dataset of Object deTection
<b>DNN</b>	Deep Neural Network
<b>ELU</b>	Exponential Linear Unit
<b>PoAF</b>	Portuguese Air Force
<b>FAA</b>	Federal Aviation Administration
<b>FCN</b>	Fully Convolutional Networks
<b>FOD</b>	Foreign Object Damage/Debris
<b>FoV</b>	Field of view
<b>FPN</b>	Feature Pyramid Network
<b>fps</b>	frames per second
<b>FSF</b>	Flight Safety Foundation
<b>HOG</b>	Histograms of Oriented Gradients
<b>IATA</b>	International Air Transport Association
<b>ICAO</b>	International Civil Aviation Organization

<b>ILSVRC</b>	ImageNet Large Scale Visual Recognition Challenge
<b>LBP</b>	Local Binary Patterns
<b>LWIR</b>	Long-Wave Infrared
<b>mAP</b>	Mean Average Precision
<b>NASEM</b>	National Academics of Sciences, Engineering and Medicine
<b>NATO</b>	North Atlantic Treaty Organization
<b>NIR</b>	Near-Infrared
<b>NN</b>	Neural Network
<b>NTP</b>	Network Time Protocol
<b>R-CNN</b>	Region-Based Convolutional Neural Network
<b>RAM</b>	Random Access Memory
<b>ReLU</b>	Rectified Linear Unit
<b>RoI</b>	Region of Interest
<b>RR</b>	Recall Rate
<b>RPN</b>	Region Proposal Network
<b>SIFT</b>	Scale Invariant Feature Transform
<b>SS</b>	Selective Search
<b>SSD</b>	Single Shot Multibox Detector
<b>SSH</b>	Secure Shell
<b>STN</b>	Spatial Transformer Network
<b>SUN</b>	Scene UNDERstanding
<b>SVM</b>	Support Vector Machine
<b>USAF</b>	United States Air Force
<b>VGG</b>	Visual Geometry Group

**VOC**      Visual Object Classes

**YOLO**     You Only Look Once

## List of Symbols

### Measurement units

Symbol	Designation	Measure of
m	Metre	Length
m <sup>2</sup>	Squared Metre	Area
h	Hour	Time
km/h	Kilometres per Hour	Speed

*intentionally left blank*

*intentionally left blank*

# CHAPTER 1

---

In Chapter 1, we introduce the reader to our problem and explain why we will investigate a solution to it.

# 1. Introduction

## 1.1. Background and Motivation

The crash of Concorde at Paris Charles de Gaulle was a turning moment for the aviation industry (Cao *et al.*, 2018), establishing new procedures regarding Foreign Object Debris (FOD) and movement area inspection. In addition, the final accident report of Air France Flight 4590 mentions the importance of the development and installation of automatic detection systems in airports to increase runway operation safety (BEA, 2000). Adding up to the safety risk, FOD also carries a substantial economic loss to aircraft operators (Graves & Buttlar, 2013; H. Xu *et al.*, 2018).

Two main factors lead to FOD related occurrences: maintenance and conditions from the physical environment of aircraft operation (ATSB, 2010; FAA, 2009; Kraus & Watson, 2001). According to the Australian Transportation Safety Bureau (ATSB) the environment's conditions are the main contributor to FODs, second to maintenance and the human factors involving it. According to Kraus and Watson (2001, p. 4), Foreign Object Damage “ranked as the most likely potential ground-based cause that could lead to a catastrophic aviation event.” Out of four runway safety risk categories: bird strikes, incursions, excursions and FOD strikes, the latter carries the highest costs to operators (McCreary, 2010). Operational Damage<sup>1</sup>, where FOD occurrences are included, was fourth in accidents by category – 16 out of 135 (12%) – on International Civil Aviation Organization's (ICAO) 2020 Safety Report (ICAO, 2020). Although International Air Transport Association (IATA) and Flight Safety Foundation (FSF) make almost no claim when it comes to the impact of FOD on their reports, McCreary (2010), disagrees with the statistics presented, who claims that at airports with 300,000 aircraft movements per year, an estimated total of 82 damage-causing incidents occur on the runway every year, being 62.1 of these caused by FODs. This translates into one event at every 4,831 movements. Considering the years 2013 through 2019, the PoAF accomplished, on average, 11,988 missions (EMFA, 2017, 2018, 2019). This corresponds, at least, to 2.06 FOD strikes on the runway at every 10,000 missions. Nevertheless, every mission has at least one take-off and one landing, increasing the number of movements. However, they do not provide the number of movements.

---

<sup>1</sup> (ICAO, 2020, p. 34) “Damage sustained by the aircraft while operating under its own power. This includes in-flight damage, foreign object debris (FOD) and all system or component failures.”

Although it is unlikely to eliminate all FOD from airports, FOD control methodologies and detection technologies could reduce the number of events related to this problem (BEA, 2000; Hussin *et al.*, 2016). The platforms for automatic FOD detection have been on the market since the early 2010s (Herricks *et al.*, 2012; Stratech, 2016; Trex Aviation Systems, 2016) and have accuracies of over 95%. Although a manual/visual inspection accuracy is around 80%, a single inspection per day can only find and remove around 4% of the total number of items present on the runway throughout an operational day (McCreary, 2010). These systems rely on electro-optical and millimetre wavelength radar sensors, but they are expensive and require special permissions due to the location of installation and the frequencies used (H. Xu *et al.*, 2018).

Artificial Intelligence (AI) is embedded in today's world and research in the field is non-stopping. Self-driving cars, leading-edge banking security, facial recognition and cancer diagnoses are some of the AI applications. In a broader sense, AI aims to outperform humans at a given task (Poola, 2017). Computer vision is one of the research areas of artificial intelligence. It allows computers to take information from visual inputs such as image and video and take actions or make recommendations based on the information received (IBM, 2021). The human eye system is comparable to computer vision because the ultimate goal of both is to interpret what surrounds them. However, man has a head start of millions of years of training and adaptation in the face of these systems.

The volume of data the human visual system can absorb from the perceived world and consecutively interpret is exceptionally high. For a computer vision system to perform a given task with the same data, it must handle large amounts of data with effective solutions. Before the introduction of the concept of deep learning by Hinton *et al.* (2006), the detection of objects was challenging because the features that helped classify the objects had to be hand-crafted by feature designers. With deep learning and Neural Networks (NNs), features are learned from data without the designer's intervention. These networks are capable of "delivering near-human accuracy" (Hassaballah & Awad, 2020, p. 2) to computer vision (*e.g.*, detection and classification), consuming less time and computer memory. From then on and, especially with the set concepts blended in AlexNet (Krizhevsky *et al.*, 2012), which provided a significant reduction in error rate and improvement in mean average precision (mAP), Convolutional Neural Networks (CNNs) have been the backbone of computer vision systems.

## **1.2. Scope**

The very own term ‘Foreign Object’ is explicit in the sense that anything alien to the aerodrome can pose a threat to an aircraft or people in the movement area. The two main consequences of this definition are:

- the large number of categories of FOD;
- the need for near real-time performance of the detection system.

Due to the potentially large number of FOD categories, this work’s scope needs to be circumscribed. The foreign objects in this work are limited to the military context. The most common categories of FODs cut across military and civil aviation and are pieces from aircraft, maintenance tools, fractured pavement parts and natural elements (McCreary, 2010; NASEM, 2011).

Additionally, we will consider solutions that augment safety and do not replace any existing procedures or products. Thus, the considered solution should have minimal negative impact on aerodrome users. This work will consider only passive electro-optical solutions and use the advantages offered by AI to process and gather information in search of foreign objects.

## **1.3. Objectives**

This dissertation aims to study passive and autonomous solutions based on computer vision to detect Foreign Object Debris on military aerodromes. To achieve this goal, the present work will meet the following objectives:

1. Study adequate detection techniques;
2. Create a hardware and software platform that enables field trials;
3. Build a dataset containing images with positive and negative examples of FODs;
4. Evaluate the implemented technological and scientific solutions.

## **1.4. Method**

The present dissertation uses quantitative research methods. This work starts by reviewing the FOD problem at aerodromes and the existing methods to mitigate it. This allows a better characterisation of the objects to be detected. Afterwards, this work will be experimental, with a first stage requiring the preparation of a computational solution (hardware

and software) and a second stage dedicated to data collection. Based on the gathered data, different solutions are implemented and evaluated.

## **1.5. Outline of the document**

The dissertation is divided into six chapters. Chapter 1 covers the background and motivation, objectives, method and outline of the dissertation. It states and motivates the development of a low-cost FOD detection system based on computer vision. The method helps to describe the way the objective will be achieved.

Chapter 2 recaps the state-of-the-art of the FOD problem, electro-optical and millimetre wave radar solutions, computer vision and FOD detection involving artificial intelligence techniques. It congregates the main topics and theories that constitute the base of this dissertation.

Chapter 3 describes the development of the hardware solution implemented on the vehicle we used to capture the images of our dataset. It defines the software based on computer vision and the mobile platform with electro-optical sensors that will have to be integrated and developed according to the identified needs and the system's different features. The proposed architecture is a raw version of what a future implementation would look like. Here, we detail the neural networks we employed to test the dataset and the metrics that help find the best approaches according to the objective of the present dissertation.

In Chapter 4, we explain why we chose to create our dataset and which documents we followed to have a dataset that corresponds to the FODs that are most common. Lastly, we detail the experimental trials we run to capture the images that constitute our dataset and the final dataset.

Chapter 5 describes the training and evaluation process applied to supervised classification and detection techniques. At last, it establishes the experiments and the metrics used in the evaluation process.

Finally, Chapter 6 is a comprehensive review of the dissertation. It draws and summarises the main conclusions and limitations obtained from the previous chapters. This chapter also provides recommendations and ideas that emerged during the dissertation to further develop and improve the system.

*intentionally left blank*

# CHAPTER 2

---

In this Chapter, we will review the state-of-the-art of the main themes discussed in this dissertation. Firstly, we look at studies and reports about FOD in terms of taxonomy, location, resulting damage and the specific case of military aviation. Then, we assess the current systems available for FOD detection, which rely on radar or optical sensors. Finally, we review the evolution of machine learning applied to computer vision and its application to FOD detection with deep learning.

## 2. State-of-the-Art

### 2.1. FOD Problem

#### 2.1.1. FOD Characterisation

The characterisation and definition of FOD are broad since anything that should not be in the movement area of an aerodrome is foreign to that place. FAA gives a broad description: “FOD can be composed of any material and can be of any color and size” (FAA, 2009, p. 3). ICAO (2010) proposes to group FOD by its location and respective persistence, the potential to cause harm either to anyone or anything and size. On the other hand, the Portuguese Air Force (PoAF) Prevention Plan for Foreign Objects Debris<sup>2</sup>, more specifically Airbase no. 5, divides FOD into ‘categories’ and ‘types’. The categories are: component, tool, environmental and others. The types are: metallic, plastic, rubber, textile, organic, mineral and others (PoAF, 2018).

According to ATSB’s study (ATSB, 2010), from 1998 to 2008, aircraft components made up 33% of FODs and these were most likely to be found on runways rather than taxiways or aprons mainly because of vibrations and impact, as an Atlanta Hartsfield Airport study showed (2009 as cited in McCreary, 2010). Tool pieces or ground equipment accounted for 19% of the debris, unidentified objects 16% and metal from unknown sources 12% (ATSB, 2010). For 22 months, a study at Chicago O’Hare Airport identified 1,146 hazardous FODs where 60% were smaller than 3.81 cm and made of metal. In the medium ]3.81; 7.62[ cm and large [7.62[ cm category sizes, the most commonly found FODs were concrete and asphalt chunks (Herricks *et al.*, 2015). Another study by Eurocontrol (2008, as cited in FAA, 2009) found that more than 60% of FODs are made of metal and 18% of rubber where nearly 50% of all items were dark-coloured. The same study ascertained that the most common FOD dimensions are 2.5 cm by 2.5 cm or smaller. Nonetheless, data from United Kingdom’s Civil Aviation Authority (CAA) study (2004, as cited in McCreary, 2010) shows that although FODs of 2 cm or smaller were found, these were present on larger aggregates such as gravel from deteriorating pavement. Although the two studies do not concur at first sight, we believe that CAA’s analysis of small debris by clusters matches Eurocontrol’s depiction by individual size. Moreover, they learned that 80% of the debris will have one of the dimensions larger than 13 cm, which matches McCreary’s (2010) evidence that 90% of FODs are larger than 2 cm, and 99% are found in clusters larger than 3 cm (Figure 1).

---

<sup>2</sup> Plano de Prevenção de Danos por Objetos Estranhos (PPDOE)

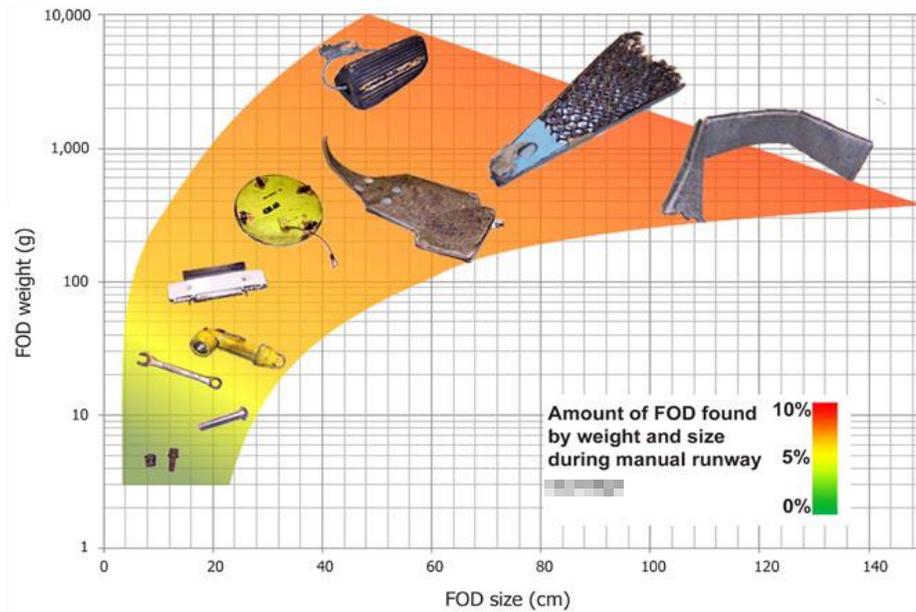


Figure 1 – FOD distribution by size and weight (adapted from: McCreary, 2010, p. 111).

When it comes to size, large FODs tend to represent safety issues, whereas small FODs have more impact on the operator’s costs (McCreary, 2008, 2010). ICAO also admits that “the absence of a globally accepted FOD definition and taxonomy” (ICAO, 2010, p. 5) makes it hard to record FOD in a consistent way leading to a poor and lack of analysis and costing of this issue (NASEM, 2011).

### 2.1.2. FOD Sources and Location

As previously seen, anything can be an FOD (FAA, 2009) and these can come from many different sources (NASEM, 2011), which makes it complicated to control FODs at an airfield (FAA, 2010; McCreary, 2010). Concerning the following locations where FOD are found: runways, taxiways, aprons and hangars (NASEM, 2011, p. 29) – see [Annex A](#) –, suggests manoeuvring area debris, natural materials, apron items, and aircraft parts are the most common type of FOD found there.

It would be expected that most FOD occurrences happened where their presence is the most significant – aprons/gates (McCreary, 2010). In addition, a report from CAA (2004, as cited in Hussin *et al.*, 2016) shows that 55% of FOD are found there and 30% on taxiways. The remaining 15%, found on runways, represent 50% of FOD related damage (Delta Airlines, 2005 as cited in McCreary, 2008). Taxiways and ramps take the remainder of FOD related damage at 30% and 20%, respectively. Moreover, merging data from several airports, the same author concludes that airports that rely exclusively on human inspections miss 97% of FODs.

From a safety perspective, these values show that it is crucial to have effective FOD monitoring on runways and taxiways, primarily when most airports and aerodromes solely rely on visual inspections. In addition, runways and taxiways are where aircraft are more susceptible to FOD damage (FAA, 2010) since aircraft speeds and engine regimes, especially on runways, can induce severe damage to the aircraft or people around.

### **2.1.3. Damage Resulting from FODs**

The study led by ATSB (2010) found that 80% of the occurrences related to FOD do not affect regular aircraft operations. However, when it does, the most common consequences are: rejected the take-off (2%), return after take-off (3%) and aborted landing (14%). Out of all FOD occurrences, 11% lead to wheel, engine and airframe damage.

Combining several data from different airports and airlines, McCreary (2010) concluded that FOD strikes occur 4.0 times per 10,000 movements, and 79% of those (3.2/10,000) inflicted damage to the aircraft. Since, as previously stated, half of the registered damage occurred on the runway, this means that 1.6/10,000 movements suffer from FOD damage on that exact location. The most relevant damages inflicted by FODs are on tyres and gas turbines' blades.

FOD tyre events can be divided into two categories: tyres visibly torn or punctured by FOD, which need replacement, and tyres with embedded FOD that are not visible during inspection but can fail the retread process. The first category implies 1.3 tyre replacements per 10,000 flights at the cost of US\$2,346<sup>3</sup>. Embedded FOD leads to an average US\$4,214 cost at a rate of 1.9 per 10,000 movements. Considering these two categories, McCreary (2010) calculated an average of US\$6,570 for every 10,000 movements, including man hours.

Engine damage comes to “nicked or broken blades” (McCreary, 2010, p. 129) since catastrophic failure due to FOD is rare (Chauhan *et al.*, 2020). However, FOD is considered one of the most significant factors affecting fan and compressor blades' life (CAA, 2020; Spanrad & Tong, 2010). Of the 4.0/10,000 movements where FOD strikes occurred, 20% were related to the engine (McCreary, 2010). We can divide this figure into three: engine strikes with no damage (10%), minor damage (9%) and substantial damage (1%). Minor damage means one or two damaged blades and the average cost per 10,000 movements is \$11,640, while three or more damaged blades mean substantial damage and cost \$108,843 for the same reference.

---

<sup>3</sup> All values derived from McCreary (2010) refer to 2010 US dollars.

Adding up the numbers, McCreary (2010) got to the conclusion that the average direct cost<sup>4</sup> of FOD strikes per 10,000 movements is US\$32,333 or US\$10,366 per strike. The same author, considering yearly indirect costs of US\$5,173,302,215, predicts a total cost of US\$5,690,632,437 worldwide per year relating to FOD damage, similar to Boeing's US\$4 billion figure (1998).

#### **2.1.4. FOD in Military Aviation**

Even though there are no official statistical reports regarding FODs in military aviation, studies admit the risk is the same as in civil aviation (NASEM, 2011) and is more expensive per damage (McCreary, 2010). NATO (2004) poses FOD as one of the leading military aircraft degraders and, consequently, one of the main reasons for losing readiness.

At the Portuguese Air Force's Airbase no. 5 (BA5), from 2014 to 2018, 54 incidents related to FOD were registered (PoAF, 2018). The United States Air Force reported 800 FOD damage events totalling US\$240,000,000 between 1995 and 2004 at an average cost of US\$300,000 per event minding that three aircraft were lost (USAF, 2005 as cited in Procaccio, 2008).

Turbine engines of military aircraft, similar to commercial aviation, suffer from FOD, and some characteristics of the aircraft and military operation are detrimental to this problem (NATO, 2004). The design and placement of the engine's position on the aircraft will make a difference in this same issue. For example, the F-16's air intake position is more prone to ingesting FOD (NATO, 2004).

#### **2.1.5. Portuguese Air Force**

We reviewed the Prevention Plan for Foreign Objects Debris of Airbases no. 1, 5, 6 and 11 to better understand the regulations created by the PoAF regarding FODs. Furthermore, we did several interviews with officers whose job is related to FOD. The interviews were conducted via e-mail, personally and by video teleconference following a semi-structured method. The transcripts of the communications are in [Annex B](#) and translated into the English language by the author of this dissertation. The interviewees are three Chiefs of the Accident Prevention Group (Airbases no. 1, 5 and 6) and one Flight Safety Officer (Airbase no. 11). They are all

---

<sup>4</sup> "means the cost of repairs and part replacement" (McCreary, 2010, p. 151).

pilots at the PoAF who fly different aircraft, which enriches the interviews with various operational backgrounds.

The prevention plans raise awareness of the theme among all personnel, define responsibilities, procedures, and inspections, give a brief taxonomy of FODs and inform about the reporting methods. We will focus on the inspections of the movement area, taxonomy, and reporting methods. Inspections are carried out every day before, after, and, desirably, during aerial activity. If requested, a sweeper truck must be available to go clean. The taxonomy given in all four plans is similar and corresponds to the one mentioned in the first paragraph of 2.1.1. Personnel may report FOD occurrences via e-mail, telephone, paper, digital forms, or radio in the case of pilots flying. All reports must be issued to the airbase's Chief of Accident Prevention Group. These accounts are registered but not centralised for data analysis.

The interviews' objective is to understand the impact of FOD in the PoAF rather than the procedures or awareness-raising actions carried out by personnel. The interviews revealed that there is no data compiling or statistical analysis of FOD occurrences (CAP Rodrigues, personal communication, November 18th, 2021). The main objects found are, either, parts of aircraft and ground vehicles (*op cit.*), organic materials, safety wire and debris from the pavement, which range "from a few millimetres to 4-5 cm" (MAJ Virgílio, personal communication, November 16<sup>th</sup>, 2021). At aerodromes with shared infrastructures, such as Air Base no. 11, there is the addition of FOD produced by commercial aviation (CAP Almeida, personal communication, November 25<sup>th</sup>, 2021).

The interviews show that most FODs are found near parking spots, taxiways, and runways. This means that their location is not concentrated in a specific area of the aerodrome. However, when asked where most of the incidents related to FOD occur, CAP Rodrigues (2021) states that it is difficult to correlate the damage to the location of impact because, most of the times, the damage is only found some time after the impact. In terms of aircraft, the interviewees agree that the ones equipped with turbofan or turbojet engines suffer more damage due to FODs. For example, MAJ Virgílio (*op cit.*), who is also an F-16 pilot, points out the F-16 as one of the military aircraft more susceptible to FOD damage. According to him, this is due to the intake's position and being a single-engine aircraft, which might imply, in some conditions, the total loss of engine power if an FOD hits the turbine. This directly impacts the operation of that aircraft as it is limited at some airfields due to the deteriorating conditions of the pavement. MAJ Andrade confirms this information (personal communication, November 23rd, 2021),

stating that the pavement at one of the airfields, due to water infiltration, vegetation growth and vehicles has several cracks, resulting in the loosening of small debris. Moreover, the sealant applied to cover the cracks becomes parched and loose, developing into more FODs.

## **2.2. Current FOD Detection Systems**

Current FOD detection systems mainly rely on human/visual support for scheduled, periodic and special inspections (NASEM, 2011). However, aviation authorities (FAA, 2009; ICAO, 2020) acknowledge automated and automatic FOD detection systems which can either be installed, delivering continuous surveillance, or mobile, providing occasional surveillance. FAA (2009) recognises four systems: human/visual, radar, electro-optical and a combination of the last two and stipulates the required characteristics of each one of them.

Human/visual observation is the primary system to detect FOD and provides the best capability of judgement and assessment to assure safety (FAA, 2009). One of the best ways to detect FOD is through FOD walks which, historically, is standard military practice (McCreary, 2010). However, it requires closing the runway for extended periods, which is not always possible, especially at aerodromes with considerable traffic. Radar-based detection was the first automated FOD detection system to be implemented. It operates with millimetre waves and can be associated with a fixed or a mobile platform. While fixed platforms provide continuous surveillance, acquiring several units to cover the intended area is needed. With mobile platforms, this does not verify, but the coverage is not continuous. The mobile platform FOD Finder<sup>TM</sup> by Trex Enterprises has a detection range of around 200 m and a detection cell of 1 m<sup>2</sup> (NASEM, 2011; Trex Aviation Systems, 2016). It has a 100% detection capability certified by the FAA for all weather conditions (Herricks *et al.*, 2011). Electro-optical sensors consist of image and video data processing to detect the debris. One of the systems is iFerret<sup>TM</sup> by Stratech, a fixed equipment which provides a 330 m runway coverage per sensor and can work satisfactorily at night and in degraded meteorological conditions (Herricks *et al.*, 2012) with an overall detection rate of 90% (Stratech, 2016). FODetect<sup>®</sup> by XSight is a fixed system that provides millimetre radar and electro-optical sensors (FAA, 2009). This system in particular, has a range of 60 m and is installed alongside the runway edge lighting system providing a detection average of 98% (Herricks *et al.*, 2012).

The systems mentioned above bring several advantages over visual detection. The most important is that fixed systems provide continuous coverage and mobile equipment requires less runway closure time (McCreary, 2010; NASEM, 2011). This does not mean that visual

inspections and prevention should not be taken seriously or at all. In addition, although it is improbable to have an FOD-free aerodrome and detect every single threat, FOD detection systems greatly minimise this issue (Hussin *et al.*, 2016; McCreary, 2010; NASEM, 2011). These systems' major downside lies in their acquisition, maintenance costs, and permissions, making them less suitable for medium/small airports.

### **2.3. Computer Vision and Machine Learning Background**

According to Huang (1996) and Lakshmanan (2021), computer vision aims to develop computational models that imitate the human visual sensory and cognitive systems to develop autonomous systems that can complete image formation and machine perception tasks. Computer vision is a subset of machine learning which, in its turn, is a subset of artificial intelligence (Alloghani *et al.*, 2020; Sandhu, 2018).

Computer vision as we know it started with Larry Roberts at the Massachusetts Institute of Technology in the 1960s (T. S. Huang, 1996; Shapiro, 2020) with his dissertation and an article from 1965 (Aloimonos, 1990). The goal of Roberts (1963) was to “process a photograph into a line drawing, transform the line drawing into a three-dimensional representation (...) and display the three-dimensional structure (...) from any point of view” (Roberts, 1963, p. 2). His work contributed to fundamental aspects of image understanding, such as edge finding, line fitting and model-based object recognition (Shapiro, 2020).

On what concerns image understanding, there are two relevant problems that, despite their intrinsic connection, must be explained – image classification and object detection. An image classification algorithm classifies the whole image and outputs the probability of matching a class or set of classes (Benali Amjoud & Amrouch, 2020). On the other hand, object detection frameworks can recognise more than one object in one image and its coordinates. These algorithms comprise a two-stage process involving object localization and classification. Hence, object detection problems are more demanding than image classification (Everingham *et al.*, 2010; Russakovsky *et al.*, 2015).

Although image understanding has always had a significant role in computer vision (Cruz, 2019), researchers only turned their attention to it in the late 1990s. It was when more “powerful machine learning techniques and large databases of training images, as well as modern image features” (Shapiro, 2020, p. 4) became available, especially with Graphics Processing Units (GPUs) (Lundervold & Lundervold, 2019). From this decade up to the 2010s,

the image understanding pipeline was relatively standardised and comprised three steps: pre-processing, feature extraction and classification. The last two stages were the most complex and are now referred to as conventional methods. These will be addressed in 2.3.2.

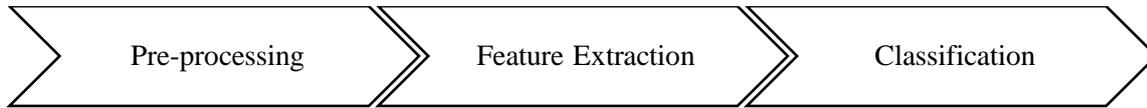


Figure 2 – Traditional methods' image understanding pipeline.

Competitions such as PASCAL Visual Object Classes (VOC) (Everingham *et al.*, 2010) and ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky *et al.*, 2015), Microsoft Common Objects in Context (MS COCO) (T. Lin *et al.*, 2015) are an essential part of the driving force on the development of image understanding (Shapiro, 2020). In conjunction with the advancements in learning algorithms, these competitions led to what is commonly called deep learning and will be addressed in 2.3.3.

The learning techniques used in conventional methods and deep learning can be classified into four types: supervised, semi-supervised, reinforcement and unsupervised learning. The type of algorithm to choose depends on the abundance of available data and the nature of the problem. The characteristics of each will be discussed in 2.3.1.

### 2.3.1. Learning Methods (supervised, semi-supervised, reinforcement, unsupervised)

The four main machine learning methods are supervised, unsupervised, semi-supervised, and reinforcement learning. Supervised and unsupervised learning methods are two of the most relevant (Alloghani *et al.*, 2020), and the primary difference between them is that the former needs labels to be provided with the training data, whereas the latter does not.

Supervised learning aims to establish a mapping between inputs and outputs to predict outcomes for new data where a known set of results are expected (Mohamed, 2017; van Engelen & Hoos, 2020). Therefore, supervised learning frameworks must be good at generalising on a training set to predict unseen data correctly. The labelled training examples are fed to the framework, and its parameters are updated. These parameters are optimised through an iterative process to minimise “the difference between the target output and the computed output” (Jo, 2021, p. 11) with the help of a loss function and an optimiser. Classification and regression are the two main problems addressed in supervised machine learning algorithms (Nasteski, 2017; Sen *et al.*, 2020). While in regression models the outputs are continuous, in classification models, they are discrete or divided by classes (Nasteski, 2017). Some of the most utilised

supervised machine learning algorithms are Linear and Logistic Regression, Naïve Bayes, Support Vector Machine (SVM), Bayesian Networks and Neural Networks, along with others (F. Y *et al.*, 2017). Classification and regression algorithms aim to label new data to a previously known target value or range. A loss function combined with an optimiser is required to penalise errors during training to achieve the final goal (Mohamed, 2017).

Often, the scarcity of labelled data as well as the laborious task of the labelling process are significant barriers to the development of supervised machine learning solutions (Alzubaidi *et al.*, 2021; Lakshmanan *et al.*, 2021). Additionally, dataset size greatly impacts the accuracy of machine learning algorithms (Prusa *et al.*, 2015; Sordo & Zeng, 2005). Contrarily, unlabelled data vastly outnumbers labelled data, making unsupervised learning a potential solution to solve some of the constraints of supervised learning (Géron, 2019).

Unsupervised learning algorithms look for uniformities in input data (Karazi *et al.*, 2019) without assigning a target attribute, learning hidden patterns and acting on data without supervision. Thereby, this learning technique helps analyse large raw data so that users can better characterise the problem they are working with. The most commonly mentioned machine learning techniques in unsupervised learning are clustering and association rules (Alloghani *et al.*, 2020; Dike *et al.*, 2018; Doshi *et al.*, 2021). However, hierarchical learning with NNs, dimensionality reduction, anomaly detection and latent variable models are equally important (Géron, 2019; Usama *et al.*, 2019). When used in a stand-alone application, unsupervised learning tends to be less accurate than supervised (Schmarje *et al.*, 2021). Nonetheless, when applied to pre-training and feature extraction, it boosts the model's accuracy (Misra *et al.*, 2016; J. Xu *et al.*, 2016) since this method is better at generalising and acts as a regulariser (Erhan *et al.*, 2010). However, this is a largely unexplored and complex field of machine learning, as Yann LeCun (2016) explained at Neural Information Processing Systems 2016 (NIPS).

Both supervised and unsupervised machine learning have their drawbacks. In the case of supervised machine learning, the problem lies in the lack of labelled data and the expertise to do so. On the other hand, in unsupervised learning, one cannot access in detail the data sorting process (Doshi *et al.*, 2021). Moreover, unsupervised learning tends to be computational and statistically expensive. This is a result of the high dimensionality introduced by the number of different variables, which the model does not know if they will be useful for the downstream application as in supervised learning (Goodfellow *et al.*, 2016; Rasmus *et al.*, 2015). The two

other machine learning methods – semi-supervised and reinforcement learning – can fill in some gaps of the previous two, as we will see.

The idea behind most semi-supervised algorithms is to combine the main tasks of supervised and unsupervised machine learning frameworks (Chapelle *et al.*, 2006; Zhu & Goldberg, 2009). Most semi-supervised frameworks' layouts are composed of an unsupervised followed by a supervised algorithm (Géron, 2019) or an adaptation of an unsupervised algorithm (Jo, 2021). There are a few different semi-supervised settings, but two of the most common are semi-supervised classification and constrained clustering (Zhu & Goldberg, 2009). The first and most established method is an appendage to supervised classification where both labelled and unlabelled inputs constitute the training data. The second is a branch of unsupervised learning where the basic idea is to feed the supervised algorithm insight about the clusters. Adding unlabelled data does not guarantee better performance, as identified by authors (Chapelle *et al.*, 2006; Y.-F. Li & Zhou, 2015; Oliver *et al.*, 2019). However, Oliver *et al.* (2019), who focused on image classification, found this performance degradation was verified in cases where the classes of unlabelled data did not match the classes of existing labelled data. Moreover, as van Engelen and Hoos (2020) mentioned, the results obtained are encouraging in the sense that unlabelled data can increase image classification tasks based on neural networks.

Reinforcement learning systems' goal consists of identifying by trial-and-error which actions yield the maximum numerical reward signal (Sutton & Barto, 2018). Here, the agent is in a state (or situation) which is limited to a finite set of states and can take actions, which are also limited to a finite number – where both are determined by the environment (Alharin *et al.*, 2020). According to its decision, the agent can be rewarded or penalised. That action will affect the following states and the following rewards (Géron, 2019; Sutton & Barto, 2018). The reinforcement learning process is iterative; therefore, with the increasing number of iterations, the greater the feedback is, increasing the chances of a greater final reward. It can be distinguished from supervised learning in its application because the latter is not suitable to learn from interaction. This has to do with the impracticability of gathering a vast spectrum of examples “of desired behaviours that are both correct and representative of all the situations in which the agent has to act” (Sutton & Barto, 2018, p. 3). Equally to unsupervised learning, reinforcement learning does not rely on labels. However, while the former aims to find hidden patterns, the latter tries to take the best action in each state or situation in order to maximize, in the long-term, the numerical signal the agent receives.

### 2.3.2. Conventional Methods

Until the early 2010s, the paradigm of object recognition in images was mainly based on approaches ‘hand-crafted’ feature extractors such as Scale Invariant Feature Transform (SIFT) (Lowe, 1999) and Histograms of Oriented Gradients (HOGs) (Dalal & Triggs, 2005). These techniques extract visual features from image databases (Cruz, 2019) and, subsequently, apply them to a trained classifier such as SVM (Boser *et al.*, 1992) or a cascaded detector (Viola & Jones, 2001). For example, HOG on PASCAL VOC 2007 scored a mAP<sup>5</sup> (mean Average Precision) of 17%, which, with continuous improvement, reached 41% in 2011 (Shapiro, 2020).

The object recognition method introduced with SIFT converts images into a set of local feature vectors called SIFT Keys which are invariant to “translation, scaling and rotation and partially invariant to illumination changes and affine or 3D projection” (Lowe, 1999, p. 1). The algorithm can be divided into four steps: scale-space peak selection, feature point localisation, orientation assignment and keypoint descriptor. The final product of SIFT is a set of vectors which describe key points in the image. These key points characterise features around them of the image in terms of location, scale and orientation (Lowe, 1999).

Histogram of Oriented Graphs is a feature descriptor similar to SIFT used for object recognition. Its descriptor gives particular focus to the shape or structure of an object. This algorithm obtains the gradient of each pixel and groups them into blocks from which a histogram is calculated (Dalal & Triggs, 2005). A feature vector is then calculated for each block, from where the HOG features of the image are obtained. In its first paper, HOG was tasked with pedestrian classification in two datasets with the help of SVM.

Support Vector Machine was firstly developed by (Boser *et al.*, 1992) for binary classification of linearly separable data or regression problems. This first version of the classifier creates a hyperplane and maximizes its distance to the points of the two classes. The nearest points to the plane help define its position and orientation and are called support vectors. Later developments of the algorithm allow for multiclass classification (Mayoraz & Alpaydin, 1999) by dividing the problem into multiple binary classification problems, creating an N-dimensional hyperplane.

---

<sup>5</sup> Mean Average Precision (mAP) derives from Average Precision (AP) which is the area under the precision-recall curve. mAP is the AP averaged across all classes, which is relevant for multiclassification problems.

These conventional methods were employed for several years and developed to achieve good performances. However, their results were surpassed when deep learning methods resurged in line with more powerful computers.

### 2.3.3. Deep Learning Methods

In this subsection, we will focus on how deep learning methods work. More specifically, we emphasise the process of feature extraction and image classification in supervised learning with CNNs based on their architecture.

The two neurobiologists Hubel and Wiesel (1959, 1962) studied how the visual system of mammals process information. The study's goal was to have a reference to the human system by analysing the primary visual cortex of cats and how its neurons react to visual stimuli. The researchers found that the brain interprets a scene in several stages of hierarchical order. Firstly, it learns the coarser structures such as orientations and edges, and as it continues to observe the scene, the features become finer. The main idea is that the coarser structures or low-level features cluster to create more complex high-level features. Deep learning algorithms and, more specifically, deep neural networks work in a similar way (Janke & Castelli, 2018). The deeper the network's layer, the finer the extracted feature is. The idea behind neural networks comes from the inspiration on neurons, but these are simplified mathematical and statistical models of how a human brain works (Ajit *et al.*, 2020).

The image classification task assumes that an image can be represented by a set of features with different hierarchies: low, medium and high-level features (Sethi & Huang, 2006; Sitaula *et al.*, 2019). According to Lakshmanan *et al.* (2021), CNN's convolutional layers learn low-level features on the first layers, and as the layers in the network go deeper, they begin to learn features with progressively more semantic meaning as suggested in [Figure 3](#). Low-level features are generally related to colour intensity and pixels.

Middle-level features hold information about space and general shapes, which yield semantic meaning from the image's parts or objects (Juneja *et al.*, 2013). In other words, these features comprise “discriminative parts, such as beds in a bedroom, washing machines in laundry, and other distinct components (...) important to identify scenes” (D. Lin *et al.*, 2014, p. 3726). The combination of low and middle-level features is similar to the features that conventional methods such as SIFT and HOG extract (Sitaula *et al.*, 2019). However, these features lack generalised information on the whole image, which is one reason why conventional methods offer poorer accuracy results than deep learning (Tang *et al.*, 2017).

At last, high-level features contain more semantic meaning and structure information because, at this stage, the image has suffered more linear and non-linear transformations (Tang *et al.*, 2017). These features will help with the final classification process since they, with the help of lower-level features, explain the image (Sitaula *et al.*, 2019).

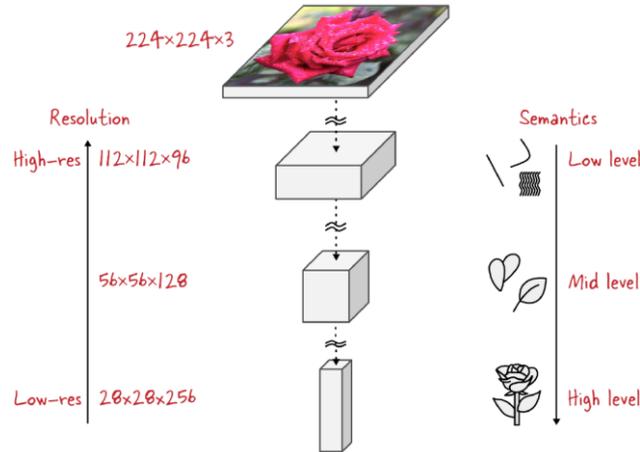


Figure 3 – As the network goes deeper, its spatial resolution decreases, but the semantic content increases from low-level to high-level features (adapted from Lakshmanan, 2021, p. 139).

LeCun and Bengio (2003) identified three architectural ideas which help CNNs achieve distortion and shift invariance – local receptive fields, shared weights and spatial subsampling. In CNNs, a hidden layer neuron is responsible for receiving information from a small region of the input or the previous layer. That neuron or region of neurons in the hidden layer is called receptive field, which is responsible for extracting low-level features at any input location. In a given layer of the CNN, the weights and biases are the same for every neuron, which means that the kernel is extracting the same feature in different locations of the image. This makes the CNN tolerant to feature translation. Since the learned features are translation invariant, we can now reduce the feature map’s resolution, further “reducing the sensitivity of the output to shifts and distortions” (Bengio & LeCun, 2003, p. 277) and the number of parameters. This process is named spatial subsampling, and it usually is a combination of a pooling layer and an activation function.

The typical modern convolutional neural network comprises convolutional layers, pooling layers and fully connected layers (Alzubaidi *et al.*, 2021; Zhou *et al.*, 2017). The result of a convolutional layer is the dot product of the input – which is usually a 2D array – and one or more kernels (=filter or feature detector,) which are slid through the input (Figure 4). This process generates activation maps which store the features of the input image. The results of the convolution are then passed through an activation function – usually, ReLU (Alzubaidi *et*

*al.*, 2021) – to add non-linearity and complexity to the network (Aloysius & Geetha, 2017). The pooling layer succeeds the activation function, and its objectives are to retain the most essential features and subsample the input matrix, reducing the number of parameters and, consequently, the likelihood of overfitting (P. Wang *et al.*, 2021).

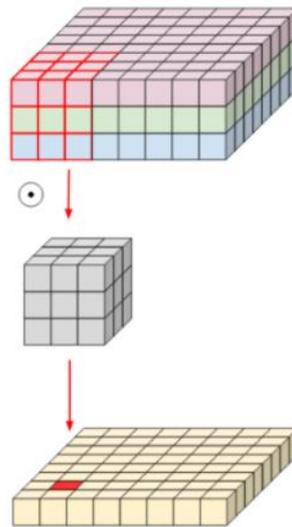


Figure 4 – Normal convolution (Bendersky, 2018).

Finally, the matrix is flattened on the fully connected layer, where all neurons are connected to the previous layer. Fully connected layers “learn non-linear combinations of learned features” (Akbar *et al.*, 2017, p. 2). The classification occurs after this layer or block of layers with the help of the respective activation function, which outputs one or more class probabilities. The most common activation function for this stage is softmax, the multiclass version of sigmoid, which, in turn, is the inverse of logit (Lakshmanan *et al.*, 2021).

To control the output of the deep neural network, one must be able to measure its distance to the ground truth. The job of the loss function or cost function is to calculate that distance. The result works as a feedback signal for the neural network’s training, whose objective is to lower the loss value via fine adjustments to the network’s weights (Chollet, 2021). Some of the most common loss functions are binary and categorical cross-entropy and mean squared error (Q. Wang *et al.*, 2022). The optimisers such as Stochastic Gradient Descent, RMSProp and Adam make the fine adjustments to the neural network. The underlying concept behind the optimisers is in calculating the error and the gradient produced by each weight (Lakshmanan *et al.*, 2021). This whole process is named backpropagation algorithm, which is central to deep neural networks (Alzubaidi *et al.*, 2021; Chollet, 2021).

The development of neural networks has been the driving force of many new applications in computer vision. Their robustness makes these networks work as good backbones or feature extractors for many different tasks such as object detection and image segmentation (Aloysius & Geetha, 2017; X. Li *et al.*, 2016).

### **2.3.3.1. Backbone Neural Networks**

Although they were not new, the development of deep neural networks started to take over and ILSVRC in 2012 was the turning point (Alom *et al.*, 2018; Cruz, 2019) with AlexNet (Krizhevsky *et al.*, 2012) – a Deep Convolutional Neural Network (CNN).

Neural Networks (NNs) have existed for a long time with studies by McCulloch and Pitts (1943) and Rosenblatt (1958) establishing the fundamentals. Work by Fukushima (1980, 1988) with Neocognitron, Ackley *et al.* (1985) with backpropagation algorithm, LeCun *et al.* (1998) “with the modern CNN architecture” (Cruz, 2019, p. 12), and Hinton *et al.* (2006) with Deep Neural Networks (DNN) revived and set significant milestones for Computer Vision and CNNs (Alom *et al.*, 2018; Cruz, 2019). Since then, image classification competitions have been dominated by CNNs, which can halve the errors of previous networks (Cruz, 2019) and achieve accuracies of 96.43% with ResNet (He *et al.*, 2016), greater than the human accuracy for the same required task – 95% (Alom *et al.*, 2018; Chollet, 2021).

The reason why AlexNet was so successful has to do with its revolutionary architecture. This network has a total of five convolutional layers, three overlapping max-pool layers interspersed with the convolutions and three fully connected layers. The Rectified Linear Unit (ReLU) is used as an activation function instead of sigmoid or tanh for faster training. Although this network is still broadly used for some applications, some architectural characteristics, such as overlapping max-pool, are not as common nowadays.

Visual Geometry Group (VGG) (Simonyan & Zisserman, 2015) was introduced at ILSVRC-2014 as VGG-16 and, albeit based on AlexNet, it surpassed the latter by approximately 13% on top-1 errors and more than halved top-5 errors. The increase from five to thirteen convolutional layers and the reduction of kernel size are two of the main reasons why VGG performed better. Another key network developed in 2014 was GoogLeNet (or Inception V1) (Szegedy *et al.*, 2015). This neural network includes a new module named Inception, repeated throughout the pipeline and more convolutional layers than the previous two. Inception modules concatenate a series of filters of several sizes ( $1 \times 1$ ,  $1 \times 3$ ,  $5 \times 5$ )

and dimensionality reduction via a  $1 \times 1$  convolution to restrict the growth of channels' number.

The search for deeper networks was evident as a result of the intuition that they could learn more complex features. Nonetheless, simply making a network deeper by adding layers brought two problems. Firstly, the vanishing gradient problem (Glorot & Bengio, 2010) inhibits convergence. Secondly, after the previous problem was solved, a second one was unveiled – accuracy saturation (He & Sun, 2014; Srivastava *et al.*, 2015), leading to a more significant training error and accuracy degradation. The network designed by He *et al.* (2016) – ResNet – introduced the idea of Residual Learning where the previous layer is directly connected to the layer or block of layers ahead. This technique prevents the two problems aforementioned.

As mentioned above, conventional methods require engineered features, which are laborious and expensive processes in terms of competence and time (Jmour *et al.*, 2018). Deep neural networks changed the paradigm with learning-based features as they accomplish the feature extraction task without human supervision faster and more accurately (Alzubaidi *et al.*, 2021).

### **2.3.3.2. Detection Networks**

The rapid success and advances portrayed by deep neural networks allowed for the development of object detection frameworks (Y. Liu *et al.*, 2021). Object detection is the combination of two processes: object localisation and classification. This task involves drawing a bounding box that contains the object or objects in the image and assigning them a label.

Elschlager and Fischer (1973) made the first steps in object recognition with a part-based model. However, the two most relevant milestones in object detection came with conventional feature extractors such as HOG and SIFT and later with deep neural networks such as AlexNet (L. Liu *et al.*, 2020; Y. Liu *et al.*, 2021). Nowadays, classification networks – with some adaptations – pose as the de facto feature extractors or backbones for object detection. Observations made by (J. Huang *et al.*, 2017) sustain this idea by establishing a link between the detector's performance and the classifier from which it was adopted. Adapting a classification network to work as a backbone usually involves removing the last layers used for classification (Y. Liu *et al.*, 2021).

The main goal of object detection is similar to that of image classification: to develop an algorithm of high efficiency and high accuracy (L. Liu *et al.*, 2020). Although these two goals

are not mutually exclusive, creating an object detection framework that performs well in both efficiency and accuracy is challenging. Consequently, the two main object detection frameworks divide into detectors that privilege speed and efficiency – single-stage detectors – and detectors that focus on accuracy – double-stage detectors – which we will explain.

### **Single-Stage Detectors**

Single-stage detectors yield object localisation and classification in one step. The two most well-known single-stage detectors are YOLO (You Only Look Once) (Redmon *et al.*, 2015) and SSD (Single Shot MultiBox Detector) (W. Liu *et al.*, 2016). These approaches handle detection as a regression problem as they predict the bounding boxes position via regressed offsets between the priors and the ground truth boxes. YOLO has suffered several improvements over the past years, but YOLOv3 (Redmon & Farhadi, 2018) is one of the most widely used. This detector divides three feature maps with different scales into a grid and assigns three bounding boxes of varying aspect ratio and size to each grid cell. During training, the network will regress the position, height and width with the help of anchor boxes or priors as well as the objectness score and class probability of each bounding box. A loss function is used during training in order to penalise the regressed values of the anchor boxes assigned with ground truth objects. SSD is also a multi-scale object detector that uses various feature maps retrieved at different network depths. It employs a small kernel at the selected features' layer to determine the class probabilities and location offset for objects. Every cell of these feature maps contains a set of varying aspect ratio and scale default boxes, where a predefined number of classes and four offset dimensions relative to the default boxes are calculated. The detector employs two different loss functions, one for localisation difference between the predicted and the ground truth boxes and another for confidence over multiple class confidences.

### **Double Stage Detectors**

Double stage detectors usually integrate two steps: first, a Region of Interest (RoI) algorithm passes through the whole image, and it selects the areas of the image where it believes there is a greater chance of having objects. Secondly, a CNN is applied to the previously known RoIs to classify them and refine their location generated in the previous pass. Faster Region-Based Convolutional Neural Network (R-CNN) (Ren *et al.*, 2016) and its predecessors – R-CNN (Girshick *et al.*, 2014) and Fast R-CNN (Girshick, 2015) – are some of the leading works in double stage detectors. In R-CNN, Selective Search (SS) (Uijlings *et al.*, 2013) executes the

region proposal phase, and a CNN classifies each proposed RoIs. This process was slow, and in the following iteration of the network – Fast R-CNN – it was inverted so that the region proposal algorithm, again SS, would only be employed on the feature maps, drastically reducing the computational cost and inference time. However, the region proposal phase with SS was still relatively slow, taking the whole model pipeline around two seconds to process an image. Faster R-CNN discarded SS and employed a small network slid over the convolutional feature maps that shares layers with the feature extractor. The output is then fed to two sibling fully-connected layers for both bounding box regression and respective classification. Later developments in this field with R-FCN (Dai *et al.*, 2016) replace the last classification fully connected layers by maintaining convolutional layers. Since convolutional neural networks are translation invariant and object detection must be translation variant, they introduced a position-sensitive pooling layer.

#### **2.4. FOD Detection with Computer Vision**

Current FOD detection systems require a sensor that can only classify foreign objects as threats. Subsequently, the final confirmation stage requires human confirmation, which, in turn, must have an official dedicated to that process (H. Xu *et al.*, 2018). More importantly, the systems mentioned in 2.2 are expensive, making them a not so good product for airports with medium and low traffic volume. On the other hand, an automatic classification system allows for better decision making on whether the detected object is harmful or not, optimising runway movement rate and reducing costs with additional specialised staff.

In order to detect and recognise FOD on aerodrome surfaces, traditional feature extractors such as HOG, SIFT, and Local Binary Patterns (LBP) (Ojala *et al.*, 2002) are not able to cope with the background variation (Cao *et al.*, 2018). Thus, data-driven methods like deep learning should be applied to deal with large amounts of data that are difficult to model analytically or have unknown statistical distributions. For example, although Fang *et al.* (2015) had significantly accurate results of 83.4% with single SVM up to 91.4% with LPB+SVM and 93.4% with HOG+SVM, they only had a 277 FOD images database and four classes. In a similar study, with five classes and 320 images, Han *et al.* (2015) introduced a mixed feature composed of SIFT and colour feature extraction – ‘mixed feature’ – resulting in a 91.6% accuracy.

On the other hand, more recent studies carried out by Cao *et al.* (2016, 2018), Liu *et al.* (2018), and Li and Li (2020) take advantage of CNN’s properties in order to detect FOD. The main characteristics and results of these studies will be explained in the subsequent sections.

#### 2.4.1. FOD Detection with Region Proposal and Spatial Transformer Network

Cao *et al.* (2016, 2018) developed a framework for FOD detection parted into two stages. In the first stage, region proposal techniques involving Region Proposal Network (RPN) (Ren *et al.*, 2016) are applied.

This technique was used in detriment of sliding windows to reduce target detection time and computing memory and increase recall rate (RR) on finding objects, enhancing the efficiency of object location. The second stage, also named ‘FOD Detector’, combines a Spatial Transformer Network (STN) “to obtain translation, scale, rotation and warping invariance and to judge which is FOD” (Cao *et al.*, 2016, p. 1) with a CNN for classification based on VGG (Simonyan & Zisserman, 2014). At the end of the process, the CNN is supposed to identify the proposed bounding boxes as screw, stone or background. To reduce the number of candidates from the RPN, the authors set three empirical rules that better fit the classes intended to be detected.

The physical support for this detection system was a vehicle employing a structure containing four digital cameras with a resolution of 2048×2048 px mounted on a trunk. In their configuration, the cameras scan a width of 5 m at a time. The dataset of images contained 12,231 images – 3,562 screws and 4,202 stones. The classification results with STN were 96.31% without fine-tuning and 97.67% with fine-tuning. Next, the proposed method was compared to other detection algorithms: Faster R-CNN, SSD and Selective Search. The results of the proposed method for false alarm rate (FAR), screw and stone RR and mAP were the best among the selected four (Table 1).

Table 1 – False alarm rate, screw and stone recall rates, and mean average precision of the tested methods (adapted from Cao *et al.* 2018, pp. 10-11).

<i>Method</i>	<i>FAR</i>	<i>Screw RR</i>	<i>Stone RR</i>	<i>mAP</i>
Faster R-CNN	11.02%	83.51%	93.84%	89.43%
SSD	8.19%	87.72%	88.63%	89.92%
Selective Search + FOD Detector	1.21%	80.63%	81.46%	96.65%
<b>RPN + FOD Detector</b>	<b>0.66%</b>	<b>96.90%</b>	<b>96.40%</b>	<b>98.41%</b>

### 2.4.2. FOD Detection with Region Proposal and Focal Loss

Although the market solutions mentioned in 2.2 are exceptional for detecting FOD larger than  $3\text{ cm} \times 3\text{ cm}$ , smaller items might go undetected. Liu *et al.* (2018) developed a framework based on Faster R-CNN, RPN and Focal Loss (T.-Y. Lin *et al.*, 2018), focusing on small targets. The proposed model is based on Faster R-CNN where DenseNet (G. Huang *et al.*, 2017) is used for feature extraction. After DenseNet-169, a RPN is applied and followed, on its last layer, by Focal Loss to achieve high-accuracy classification.

After the framework is applied, it is supposed to identify four classes: small steel balls, small screws, big screws and metal nuts. The dataset comprises 4,995 images of the mentioned classes and the experiments were undertaken during sun hours. The recall rates oscillated between 91.45% for small screws and 98.21% for big screws, averaging 95.60% for all four classes and a precision average of 98.82%. In order to have means of comparison, the authors compared their framework with other methods focusing on the feature extraction algorithm. Table 2 highlights those results.

Table 2 – Recall rates for several methods (adapted from Liu *et al.*, 2018, pp. 552-553).

<i>Method</i>	<i>Recall Rate</i>			
	<i>Steel ball</i>	<i>Big screw</i>	<i>Small Screw</i>	<i>Small nut</i>
ZF Net	55.85%	75.00%	56.48%	64.56%
VGG-16	61.11%	77.17%	72.87%	72.15%
DenseNet-169	68.18%	85.15%	78.57%	84.18%
<b>DenseNet-169 + Focal Loss + RPN</b>	<b>97.77%</b>	<b>98.21%</b>	<b>91.45%</b>	<b>95.42%</b>

### 2.4.3. FOD Detection with YOLOv3

Li and Li (2020) developed a framework with YOLOv3 (You Only Look Once) for object detection, taking advantage of the residual connections and Feature Pyramid Network (FPN) like structure. The multi-scale feature fusion method embedded in YOLOv3 is of great use in improving the detection of small-scale FOD.

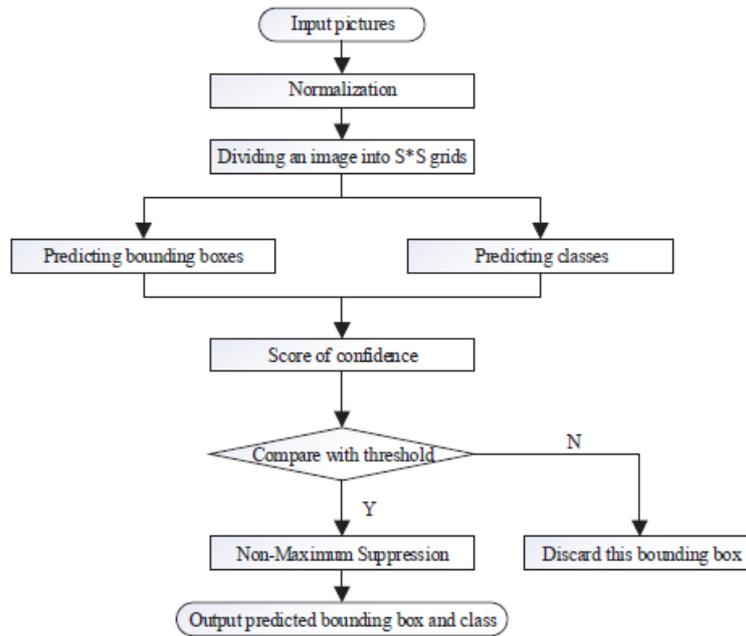


Figure 5 – FOD Detection process with YOLOv3 (Li & Li, 2020, p. 7098).

The dataset contains 2,000 images and nineteen classes, totalling an average of 100 images per class of FOD. Then, the same images are flipped, rotated, applied Gaussian noise, brightened, blurred and changed in contrast, making a 40,000 samples dataset. Finally, the proposed framework is tested and compared with other algorithms – Faster R-CNN and SSD300. Using the same hardware and software, YOLOv3 achieved the best performance in relation to the other two. The results are summarised in [Table 3](#).

Table 3 mAP, average RR and FPS results from SSD300, faster R-CNN and YOLOv3 (adapted from Liu & Liu, 2020, p. 7099).

<i>Method</i>	<i>mAP</i>	<i>Average RR</i>	<i>FPS</i>
SSD300	83,6%	85,3%	24
Faster R-CNN	80,1%	82,3%	9
<b>YOLOv3</b>	<b>92,2%</b>	<b>93,1%</b>	<b>32</b>

# CHAPTER 3

---

This chapter describes the system's architecture used to capture and compile the dataset and the software pipeline. We describe the platforms used to support the cameras and the required equipment to perform our acquisitions. The software pipeline summarily describes the steps made from image capturing until the neural networks' application. We explain the reasoning behind the NNs used and describe their architecture. For image classification, we used Xception, a recent neural network. We opted for a single-stage detector – YOLOv3 – for object detection, which delivers a good balance between inference time and accuracy.

### 3. System Architecture

The radar and electro-optical-based systems reviewed in 2.2 perform very well in several weather conditions, but they are costly and require many specialised hardware and restrictive permissions. These permissions have to do with the frequency employed by the radars, which might conflict with other frequencies used in aeronautics and building restrictions at aerodromes. On the other hand, the FOD detection systems based on computer vision models tested by the authors in 2.4 seem to perform well. Particularly, those systems do not require costly material and maintenance nor frequency allocation since they only work in the electro-optical range and are passive.

Taking into consideration the introductory paragraph and that the datasets created by the authors aforementioned are not publicly available, we created our dataset with electro-optical sensors. Moreover, we opted for the mobile platform for two reasons. Firstly, several vehicles move around the aerodrome's movement area daily. Secondly, there is no need to construct new infrastructures close to the runway, avoiding extra costs, beurocracies and safety risks. Additionally, mounting the sensors pointing to the back of the vehicle brings the advantage of them detecting an FOD that might get loose from the own platform. Another platform that could serve our interest are Unmanned Aerial Systems, but these require a specialised operator and other licenses to operate at an aerodrome.

A passive electro-optical system installed in a vehicle can be implemented in two distinct approaches. The first is to have the system collect the images, transmit, and test them at a server. The second is to have an embedded system, collecting and testing the images on the mobile platform. On the one hand, a server has more memory and computing power, which means it can handle more data and heavier NNs. On the other hand, running the neural network on a server would require data transmission between the mobile platform and the server. This would also require frequency allocation and might introduce artifacts in the images during transmission.

Conversely, the advances in small but powerful computational boards like NVIDIA® Jetson TX2 (Nvidia, 2022) allow the deployment of simple embedded systems such as the one we want to implement. However, these are not as computationally powerful as a server. In the bigger picture, these factors lead to one of our goals: building a low-cost passive system for FOD detection. In Figure 6, we can observe both training and deployment methodologies we want to employ in our system.

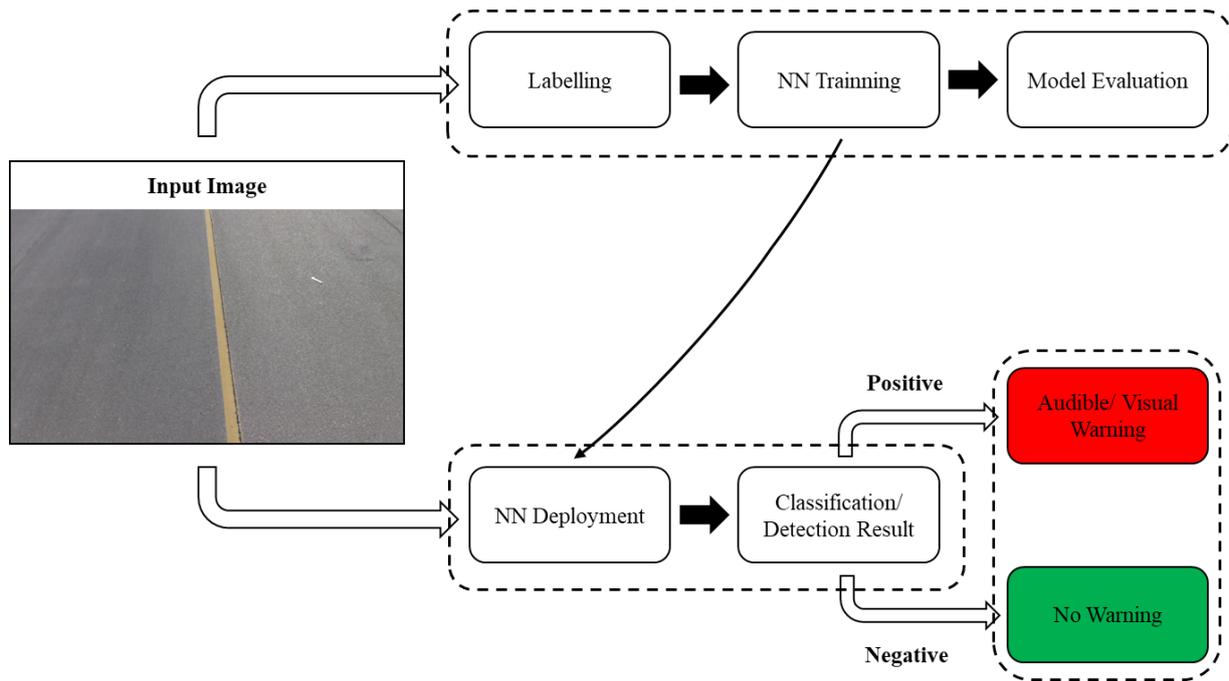


Figure 6 – System training (top) and deployment (bottom) methodologies. After labelling, training and evaluating a model, the latter should be uploaded to the system for its deployment. When the vehicle passes through an FOD in the field of view of the sensor, it should warn the driver of its presence recurring to a visual and/or audible warning.

Since this work is a first iteration and we had to create our own dataset, we needed to build a platform to meet our goals. This involves having a mobile platform that simulates as close as possible the intended deployment of the system, hardware and connections and creating a software pipeline. All the hardware used in this work was already available at the Portuguese Air Force Academy Research Centre as well as the mobile platform and electro-optical sensors.

### 3.1. Image Acquisition Platform

#### 3.1.1. Vehicle

The mobile platform used to support the cameras was a Volkswagen™ Crafter van from the Portuguese Air Force Research Centre with a modified roof to support structures on its top. The reason we chose this platform and structure is closely related to the intended implementation of the system. We want to install this solution on the vehicles – such as fire trucks – that, by their nature of operation, move across aprons, taxiways and runways. In [Figure 7](#), we can observe the mobile platform used to support the cameras during all the acquisitions of this project.



Figure 7 – Mobile platform used in our project to capture images while moving with the cameras mounted on its top.

### 3.1.2. Cameras

We used three different electro-optical sensors to obtain a larger wavelength spectrum during the data acquisition process. The cameras used were one visible spectrum, one Near-Infrared (NIR) plus Visible spectrums and a Long Wave Infrared (LWIR) spectrum.

The visible spectrum camera was a Sunny P5V27C, and it was coupled to a NVIDIA® Jetson TX2. This camera is equipped with a 1/4” OmniVision OV5693 (Sunny, 2012) CMOS sensor (sensor 01) and has a maximum resolution of  $2592 \times 1944$  px. The still resolution is 5 Megapixels, and the horizontal and vertical field of view (FoV) of  $70^\circ$  and  $45^\circ$ , respectively. The saving frame rate was 3 fps on jpg format with a  $1920 \times 1080$  px resolution from a video captured at 30 frames per second (fps) (Sunny, 2012). During all the image acquisitions, we opted for 3 fps in order to provide a good balance between storage memory space and not missing the objects – considering a speed of 15 km/h.

The visible+NIR camera was a PiCamera NoIR v2 (Raspberry Pi, 2016) connected to a Raspberry Pi v3 module (Raspberry Pi, 2017). The PiCamera, equipped with a 1/4” Sony IMX219 CMOS sensor (sensor 02), delivers a maximum resolution of  $3280 \times 2464$  px. Its still resolution is 8 Megapixels and has a horizontal and vertical field of view of  $62.2^\circ$  and  $48.8^\circ$ , respectively. The NIR capability of this camera comes from the pre-removed NIR filter, making it sensitive to radiation in that spectrum. The images are in RGB with a pink tonality. With PiCamera, we opted for image capturing instead of saving frames from the video source because of image quality. However, this module could run at the same 3 fps saving rate of Jetson TX2, from a video source, with the downside of losing considerable image quality. Because of the

method we chose, we ended with a 2 fps saving rate of 1920×1080 px resolution images. However, we must point out that during prolonged image capturing, by several times, image recording halted for more than six seconds due to the limited Raspberry Pi’s processing capacity.

A Xenics GOBI 384 camera (Xenics, 2008) (sensor 03) was used to capture the LWIR frequency band images. The wavelength captured by the sensor is 8 to 14  $\mu\text{m}$ , corresponding to most of the thermal radiation that bodies emit at ambient temperature. The horizontal and vertical field of view are 25.5° and 20°, respectively, and the resolution is 384×288 px (Xenics, 2008). Similarly to what we did with sensor 02, we captured images at 3 fps.

### 3.1.3. Setup

Sensors 01 and 02, as aforementioned, were connected to a Jetson TX2, a Raspberry Pi and (computational boards) and sensor 03 to a laptop. These platforms were connected to the internet to get a reference time from Network Time Protocol (NTP) to name the images according to their acquisition instant. The capability of working with the internet also allowed to control Jetson TX2 without peripherals – mouse and keyboard – via Secure Shell (SSH) protocol, reducing the number of cables inside the vehicle. Raspberry Pi and Jetson TX2 were connected to monitors to allow for a more effortless operation of the image capturing scripts (Figure 8).

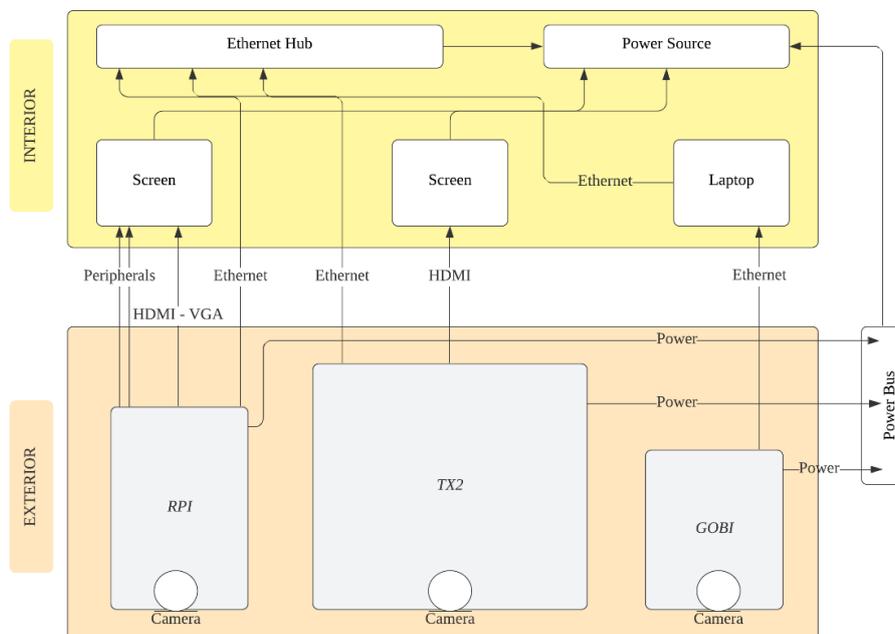


Figure 8 – Simplified setup architecture. The setup is divided into two main areas: interior and exterior. In the interior, we placed computer screens to monitor the image capturing process and help control the cameras. In the exterior, the computational boards and cameras are fixed to a support structure and connected to a power bus.

For both computational boards, we printed an individual case for weather, impact protection and securing (Figure 9 and Figure 10). All cameras were rigidly mounted to a structure mounted on the vehicle's top at 2.55 m with a fixed angle of 38° from the horizontal plane. This angle permitted a reasonable field of view of the ground and discarded any image contamination with the vehicle's structure, except its own shadow and the horizon.



Figure 9 – Raspberry Pi with 3D printed protective case and Visible+NIR sensor mounted.

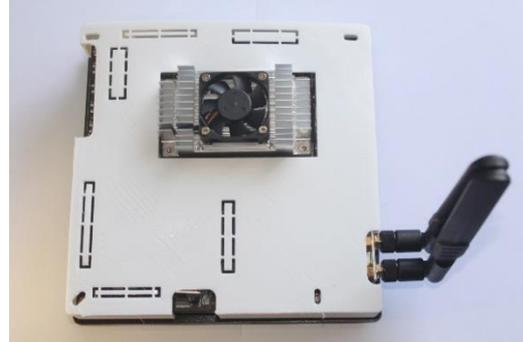


Figure 10 – NVIDIA® Jetson TX2 with 3D printed protective case.

Figure 11 is a visual representation of the areas captured by each camera considering their respective field of view, height and angle at which they were installed. In the case of sensor 01, the height of the trapezoid (in blue) is 7.42 m and the width of the larger base 12.45 m. For sensor 02, the height of the trapezoid (in red) is 9.21 m and the width of the larger base 13.08 m. The sensor 03 (in green) has the smallest field of view with a height of 2.50 m and a width of the larger base of 2.59 m.

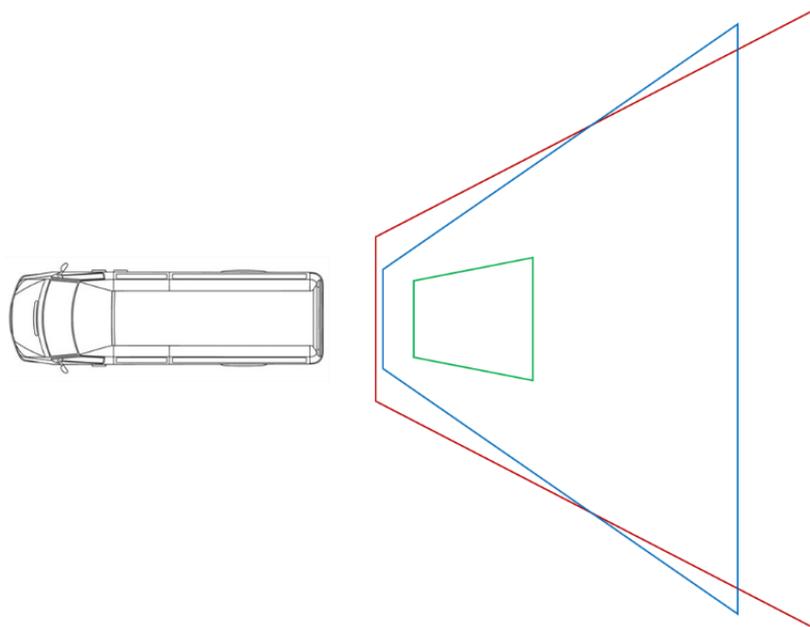


Figure 11 – Field of view of each sensor: 01 (in blue), 02 (in red) and 03 (in green). Image to scale.

The minimum speed for mobile platforms defined by FAA (2009) is 30 km/h. Considering the vertical field of view of the sensors, and for them to capture, at least, twice the same FOD, they must work, at least, at the following fps: 2.25 for sensor 01, 1.81 for sensor 02 and 6.67 for sensor 03.

### 3.2. Software Pipeline

In order to create a dataset and test several machine learning algorithms, we needed tools that could help automatise and accelerate trivial processes. This necessity mainly came from the large quantity of data that composes a dataset. The purpose of the scripts and tools ranges from the image capturing and cropping to image labelling, and they will be described in this Section. For the scripts, we used Python's versions 3.6 and 3.8 due to its great compatibility with several platforms and libraries.

At first, except for Gobi-384, the cameras need a setup script to capture and save the images from the sequences. Accordingly, we created two separate scripts; one for Raspberry Pi and another for Jetson TX2. Raspberry's script consists of opening the camera, capturing and saving the image at a rate between, but not including, 0 and 2 fps. Similarly, the script for Jetson TX2 starts recording a video at 30 fps and then we choose the rate at which we save the frames, discarding the remaining. The file naming is based on clock time from the NTP server and elapsed time from the Central Processing Unit CPU from the beginning of each capturing sequence until its end. Although more complex, Jetson's script for image capturing follows a similar pattern.

Then, we crop the images in tiles to feed the NNs, avoiding resizing, cropping, and altering aspect ratio problems. In this script, one can select the cropped image size in pixels and the horizontal and vertical overlap ratio ranging  $]0; 1]$ . We use an open-source image labelling tool – Label-Studio – to manually label all the original images we pre-selected from each camera containing FOD (Figure 12). We then export a .json file containing the coordinates of each FOD associated with an image with every image labelled. The script crops every image, whether it contains FOD or not. If the image being cropped is listed in the exported .json file, the portion of the image being cropped that contains the FOD's coordinates will be saved into a folder. The remaining tiles of those images and the images that do not contain FOD are saved to a separate folder. This way, the folders will be ready to deploy in the neural networks' scripts. In the case of the images for detection, the script works similarly, but it also creates annotation files. To this script, we only provide images with positive examples, and it only saves the tiles

which contain FODs. At the same time, it creates the annotation files of each image and saves them to a .xml file in PASCAL VOC format with the same name as the corresponding image.

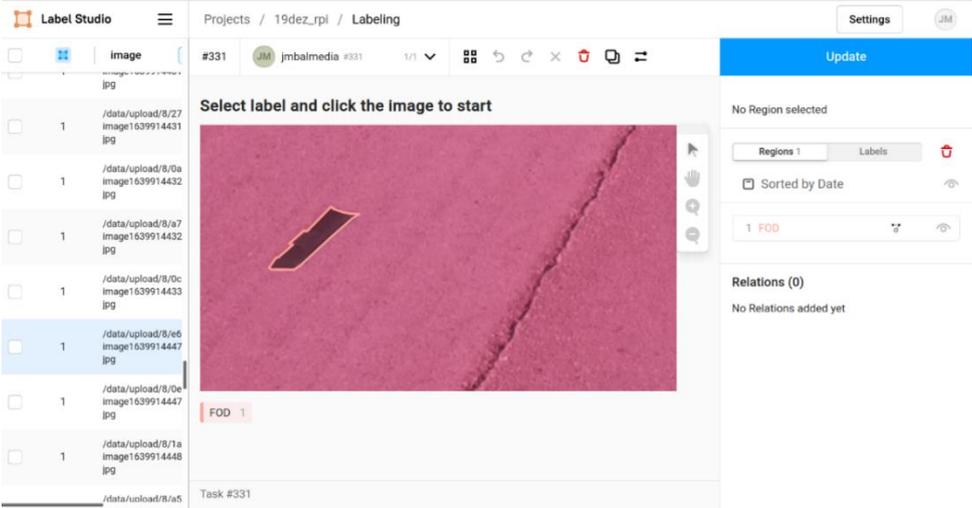


Figure 12 – Label Studio image labelling example.

The fourth step of the pipeline (Figure 13) is the neural network script. We tested two different approaches to our problem: image classification and object detection. In the first case, we created our scripts for training, evaluating, and predicting. In the second case, we cloned a GitHub repository and made the adaptations to our dataset. The last stage consists of a script to evaluate the neural networks and predict the classes of the test set.

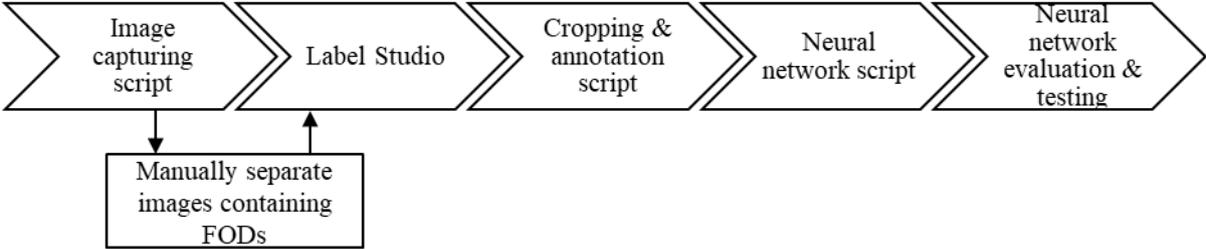


Figure 13 – Software pipeline representation.

In these scripts, we recurred to Keras, a high-level NN Application Programming Interface (API) written in Python. It is an open-source NN library which allows its users to readily specify network’s configurations, tweak training parameters, evaluate and easily debug. We also employed TensorFlow in its 2.8 version, which worked as the ecosystem on which Keras runs. TensorFlow is a framework that allows efficient parallel computation and, therefore, is adequate for NNs. Moreover, it allows Keras to run on GPU in conjunction with CUDA, which significantly expedites the training process.

### 3.3. Neural Networks

We propose to tackle this problem with supervised machine learning methods. Within supervised machine learning methods, we believe that classification and detection are the methods that better suit this problem. This has to do with several factors where speed in real-time classification and detection is necessary, and the computational board's processing capability is limited.

Moreover, a method like segmentation does not fit our interest because it would not bring any advantage in relation to detection and would be more computationally expensive. Another approach that seems adequate is anomaly detection with unsupervised learning. We tried it, but the results were not good. We believe that this resulted from poor framework implementation, which is why this approach must be revised in depth in a second iteration. The neural networks we want to work with have to achieve a general performance capable of meeting a reasonable goal within the computational and speed limitations aforementioned. In other words, considering a real-world system deployment, we want to minimise the prediction error and maximise the prediction speed or inference time.

Keeping this in mind, we will be looking at this multi-objective optimisation problem intuitively and empirically. This means that both in the cases of the classification and detection networks, they should be capable of being deployed on an embedded platform while retaining the performance adequate to the problem.

#### 3.3.1. Classification Network

More than anything, the present dissertation is establishing a benchmark for future developments using the dataset. Thereof, the choice of the classification network, although important, is not of the greatest priority. The main goal is to establish a baseline for testing whether classification suits our problem or not and comparing the performance of different strategies.

As mentioned in the previous section, we want to implement a neural network with a good trade-off between accuracy and inference time. These characteristics can be partially sorted out *a priori* by comparing the network's performance on benchmark classification datasets. We looked at the performance of the neural networks on the ImageNet contest from ILSVRC as an indicator for choosing which NNs to use. In recent years, problems related to ImageNet have been raised. Among others: labelling quality, being single-labelled on the training set, not

having adversarial examples, and overfitting to label bias (Beyer *et al.*, 2020; Stock & Cisse, 2018; Yun *et al.*, 2021). Despite this, ImageNet is one of the datasets where most neural networks are trained and tested for image classification according to Papers with Code (2022) website and is still a benchmark dataset.

We started testing the dataset with AlexNet and GoogLeNet architectures, which did not yield poor results in Digits<sup>6</sup>. While exploring more capable network configurations, we opted for Xception since it was included in Keras' library, which we were already using. Moreover, its reduced number of parameters is indicative of low memory usage, which is essential for its deployment in embedded computational boards. Additionally, its performance in ILSRVC is close to the state-of-the-art results.

### 3.3.1.1. Xception Architecture and Characteristics

For a classification network, we opted for Xception (Chollet, 2017), which stands for Extreme Inception (Ioffe *et al.*, 2016), another neural network developed by Google. The main feature of Xception is in the application and rethinking of the depthwise separable convolution (Figure 14) introduced by Sifre (2014).

This convolution comprises a depthwise convolution (channelwise) and a pointwise convolution (spatialwise), breaking the 'conventional' convolution process in two. The first step – depthwise convolution – is to apply the kernel of dimensions ( $k \times j \times 1$ ) to each channel, which results in a stack of  $n$ -channels. The second step – pointwise convolution – is to apply a kernel of dimensions ( $1 \times 1 \times n$ ). This method can drastically reduce the number of learnable parameters and the computational cost of training and testing multiplications and, consequently, connections the computers must do, making the model lighter (Lakshmanan *et al.*, 2021). In the case of Xception, there are two new features:

- The operations' order reverses: the first operation is the pointwise convolution, and the second is the depthwise;
- The inclusion or not of non-linearity between the two operations: depthwise separable convolutions rarely implement it.

---

<sup>6</sup> Digits is a web-based frontend for software that implements neural networks provided by NVIDIA®. Specifically, this tool allows the users to create their models and datasets on different neural networks.

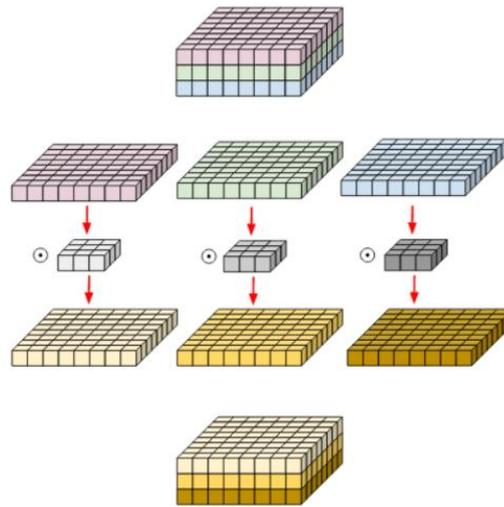


Figure 14 – Depthwise convolution. The image and filters are divided into three different channels to be convoluted separately and regrouped afterwards (Bendersky, 2018).

Firstly, we must understand that Xception derives from Inception, known for its Inception modules. These modules still apply the conventional convolution but break it into smaller steps that simplify the process. Taking an Inception module to the extreme, resembles a depthwise separable convolution. The author claims the first feature to be “unimportant” (Chollet, 2017, p. 3), but including or not non-linearity between operations has an impact. This idea is based on what is done in Inception modules as it improves convergence. According to their results, applying Exponential Linear Unit or Rectified Linear Unit between depth and pointwise convolutions degrades convergence and performance of Xception.

The architecture of the neural network is based on the depthwise convolution layers. It contains 36 convolutional layers which perform feature extraction arranged in 14 modules with skip connections inspired in ResNet, excluding the first and last modules. Batch normalization is applied after every convolution and separable convolution (Ioffe & Szegedy, 2015). As suggested in Figure 15, Xception can be divided into three major blocks: entry, middle and exit flows. Whilst middle flow repeats itself eight times, entry and exit flow do not repeat themselves. As evaluated by Chollet (2017) on ImageNet, the addition of residual connections improves convergence in matters of speed and classification results. The activation function employed in the network is ReLU, and it is applied after every convolution and pooling, except when a pooling layer immediately follows a convolution. All pooling layers are max-pool type. Optionally, fully connected layers can be added to the network prior to the logistic regression layer, which is suitable for classification tasks.

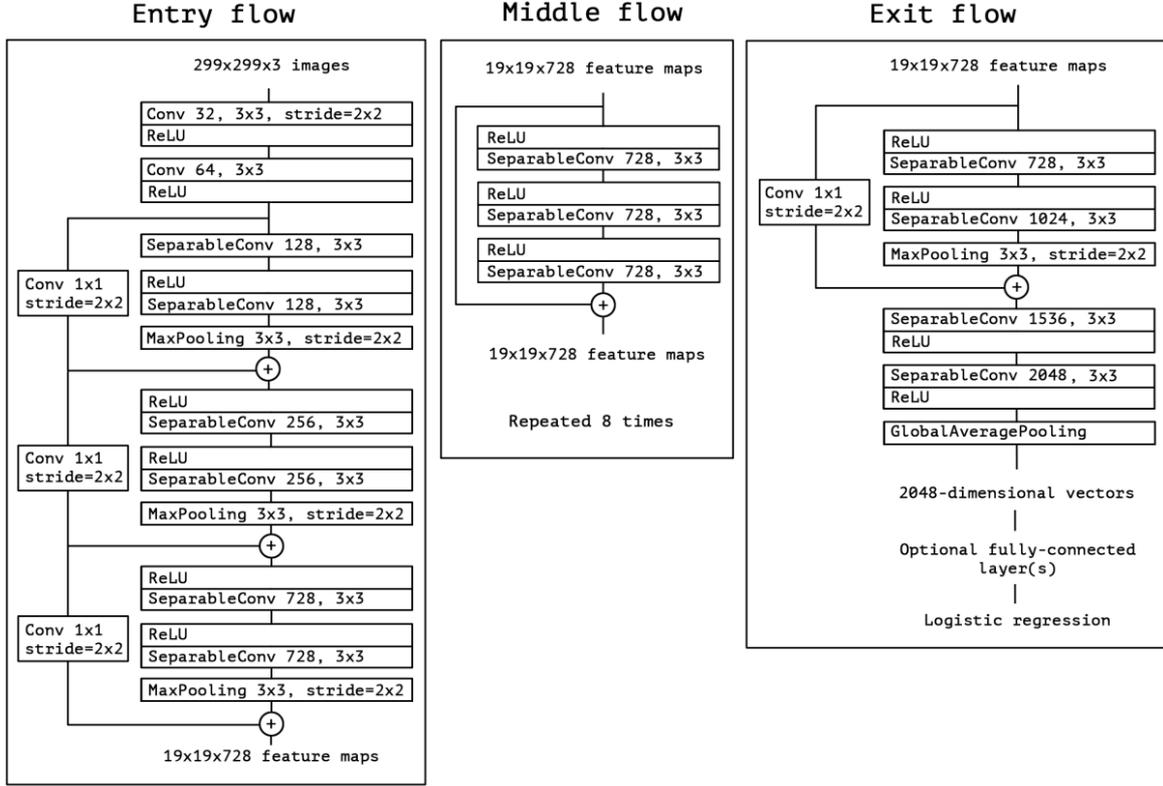


Figure 15 – Xception architecture (Chollet, 2017, p. 5).

The loss function we applied to Xception was binary cross-entropy due to the binary nature of our problem. In a binary problem, the two classes are attributed a value of 0 or 1, and the outputs – the predicted probabilities of each class – are compared to those ground truth values. Binary cross-entropy is calculated as:

$$H_p(q) = -\frac{1}{N} \sum_{i=1}^N y_i \times \log(p(y_i)) + (1 - y_i) \times \log(1 - p(y_i)). \quad (1)$$

In eq. 1,  $y_i$ , is the label associated with the input – in our case, 0 is for the label ‘fod’ and 1 for ‘no fod’ – and  $p(y_i)$  the probability attributed to the  $i^{\text{th}}$  input by the classifier. Hence, looking at the equation, only one of the terms will be activated because either  $y_i$  or  $(1 - y_i)$  will be equal to 0. The term  $\log(p(y_i))$  is the log probability of the input being ‘no\_fod’ and  $\log(1 - p(y_i))$  is its complementary.

One of the reasons why the logarithm is applied has to do with its output value. Taking the example of  $\log(p(y_i))$ , when the probability associated with a prediction is correct (=1), we do not want to penalise the loss function. Therefore, the output value of  $\log(p(y_i))$  will be

0. On the other hand, when the probability is low, we want to penalise the loss function heavily. This can be achieved with the *log* of small values. Furthermore, since  $p(y_i) \in [0,1]$ , the output of the *log* function will be negative. This will suit well the loss function since the term  $-\frac{1}{N}$  will turn the value of loss positive.

### 3.3.2. Detection Network

Object detection also presented itself as an interesting approach to our problem. We wanted to employ a single-stage framework within detection due to its inference time *versus* accuracy characteristics. Keeping this in mind, three benchmark single-stage detectors pose as good options for our problem: RetinaNet (T.Y. Lin *et al.*, 2017), SSD and YOLOv3. We selected the detector based on their results on two benchmark datasets – MS COCO and PASCAL VOC when applicable. However, we are aware that those results might be different when we apply the detectors to our dataset.

From the start, we excluded RetinaNet<sup>7</sup> as a candidate. Although it delivers good AP (Average Precision) (37.8%) on COCO, the inference time (5 fps) is undesirable for real-time object detection<sup>8</sup>. However, this detector achieves satisfactory AP on small and medium objects –  $\text{area} < 32^2 \text{ px}^2$  and  $32^2 \text{ px}^2 < \text{area} < 96^2 \text{ px}^2$  – at 20.2% and 41.1%, respectively. When the backbone network trains for longer periods, the AP increases to 39.1% and to 40.8% in the same condition if ResNeXt-101-FPN (Xie *et al.*, 2017) is used.

SSD has two main variants, which depend on the input shape: SSD300 and SSD512. The main differences between the two SSD variants lay in inference time and AP due to the input size. While SSD300 has a better inference time than SSD512, the latter has a higher AP. The inference time of SSD300 and SSD512 are 59 fps and 22 fps, respectively, on PASCAL VOC 2007. The AP of SSD512 is greater than that of SSD300 in all MS COCO metrics. The AP of SSD300 and SSD512 are 23.2% and 26.8%, respectively, both lower than RetinaNet. The AP for small and medium objects are 5.3% and 23.2% for SSD300 and 9.0% and 28.9% for SSD512 – both lower than RetinaNet.

YOLO was first introduced in 2015 by Redmon *et al.* and developed into its second version in the following year: YOLO9000 (Redmon & Farhadi, 2016). The second version of

---

<sup>7</sup> All RetinaNet results refer to RetinaNet using ResNet-101 as backbone.

<sup>8</sup> These values are valid for images of size 800×800. When the images are scaled down to 400×400, the AP reduces to 31,9% and inference increases to 12 fps. For 600×600 images, the AP is 36,0% and inference 8 fps.

YOLO outperformed the first in terms of mAP and inference time on PASCAL VOC 2007 and 2012, MS COCO and ImageNet detection set. Moreover, it surpassed SSD300 and SSD512 on PASCAL VOC 2007 in terms of mAP and inference time and performed as good as these two in PASCAL VOC 2012. However, in terms of all AP metrics on MS COCO, YOLO9000 was still surpassed by SSD512. Two years later, YOLOv3 (Redmon & Farhadi, 2018) was released and performed better than the two previous versions both in terms of AP and inference time on MS COCO test set. Furthermore, YOLOv3 achieved the same AP as SSD321 on MS COCO, but in a third of the inference time. YOLOv3 performed worse than RetinaNet in all COC AP metrics. Nevertheless, it achieved the same mAP-50 with inference times of 19 fps to 45 fps – three to four times faster for similar input sizes and equivalent GPUs. YOLOv3 achieved APs of 18.3% and 35.4%, lower than those achieved by RetinaNet 24.1% and 44.2% on small and medium-sized objects.

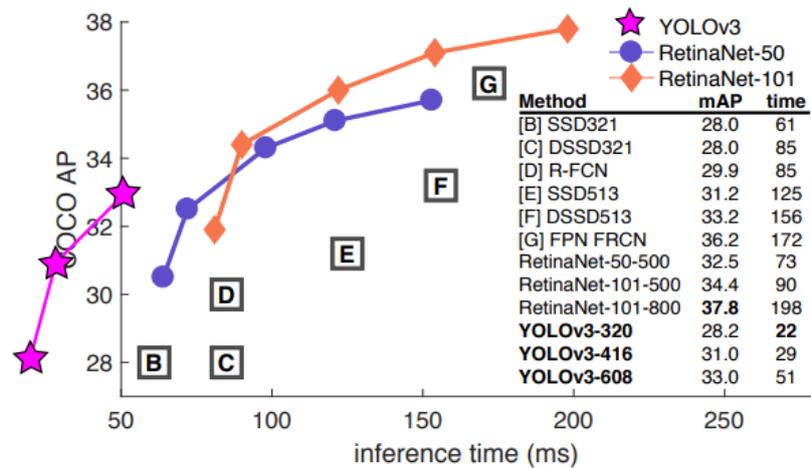


Figure 16 – Visual representation of the inference time versus MS COCO AP for YOLOv3, RetinaNet and SSD (Redmond, J. and Farhadi, A., 2018, p.1).

Liu *et al.* (2021) and Nguyen (2020) evaluated and compared one-stage and two-stage object detectors with varying backbones and input sizes for small object detection. Liu *et al.* compared Faster R-CNN, SSD and YOLOv3 on three datasets: Dataset of Object deTection in Aerial images (DOTA) (Xia *et al.*, 2018), MS COCO+SUN and Wider Face (Yang *et al.*, 2016). In terms of object size in DOTA, 57% of the instances are 10-50 px and 41% 50-300 px. The median relative area<sup>9</sup> of MS COCO+SUN dataset is between 0.08% and 0.58% (Chen *et al.*, 2017). On Wider Face, the authors only evaluated a small subset of faces smaller than

<sup>9</sup> The median relative area is the median of the areas of the bounding boxes that enclose the objects relative to the area of the image or frame where the object is in.

50×50 px on 512×512 px images. The evaluation metric employed for comparison was mAP as defined in (Everingham *et al.*, 2010).

As expected, Faster R-CNN achieved the best mAP scores because it is a double stage detector (Table 4). However, we must note that YOLOv3 had similar results, which become even more impressive when compared to another single-stage detector – SSD. Liu *et al.* do not provide information about the inference time except for Wider Face, where Faster R-CNN achieved 4 fps, SSD 18 fps and YOLOv3 40 fps. Although Liu *et al.* (2021) do not detail the input size of any of the detection networks, we can conclude that YOLOv3 performs better than its direct competitor SSD and its scores are close to Faster R-CNN – a double stage detector. Moreover, YOLOv3 is the only one capable of performing the task in real-time in one of the datasets.

Table 4 – mAP scores from three different object detectors and datasets. Adapted from Liu *et al.* (2021).

<i>Method</i>	<i>mAP</i>		
	<i>DOTA</i>	<i>MS COCO+SUN</i>	<i>Wider Face</i>
Faster R-CNN	<b>0.35</b>	<b>0.241</b>	<b>0.336</b>
SSD	0.24	0.17	0.246
YOLOv3	0.32	0.23	0.31

Nguyen *et al.* (2020) studied the performance of more detectors: YOLOv3, SSD, RetinaNet and all three versions of R-CNN. Moreover, they tested different input sizes and backbone networks on five datasets. One of the testing datasets was the MS COCO+SUN – the same as (Y. Liu *et al.*, 2021). The other datasets are four subsamples of PASCAL VOC 2007, divided by the mean relative area of the instances. From those, we will only look at the results of the ones with a mean relative area smaller than 0.58% (VOC\_MRA\_0.58) and 10% (VOC\_MRA\_10). The primary metric used to compare the methods is mAP, as defined in (Everingham *et al.*, 2010). Scoring 41.2% on mAP, Faster R-CNN has the best performance on the MS COCO+SUN small objects dataset, and it is followed by YOLOv3 608 and 416 with 33.1% and 30.28%, respectively. In single-stage detectors, excluding YOLOv3, RetinaNet had the best results and SSD the worst.

Table 5 – mAP scores of single and double stage object detectors with different backbones on MS COCO+SUN small objects. Adapted from (Nguyen, 2020).

<i>Method</i>	<i>Backbone</i>	<i>mAP (%)</i>
YOLOv3 416	Darknet-53	30.28
YOLOv3 608	Darknet-53	<b>33.1</b>
SSD300	ResNet-101	8.25
SSD512	VGG16	11.32
RetinaNet	101-FPN	25.1
RetinaNet	ResNeXT-101-64 × 8d-FPN	30
Faster R-CNN	ResNeXT-101-64 × 4d-FPN	<b>41.2</b>

On the two PASCAL VOC subsets, we discuss the same versions of the detectors. As for VOC\_MRA\_0.58, YOLOv3’s performance is similar to the slower RetinaNet but outperforms SSD by a significant score. In fact, YOLOv3 608 outperforms the double stage detector Faster R-CNN. On the other hand, on VOC\_MRA\_10, Faster R-CNN gets the best mAP (47,3%), but RetinaNet and SSD512 closely follow it. In this subset, YOLOv3’s 608 performance is close to SSD512, but the mAP score decreases by 4% when the input size changes to 416×416.

Table 6 – mAP scores of single and double stage object detectors with different backbones on the subsets VOC\_MRA\_0.58 and VOC\_MRA\_10. Adapted from (Nguyen, 2020).

<i>Method</i>	<i>Backbone</i>	<i>mAP (%)</i>	
		<i>VOC_MRA_0.58</i>	<i>VOC_MRA_10</i>
YOLOv3 416	Darknet-53	10.2	38.97
YOLOv3 608	Darknet-53	<b>11.7</b>	42.65
SSD300	ResNet-101	1.71	32.76
SSD512	VGG-16	2.9	<b>43.46</b>
RetinaNet	ResNet-101-FPN	8.95	42.5
RetinaNet	ResNeXT-101-64 × 4d-FPN	10.71	45.5
Faster R-CNN	ResNeXT-101-32 × 8d-FPN	<b>11.64</b>	<b>47.3</b>

Another interesting metric studied by Nguyen *et al.* (2020) is the inference time and amount of Random Access Memory (RAM) required for training and testing. These metrics are presented in Table 7 for both MS COCO+SUN small objects dataset and PASCAL VOC subsets. For the sake of comparison, we review the same architectures we chose for mAP scores in Table 5 and Table 6. Nguyen *et al.* (2020) do not provide data for any of SSD’s performance on these metrics, but (Pham *et al.*, 2017) evaluated their performance MS COCO+SUN small objects dataset. YOLOv3 scores the best results in all categories except on test RAM for the MS COCO+SUN small objects dataset, where SSD300 surpasses it. Moreover, YOLOv3 is the only detector capable of performing in real-time with 30 fps on MS COCO+SUN and 37 fps on PASCAL VOC subsets.

Table 7 – Performance of single-stage and double stage methods in terms of inference time, Test RAM and Train RAM on MS COCO+SUN small object dataset and PASCAL VOC 2007 subsets. Adapted from (Nguyen *et al.*, 2020; Pham *et al.*, 2017).

<i>Method</i>	<i>Inference time (s)</i>		<i>Test RAM (MiB)</i>		<i>Train RAM (MiB)</i>	
	<i>MS COCO+SUN</i>	<i>VOC</i>	<i>MS COCO+SUN</i>	<i>VOC</i>	<i>MS COCO+SUN</i>	<i>VOC</i>
YOLOv3	<b>0.0331</b>	<b>0.027</b>	1,825	<b>1,645</b>	<b>4,759</b>	<b>4,079</b>
SSD300	0.167	N/A	<b>962</b>	N/A	N/A	N/A
SSD512	0.25	N/A	2,434	N/A	N/A	N/A
RetinaNet	0.127	0.116	2,947	2,585	5,627	5,435
RetinaNet	0.229	0.222	3,767	3,641	4,961	7,723
Faster R-CNN	0.256	0.225	4,027	3,943	5,333	5,087

Our objective is to use embedded computational boards to deploy our system, and those do not have the same computing capabilities as common training setups such as desktops and servers. Given the overall computational efficiency and performance of YOLOv3 in terms of speed *versus* accuracy and the computing limitations aforementioned, we decided to implement it as our detection network.

### 3.3.2.1. YOLOv3 Architecture and Characteristics

YOLOv3 is a single-stage fully convolutional object detector which uses Darknet-53, trained on ImageNet, for feature extraction plus 53 layers for detection. The 53 on Darknet stands for 53 convolutional layers, which are arranged in consecutive  $1 \times 1$  and  $3 \times 3$  convolutional layers followed by batch normalization, Leaky ReLU and residual blocks inspired in ResNet. The architecture of Darknet-53 was originally built for image classification because of the last three layers: Average Pooling, Fully Connected and SoftMax activation function.

As presented in [Figure 17](#), these last three layers are removed when applied to object detection. In its whole architecture, YOLOv3 does not have pooling layers; instead, it applies convolutional layers of stride 2 for down sampling. This characteristic further helps detect smaller objects since it preserves low-level features, which does not happen with pooling layers.

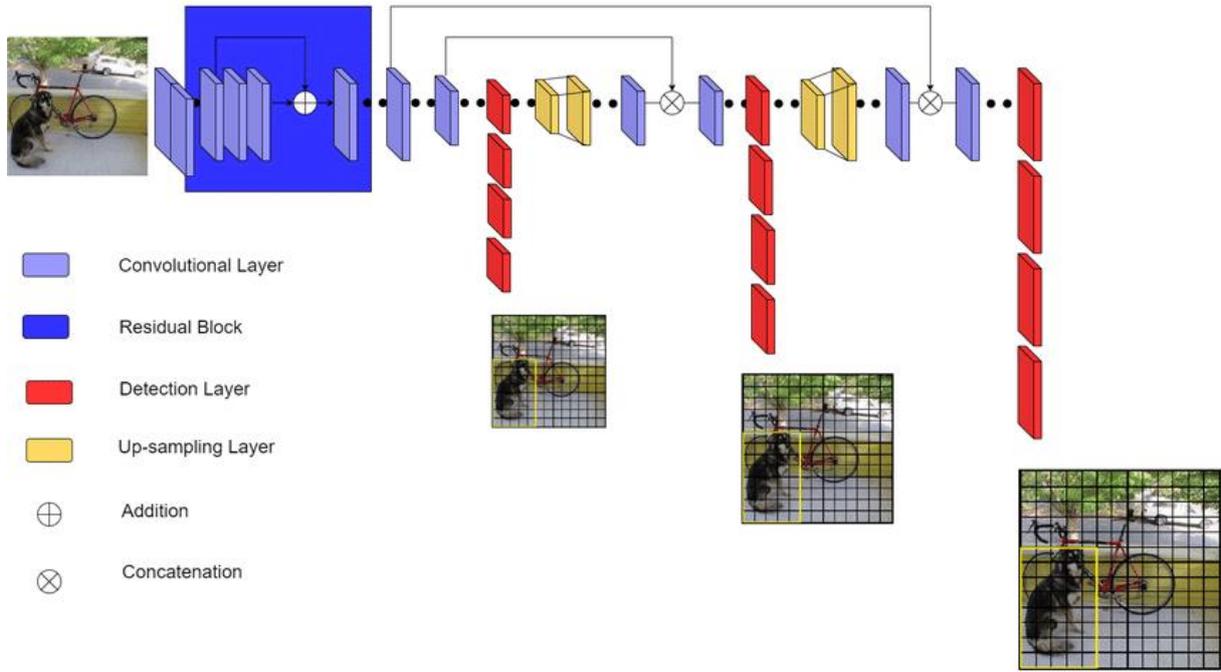


Figure 17 – YOLOv3 network architecture. When employed for detection, the Average Pooling, Connected, and SoftMax layers are removed (Valdez, 2020).

YOLOv3 is a multi-scale detector, meaning that it can detect objects at different scales in the images, similar to FPN. More specifically, the three stages at which the network makes the detections are the 82<sup>nd</sup>, 94<sup>th</sup> and 106<sup>th</sup> layers. The deeper the layer, the smaller the object it can detect since the layers' stride<sup>10</sup> decrease with successive convolutions. The concatenation between the up sampled layers in green (Figure 17) and the previous layers help detecting smaller objects by providing larger feature maps and information on finer features closer to the input image. The strides at the aforementioned detection layers are, respectively, 32, 16 and 8, which means that for an input image of 416×416 px, the resulting feature maps are, respectively, of size 13×13, 26×26 and 52×52. Subsequently, YOLOv3 applies 1×1 kernels to the feature maps of these three layers for detection. The depth of these kernels varies with the number of training classes  $c$ , bounding boxes  $B$  and attributes, which are five, as expressed in the following equation:

$$depth = B \times (5 + c) . \quad (2)$$

Then, the network will predict three different bounding boxes at each cell of the feature map generated by the kernels. These bounding boxes contain information about the predicted centre

<sup>10</sup> Ratio between the feature map and the input dimensions.

coordinates, their height and width, objectness score and a list of confidences for each candidate class. The total number of parameters generated at each feature map will be

$$\# \text{ parameters} = N \times N \times 3 \times (5 + B), \quad (3)$$

where  $B = 3$  and  $N$  is the size of the feature map. Afterwards, if the receptive field of a cell lies in the centre of an object, it will be responsible for its prediction. This is where the ground truth bounding box comes into action: to assign that centre cell as the responsible for the prediction. This architecture incorporates anchor boxes, which are selected based on a k-means clustering algorithm passed on the train annotations before training. For each of the three detection scales, the algorithm calculates three anchor boxes of varying aspect ratios and sizes. During training, the network assigns three anchor boxes to each cell of the three feature maps generated on the detection layers by the  $1 \times 1$  kernels. These will help regress the predicted bounding boxes whose parameters are given in normalized offsets – through a sigmoid function – to the anchor boxes. This technique helps reduce unstable gradients. The objectness score ( $P_0$ ) is predicted for each bounding box using logistic regression. It represents how likely the predicted bounding box is to contain an object. It varies between 0 and 1 according to the overlap between the prior and the ground truth object. The objectness score is given by

$$P_0 = P_{object} \times IoU = \sigma(t_0), \quad (4)$$

where IoU is the ratio between the intersection of the prior and the ground truth bounding boxes and their union. If the IoU is less than a given threshold, that prediction is ignored and will only affect objectness loss without changing class and coordinate losses. The value of objectness is then normalised using a sigmoid  $\sigma(t_0)$ . Following this step, YOLOv3 applies non-maximum suppression because more than one box may be detecting the same object more than once. What non-maximum suppression does is to leave out boxes with a low confidence score and select the box, among the several overlapping boxes detecting the object, with the highest score.

In order to calculate its loss value, YOLOv3 employs the sum of mean-squared errors to determine the error between the ground truth and the predictions. The function (eq. 5) incorporates three main blocks that correspond to localisation, confidence and classification losses and is written as

$$\begin{aligned}
loss = & \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
& + \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{obj} \left[ (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\
& + \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^B 1_{ij}^{noobj} (C_i - \hat{C}_i)^2 \\
& + \sum_{i=0}^{s^2} 1_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 .
\end{aligned} \tag{5}$$

The localisation loss is composed of two parts, where the first component represents the loss of the centre coordinates  $(x_i, y_i)$  and of the width and height  $(w_i, h_i)$  of the ground truth bounding box in relation to the predicted one. The width and height are square rooted to partially penalise more the errors in predictions of small bounding boxes. This technique is accompanied by the constant  $\lambda_{coord}$ , which is set, by default, to 5. Secondly, we have two terms representing the confidence loss, which measures the objectness of the box. The first term will be activated if an object is detected in the box, whereas the second will be employed if no object is detected.  $\hat{C}_i$  is the confidence score of the predicted box  $j$  in cell  $i$  and  $1_{ij}^{obj}$  will be 1 or 0 whether the  $j^{th}$  bounding box in cell  $i$  is responsible for detecting an object or not, respectively.  $1_{ij}^{noobj}$  is the complementary of  $1_{ij}^{obj}$  and  $\lambda_{noobj}$  will be lower when the model detects background and is set by default to 0.5. Finally, the classification loss is based on cross-entropy and will come into action if an object is detected because of the parameter  $1_{ij}^{obj}$ .  $\hat{p}_i(c)$  symbolises the conditional class probability for class  $c$  in cell  $i$ .

# CHAPTER 4

---

The following Chapter starts by reasoning why we did not choose to use publicly available datasets and continues to describe the characteristics of the dataset we created. We explain the selection of the objects selected as FOD and detail the procedures we followed to capture the dataset's images. Finally, we describe the three acquisitions we made during this project and the final dataset.

## 4. FOD Dataset

### 4.1. Existing Datasets

Thanks to large datasets such as ImageNet (Deng *et al.*, 2009), more and more machine learning-based approaches to real-world problem solving are possible. As aforementioned, many classes are suitable to integrate as FOD; hence many datasets would be helpful to achieve our objective. However, some problems arise when applying those datasets to the specific environment where FOD exist.

Firstly, albeit some datasets provide millions of images and thousands of classes, some corresponding to FOD, the majority of the image is occupied by the target class(es). In our problem, the object of interest represents a small portion of the image due to the practicality of scanning large areas like aerodromes' surfaces without interrupting traffic. Furthermore, since the goal of this project is to detect FOD in real-time using aerodrome vehicles, onboard cameras are limited to a set of positions.

Some authors had already addressed the FOD issue through machine learning when this project started, but most datasets are not publicly available. A private dataset that could be useful to our problem is Cao *et al.*'s (2018), whose image acquisition process is like ours. Even though it has three classes – screw, stone, and background – 7,764 out of 12,231 images (64%) contain instances of two classes – screw and stone. On the other hand, Xu *et al.*'s (2018) dataset is publicly available, but it is oriented to detect objects using material recognition which substantially differs from our objective. In the meantime, Munyer *et al.* (2022) released a large, publicly available, dataset compared to private ones, comprising 30,000 images and 31 object categories in three different lighting and two weather conditions. However, the way the images were captured does not match how we expect to deploy our system. The main reasons why are: the camera's position varies significantly during the capture and is always relatively closer to the objects as observed in [Figure 18](#). Consecutively, the sensors will not capture the objects in a similar way to how we want our mobile system to work, reason why we discarded this dataset.

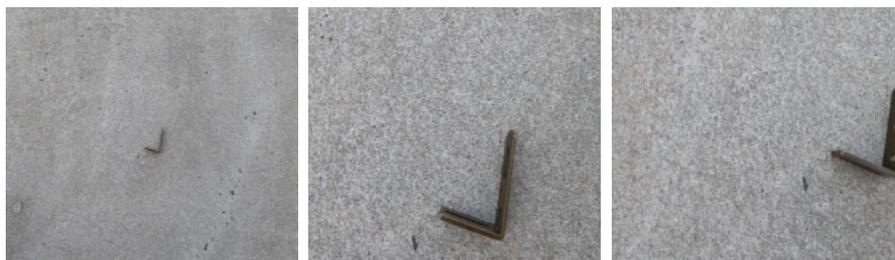


Figure 18 – Frames from Munyer *et al.*'s (2022) 'MetalPart21'. The frames are saved from a video where the camera moves around the object, providing different perspectives of it.

Categorically, we want to implement the cameras at the back of several vehicles, limiting the camera's point of view in relation to the objects (FOD) in terms of angle and height. Combining the factors and the limitations of the datasets mentioned above, we decided to create our dataset.

## 4.2. Objects Selected for the Dataset

In order to guide our selection of FOD, we relied on FAA's Advisory Circular 150/5220-24 (2009) – [Annex C](#). Consequently, we chose objects that would resemble as much as possible the descriptions given on the Advisory Circular (AC), and some other objects we found relevant based on other reports, papers and PoAF's prevention plans (ATSB, 2010; FAA, 2009; Herricks *et al.*, 2015; McCreary, 2010; PoAF, 2018). We provide a comprehensive table with the objects selected in [Annex D](#).

AC 150/5220-24 defines several parameters that the detector must achieve and objects it must detect when it comes to object detection. Firstly, the document establishes that mobile radar platforms must perform object detection at a minimum speed of 30 km/h. The system must detect a metal cylinder, a sphere like a golf ball and 9 out of 10 of a group of objects placed within a 30 m by 30 m square within the coverage area. Moreover, if the two objects are placed more than 3 m apart, they must be separately identified. These ten items must be smaller than 10 cm in any dimension unless otherwise specified. In short, the items are a 'chunk' of asphalt, portion of a runway light fixture, crescent wrench, deep socket, piece of aircraft tyre rubber, distorted metal strip, fuel cap, lug nut, hydraulic line, and PVC pipe.

For the first image acquisition, there was no specific criteria to select the objects representing FODs except that they fit into the aeronautical context and were mentioned in most FOD related documents (Cao *et al.*, 2018; Herricks *et al.*, 2015; Y. Liu *et al.*, 2018; McCreary, 2010). Some of the objects were nuts, bolts, various metals, paint, and asphalt. The FODs selected for the second and third acquisitions followed a more thorough selection process mainly based on the AC mentioned above. Most of the objects from the first acquisition were kept, and we added a few more objects to partially correspond to AC 150/5220-24's list. [Table 8](#) briefly describes the selected FOD for each capture with their dimensions and assigned class.

We did not divide the images per object class for two reasons. Firstly, the number of images is small, forcing the framework to learn their features with fewer examples. Secondly, that approach would lead the algorithm to learn each object's features and fail to detect different

objects from those provided during training. The objects used for the first capture range from object number 1 to 15, and the objects used for the subsequent captures range from number 2 to 23.

Table 8 – Object description by class and dimensions.

<i>Object No.</i>	<i>Object</i>	<i>Class</i>	<i>Width/ Diameter [cm]</i>	<i>Height [cm]</i>
1	Car Tyre	Rubber Tyre	67	23
2	Rubber Stripe	Rubber	32	8
3	Carbon Fibre	Carbon Fibre	49,5	28
4	Metal Bar	Large Metal	29	4
5	Bolt 1	Bolt	1,5	10
6	Small Metal Tube	Metal Tube	2	11
7	Paint Chip	Paint	9	8
8	Small Nut	Nut	3	1,5
9	Bolt 2	Bolt	1,5	6,5
10	Bolt 3	Bolt	1,5	8,5
11	Bolt 4	Bolt	2	2,5
12	Wrench	Tools	21	3,5
13	Bent Carbon Fibre	Carbon Fibre	21,5	5
14	Rock	Rock	8	4
15	Asphalt	Pavement	6,5	4
16	Safety Wire	Wiring	19	6,5
17	Lug Nut	Nut	3	3,5
18	PVC Tube	Tube	9	10
19	Hydraulic Line	Line	19	12
20	Brown Metal Stripe	Small Metal	13	3
21	Wide Metal Tube	Metal Tube	4	8
22	Golf Ball	Golf Ball	4,3	
23	Bent Metal Stripe	Small Metal	8	6

### 4.3. Data Acquisition

We deployed our vehicle with the cameras three times during this project to capture images to build a dataset. Along the three deployments, we made several improvements to the sequence capturing. These improvements had to do with FOD placement and securing, image capturing scripts, camera positioning and overall workflow.

The first acquisition was the simplest, where the image capturing took place with the vehicle stopped at three locations in the aerodrome. For the following acquisitions, we built a structure that allowed for capturing images while moving. Due to some events on the second acquisition, we made a third one to have a better dataset. The final dataset only contains images from the third acquisition.

All three acquisitions took place at the Portuguese Air Force Base no. 1 (LPST) in Sintra, Portugal, during different times of the day with varying cloud coverage.

#### 4.3.1. First Acquisition

The first deployment took place on October 5<sup>th</sup>, 2021. The objects representing FOD were placed in three different locations – apron, taxiway, and runway (Figure 19). The cameras were installed on the upper part of the van’s back bumper parallel to the ground.



Figure 19 – Locations of image capturing during the first acquisition (Map data ©2022, CNES/Airbus, IGP/DGRF, Maxar Technologies).

Due to it, the images are ‘divided’ into two parts ground and sky (Figure 20). This leads to two events that present downsides to our goal. Firstly, since the background is more complex, it is harder to distinguish the objects from the background, making the classification and detection tasks more challenging. Secondly, by capturing a scene with different luminosities, we will have a higher dynamic range as seen in Figure 20, which will further difficult the observation of the FODs. The image acquisition was made with the van stopped, where a sequence of ten images was taken. At the end of each sequence, one or more FODs were retrieved until none were left. These acquisitions resulted in 1,176 images where 1,082 contained FOD and 94 did not.

Table 9 – Number of FODs in each sequence per capturing location.

<i>location</i>	<i>seq01</i>	<i>seq02</i>	<i>seq03</i>	<i>seq04</i>	<i>seq05</i>	<i>seq06</i>	<i>seq07</i>
apron	15 <sup>11</sup>	14	13	10	7	5	2
taxiway	15	14	13	10	7	5	2
runway	15	14	13	10	7	4	1



Figure 20 – Image from runway sequence with 10 FODs (sensor 01). The background is complex when compared to our target – the runway –, leading to a higher difficulty in observing the objects due to the high dynamic range.

### 4.3.2. Second Acquisition

The second deployment was on December 5<sup>th</sup>, 2021. This was the first time we used the cameras’ support structure mentioned in 3.1.3 and depicted in Figure 21. It allowed us to safely capture images while the van was moving, which, in turn, allowed more images to be captured in a greater area in a short time.

In this acquisition, we made three passages exclusively on a taxiway (Figure 22) to cover its whole area – one on each edge and another on the centre line. In each passage, we covered 1,500 m at a varying speed between 10 and 25 km/h. By approaching the image capturing method to what the expected deployment of the system will be, we make the dataset more representative, increasing the chances of higher accuracy in classification and detection.

---

<sup>11</sup> No images were captured with Jetson’s camera in this sequence.



Figure 21 – Structure used to secure the cameras to the mobile platform.

The acquisition occurred during the afternoon with few to scattered cloud coverage, which is a setback in terms of lighting conditions and, consequently, lowered image quality. Moreover, due to sunlight hours, the last two sequences were captured during dusk, which further diminished the images' quality. These factors impacted the visible and NIR spectrums and the LWIR spectrum due to less solar radiation being reflected and less thermal radiation emission caused by lower temperatures. When we reviewed the images to extract the ones with FOD for labelling, we could not identify some of the objects in the images where they were supposed to be.

The wind also figured a problem because it would drag the lighter objects. This posed two problems. Firstly, losing an object on a taxiway is a major safety risk. Secondly, the objects move from their places, which hinders the capturing and can reduce the number of object occurrences on the dataset if they fall off the cameras' field of view. Taking this into account, we taped the objects to the pavement with dark tape. Another important aspect related to the image's quality was the vehicle's speed, where we noticed that at speeds close to 20 *km/h* and above, the objects would appear distorted like a blur. The setbacks described were fundamental to improving the quality of the subsequent acquisition.



Figure 22 – Location of image capturing during the second acquisition. (Map data ©2022, CNES/Airbus, IGP/DGRF, Maxar Technologies).

### 4.3.3. Third Acquisition

The third image acquisition took place in the same location as the second one (Figure 22) but during the morning and without any cloud coverage. This time, we used three different tapes – transparent, white, and black – to disguise the securing of the objects to the ground as much as feasible. Although this technique is not desirable, especially in the ‘thermal’ spectrum, we could not hide the tape from the cameras in some objects because of their size or shape. Moreover, it would be a high safety risk to let unsecured and unwatched objects on a taxiway. To further improve the safety of the acquisition and workflow, we marked the spot of the objects with stakes on the side of the taxiway.

During this acquisition, we made six passages along the taxiway – three in each direction to cover it in whole. This allowed us to capture more instances and have greater variety because the southeast-bound passages made the van’s shadow cover part of the images. Additionally, we drove the vehicle at 15 km/h to reduce the distortion and blur of objects in the acquired images.

In Figure 23, we can observe the number of occurrences of each object class by camera sensor. The camera coupled to NVIDIA® Jetson TX2 captured the most images, followed by Raspberry Pi and Gobi-384, respectively. This is a consequence of the differences in processing power of the assigned computational boards and cameras’ field of view. Bolt, nut, and metal

stripe are the classes with a bigger representation because they are the ones with more assigned objects. Some classes are not represented on the LWIR spectrum because the assigned photosensor did not have sufficient horizontal field of view to capture the corresponding objects. Moreover, we must note that in this acquisition, the large metal tube rolled out of its place before the image capturing process, meaning that the metal tube class is fully represented by the small metal tube.

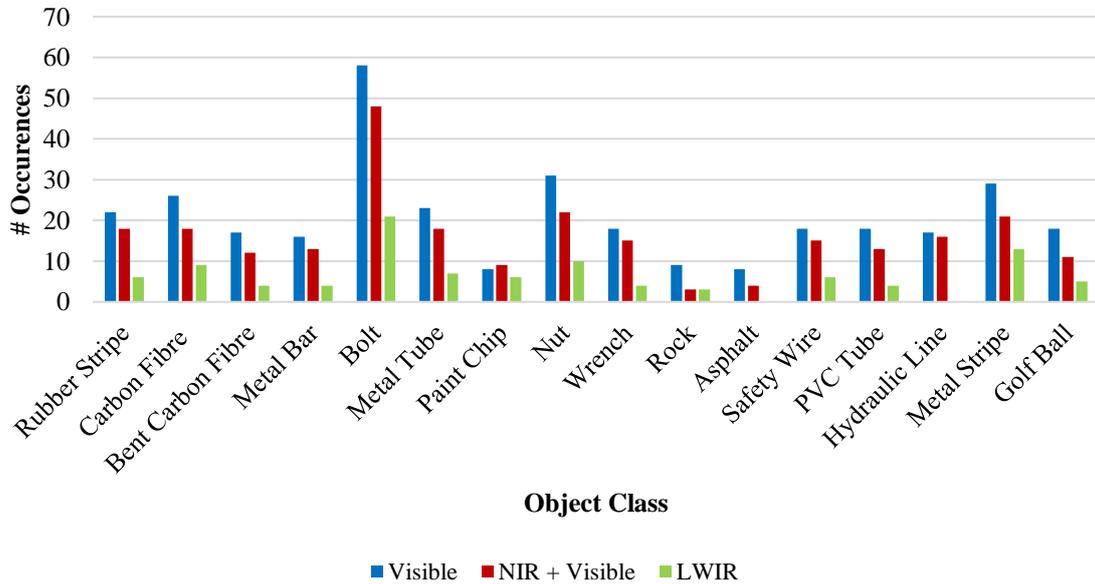


Figure 23 – Object occurrences by class and sensor for the third acquisition.

Table 10 provides a general perspective on the number of frames, labels and objects captured by each camera and the dimensions of the objects in pixels used in the dataset. From the table, we can observe that although Gobi-384 captured the largest number of frames, it lacks two labels and two objects due to the problem aforementioned. On the other hand, the remaining two cameras captured fewer images but enclose all the labels and objects. The average size, size range and standard deviation of the width and height of sequences 01 and 02 are similar. The differences are due to the largest object captured nearer the van by sensor 01 than by sensor 02. In the case of sensor 03, the values decrease because many of the objects are partially cut from the field of view. The average dimensions of the objects, as seen in Table 10, are 40×27 px, 37×24 px and 22×16 px.



of those rectangles and the image frame is less than 0.2% in every sequence and less than 0.03% in the first two sequences.

Table 11 – Area of the objects compared to the image size from the third acquisition.

Name	Spectrum	Frame		Objects' area				Original median relative area (%)
		Resolution	Area	Ave.	Median	Range	Std. Dev.	
Sen01	Visible	1920×1080	2,073,600	1,742	500	[20; 33,654]	±3834	<b>0.0241</b>
Sen02	NIR + Visible	1920×1080	2,073,600	1,475	391	[25; 31,548]	±3362	<b>0.0189</b>
Sen03	LWIR	384×288	110,592	491	200	[16; 4,371]	±838	<b>0.180</b>

There is no standardised definition of a small object in machine learning, leading some authors to take their approaches. Sambolek and Ivasic-Kos (2021) define a small object as one with a bounding box of width and height smaller than  $32^2$  px<sup>2</sup> in 1920×1080 px images. Torralba *et al.* (2008) consider 32×32 px colour images as being an acceptable minimum for accurate human recognition. Moreover, on COCO detection evaluation metrics, small objects are considered to have less than  $32^2$  px<sup>2</sup>. Chen *et al.* (2017) consider the median relative area of the objects between 0.08% and 0.58%, extracted from MS COCO and Scene UNderstanding (SUN) (Xiao *et al.*, 2010) datasets, as being small instances.

Considering the metrics mentioned above and the evidence shown in Table 10 and Table 11 and Figure 23 we can say our objects of interest are small. The dataset is publicly available on Harvard Dataverse<sup>12</sup>.

<sup>12</sup> <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi%3A10.7910%2FDVN%2FGMJPSR>

*intentionally left blank*

# CHAPTER 5

---

Chapter 5 presents the results obtained for image classification and object detection using the neural networks described in Chapter 3 and the dataset from Chapter 4. The evaluation of these frameworks is based on a subset similar to the training set and a subset of objects previously unseen by the networks. Furthermore, we describe the image preparation steps and the metrics used for evaluation. Lastly, we discuss the results from both approaches.

## 5. System Training and Testing

We conduct the training of the models on a desktop equipped with core i7 with 32GB of RAM and two GPUs. One NVIDIA® Titan XP with 12GB of RAM and another NVIDIA® 1060 with 6GB of RAM. We only used NVIDIA® Titan XP for evaluation to have the same comparison base for all results.

### 5.1. First Steps

After the first data acquisition, the images were tested on Digits to understand the viability of this project. Digits is a software for deep learning models training provided by NVIDIA® with a frontend based on a webapp. Specifically, this tool allows the users to create their deep neural network models and datasets on different NNs for image classification, object detection and segmentation.

In Digits, we used Alexnet and GoogLeNet, which are the pre-defined NNs ready for training and evaluation. The user must select the images and which data augmentation he wants to apply and tune the hyperparameters of the network, such as the loss function, learning rate, and optimizer to produce different results. This process concludes with fine-tuning the network's weights by the input images since these are already pre-trained with the ImageNet dataset.

The results were poor and preliminary. We believe that the small number of images in general and particularly the small number of images containing FODs were two of the main factors contributing to poor and erratic prediction results. However, as previously said, the position of the cameras, and consequently, the resulting images, did not represent a real scenario; thus, we discarded them. The intent of using Digits was to use the weights of the trained networks to have a working model running on a computer. However, this was a slow and meticulous process (*e.g.* changing the name of layers from dense to fully connected or determining which data augmentation was applied to replicate the new framework), so we abandoned the platform.

Nonetheless, the results obtained with the Digits models and the first dataset narrowed the path by helping to perceive what could work or not in terms of the network's hyperparameters and images.

## 5.2. Image Preparation

Except for Gobi's images, the cameras' resolution is  $1920 \times 1080$  px, which is large compared to most neural networks' input size. Moreover, feeding images this big to a NN comes at the cost of computing capacity that we do not have nor intend to implement on a low-cost embedded solution. On the other hand, the bigger the resolution of the original image, the greater the network's performance is because the detail also increases (Dong *et al.*, 2014).

Fully Convolutional Networks (FCN) (Long *et al.*, 2015) accept images of virtually any and varying size, and thus we could just feed them our original images at the cost of computing capacity. On the contrary, networks with fully connected (dense) layers – which are common in most CNNs – require fixed input size values (He *et al.*, 2014). Changing a network's architecture to accept a different input size is possible, but it is time-consuming. Two possible solutions to deal with high-resolution images are: resizing the images via down-sampling or cropping to the input size.

By sampling an image, we risk losing some information or degrading it. When up sampling, the process adds information to the image via traditional interpolation-based algorithms (Hsieh Hou & Andrews, 1978; Rifman, 1973) or deep learning-based models (Dong *et al.*, 2014; Z. Li *et al.*, 2019), which may introduce blur or amplify image noise (Z. Wang *et al.*, 2021). When down sampling, the image will, inherently, lose pixels (Yongbing Zhang *et al.*, 2011) which may cause the loss of small details (Wu *et al.*, 2015) and even quality (García Aranda *et al.*, 2021; Shorten & Khoshgoftaar, 2019; Yan *et al.*, 2019). The last scenario is particularly unwanted for our problem since the objects' size is small compared to the images. Richter *et al.* (2021a) found that the input size directly impacts classification accuracy and information processing. They further demonstrate that simply sampling images to the expected input size of the network delivers better results than feeding the original images, especially when up sampling. The same follows when zero padding is applied (Hashemi, 2019). Nevertheless, the results are different from when the original image has the same size as the networks' input size.

Changing the aspect ratio of the training set will deform the shape of the objects. When testing the model with images that were not distorted, the accuracy will drop (Zheng *et al.*, 2016). Likewise, random cropping of the image is not recommended because, in the images containing objects, we might be cropping an area that does not contain the target object (Richter *et al.*, 2021b; Shorten & Khoshgoftaar, 2019). Additionally, the context of the objects is quite

relevant (Mottaghi *et al.*, 2014; Torralba *et al.*, 2003), especially when the targets are small objects (Hu & Ramanan, 2017). Due to their constrictions, image sampling by large factors and random cropping are not desirable, and we still want to preserve the details of the original image, especially the FOD.

We solve this problem with the tiling script mentioned in 3.2, which divides the images into tiles of configurable size. It artificially reduces image size and required computational power while ensuring we keep the objects and their original features. One worrying aspect of this procedure is the loss of context. However, the background of our images is almost constant – grey asphalt and patches of white and yellow paint – meaning that there is practically no loss of context of the object. We did not crop the LWIR images because their original size (384×288 px) is close to the input size of the neural networks we used. Furthermore, we did not use these images for training and testing. This decision was made based on the small number of images with FODs, the camera's small FoV, and our difficulty finding the objects and labelling them.

Since one of the goals of the present dissertation is to determine which frameworks better suit our problem, we opted for a fixed tile size for image classification and object detection. We did this because the tiling process results in datasets of hundreds of thousand images, which require large storage space. We cropped them in 256×256 px squares with horizontal and vertical overlap ratios of 0.5. This means that the objects will now have an apparent area 31.6 times larger than in the original images. Nonetheless, the height and width will not increase in the same proportion. Referring to Table 11 we observe that the median relative area of the objects in relation to the original image area is between 0.0189% and 0.180%. By cropping the image into tiles of 256×256 px, that ratio will now range between 0.180% and 0.763% (Table 12).

Table 12 – Comparison of the median relative area of the objects for the original images and their cropped versions for image classification.

Name	Original frame	Object's area		Original median relative area (%)	Cropped frame	New median relative area (%)
	Area	Average	Median		Area	
Sensor01	2,073,600	1,742	500	<b>0.0241</b>	65,536	<b>0.763</b>
Sensor02	2,073,600	1,475	391	<b>0.0189</b>	65,536	<b>0.597</b>
Sensor03	110,592	491	200	<b>0.180</b>	N/A	<b>N/A</b>

We followed the same principle for object detection, but we divided the image into tiles of 416×416 px with horizontal and vertical overlap ratios of 0.5. The apparent size of the objects, in this case, will be increased by a factor of 12 relative to the same objects in the

original images. We can see these figures in Table 13, which shows that the range of the median relative area changes from 0.0189% to 0.180% to 0.180% to 0.289%.

Table 13 – Comparison of the median relative area of the objects for the original images and their cropped versions for object detection.

Name	Original frame	Object's area		Original median relative area (%)	Cropped frame	New median relative area (%)
	Area	Average	Median		Area	
Sensor01	2,073,600	1,742	500	<b>0.0241</b>	173,056	<b>0.289</b>
Sensor02	2,073,600	1,475	391	<b>0.0189</b>	173,056	<b>0.226</b>
Sensor03	110,592	491	200	<b>0.180</b>	N/A	N/A

### 5.3. Metrics for Evaluation

In order to evaluate the performance of neural networks, we must recur to metrics. Accuracy, precision, recall, and AP are some of the most popular. Beforehand, we must know four essential concepts that are present in binary classification problems with labelled data, which allow us to calculate those parameters:

<b>Actual Class</b>	<b>Negative</b>	True Negative (TN)	False Positive (FP)
	<b>Positive</b>	False Negative (FN)	True Positive (TP)
		<b>Negative</b>	<b>Positive</b>
		<b>Predicted Class</b>	

Figure 25 – Confusion matrix for binary classification problems.

The accuracy of a model depicts the ratio of predictions correctly classified by the model out of all predictions. This metric is widely used in deep learning, but one must be careful when applying it to unbalanced datasets due to the number of TN, as we can see in the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\# \text{ correct predictions}}{\# \text{ predictions}} \quad (6)$$

Recall measures the model's aptitude to correctly detect an object, TP, out of all possible correct predictions, TP and FN, as shown in eq. (7).

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{\# \text{ ground truths}} . \quad (7)$$

Precision is different from recall, and it measures the correctness of predictions. It returns the ratio between the correct predictions (TP) and all the predictions made by the model (TP and FP) as shown in eq. (8).

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{\# \text{ detections}} . \quad (8)$$

The precision-recall curve helps analyse the trade-off between a sensitive model that produces detections with low confidence and a model whose detections are scarcer but with more confidence. This plot aids in understanding the model’s performance across various thresholds. One advantage of the precision-recall curve is that it allows for efficiently analysing models when the class distribution of the dataset is imbalanced (Davis & Goadrich, 2006). This happens because precision and recall do not consider TN. Here, the recall is on the x-axis and precision on the y-axis. The precision-recall graph’s Area Under the Curve (AUC) is the average precision (AP), a good tool to compare models. This metric indicates whether the model can correctly classify the positive examples without classifying many false ones as positive.

IoU is a metric used for detection. It computes the ratio between the intersected area of a ground truth bounding box (A) and the predicted bounding box (B) and the union of those same boxes (eq. 9). The result of the IoU is then compared with a predefined threshold to determine if the detection is valid. The most common threshold used in detection networks is 0.5.

$$IoU(A, B) = \frac{A \cap B}{A \cup B} . \quad (9)$$

When comparing the performance of detection models, the AP is one of the most widely used metrics. However, it is common to use the AP in conjunction with IoU, which defines a minimum threshold where detections with IoU below that value do not count as TP. The two benchmark competitions – VOC and MS COCO – employ different definitions of AP. In the case of VOC, the AP is the AUC of an adjusted version of the precision-recall curve, which we will explain in detail next. In MS COCO, the AP is interpolated by 101 equally spaced points. The metric used in MS COCO is called AP, but they do not distinguish it from mAP. The definition of mAP is the weighted sum of the APs of every class. In our case, we will not use mAP since our problem is binary.

Nevertheless, there is another difference between the APs of VOC and MS COCO. In VOC, the AP is calculated with a unique threshold of IoU=0.5, whereas in MS COCO, the primary metric employs IoUs ranging from 0.5 to 0.95. We will recur to VOC’s definition. Firstly, VOC 2010 calculates the AP “by setting the precision for recall  $r$  to the maximum precision obtained for any recall  $r' \geq r$ ” (Everingham & Winn, 2010). Lastly, it calculates, via numerical integration, the area under the newly obtained curve as shown in equations 10 and 11.

$$AP = \sum_{n=0} (r_{n+1} - r_n) \times p_{interp(r_{n+1})} \quad (10)$$

where

$$p_{interp(r_{n+1})} = \max_{\tilde{r}: \tilde{r} \geq r_{n+1}} p(\tilde{r}) \quad , \quad (11)$$

and  $p(\tilde{r})$  is the precision measured at recall  $\tilde{r}$ .

### 5.3.1. Xception Implementation and Results

We implemented Xception based on Keras Applications, where we import a library with the network model trained on ImageNet. In our case, we opted for transfer learning since the dataset is small. We used balanced and imbalanced subsets of the datasets where both have two labels: ‘fod’ and ‘no fod’. We present in [Table 14](#), divided into balanced and imbalanced subsets, the classes and the corresponding number of images per class. On the imbalanced subsets, we used five times more images of ‘no fod’ than of ‘fod’.

We trained the model on seven different subsets with the same tile size of 256×256 px. The subsets are composed of images from a single or two sensors. The number of tiles is limited by the quantity of the tiles labelled with ‘fod’. The train/validation/test split is in a proportion of 80/10/10, respectively. The choice of training on imbalanced subsets was to have more images since, presumably, more images provide better results. The proportion of ‘no fod’ to ‘fod’ on the imbalanced datasets is 5:1, reason why we changed the bias of the class ‘fod’ to 5. We also created a balanced subset (1+2)\* with half of the images from subsets 1 and 2 in order to test if a combination of images from two sensors would generate better results. The names of the subsets are directly linked to the number attributed to each sensor in [3.1.2](#).

Table 14 – Number of images per label used in each dataset for classification.

<i>subset(s)</i>	<i># images per label</i>		<i>train</i>		<i>validation</i>		<i>test</i>	
	<i>fod</i>	<i>no fod</i>	<i>fod</i>	<i>no fod</i>	<i>fod</i>	<i>no fod</i>	<i>fod</i>	<i>no fod</i>
1	1,264	1,263	1,011	1,011	126	126	127	126
2	951	961	761	773	95	94	95	94
1+2	2,215	2,224	1,772	1,784	221	220	222	220
(1+2)*	1,105	1,111	885	891	110	110	110	110
1	1,264	5,811	1,011	5,054	126	631	127	126
2	951	3,994	761	3,805	95	475	95	94
1+2	2,215	10,185	1,772	8,859	221	1,106	222	220

We conduct for an unlimited number of epochs until convergence, where we would monitor the validation loss with a patience of 50 epochs for balanced subsets and 25 for imbalanced ones. This means that if the value of the validation loss did not decrease below the lowest registered value for 50 or 25 consecutive epochs, the training would halt. We reduce the patience for the second case to accelerate the training process since more images take more time to train.

To evaluate the performance of a model during training, we analyse their train and validation loss curves along the epochs. Additionally, we evaluate the model on a balanced test subset for its loss and accuracy values. At first, we trained the network on the balanced subsets with the same learning rate. The focus here was on understanding which data augmentation would produce good results without considerably increasing training time. Taking that into account, we figured that normalising the input values of the pixels, flipping the images horizontally and setting the mean value of the dataset to zero and its standard deviation to one in the different channels would produce better results without changing the training time by a large margin. We double the number of images on the dataset with the horizontal flipping.

Additionally, we tested whether freezing every layer of the network except the classification head would bring better results. It delivered worse results in terms of accuracy and loss than not doing it so. Lastly, and with the variables mentioned above, we tested three learning rates –  $1 \times 10^{-3}$ ,  $1 \times 10^{-4}$  and  $1 \times 10^{-5}$  – to understand which one produces the best accuracy and loss values. A summary of the best results obtained with each subset of images for the trained models is in [Table 15](#).

Table 15 – Summary of the results obtained during training and testing for each model.

<i>subset(s)</i>	<i># epochs</i>	<i>test set</i>		<i>learning rate</i>	<i>fps</i>
		<i>accuracy</i>	<i>loss</i>		
1	23	0.9603	0.1310	1.00E-05	
2	34	0.9521	0.2522	1.00E-04	<b>90.9</b>
1+2	13	<b>0.9614</b>	<b>0.1291</b>	1.00E-05	
(1+2)*	32	0.9500	0.2259	1.00E-05	
1	11	0.9839	0.0642	1.00E-04	
2	15	0.9728	0.1366	1.00E-04	<b>90.9</b>
1+2	24	<b>0.9886</b>	<b>0.0338</b>	1.00E-04	

Given the results in [Table 15](#), we can conclude that the dataset which contains all the images from the two balanced subsets delivers the best results in terms of accuracy and loss. However, we believe that this result is linked to the fact that that subset contains more images than the rest because the subset (1+2)\* presents the worst results of all subsets. Nonetheless, all models seem to have a good performance considering that their results are similar. Comparing subsets 1 and 2, the latter has lower results on both balanced and imbalanced subsets. However, we can notice that the number of epochs until convergence is low, which we believe that is directly linked to the small number of images on the subsets. In [Figure 26](#), we can observe the training and validation plot for accuracy and loss values of the model with the best result.

Analysing the plots, we see how the train loss curve is approximately monotonically decreasing. On the other hand, the validation loss curve oscillates around a fixed value from start to finish, indicating an under representative validation set or overfitting. We run the training with automatic data split between the training and the validation sets, which means that the composition of these subsets varies with each model we train. This technique should eliminate the effect observed, but, analysing the curves of other models, the validation line does not change much from what we see in [Figure 26](#). In our perspective, the problem has to do with overfitting, the limited number of images with FODs and little data augmentation. This allows the model to perform very well in the training examples – high train accuracy –, losing generalisation capability.

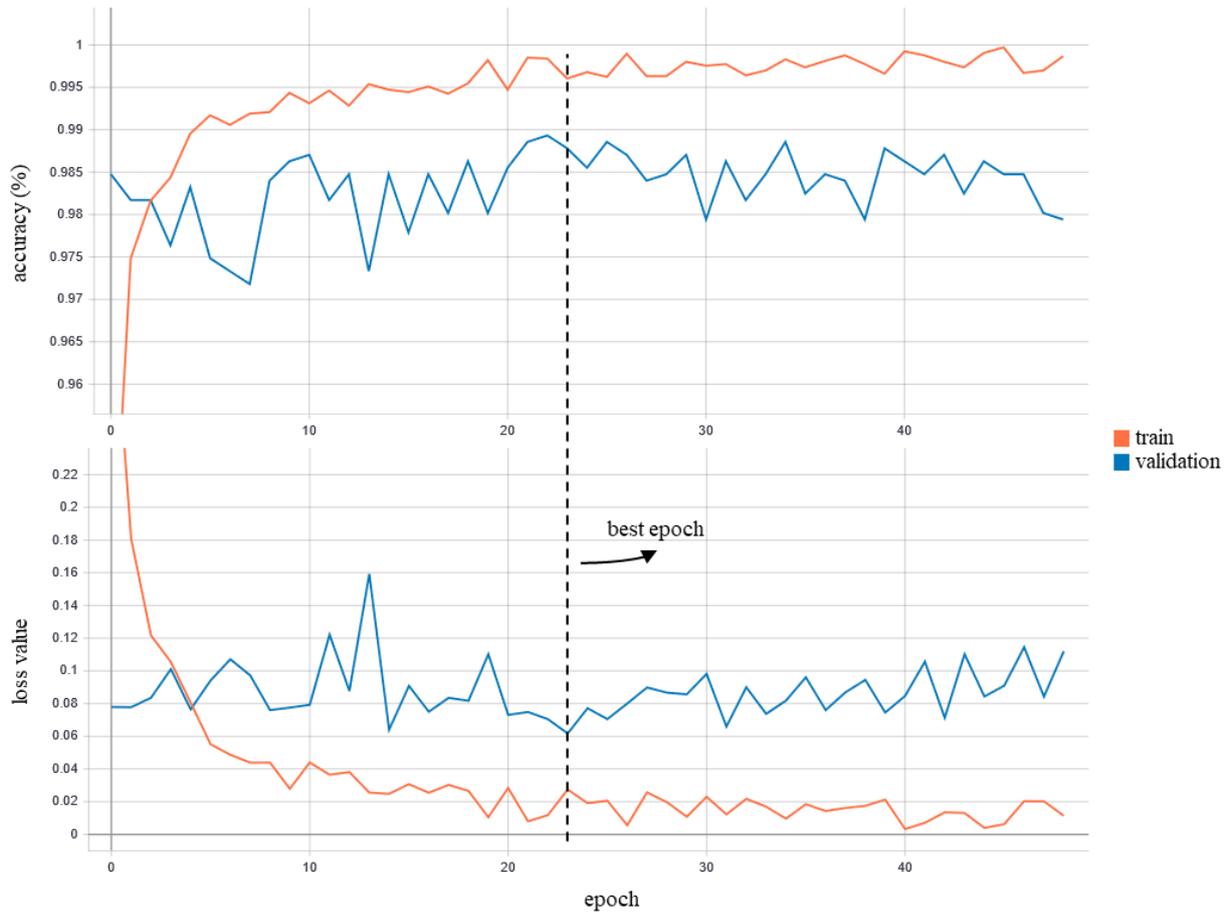


Figure 26 – Training plot of the curves of loss and accuracy for training and validation.

### 5.3.2. YOLOv3 Implementation and Results

In this project, we use Python for our scripts and TensorFlow and Keras as libraries, but Darknet-53 is written in C, which raises some difficulties. While Python is a high-level object-oriented programming language, C is a middle to high-level one. This means that Python is more programmer-friendly and less memory efficient than C. The loss of memory efficiency at the present stage of this project is not as relevant as the programming language is simpler to use, so we opt for a Python version of YOLOv3. We used @experincor's /keras-yolo3<sup>13</sup> version available in his GitHub repository.

As mentioned in 3.3, the tile size employed for object detection purposes was 416×416 px. This image size allows YOLOv3 to infer in real-time (Figure 16) while scoring good AP results compared to 608×608 px tiles (Table 5 and Table 6). Like we did for Xception, we opted for transfer learning with pre-trained weights from MS COCO. For training, we only

<sup>13</sup> <https://github.com/experincor/keras-yolo3>.

provided tiles containing FODs. The train/validation/test split is in a proportion of 80/10/10, respectively. We tested the model in four different subsets based on each sensor, as presented in [Table 16](#).

Table 16 – Number of tiles employed in each set for object detection.

<i>subset(s)</i>	<i># images per set</i>			
	<i>total</i>	<i>train</i>	<i>validation</i>	<i>test</i>
01	1,002	800	101	101
02	746	596	75	75
1+2	1,748	1,396	176	176
(1+2)*	872	698	87	87

Similarly to what we did in Xception, we conducted the training for an unlimited number of epochs until convergence, where we would monitor the loss value with a patience of 15 epochs. We did not change the default settings from the repository’s model in terms of data augmentation. This process includes random scaling, horizontal flipping, distortion and sampling plus cropping. The first parameter we tested was the initial learning rate between  $1 \times 10^{-2}$  and  $1 \times 10^{-3}$ , where the latter delivered the best results for AP. Next, we tried different combinations of learning rate decay and corresponding patience. The learning rate would decay by a constant factor if the loss value of the model did not decrease for a given number of epochs. The best combination of decay factor and patience was 0.5 and 7. In the case of YOLOv3, we did not change the pre-defined data augmentation. The best results for each dataset and the respective training parameters are displayed in [Table 17](#), where the AP is for an IoU=0.5.

Table 17 – Summary of the model's performance on the test set.

<i>subset</i>	<i># epochs</i>	<i>initial learning rate</i>	<i>AP (%)</i>	<i>fps</i>
1	150	0.001	83.79	
2	153	0.001	83.91	
<b>1+2</b>	<b>166</b>	<b>0.001</b>	<b>91.08</b>	<b>11.5</b>
(1+2)*	170	0.001	76.34	

Analysing the results in [Table 17](#), we can see that the subset with more images got the best results, scoring a 91.08% AP at 11.5 fps. The subset (1+2)\* achieves the lowest AP, which means that mixing images from the two sensors does not provide better results than single sensor subsets. The difference between subsets 1 and 2 is residual, and we can say that they perform equally. Moreover, given the limitation established by the FAA of 30 km/h for mobile platforms in conjunction with capturing the FODs at least twice, we can say that the system can work in real-time.

Table 18 – Model's fps versus minimum required system's fps.

<i>subset</i>	<i>required fps</i>	<i>model's fps</i>
1	2.25	
2	1.81	
1+2	2.25	<b>11.5</b>
(1+2)*	2.25	

Furthermore, we tested different IoUs in order to study their impact on the AP. Table 19 summarily describes the results obtained. As expected, as the IoU grows, the AP decreases. Figure 27 shows that until an IoU of 0.5, the AP remains above 90% but starts decreasing linearly after that.

Table 19 – Average precision for each IoU on the images of the third acquisition.

<i>IoU</i>	<i>AP (%)</i>
0.3	95.88
0.45	91.17
<b>0.5</b>	<b>91.08</b>
0.6	75.12
0.7	58.55
0.75	44.77

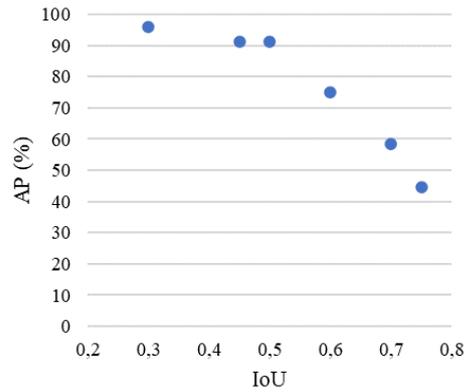


Figure 27 – Plot of the AP versus IoU on the images of the third acquisition.

In Figure 28 and Figure 29, we present some of the results of the best model on the test set. We can observe how the model does not struggle to identify objects closer to the edges nor misses some smaller objects that blend well with the background, especially cracks. An overall qualitative evaluation of the results allows us to say that they are good since most of the false negatives and false positives are difficult to be correctly classified by a human. We believe that these results are the product of an overfitted model.

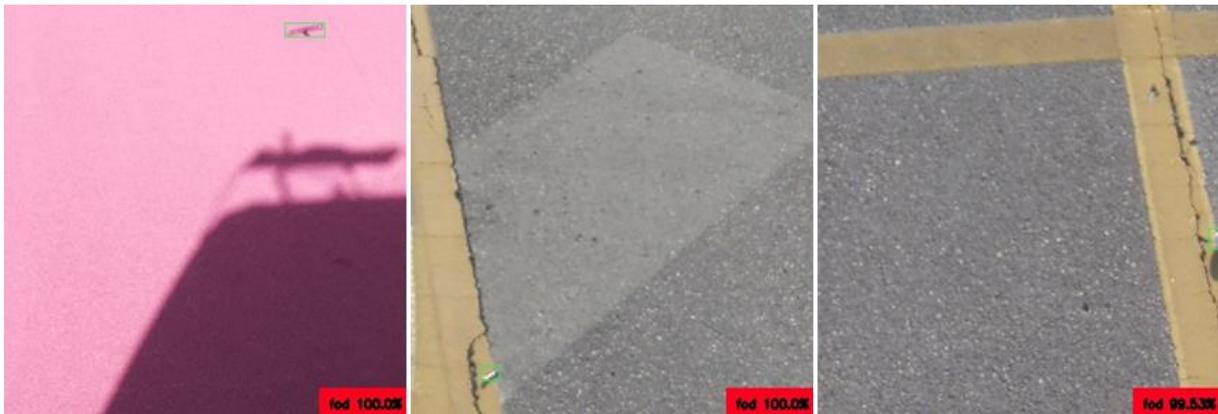


Figure 28 – Correct detections of FODs by the classifier. It can detect small objects on the edges with a high degree of certainty, as seen in the middle and right images (sensors 02 and 01).

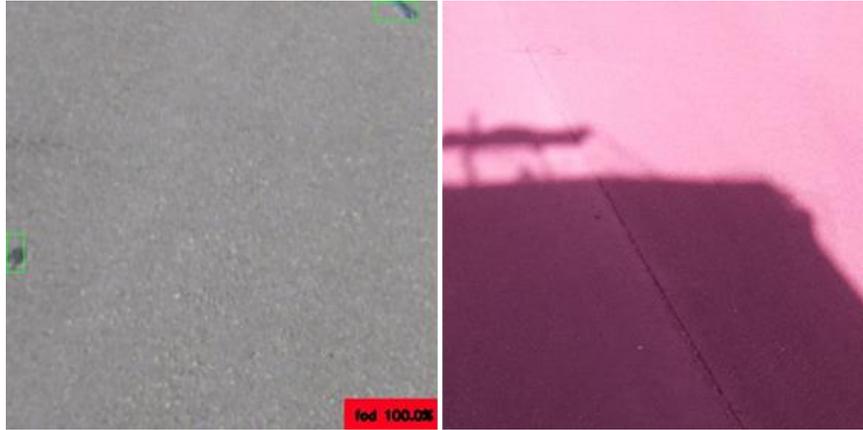


Figure 29 – Example of a false positive detection on the left image (sensor 01) and false negative detection on the right image (sensor 02).

#### 5.4. Evaluation with New Objects

Given the results obtained in the previous sections, we wanted to test the robustness of the models by analysing their reaction to previously unseen objects. We did this because we believe that the images from the test set do not differ much from the images on the training set, leading to the results observed in 5.3.1 and 5.3.2.

Considering operational restrictions at the aerodrome, we could not accomplish the acquisition at an aerodrome’s surface such as a runway or taxiway. Consequently, we collected new samples on a road-like surface to imitate the background of the previous acquisitions as close as possible. This acquisition took place on May 24<sup>th</sup> during the afternoon under clear sky conditions. We only captured images with sensors 01 and 02. In Table 20, we provide a summary description of the objects used during the acquisition – see Annex E for a detailed description.

Table 20 – Object description by class and dimensions of the fourth acquisition.

<i>Object No.</i>	<i>Object</i>	<i>Class</i>	<i>Width/ Diameter [cm]</i>	<i>Height [cm]</i>
1	Tree Branch	Organic	86	60
2	Metal Plate	Small Metal	7	7
3	Bolt	Bolt	1.5	10
4	Plastic Tube	Plastic Tube	3	13
5	Bent Metal	Small Metal	4	4
6	Grass Cutter Wire	Plastic Line	23	14
7	Cloth	Textile	17	10
8	Wooden Stick	Organic	19	2

Table 21 provides a general perspective on the number of frames, labels and objects captured by each camera and the dimensions of the objects in pixels used in this acquisition. The table shows that sensor 01 captured the largest number of frames, and sensor 02 failed to capture one of the objects. The average dimensions of the objects, as seen on, are 43×31 px and 70×51 px.

Table 21 – Capture sequence and objects’ characteristics of the fourth acquisition.

Name	Spectrum	Resolution	Frames	Labels	Objects	Objects’ width			Objects’ height		
						Ave	Range	Std. Dev	Ave	Range	Std. Dev
Sen01	Visible	1920 × 1080	237	7	8	43	[4;227]	±35	31	[9;278]	±49
Sen02	NIR + Visible	1920 × 1080	186	7	7	70	[13;320]	±74	51	[7;276]	±53

Calculating the median area of the rectangles that enclose the objects (Table 22), they take up a small portion of the image frame. The ratio between the median area of those rectangles and the image frame is less than 0.62% in both sequences. Comparing these values to the ones obtained from the third acquisition, we can see that the median relative area of the objects captured by sensor 02 more than tripled. This is due to a new, larger, object – tree branch – which was captured in its whole closer to the sensor, which is observable on the objects’ area maximum range and respective standard deviation. On the other hand, the value of the sensor 01 remained almost the same.

Table 22 – Area of the objects compared to the image size from the fourth acquisition.

Name	Spectrum	Objects’ area				Original median relative area 3 <sup>rd</sup> acquisition (%)	Original median relative area 4 <sup>th</sup> acquisition (%)
		Ave.	Median	Range	Std. Dev.		
Sensor01	Visible	2,958	589	[72; 62,828]	±8,861	<b>0.0241</b>	<b>0.0284</b>
Sensor02	NIR + Visible	7,278	1,276	[91; 88,320]	±16,565	<b>0.0189</b>	<b>0.0615</b>

We employed these datasets only for testing the two best models obtained in 5.3.1 and 5.3.2. In other words, we only used the models that resulted from subset 1+2, so the test subset is composed of a balanced number of images with and without FODs.

#### 5.4.1. Results on the new Dataset with Image Classification

The subset used for image classification is limited by the number of images with instances of FODs. Like we did before, we divided the images into tiles of 256×256 px with a horizontal

and vertical overlap ratio of 0.5. From here, resulted 1,134 images for testing which contain FODs and 1,132 that do not, where 721 and 720, respectively, correspond to sensor 01 and the remaining to sensor 02.

Referring to [Table 23](#) we observe that the median areas of the objects in the original image are 0.0284% and 0.0615%. By cropping the image into tiles of 256×256 px, that ratio will now range between 0.899% and 1.95%. Concerning the third acquisition, the median relative area of the objects in the tiles from the fourth acquisition is larger.

Table 23 – Comparison of the median relative area of the objects for classification on the third and fourth acquisitions.

Name	Object's area		Original median relative area 4 <sup>th</sup> acquisition (%)	New median relative area 3 <sup>rd</sup> acquisition (%)	New median relative area 4 <sup>th</sup> acquisition (%)
	Average	Median			
Sensor01	2,958	589	0.0284	<b>0.763</b>	<b>0.899</b>
Sensor02	7,278	1,276	0.0615	<b>0.597</b>	<b>1.95</b>

We can see a drop in the classifier's performance by testing the images from the fourth acquisition on the trained model. This result was expected since most of the objects presented to the NN are novelties. However, the obtained result is uplifting because we consider that a drop to 77.92% of accuracy is a good result considering the size of the training dataset and the shape of training curves obtained for the model. In [Table 24](#), we compare the result obtained with the test set from the third acquisition and the test set from the fourth one with accuracy and fps.

Table 24 – Comparison of the performance of the classifier in the test subset of the third and fourth

	accuracy	fps
3 <sup>rd</sup> acquisition	98.86%	90.9
4 <sup>th</sup> acquisition	<b>77.92%</b>	

In [Figure 30](#) and [Figure 31](#), we can visually interpret the classifier's results by looking at the images provided. Objects similar to those used in the third acquisition, such as the bolt, the plastic tube and the metals are correctly classified ([Figure 30](#)). It is clear that the model tends to classify novelties as FODs. As depicted in red in [Figure 31](#), plants and cracks tend to be detected over other, more dangerous, FODs such as the tree branch in orange. This particular result is not as bad as it would seem at first sight because loose plants may constitute an organic FOD, especially in large quantities.

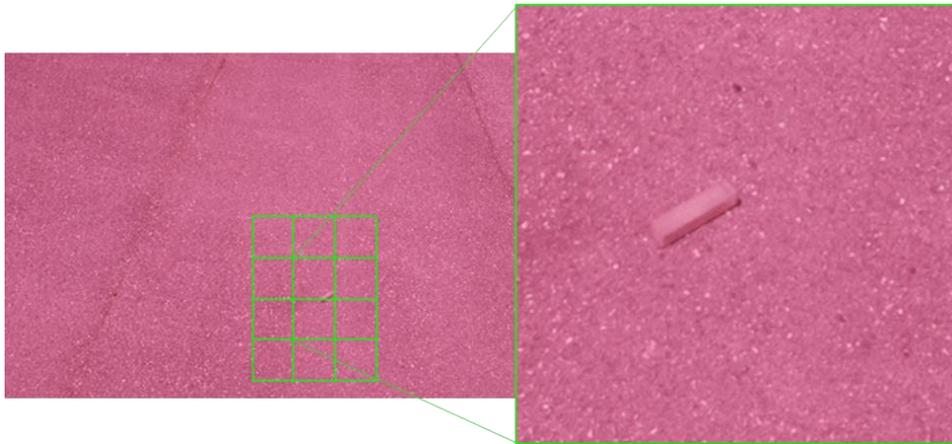


Figure 30 – Correct classification of the plastic tube (sensor 02).

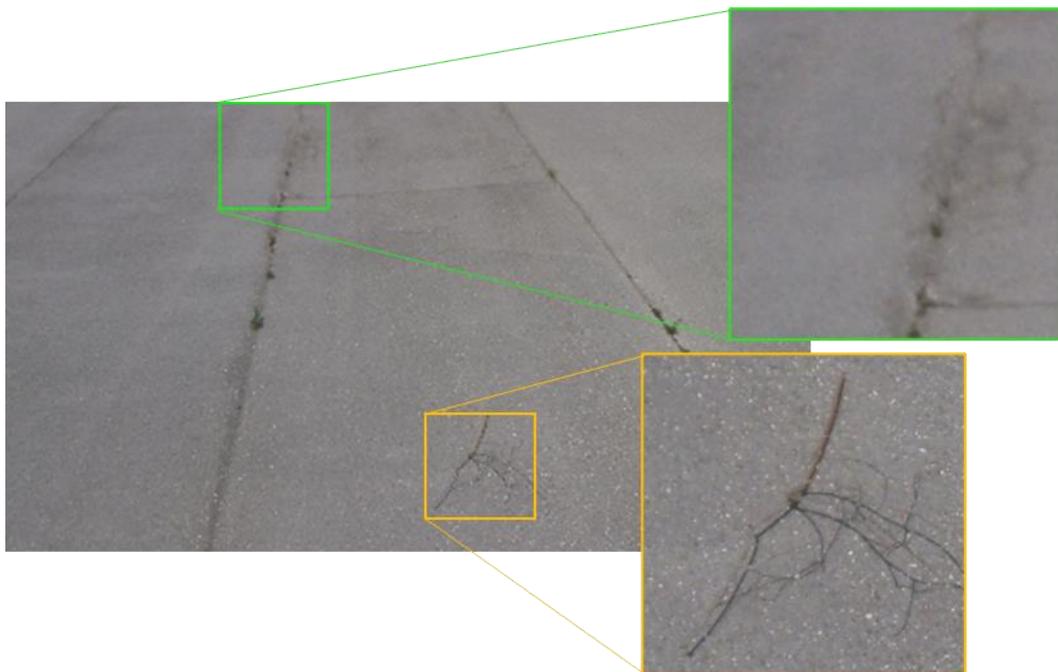


Figure 31 – False positive classification of plants as FOD (in green) and the true positive classification (in orange). Image from sensor 01.

#### 5.4.2. Results on the new Dataset with Object Detection

As in the previous Section, the subset used for object detection is limited by the number of images with instances of FODs. We divided the images into tiles of  $416 \times 416$  px with a horizontal and vertical overlap ratio of 0.5. From here, resulted a subset of 423 images, where 248 contain FODs and 175 do not, and 237 belong to sensor 01 and 186 to sensor 02.

In [Table 25](#), we observe that the median areas of the objects in the original image area are between 0.284% and 0.615%. By cropping the image into tiles of  $416 \times 416$  px, that ratio will

now drop to 0.340% and 0.737%. Concerning the third acquisition, the median relative area of the objects in the tiles from the fourth acquisition is larger.

Table 25 – Comparison of the median relative area of the objects for detection on the third and fourth acquisitions.

Name	Object's area		Original median relative area (%)	New median relative area 4 <sup>th</sup> acquisition (%)	New median relative area 3 <sup>rd</sup> acquisition (%)
	Average	Median			
Sensor01	2,958	589	0.0284	<b>0.340</b>	<b>0.289</b>
Sensor02	7,278	1,276	0.0615	<b>0.737</b>	<b>0.226</b>

Similarly to the behaviour of the classifier, the detector registered a performance drop. However, the performance decreased significantly more than that of the classifier, from 91.08% to 37.49%. In terms of inference time, it remained unchanged at 11.5 fps, the same value as in the third acquisition. Table 26 compares the AP results on the test sets from the third and fourth acquisitions at different IoUs. Transposing these results to a plot (Figure 32) and comparing them with the model's results in the third acquisition, we can see a more significant drop from the IoU=0.3 to IoU=0.5. However, the results obtained in the third acquisition follow a steeper decline from that point on. Additionally, with an IoU equal to 0.3, the model is still able to deliver an AP over 50%. Moreover, in our case, lowering the IoU has small impact on the operation of the system because the individual driving the vehicle only needs to know that he just passed the FOD, meaning that the object will be in a close range to him.

Table 26 – Average precision for each IoU on the images of the third and fourth acquisitions.

IoU	AP (%)	
	3 <sup>rd</sup> acquisition	4 <sup>th</sup> acquisition
0.3	95.88	52.66
0.45	91.17	42.16
<b>0.5</b>	<b>91.08</b>	<b>37.49</b>
0.6	75.12	23.81
0.7	58.55	13.45
0.75	44.77	8.24

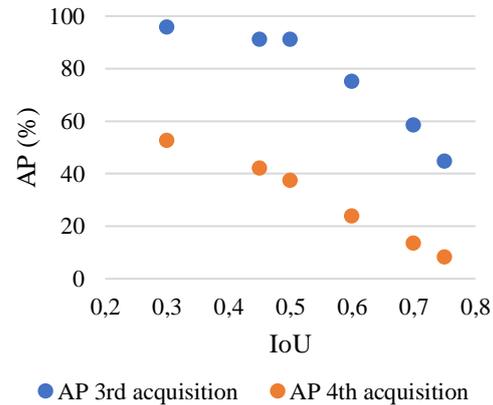


Figure 32 – Plot of the AP versus IoU on the images of the third and fourth acquisitions.

In Figure 33 and Figure 34, we can visually interpret the detector's results by looking at the images provided. It is clear that the model tends to classify novelties as FODs, especially plants. This particular result is not as negative as it would seem at first because plants constitute an organic FOD, especially in larger quantities.

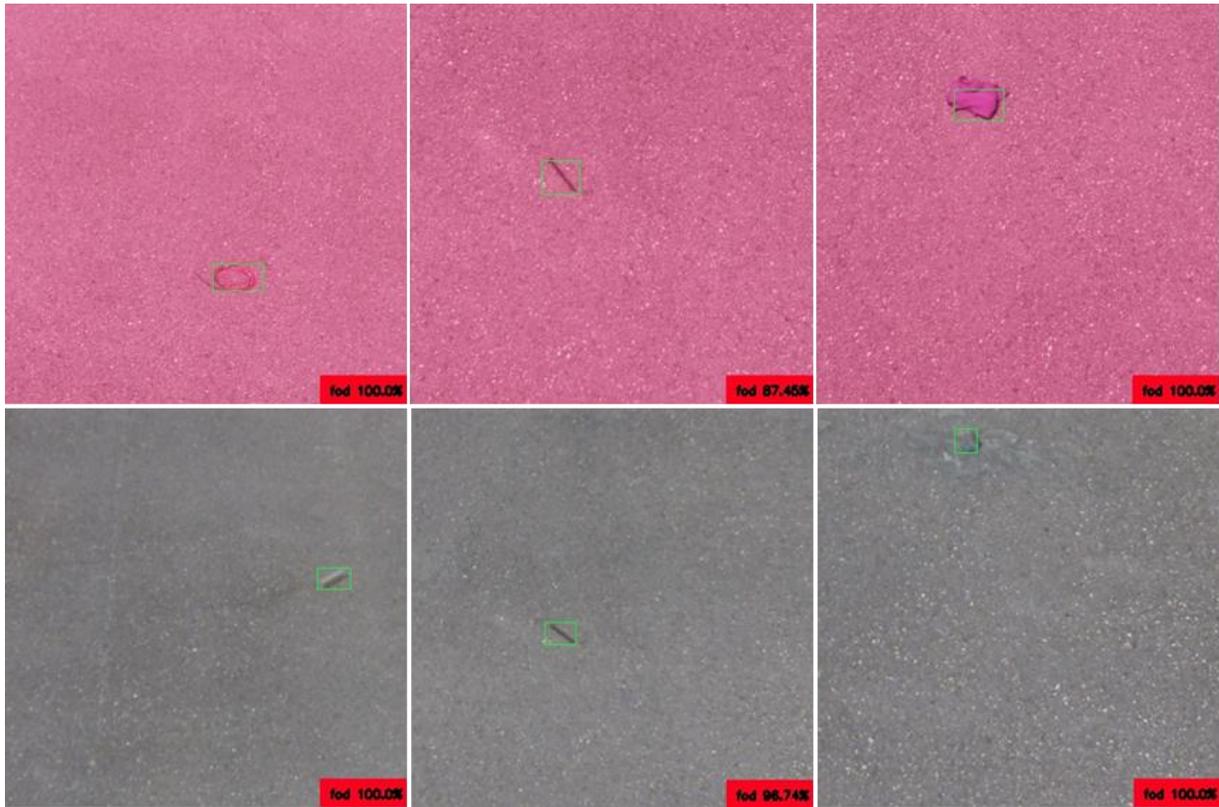


Figure 33 – True positive detections of different objects previously unseen by the model. Top images (sensor 02), bottom images (sensor 01).



Figure 34 – Depiction of incorrect detections. False negative (left) and false positive (right).

# CHAPTER 6

---

In this last chapter, we resume the main results achieved and provide suggestions for future work.

## 6. Conclusions and Future Work

### 6.1. Conclusion

FODs pose a double threat to aviation, where the first and most important has to do with safety and the second with the associated costs. Although airports try to mitigate this problem through several visual inspections every day, it is clear that this traditional method is insufficient to satisfy the problem we face. This is evident considering the occurrences due to FODs and the direct and indirect costs associated. In recent years, radar and optical systems have been employed at several airports for early detection of FODs. These systems are either installed on the side of runways and taxiways – fixed – or coupled to a vehicle that moves around the airfield – mobile. Even though these systems are effective, they are also costly in terms of acquisition and maintenance.

On the other hand, the development of computer vision and machine learning brought several solutions to different fields of research where human inspection is necessary. Among others, these systems have the advantage of requiring low-cost solutions compared to the traditional systems, as is our case. Computer vision application to FOD detection is not a new field of research. Earlier studies investigated the possibility of detecting FODs with traditional computer vision algorithms such as HOGs and SIFTs. The training and testing of these algorithms were performed with a small number of images and faced some difficulties with the varying background. Moreover, the defining characteristic of the conventional methods – handcrafted features – is a major setback due to the number of images required to have good generalisation.

However, the development of machine learning introduced a powerful tool called deep neural networks. These frameworks eliminate the need to extract features by hand, making the training process substantially faster facing conventional methods. Some authors tested solutions with deep neural networks, employing supervised object detection to identify FODs on the aerodrome's surface. Their results were good, but we identified two drawbacks: their dataset was small and/or had a small variety of FODs. Moreover, their datasets were not publicly available.

Our approach to this problem is twofold. Firstly, we build a dataset of images with three different sensors which operate in different wavelengths. This dataset resembles as close as possible the way a fully deployable system would work: with cameras mounted on the top of vehicles which regularly drive around the aerodrome. Secondly, we test classification and

detection based supervised learning techniques to evaluate the pros and cons of each. The main goal of the present work is to build a preliminary system that resembles a full deployment. This system must be low-cost and non-intrusive to the regular operation of an aerodrome.

We developed a system architecture with a mobile platform providing the base for installing electro-optical sensors for the first part. We opted for this system architecture because it does not require acquiring new platforms, is considerably less costly than radar solutions and does not require as many permissions as radar-based technology does. We chose three different sensors which work in three different electromagnetic spectrum ranges: visible, visible plus near-infrared and long-wave infrared. The cameras were mounted on the top of a van at 2.55 m of height at an angle of 38° with the horizontal plane. This angle allows for a good trade-off between the field of view and no image contamination with background other than the target. In total, we performed three image acquisitions, where the first two ended up working as testbeds and being essential to improve the quality of the definite acquisition.

Our dataset resulted from the third acquisition. It contains 9,260 images from the visible sensor, 5,672 images from the visible plus near-infrared sensor and 10,388 images from the long-wave infrared sensor. From the first sensor, 336 images contain FODs, 256 from the second, and 102 from the third. The median average area of the objects is between 0.0189% and 0.180% and, based on other authors and MS COCO dataset, we can consider our targets as being small. The dataset is publicly available on Harvard Dataverse.

Finally, we evaluated the performance of the neural networks on the test subsets of images from the corresponding training subsets. Firstly, we trained the classification network on seven different subsets, where four were balanced and three imbalanced by a factor of 5 images of the class ‘no\_fod’ to 1 of the class ‘fod’. The fourth balanced subset results from half of the subsets’ images from both sensors. We created it to test whether mixing images from different sensors would provide better results, and it did not. Then, we tested the trained models on the respective test balanced subsets. The model with the best performance achieved an accuracy of 98.86% at 90.9 fps, trained on the imbalanced dataset with images from both sensors. However, the training curve of the models, especially the validation loss in conjunction with the training accuracy, show signs of overfitting. Secondly, we trained the detection network on four different subsets, which followed the same logic as the subsets for classification, but, in this case, only with one label – ‘fod’. Again, the subset with every image from both sensors provided the best results with an AP of 91.08% at 11.5 fps. Despite the promising results, we suspected

that the detector was overfitting because of the few images and the excellent results in the testing set. Additionally, the results obtained from subsets 1 and 2 for classification and detection reveal that the sensors deliver similar performances to the models.

This led to a fourth image acquisition with sensors 01 and 02 that intended to test the NNs on entirely new objects. As expected, the accuracy and average precision of the classifier and the detector dropped while retaining the fps. The classifier went from an accuracy of 98.86% to 77.92%, while the detector dropped from an AP of 91.08% to 37.49%.

In conclusion, the preliminary results obtained in the present dissertation constitute a good base to develop the system in future iterations.

## **6.2. Future Work**

As mentioned in the present dissertation, this work constitutes the first iteration of a system that is meant to be deployable and easy to reproduce. Moreover, the results obtained are preliminary and establish a starting point for narrowing the path for future work.

One of the most important steps to follow is to increase the number of images containing FODs. We believe that the small number of this image class was one of the main reasons for the results in classification. Moreover, having more objects of the same and different classes will make the model more robust to adversarial examples. Our image acquisition took place at a taxiway, which is not fully representative of an aerodrome's movement area ground. This is especially important in the case of runways since these have a wider variety of markings.

In this work, we only tested one image classification model and another object detection model because the focus was mainly on building the dataset, which was time-consuming because it took three acquisitions. That being said, we recommend testing newer and faster algorithms for image classification and object detection since their performance in benchmark competitions is better than the ones we used.

Given the definition of FOD – anything alien to the aerodrome surface which might cause harm to someone or an aircraft – we believe that anomaly detection with algorithms such as autoencoders would be an interesting method to try. Another reason why we think it would be a good method to implement is because the number of images without FOD vastly surpasses the number of images with, making it easier to have more examples of the former. Moreover, FODs constitute rare events given the area of an aerodrome's surface, which is why we believe they can be considered an anomaly.

Lastly, we believe that increasing the quality of the cameras would be advantageous because it would allow for the objects to be captured with less blur and at higher speeds. Particularly, the acquisition speed was primarily limited by this factor because the cameras are rolling shutter and have a small dimension of the sensors could not keep up with the vehicle's speed, degrading the quality of the images. Notwithstanding, one must be careful not to overload the embedded system because of its processing memory limitations.

## Bibliography

- [1] Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A Learning Algorithm for Boltzmann Machines. *Cognitive Science*, 9(1), 147–169. [https://doi.org/10.1207/s15516709cog0901\\_7](https://doi.org/10.1207/s15516709cog0901_7)
- [2] AeroMorning. (2015, November 30). Sea-Tac Airport Becomes Second in U.S. to Install Runway Debris Safety Detectors. AeroMorning. <https://aeromorning.com/en/sea-tac-airport-becomes-second-in-u-s-to-install-runway-debris-safety-detectors/>
- [3] Ajit, A., Acharya, K., & Samanta, A. (2020). A Review of Convolutional Neural Networks. 2020 International Conference on Emerging Trends in Information Technology and Engineering (Ic-ETITE), 1–5. <https://doi.org/10.1109/ic-ETITE47903.2020.049>
- [4] Akbar, S., Martel, A., Nofech-Mozes, S., Peikari, M., & Salama, S. (2017). Transitioning Between Convolutional and Fully Connected Layers in Neural Networks. In T. Arbel, V. Belagiannis, A. Bradley, J. S. Cardoso, M. J. Cardoso, G. Carneiro, H. Greenspan, Z. Lu, A. Madabhushi, M. Moradi, J. C. Nascimento, T. Syeda-Mahmood, J. P. Papa, & J. M. R. S. Tavares (Eds.), *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (Vol. 10553, pp. 143–150). Springer International Publishing. [https://doi.org/10.1007/978-3-319-67558-9\\_17](https://doi.org/10.1007/978-3-319-67558-9_17)
- [5] Alharin, A., Doan, T.-N., & Sartipi, M. (2020). Reinforcement Learning Interpretation Methods: A Survey. *IEEE Access*, 8, 171058–171077. <https://doi.org/10.1109/ACCESS.2020.3023394>
- [6] Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., & Aljaaf, A. J. (2020). A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science. In M. W. Berry, A. Mohamed, & B. W. Yap (Eds.), *Supervised and Unsupervised Learning for Data Science* (pp. 3–21). Springer International Publishing. [https://doi.org/10.1007/978-3-030-22475-2\\_1](https://doi.org/10.1007/978-3-030-22475-2_1)
- [7] Aloimonos, J. (1990). Purposive and qualitative active vision. [1990] Proceedings. 10th International Conference on Pattern Recognition, 1, 346–360. <https://doi.org/10.1109/ICPR.1990.118128>
- [8] Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esesn, B. C., Awwal, A. A. S., & Asari, V. K. (2018). The History Began from AlexNet:

- A Comprehensive Survey on Deep Learning Approaches. ArXiv:1803.01164 [Cs]. <http://arxiv.org/abs/1803.01164>
- [9] Aloysius, N., & Geetha, M. (2017). A review on deep convolutional neural networks. 2017 International Conference on Communication and Signal Processing (ICCSP), 0588–0592. <https://doi.org/10.1109/ICCSP.2017.8286426>
- [10] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), 53. <https://doi.org/10.1186/s40537-021-00444-8>
- [11] ATSB. (2010). Ground operations occurrences at Australian airports 1998 to 2008 (Safety Report No. 42; ATSB Transport Safety Report, pp. 1–20). ATSB.
- [12] BEA. (2000). Accident to the Concorde registered F-BTSC operated by Air France occurred on 07/25/00 at Gonesse (f-sc000725a, pp. 169–185) [Final Accident Report]. Ministère de l'Équipement des Transports et du Logement. <https://www.bea.aero/en/investigation-reports/notified-events/detail/accident-to-the-concorde-registered-f-btsc-operated-by-air-france-occured-on-07-25-00-at-gonesse/>
- [13] Benali Amjoud, A., & Amrouch, M. (2020). Convolutional Neural Networks Backbones for Object Detection. In A. El Moataz, D. Mammass, A. Mansouri, & F. Nouboud (Eds.), *Image and Signal Processing* (Vol. 12119, pp. 282–289). Springer International Publishing. [https://doi.org/10.1007/978-3-030-51935-3\\_30](https://doi.org/10.1007/978-3-030-51935-3_30)
- [14] Bengio, Y., & LeCun, Y. (2003). Convolutional Networks for Images, Speech, and Time Series. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (Second Edition, pp. 276–279). MIT Press.
- [15] Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., & Oord, A. van den. (2020). Are we done with ImageNet? ArXiv:2006.07159 [Cs]. <http://arxiv.org/abs/2006.07159>
- [16] Boeing. (1998, January). FOD Prevention Program. *Aero Magazine*, 1(1). [https://www.boeing.com/commercial/aeromagazine/aero\\_01/s/s01/index.html](https://www.boeing.com/commercial/aeromagazine/aero_01/s/s01/index.html)
- [17] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory - COLT '92*, 144–152. <https://doi.org/10.1145/130385.130401>
- [18] CAA. (2020). CAP 562: Civil Aircraft Airworthiness Information and Procedures (CAAIP). In *Civil Aircraft Airworthiness Information and Procedures* (4th ed., Vol. 1, p. 530). CAA. <https://publicapps.caa.co.uk/modalapplication.aspx?appid=11&mode=detail&id=92>

- [19] Cao, X., Gong, G., Liu, M., & Qi, J. (2016). Foreign Object Debris Detection on Airfield Pavement Using Region Based Convolution Neural Network. 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 1–6. <https://doi.org/10.1109/DICTA.2016.7797045>
- [20] Cao, X., Wang, P., Meng, C., Bai, X., Gong, G., Liu, M., & Qi, J. (2018). Region Based CNN for Foreign Object Debris Detection on Airfield Pavement. *Sensors*, 18(3), 1–14. <https://doi.org/10.3390/s18030737>
- [21] Chapelle, O., Schölkopf, B., & Zien, A. (Eds.). (2006). *Semi-supervised learning*. MIT Press.
- [22] Chauhan, T., Goyal, C., Kumari, D., & Thakur, A. K. (2020). A review on foreign object debris/damage (FOD) and its effects on aviation industry. *Materials Today: Proceedings*, 33, 4336–4339. <https://doi.org/10.1016/j.matpr.2020.07.457>
- [23] Chen, C., Liu, M.-Y., Tuzel, O., & Xiao, J. (2017). R-CNN for Small Object Detection. In S.-H. Lai, V. Lepetit, K. Nishino, & Y. Sato (Eds.), *Computer Vision – ACCV 2016* (Vol. 10115, pp. 214–230). Springer International Publishing. [https://doi.org/10.1007/978-3-319-54193-8\\_14](https://doi.org/10.1007/978-3-319-54193-8_14)
- [24] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1800–1807. <https://doi.org/10.1109/CVPR.2017.195>
- [25] Chollet, F. (2021). *Deep learning with Python (Second Edition)*. Manning Publications.
- [26] Cruz, G. (2019). *Autonomous aerial imagery analysis in maritime surveillance scenarios* [PhD]. Instituto Superior Técnico.
- [27] Dai, J., Li, Y., He, K., & Sun, J. (2016). R-FCN: Object Detection via Region-based Fully Convolutional Networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 29, pp. 379–387). Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2016/file/577ef1154f3240ad5b9b413aa7346a1e-Paper.pdf>
- [28] Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 1, 886–893. <https://doi.org/10.1109/CVPR.2005.177>
- [29] Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*, 233–240. <https://doi.org/10.1145/1143844.1143874>

- [30] Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2009). ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, 248–255. <https://doi.org/10.1109/CVPR.2009.5206848>
- [31] Dike, H. U., Zhou, Y., Deveerasetty, K. K., & Wu, Q. (2018). Unsupervised Learning Based On Artificial Neural Network: A Review. 2018 IEEE International Conference on Cyborg and Bionic Systems (CBS), 322–327. <https://doi.org/10.1109/CBS.2018.8612259>
- [32] Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a Deep Convolutional Network for Image Super-Resolution. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014* (pp. 184–199). Springer International Publishing.
- [33] Doshi, R., Hiran, K., Jain, R., & Lakhwani, K. (2021). *Machine Learning: Master supervised and unsupervised learning algorithms with real examples. (First Edition)*. BPB Publications; [www.bponline.com](http://www.bponline.com).
- [34] EMFA. (2017). Relatório Anual de Atividades 2017 (p. 58) [[Annual Report]]. Força Aérea Portuguesa.
- [35] EMFA. (2018). Relatório de Gestão 2018 (p. 61) [[Annual Report]]. Força Aérea Portuguesa.
- [36] EMFA. (2019). Relatório de Gestão 2019 (p. 66) [[Annual Report]]. Força Aérea Portuguesa.
- [37] Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why Does Unsupervised Pre-training Help Deep Learning? *The Journal of Machine Learning Research*, 11, 625–660. <https://doi.org/10.5555/1756006>.
- [38] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2), 303–338. <https://doi.org/10.1007/s11263-009-0275-4>
- [39] Everingham, M., & Winn, J. (2010). *The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Development Kit*.
- [40] F. Y, O., J.E.T, A., O, A., J. O, H., O, O., & J, A. (2017). Supervised Machine Learning Algorithms: Classification and Comparison. *International Journal of Computer Trends and Technology*, 48(3), 128–138. <https://doi.org/10.14445/22312803/IJCTT-V48P126>
- [41] FAA. (2009). AC 150/5220-24—Foreign Object Debris Detection Equipment – Document Information (Advisory Circular No. 150). FAA.

- [https://www.faa.gov/airports/resources/advisory\\_circulars/index.cfm/go/document.current/documentNumber/150\\_5220-24/](https://www.faa.gov/airports/resources/advisory_circulars/index.cfm/go/document.current/documentNumber/150_5220-24/)
- [42] FAA. (2010). AC 150/5210-24—Foreign Object Debris (FOD) Management (Advisory Circular No. 150; p. 44). FAA. [https://www.faa.gov/airports/resources/advisory\\_circulars/index.cfm/go/document.information/documentID/391902](https://www.faa.gov/airports/resources/advisory_circulars/index.cfm/go/document.information/documentID/391902)
- [43] FAA. (2019). FAA Reauthorization Bill 2018 Foreign Object Debris (FOD) Detection Technology [[PowerPoint]]. [https://www.faa.gov/sites/faa.gov/files/2021-11/Technology\\_Use\\_Airports\\_PL115-254\\_Section\\_142b.pdf](https://www.faa.gov/sites/faa.gov/files/2021-11/Technology_Use_Airports_PL115-254_Section_142b.pdf)
- [44] Fang, Y., Han, Z., Xu, H., & Zheng, Y. (2015). A Novel FOD Classification System Based on Visual Features. In Y.-J. Zhang (Ed.), *Image and Graphics* (Vol. 9217, pp. 288–296). Springer International Publishing. [https://doi.org/10.1007/978-3-319-21978-3\\_26](https://doi.org/10.1007/978-3-319-21978-3_26)
- [45] Fischler, M. A., & Elschlager, R. A. (1973). The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, C-22(1), 67–92. <https://doi.org/10.1109/T-C.1973.223602>
- [46] Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4), 193–202. <https://doi.org/10.1007/BF00344251>
- [47] Fukushima, K. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2), 119–130. [https://doi.org/10.1016/0893-6080\(88\)90014-7](https://doi.org/10.1016/0893-6080(88)90014-7)
- [48] García Aranda, J. J., Alarcón Granero, M., Juan Quintanilla, F. J., Caffarena, G., & García-Carmona, R. (2021). Elastic Downsampling: An Adaptive Downsampling Technique to Preserve Image Quality. *Electronics*, 10(4). <https://doi.org/10.3390/electronics10040400>
- [49] Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems* (Second edition). O'Reilly Media, Inc.
- [50] Girshick, R. (2015). Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV), 1440–1448. <https://doi.org/10.1109/ICCV.2015.169>
- [51] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on

- Computer Vision and Pattern Recognition, 580–587.  
<https://doi.org/10.1109/CVPR.2014.81>
- [52] Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterton (Eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Vol. 9, pp. 249–256). PMLR. <https://proceedings.mlr.press/v9/glorot10a.html>
- [53] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning* (First Edition). MIT Press.  
<http://public.ebookcentral.proquest.com/choice/publicfullrecord.aspx?p=6287197>
- [54] Graves, S., & Buttlar, W. (2013). Electro-Optical Sensor Evaluation of Airfield Pavement. *Sustainable and Efficient Pavement*, 585–596.  
<https://doi.org/10.1061/9780784413005.047>
- [55] Han, Z., Fang, Y., & Xu, H. (2015). Fusion of low-level features for FOD classification. *2015 10th International Conference on Communications and Networking in China (ChinaCom)*, 465–469. <https://doi.org/10.1109/CHINACOM.2015.7497985>
- [56] Hashemi, M. (2019). Enlarging smaller images before inputting into convolutional neural network: Zero-padding vs. interpolation. *Journal of Big Data*, 6(1), 98.  
<https://doi.org/10.1186/s40537-019-0263-7>
- [57] Hassaballah, M., & Awad, A. I. (Eds.). (2020). *Deep learning in computer vision: Principles and applications* (First edition). CRC Press/Taylor and Francis.
- [58] He, K., Zhang, X., Ren, S., & Sun, J. (2014). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014* (pp. 346–361). Springer International Publishing.
- [59] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- [60] Herricks, E., Lazar, P., & Woodworth, E. (2011). Performance Assessment of a Mobile, Radar-Based Foreign Object Debris Detection System (Final Report DOT/FAA/AR-11/12). FAA. <https://www.airporttech.tc.faa.gov/Products/Airport-Safety-Papers-Publications/Airport-Safety-Detail/ArtMID/3682/ArticleID/67/Performance-Assessment-of-a-Mobile-Radar-Based-Foreign-Object-Debris-Detection-System>
- [61] Herricks, E., Lazar, P., Woodworth, E., & Patterson, J. (2012). Performance Assessment of an Electro-Optical-Based Foreign Object Debris Detection System (Final Report

- DOT/FAA/AR-11/13). FAA. <https://www.airporttech.tc.faa.gov/Products/Airport-Safety-Papers-Publications/Airport-Safety-Detail/ArtMID/3682/ArticleID/62/Performance-Assessment-of-an-Electro-Optical-Based-Foreign-Object-Debris-Detection-System>
- [62] Herricks, E., Mayer, D., & Majumdar, S. (2015). Foreign Object Debris Characterization at a Large International Airport (Technical Note DOT/FAA/TC-TN14/48; p. 89). FAA; <https://www.airporttech.tc.faa.gov/Products/Airport-Safety-Papers-Publications/Airport-Safety-Detail/ArtMID/3682/ArticleID/36/Foreign-Object-Debris-Characterization-at-a-Large-International-Airport>.  
<https://www.airporttech.tc.faa.gov/Products/Airport-Safety-Papers-Publications/Airport-Safety-Detail/ArtMID/3682/ArticleID/36/Foreign-Object-Debris-Characterization-at-a-Large-International-Airport>
- [63] Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7), 1527–1554.
- [64] Hsieh Hou, & Andrews, H. (1978). Cubic splines for image interpolation and digital filtering. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(6), 508–517. <https://doi.org/10.1109/TASSP.1978.1163154>
- [65] Hu, P., & Ramanan, D. (2017). Finding Tiny Faces. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1522–1530. <https://doi.org/10.1109/CVPR.2017.166>
- [66] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2261–2269. <https://doi.org/10.1109/CVPR.2017.243>
- [67] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., & Murphy, K. (2017). Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3296–3297. <https://doi.org/10.1109/CVPR.2017.351>
- [68] Huang, T. S. (1996, 21/09). Computer Vision: Evolution And Promise. CERN Yellow Reports: School Proceedings. <https://doi.org/10.5170/CERN-1996-008.21>
- [69] Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3), 574–591. <https://doi.org/10.1113/jphysiol.1959.sp006308>

- [70] Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1), 106–154. <https://doi.org/10.1113/jphysiol.1962.sp006837>
- [71] Hussin, R., Ismail, N., & Mustapa, S. (2016). A study of foreign object damage (FOD) and prevention method at the airport and aircraft maintenance area. *IOP Conference Series: Materials Science and Engineering*, 152, 012038. <https://doi.org/10.1088/1757-899X/152/1/012038>
- [72] IBM. (2021, February 19). What is Computer Vision? IBM. <https://www.ibm.com/topics/computer-vision>
- [73] ICAO. (2020). ICAO Safety Report 2020 (Safety Report No. 1; Safety Report, pp. 1–64). ICAO. <https://www.icao.int/safety/Pages/Safety-Report.aspx>
- [74] Ioffe, S., Shlens, J., Szegedy, C., Vanhoucke, V., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2818–2826. <https://doi.org/10.1109/CVPR.2016.308>
- [75] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, 37, 448–456. <http://proceedings.mlr.press/v37/ioffe15.pdf>
- [76] Janke, J., & Castelli, M. (2018). Analysis of the Proficiency of Fully-Connected Neural Networks in the Process of Classifying Digital Images: Benchmark of Different Classification Algorithms on High-Level Image Features from Convolutional Layers [MsC, NOVA IMS]. <http://www.di.fct.unl.pt>
- [77] Jmour, N., Zayen, S., & Abdelkrim, A. (2018). Convolutional neural networks for image classification. 2018 International Conference on Advanced Systems and Electric Technologies (IC\_ASET), 397–402. <https://doi.org/10.1109/ASET.2018.8379889>
- [78] Jo, T. (2021). *Machine Learning Foundations: Supervised, Unsupervised, and Advanced Learning*. Springer International Publishing. <https://doi.org/10.1007/978-3-030-65900-4>
- [79] Juneja, M., Vedaldi, A., Jawahar, C. V., & Zisserman, A. (2013). Blocks That Shout: Distinctive Parts for Scene Classification. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 923–930. <https://doi.org/10.1109/CVPR.2013.124>
- [80] Karazi, S. M., Moradi, M., & Benyounis, K. Y. (2019). Statistical and Numerical Approaches for Modeling and Optimizing Laser Micromachining Process-Review. In

- Reference Module in Materials Science and Materials Engineering (p. B9780128035818117000). Elsevier. <https://doi.org/10.1016/B978-0-12-803581-8.11650-9>
- [81] Kraus, D. C., & Watson, J. (2001). Guidelines for the Prevention and Elimination of Foreign Object Damage/Debris (FOD) in the Aviation Maintenance Environment through Improved Human Performance. FAA.
- [82] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 1097–1105.
- [83] Lakshmanan, V., Görner, M., & Gillard, R. (2021). *Practical machine learning for computer vision: End-to-end machine learning for images (First Edition)*. O'Reilly.
- [84] LeCun, Y. (2016, December 5). Predictive Learning [Keynote]. 2016 Conference on Neural Information Processing Systems, Barcelona, Spain. <https://www.youtube.com/watch?v=Ount2Y4qxQo&t=1072s>
- [85] Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324. <https://doi.org/10.1109/5.726791>
- [86] Li, P., & Li, H. (2020). Research on FOD Detection for Airport Runway based on YOLOv3. 2020 39th Chinese Control Conference (CCC), 7096–7099. <https://doi.org/10.23919/CCC50068.2020.9188724>
- [87] Li, X., Zhang, G., Li, K., & Zheng, W. (2016). Deep Learning and Its Parallelization. In R. Buyya, R. Calheiros, & A. Dastjerdi (Eds.), *Big Data – Principles and Paradigms* (1st ed., pp. 95–118). Elsevier. <https://doi.org/10.1016/B978-0-12-805394-2.00004-0>
- [88] Li, Y.-F., & Zhou, Z.-H. (2015). Towards Making Unlabeled Data Never Hurt. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1), 175–188. <https://doi.org/10.1109/TPAMI.2014.2299812>
- [89] Li, Z., Yang, J., Liu, Z., Yang, X., Jeon, G., & Wu, W. (2019). Feedback Network for Image Super-Resolution. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 3862–3871. <https://doi.org/10.1109/CVPR.2019.00399>
- [90] Lin, D., Lu, C., Liao, R., & Jia, J. (2014). Learning Important Spatial Pooling Regions for Scene Classification. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 3726–3733. <https://doi.org/10.1109/CVPR.2014.476>

- [91] Lin, T., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2015). Microsoft COCO: Common Objects in Context. ArXiv:1405.0312 [Cs]. <http://arxiv.org/abs/1405.0312>
- [92] Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. 2017 IEEE International Conference on Computer Vision (ICCV), 2999–3007. <https://doi.org/10.1109/ICCV.2017.324>
- [93] Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2018). Focal Loss for Dense Object Detection. ArXiv:1708.02002 [Cs]. <http://arxiv.org/abs/1708.02002>
- [94] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, 128(2), 261–318. <https://doi.org/10.1007/s11263-019-01247-4>
- [95] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Single Shot MultiBox Detector. *Computer Vision – ECCV 2016*, 9905, 21–37. [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
- [96] Liu, Y., Li, Y., Liu, J., Peng, X., Zhou, Y., & Murphey, Y. L. (2018). FOD Detection using DenseNet with Focal Loss of Object Samples for Airport Runway. *IEEE Symposium Series on Computational Intelligence (SSCI)*, 8. <https://doi.org/10.1109/SSCI.2018.8628648>
- [97] Liu, Y., Sun, P., Wergeles, N., & Shang, Y. (2021). A survey and performance evaluation of deep learning methods for small object detection. *Expert Systems with Applications*, 172, 114602. <https://doi.org/10.1016/j.eswa.2021.114602>
- [98] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3431–3440. <https://doi.org/10.1109/CVPR.2015.7298965>
- [99] Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 2, 1150–1157. <https://doi.org/10.1109/ICCV.1999.790410>
- [100] Lundervold, A. S., & Lundervold, A. (2019). An overview of deep learning in medical imaging focusing on MRI. *Zeitschrift Für Medizinische Physik*, 29(2), 102–127. <https://doi.org/10.1016/j.zemedi.2018.11.002>
- [101] Mayoraz, E., & Alpaydin, E. (1999). Support vector machines for multi-class classification. In J. Mira & J. V. Sánchez-Andrés (Eds.), *Engineering Applications of Bio-Inspired Artificial Neural Networks* (Vol. 1607, pp. 833–842). Springer Berlin Heidelberg. <https://doi.org/10.1007/BFb0100551>

- [102] McCreary, I. (2008). The economic cost of FOD to airlines. Insight SRI Ltd.
- [103] McCreary, I. (2010). FOD, Birds, and the Case for Automated Scanning (1st ed.). Insight SRI Ltd.
- [104] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4), 115–133. <https://doi.org/10.1007/BF02478259>
- [105] Misra, I., Zitnick, C. L., & Hebert, M. (2016). Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Computer Vision – ECCV 2016* (Vol. 9905, pp. 527–544). Springer International Publishing. [https://doi.org/10.1007/978-3-319-46448-0\\_32](https://doi.org/10.1007/978-3-319-46448-0_32)
- [106] Mohamed, A. E. (2017). Comparative Study of Four Supervised Machine Learning Techniques for Classification. *International Journal of Applied Science and Technology*, 7(2), 5–18.
- [107] Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., & Yuille, A. (2014). The Role of Context for Object Detection and Semantic Segmentation in the Wild. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 891–898. <https://doi.org/10.1109/CVPR.2014.119>
- [108] Munyer, T., Huang, P.-C., Huang, C., & Zhong, X. (2022). FOD-A: A Dataset for Foreign Object Debris in Airports. *ArXiv:2110.03072* [Cs]. <http://arxiv.org/abs/2110.03072>
- [109] NASEM. (2011). Current Airport Inspection Practices Regarding FOD (Foreign Object Debris/Damage). Transportation Research Board. <https://doi.org/10.17226/14572>
- [110] Nasteski, V. (2017). An overview of the supervised machine learning methods. *HORIZONS.B*, 4, 51–62. <https://doi.org/10.20544/HORIZONS.B.04.1.17.P05>
- [111] NATO. (2004). Best Practices for the Mitigation and Control of Foreign Object Damage-Induced High Cycle Fatigue in Gas Turbine Engine Compression System Airfoils. NATO.
- [112] NDCEE. (2011). 2011 Annual Report (National Defense Center for Energy and Environment) (p. 13) [Annual Report]. National Defense Center for Energy and Environment. <https://apps.dtic.mil/sti/pdfs/ADA574500.pdf>
- [113] Nguyen, N.-D., Do, T., Ngo, T. D., & Le, D.-D. (2020). An Evaluation of Deep Learning Methods for Small Object Detection. *Journal of Electrical and Computer Engineering*, 2020, 1–18. <https://doi.org/10.1155/2020/3189691>

- [114] Nvidia. (2022). Jetson TX2 Module. Nvidia Developer. <https://developer.nvidia.com/embedded/jetson-tx2>
- [115] Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 971–987. <https://doi.org/10.1109/TPAMI.2002.1017623>
- [116] Oliver, A., Odena, A., Raffel, C., Cubuk, E. D., & Goodfellow, I. J. (2019). Realistic Evaluation of Deep Semi-Supervised Learning Algorithms. ArXiv:1804.09170 [Cs, Stat]. <http://arxiv.org/abs/1804.09170>
- [117] Papers with Code. (2022). Machine Learning Datasets. Papers with Code. <https://paperswithcode.com/datasets?q=&v=lst&o=cited&mod=images&task=image-classification>
- [118] Pham, P., Nguyen, D., Do, T., Ngo, T. D., & Le, D.-D. (2017). Evaluation of Deep Models for Real-Time Small Object Detection. In D. Liu, S. Xie, Y. Li, D. Zhao, & E.-S. M. El-Alfy (Eds.), *Neural Information Processing* (Vol. 10636, pp. 516–526). Springer International Publishing. [https://doi.org/10.1007/978-3-319-70090-8\\_53](https://doi.org/10.1007/978-3-319-70090-8_53)
- [119] PoAF. (2018). MBA5 330-3 (A)—Programa de Prevenção de Danos por Objetos Estranhos.
- [120] Poola, I. (2017). How Artificial Intelligence is Impacting Real Life Every Day. *International Journal of Advanced Research and Development*, 2(10), 96–100.
- [121] Procaccio, F. (2008). Effectiveness of FOD Control Measures [Masters]. Embry-Riddle Aeronautical University.
- [122] Prusa, J., Khoshgoftaar, T. M., & Seliya, N. (2015). The Effect of Dataset Size on Training Tweet Sentiment Classifiers. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), 96–102. <https://doi.org/10.1109/ICMLA.2015.22>
- [123] Rasmus, A., Berglund, M., Honkala, M., Valpola, H., & Raiko, T. (2015). Semi-supervised Learning with Ladder Networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 28, pp. 3546–3554). Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2015/file/378a063b8fdb1db941e34f4bde584c7d-Paper.pdf>
- [124] Raspberry Pi. (2016). Raspberry Pi Camera Module 2 NoIR. Raspberry Pi. <https://www.raspberrypi.com/products/pi-noir-camera-v2/>

- [125] Raspberry Pi. (2017). Raspberry Pi 3 Model. Raspberry Pi. <https://www.raspberrypi.com/products/raspberry-pi-3-model-b/>
- [126] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. ArXiv:1506.02640v4 [Cs]. <http://arxiv.org/abs/1506.02640>
- [127] Redmon, J., & Farhadi, A. (2016). YOLO9000: Better, Faster, Stronger. ArXiv:1612.08242 [Cs]. <http://arxiv.org/abs/1612.08242>
- [128] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. ArXiv:1804.02767 [Cs]. <http://arxiv.org/abs/1804.02767>
- [129] Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. Lawrence, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* 28. <http://arxiv.org/abs/1506.01497>
- [130] Richter, M. L., Byttner, W., Krumnack, U., Schallner, L., & Shenk, J. (2021a). Size Matters. ArXiv:2102.01582 [Cs], 12892, 133–144. [https://doi.org/10.1007/978-3-030-86340-1\\_11](https://doi.org/10.1007/978-3-030-86340-1_11)
- [131] Richter, M. L., Byttner, W., Krumnack, U., Schallner, L., & Shenk, J. (2021b). Size Matters. ArXiv:2102.01582 [Cs], 12892, 133–144. [https://doi.org/10.1007/978-3-030-86340-1\\_11](https://doi.org/10.1007/978-3-030-86340-1_11)
- [132] Rifman, S. (1973). Digital rectification of ERTS multispectral imagery. Significant Results Obtained from Earth Resources Technology Satellite-1, 1, 1131–1142. <https://ntrs.nasa.gov/citations/19730019595>
- [133] Roberts, L. (1963). *Machine Perception of Three-Dimensional Solids* [PhD, Massachusetts Institute of Technology]. <https://dspace.mit.edu/handle/1721.1/11589>
- [134] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386–408. <https://doi.org/10.1037/h0042519>
- [135] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211–252. <https://doi.org/10.1007/s11263-015-0816-y>
- [136] Sambolek, S., & Ivasic-Kos, M. (2021). Automatic Person Detection in Search and Rescue Operations Using Deep CNN Detectors. *IEEE Access*, 9, 37905–37922. <https://doi.org/10.1109/ACCESS.2021.3063681>

- [137] Sandhu, T. H. (2018). Machine Learning and Natural Language Processing – A Review. *International Journal of Advanced Research in Computer Science*, 9(2), 582–584. <https://doi.org/10.26483/ijarcs.v9i2.5799>
- [138] Schmarje, L., Santarossa, M., Schroder, S.-M., & Koch, R. (2021). A Survey on Semi-, Self- and Unsupervised Learning for Image Classification. *IEEE Access*, 9, 82146–82168. <https://doi.org/10.1109/ACCESS.2021.3084358>
- [139] Sen, P. C., Hajra, M., & Ghosh, M. (2020). Supervised Classification Algorithms in Machine Learning: A Survey and Review. In J. K. Mandal & D. Bhattacharya (Eds.), *Emerging Technology in Modelling and Graphics* (Vol. 937, pp. 99–111). Springer Singapore. [https://doi.org/10.1007/978-981-13-7403-6\\_11](https://doi.org/10.1007/978-981-13-7403-6_11)
- [140] Sethi, A., & Huang, T. (2006). Interaction between modules in learning systems for vision applications.
- [141] Shapiro, L. G. (2020). Computer vision: The last 50 years. *International Journal of Parallel, Emergent and Distributed Systems*, 35(2), 112–117. <https://doi.org/10.1080/17445760.2018.1469018>
- [142] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6(1), 60. <https://doi.org/10.1186/s40537-019-0197-0>
- [143] Sifre, L. (2014). Rigid-Motion Scattering For Image Classification—Sifre L 2014.pdf (Publications, 2014) [PhD, Ecole Polytechnique, CMAP]. PRAIRE. [https://www.di.ens.fr/data/publications/papers/phd\\_sifre.pdf](https://www.di.ens.fr/data/publications/papers/phd_sifre.pdf)
- [144] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations. ICLR*. <http://arxiv.org/abs/1409.1556>
- [145] Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations*, 14.
- [146] Sitaula, C., Xiang, Y., Zhang, Y., Lu, X., & Aryal, S. (2019). Indoor Image Representation by High-Level Semantic Features. *IEEE Access*, 7, 84967–84979. <https://doi.org/10.1109/ACCESS.2019.2925002>
- [147] Sordo, M., & Zeng, Q. (2005). On Sample Size and Classification Accuracy: A Performance Comparison. In J. L. Oliveira, V. Maojo, F. Martín-Sánchez, & A. S. Pereira (Eds.), *Biological and Medical Data Analysis* (Vol. 3745, pp. 193–201). Springer Berlin Heidelberg. [https://doi.org/10.1007/11573067\\_20](https://doi.org/10.1007/11573067_20)

- [148] Spanrad, S., & Tong, J. (2010). Characterization of foreign object damage (FOD) and early fatigue crack growth in laser shock peened Ti-6AL-4V aerofoil specimens. *Procedia Engineering*, 2(1), 1751–1759. <https://doi.org/10.1016/j.proeng.2010.03.188>
- [149] Stock, P., & Cisse, M. (2018). ConvNets and ImageNet Beyond Accuracy: Understanding Mistakes and Uncovering Biases. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (Vol. 11210, pp. 504–519). Springer International Publishing. [https://doi.org/10.1007/978-3-030-01231-1\\_31](https://doi.org/10.1007/978-3-030-01231-1_31)
- [150] Stratech. (2016). iFerretTM. <https://www.westernadvance.com/MediaLibrary/Documents/iFerret%E2%84%A2-Product-Brochure.pdf>
- [151] Sunny. (2012). OmniVision OV 569 [Datasheet]. <https://www.ovt.com/>
- [152] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.
- [153] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1–9. <https://doi.org/10.1109/CVPR.2015.7298594>
- [154] Tang, P., Wang, H., & Kwong, S. (2017). G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition. *Neurocomputing*, 225, 188–197. <https://doi.org/10.1016/j.neucom.2016.11.023>
- [155] Torralba, A., Fergus, R., & Freeman, W. T. (2008). 80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11), 1958–1970. <https://doi.org/10.1109/TPAMI.2008.128>
- [156] Torralba, Murphy, Freeman, & Rubin. (2003). Context-based vision system for place and object recognition. *Proceedings Ninth IEEE International Conference on Computer Vision*, 273–280 vol.1. <https://doi.org/10.1109/ICCV.2003.1238354>
- [157] Trex Aviation Systems. (2016). FOD Finder XM. <http://www.fodfinder.com/pages/Linked%20items/FOD%20Finder%20XM.pdf>
- [158] Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective Search for Object Recognition. *International Journal of Computer Vision*, 104(2), 154–171. <https://doi.org/10.1007/s11263-013-0620-5>

- [159] Usama, M., Qadir, J., Raza, A., Arif, H., Yau, K. A., Elkhatib, Y., Hussain, A., & Al-Fuqaha, A. (2019). Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges. *IEEE Access*, 7, 65579–65615. <https://doi.org/10.1109/ACCESS.2019.2916648>
- [160] van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, 109(2), 373–440. <https://doi.org/10.1007/s10994-019-05855-6>
- [161] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1, I-511–I-518. <https://doi.org/10.1109/CVPR.2001.990517>
- [162] Wang, P., Fan, E., & Wang, P. (2021). Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recognition Letters*, 141, 61–67. <https://doi.org/10.1016/j.patrec.2020.07.042>
- [163] Wang, Q., Ma, Y., Zhao, K., & Tian, Y. (2022). A Comprehensive Survey of Loss Functions in Machine Learning. *Annals of Data Science*, 9(2), 187–212. <https://doi.org/10.1007/s40745-020-00253-5>
- [164] Wang, Z., Chen, J., & Hoi, S. C. H. (2021). Deep Learning for Image Super-Resolution: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10), 3365–3387. <https://doi.org/10.1109/TPAMI.2020.2982166>
- [165] Wu, R., Yan, S., Shan, Y., Dang, Q., & Sun, G. (2015). Deep Image: Scaling up Image Recognition. *ArXiv:1501.02876 [Cs]*. <http://arxiv.org/abs/1501.02876>
- [166] Xenics. (2008). Xenics Gobi-384 [Data sheet]. [https://pyramidimaging.com/specs/Xenics/XB-020\\_04\\_Gobi-384\\_industrial\\_LowRes.pdf](https://pyramidimaging.com/specs/Xenics/XB-020_04_Gobi-384_industrial_LowRes.pdf)
- [167] Xia, G.-S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., & Zhang, L. (2018). DOTA: A Large-Scale Dataset for Object Detection in Aerial Images. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3974–3983. <https://doi.org/10.1109/CVPR.2018.00418>
- [168] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., & Torralba, A. (2010). SUN database: Large-scale scene recognition from abbey to zoo. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3485–3492. <https://doi.org/10.1109/CVPR.2010.5539970>
- [169] Xie, S., Girshick, R., Dollar, P., Tu, Z., & He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. *2017 IEEE Conference on Computer*

- Vision and Pattern Recognition (CVPR), 5987–5995.  
<https://doi.org/10.1109/CVPR.2017.634>
- [170] Xu, H., Han, Z., Feng, S., Zhou, H., & Fang, Y. (2018). Foreign object debris material recognition based on convolutional neural networks. *EURASIP Journal on Image and Video Processing*, 2018(1), 21. <https://doi.org/10.1186/s13640-018-0261-2>
- [171] Xu, J., Xiang, L., Liu, Q., Gilmore, H., Wu, J., Tang, J., & Madabhushi, A. (2016). Stacked Sparse Autoencoder (SSAE) for Nuclei Detection on Breast Cancer Histopathology Images. *IEEE Transactions on Medical Imaging*, 35(1), 119–130. <https://doi.org/10.1109/TMI.2015.2458702>
- [172] Yan, Z., Han, X., Wang, C., Qiu, Y., Xiong, Z., & Cui, S. (2019). Learning Mutually Local-Global U-Nets For High-Resolution Retinal Lesion Segmentation In Fundus Images. 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), 597–600. <https://doi.org/10.1109/ISBI.2019.8759579>
- [173] Yang, S., Luo, P., Loy, C. C., & Tang, X. (2016). WIDER FACE: A Face Detection Benchmark. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5525–5533. <https://doi.org/10.1109/CVPR.2016.596>
- [174] Yongbing Zhang, Debin Zhao, Jian Zhang, Ruiqin Xiong, & Wen Gao. (2011). Interpolation-Dependent Image Downsampling. *IEEE Transactions on Image Processing*, 20(11), 3291–3296. <https://doi.org/10.1109/TIP.2011.2158226>
- [175] Yun, S., Oh, S. J., Heo, B., Han, D., Choe, J., & Chun, S. (2021). Re-labelling ImageNet: From Single to Multi-Labels, from Global to Localized Labels. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2340–2350. <https://doi.org/10.1109/CVPR46437.2021.00237>
- [176] Zheng, L., Zhao, Y., Wang, S., Wang, J., & Tian, Q. (2016). Good Practice in CNN Feature Transfer. <https://doi.org/10.48550/ARXIV.1604.00133>

## **ANNEXES**

*intentionally left blank*

## Annex A –Types of FOD encountered by zone

Table A - 1 – Most common types of FOD removed by area. Data collected from 50 airports in North America, Europe and Australia (adapted from NASEM, 2011, p. 29).

Area	1 <sup>st</sup> most common	2 <sup>nd</sup> most common	3 <sup>rd</sup> most common	4 <sup>th</sup> most common
<b>Runways</b>	Runway and taxiway materials	Natural materials	Aircraft parts	Winter ops
<b>Taxiways</b>	Runway and taxiway materials	Natural materials	Winter ops	Aircraft parts
<b>Hangar areas</b>	Aircraft parts	Winter ops	Mechanic’s tools	Apron items, construction debris and plastics
<b>Outside defined construction areas</b>	Construction debris	Plastics	Apron items	Natural materials
<b>Non-pavement areas</b>	Plastics	Apron items	Natural materials	Construction debris

*intentionally left blank*

**Annex B – Answers from the Officers to the proposed questions about FODs in military aviation**

Subject	Position
MAJ Tomás Virgílio	Chief of the Accident Prevention Group of Airbase no. 5
MAJ Pedro Andrade	Chief of the Accident Prevention Group of Airbase no. 6
CAP Nuno Rodrigues	Chief of the Accident Prevention Group of Airbase no. 1
CAP Diogo Almeida	Flight Safety Officer of Airbase no. 11

Question	Answer	Subject
Which are the most common type of FODs in terms of material (e.g., rubber, metal, plastic, etc.) and their origin (e.g., tools, parts of aircraft, etc.)?	“Besides organic debris, given the proximity of the pine forest, the 'metal screw' is the most common object. Either by loosening, vibration, or fatigue of one of the parts, these come loose from vehicles and support systems (GSE, for example). Another possible FOD, I would say, also found with some regularity, are the safety wires. Mechanics are aware and have adopted procedures, but they do not always collect all safety wires, namely when there is the need to change the wheels of the main train (a task performed in the front line/parking spots). Apart from these FOD from external sources, a major concern focuses on the pavement. Given the degradation of some parking places and paths, many pavement fragments (from a few millimetres to 4-5 cm) may cause damage. Although in Monte Real the situation is under control [...], the same does not happen in other airbases, where at this moment the operation of the F-16 is virtually impossible (Montijo).”	Major Virgílio
	“The union of the plates is always a sensitive area, and with the infiltration of water, vegetation growth and passage of vehicles, these produce loose stones due to the generated cracks. We apply black rubber sealant to cover these cracks on the pavement to mitigate the deterioration of the areas, but with time, they also become parched, break, and start to be a potential FOD. Occasionally there are areas of the airfield that present some FODs other than these, such as vegetation torn off by the downwash of helicopters and other aircraft and some pieces of equipment from the movement area of the aerodrome.”	Major Andrade
	“[...] we do not do a statistical analysis of what we catch. What is reported to me, are metal FOD, more specifically remains of aircraft and vehicles, screws or pieces from mobile equipment that has broken like landing gear springs or vehicles [...]”	Captain Rodrigues
	“Besides the common FODs such as bolts, nuts, safety wire and organic materials, we also get some animal carcasses because of the nearing farms and birds. Plus, we also found some construction debris several times while some construction was going on at the airfield. Additionally, we share some infrastructures such as the runway with a civilian airport, and it is not rare to collect one or another object from their operation.”	Captain Almeida

Question	Answer	Subject
Which are the most common type of FODs in terms of material (e.g., rubber, metal, plastic, etc.) and their origin (e.g., tools, parts of aircraft, etc.)?	<p>“Besides organic debris, given the proximity of the pine forest, the 'metal screw' is the most common object. Either by loosening, vibration, or fatigue of one of the parts, these come loose from vehicles and support systems (GSE, for example). Another possible FOD, I would say, also found with some regularity, are the safety wires. Mechanics are aware and have adopted procedures, but they do not always collect all safety wires, namely when there is the need to change the wheels of the main train (a task performed in the front line/parking spots). Apart from these FOD from external sources, a major concern focuses on the pavement. Given the degradation of some parking places and paths, many pavement fragments (from a few millimetres to 4-5 cm) may cause damage. Although in Monte Real the situation is under control [...], the same does not happen in other airbases, where at this moment the operation of the F-16 is virtually impossible (Montijo).”</p>	Major Virgílio
	<p>“The union of the plates is always a sensitive area, and with the infiltration of water, vegetation growth and passage of vehicles, these produce loose stones due to the generated cracks. We apply black rubber sealant to cover these cracks on the pavement to mitigate the deterioration of the areas, but with time, they also become parched, break, and start to be a potential FOD. Occasionally there are areas of the airfield that present some FODs other than these, such as vegetation torn off by the downwash of helicopters and other aircraft and some pieces of equipment from the movement area of the aerodrome.”</p>	Major Andrade
	<p>“[...] we do not do a statistical analysis of what we catch. What is reported to me, are metal FOD, more specifically remains of aircraft and vehicles, screws or pieces from mobile equipment that has broken like landing gear springs or vehicles [...]”</p>	Captain Rodrigues
	<p>“Besides the common FODs such as bolts, nuts, safety wire and organic materials, we also get some animal carcasses because of the nearing farms and birds. Plus, we also found some construction debris several times while some construction was going on at the airfield. Additionally, we share some infrastructures such as the runway with a civilian airport, and it is not rare to collect one or another object from their operation.”</p>	Captain Almeida

Question	Answer	Subject
In which places is it most common to find FODs?	<p>“[FODs] are usually found near parking places and taxiways (there is high traffic in taxiways given some dispersion of aircraft parking spots).”</p>	Major Virgílio
	<p>“The FODs we usually find result from the deterioration of the runways and taxiways.”</p>	Major Andrade
	<p>“These objects are mostly found near the hangars and on taxiways, but the runway is also prone to the appearance of FODs because of the impact of aircraft when landing.</p>	Captain Almeida

Question	Answer	Subject
In which places is it most common to have incidents or accidents involving FODs?	“It is hard to figure out. You have to try to correlate that the damage you had was caused by an FOD, which is difficult because most of the time you only see the damage after the moment of impact.”	Captain Rodrigues

Question	Answer	Subject
Which are the most common damages generated by FODs?	“I have no reports of major damage to the F-16 at the moment, but it is usually limited to damage to the first engine blades. These are inspected regularly. From memory, we have had some damage in foreign operations where you cannot always control the whole cleaning process.”	Major Virgilio

Question	Answer	Subject
Which aircraft are more prone to damage due to FODs?	“The F-16 is, undoubtedly, one of the most susceptible military aircraft to FOD. Not only because it is single-engine, where an FOD strike represents a possible total loss of the aircraft, but also because of the dynamics of the engine's air intake, which is relatively low above the ground [...]”	Major Virgilio
	“It is the jet aircraft; they are the most sensible due to the nature of their engines.”	Captain Rodrigues
	“Jet aircraft are more prone to suffer from FOD damage due to the characteristics and placement of their air intakes. In the case of aircraft with propellers and helicopters, it is more difficult, yet, helicopters' blades might hit light objects such as plastic bags projected by their downwash or the wind, which is a serious risk.”	Captain Almeida

*intentionally left blank*

## Annex C – Objects required by the FAA to be detected by that FOD detection systems

Figure C - 1 – Objects that FOD detection systems must detect according to FAA AC 150/5220-24 (FAA, 2009, p. 7).

**(1) Object Detection.** FOD detection systems must be able to detect the following objects (mobile systems must provide this performance at a minimum speed of 20 mph (30 km/h):

(a) An unpainted, metal cylinder, measuring 1.2 in (3.1 cm) high and 1.5 in (3.8 cm) in diameter,

(b) A white, grey, or black sphere, measuring 1.7 in (4.3 cm) in diameter (i.e., a standard size golf ball),

(c) 90 percent of the following group of objects when placed within a 100 ft by 100 ft (30 m by 30 m) square in the desired coverage area. One item from each category must be included in the group and each item must measure no larger than 4 in (10 cm) in any dimension unless otherwise specified:

- A “chunk” of asphalt or concrete,
- Any portion of a runway light fixture (in-pavement or edge light),
- An adjustable crescent wrench (up to 8 in. (20 cm) in length),
- A deep socket (at least 2 in. (5 cm) in length),
- A piece of rubber from an aircraft tire,
- A distorted metal strip (up to 8 in. (20 cm) in length),
- Fuel cap (aircraft or automotive),
- Lug nut,
- Hydraulic line (from aircraft or GSE, up to 8 in. (20 cm) in length)
- PVC pipe, white (2 in. (5 cm) in diameter), and

(d) Any two of the objects above, located no more than 10 ft (3 m) apart from each other, identified as separate objects.

*intentionally left blank*

## Annex D – Description of the FODs in our dataset

Table D - 1 – Description of the objects selected as FODs for our dataset and respective photograph.

Object No.	Object	Class	Width/ Diameter [cm]	Height [cm]	Material	Colour	Image
1	Car Tyre	Rubber Tyre	67	23	Rubber	Black	N/A
2	Rubber Stripe	Rubber	32	8	Rubber	Black	
3	Carbon Fibre	Carbon Fibre	49.5	28	Carbon Fibre	Black	
4	Metal Bar	Large Metal	29	4	Steel	Grey	
5	Bolt 1	Bolt	1.5	10	Steel	Brown	
6	Small Metal Tube	Small Metal	2	11	Aluminium	Grey	
7	Paint Chip	Paint	9	8	Paint	White	

8	Small Nut	Nut	3	1.5	Steel	Grey	
9	Bolt 2	Bolt	1.5	6.5	Steel	Grey	
10	Bolt 3	Bolt	1.5	8.5	Steel	Grey	
11	Bolt 4	Bolt	2	2.5	Steel	Brown	
12	Wrench	Tools	21	3.5	Steel	Grey	
13	Bent Carbon Fibre	Carbon Fibre	21.5	5	Carbon Fibre	Black	
14	Rock	Rock	8	4	Rock	Grey	
15	Asphalt	Pavement	6.5	4	Asphalt	Grey	

16	Safety Wire	Safety Wire	19	6.5	Steel	Grey	
17	Lug Nut	Nut	3	3.5	Steel	Grey	
18	PVC Tube	Tube	9	10	PVC	White	
19	Hydraulic Line	Line	19	12	Plastic	Yellow	
20	Brown Metal Stripe	Small Metal	13	3	Metal	Copper	
21	Wide Metal Tube	Tube	4	8	Steel	Grey	
22	Golf Ball	Golf Ball	4.3		Plastic	White	
23	Bent Metal Stripe	Small Metal	8	6	Steel	Grey	

*intentionally left blank*

## Annex E – Description of the FODs from the fourth acquisition

Table E - 2 – Description of the objects selected as FODs for testing the trained models and respective photographs.

Object No.	Object	Class	Width/ Diameter [cm]	Height [cm]	Material	Colour	Image
1	Tree Branch	Organic	86	60	Rubber	Brown	
2	Metal Plate	Small Metal	7	7	Rubber	Grey	
3	Bolt	Bolt	1.5	10	Carbon Fibre	Copper	
4	Plastic Tube	Plastic Tube	3	13	Steel	White	
5	Bent Metal	Small Metal	4	4	Steel	Grey	
6	Grass Cutter Wire	Plastic Line	23	14	Plastic	Red	
7	Cloth	Textile	17	10	Textile	Dark Blue	

8      Wooden  
Stick      Organic      19      2      Steel      Brown

