An Analysis of Motion Smoothness in Video Object Detection

by

Mohammed Hamada Gasmallah

A thesis submitted to the
School of Computing
in conformity with the requirements for
the degree of Master of Science

Queen's University
Kingston, Ontario, Canada
May 2020

Copyright © Mohammed Hamada Gasmallah, 2020

Abstract

Machine learning in computer vision has become an invaluable aspect of research on object detection and object tracking. While advancements in current research aim to improve the matching of predictions with ground truth bounding box annotations from humans, to our knowledge, very little work is currently being done on analyzing bounding box path smoothness. Bounding box path smoothness is useful as it can contribute to improve machine vision. Additionally, it provides another metric by which researchers can assess the capabilities and qualities of video object detection systems.

In this work, we investigate the problem of object bounding box path smoothness in video object detection systems. We begin by studying the fields of convolutional neural networks for object detection systems, and smoothness metrics from biokinematics research. Two smoothness metrics from this field, namely Log Dimensionless Jerk (LDLJ) and Spectral Arc Length (SAL), are adapted for usage in object bounding box paths and an analysis is done to justify the adaptations made. An in-depth analysis of two bounding box proposal generation systems is done using the two adapted smoothness metrics and validated against the ground truth bounding box paths. The analysis showed that both LDLJ and SAL can differentiate between all tested object bounding box path generation systems. Additional experiments demonstrate that the human annotations are the most smooth bounding box paths, however, the object detection systems tested can be improved naively by doing a moving average over proposed paths.

Finally, we adapt the smoothness metrics as loss functions in a video object detection system to analyze if it could be used as a regularizer on video object detection using convolutional neural networks. We propose, train and analyse a model on video object detection with 7 training regimens which vary only in the regularizer. We found that using a smoothness regularizer can improve object path smoothness by a small amount and conclude with a list of possible future work.

Acknowledgments

I would like to thank my supervisors Dr. Farhana Zulkernine and Dr. François Rivest for their high expectations, support, guidance and immeasurable help. Thank you for believing in my ideas, and pushing me to reach my limits.

Special thanks to all the members of my thesis committee, for dedicating the time to read my thesis and evaluating my work. A special thank you to all the members of the Big-Data Analytics and Management (BAM) Laboratory as well as the Natural and Artificial Adaptive Intelligent Systems Laboratory (NAAIS-SIANA) for their fruitful collaborations, the many discussions and the help that you provided.

Thanks to my family for their support, love and for bringing me to Canada and providing me the opportunity to achieve all that I have achieved.

Thanks to my friends and all those who provided me the love and support throughout the course of my thesis. Thank you for keeping me sane and for all the fun we had.

Publications

Submitted To Conference

- From Chapter 3: Mohammed Gasmallah, Francois Rivest, and Farhana Zulkernine. Quantifying Path Smoothness in Video Object Tracking. *Submitted to ECCV2020*, 2020.

Published

- Mohammed Hamada Gasmallah and Farhana Zulkernine. Video Predictive Object Detector. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 365–371. IEEE, nov 2018.

- Mohammed Gasmallah, Farhana Zulkernine, Francois Rivest, Parvin Mousavi, and Alireza Sedghi. Fully End-To-End Super-Resolved Bone Age Estimation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11489 LNAI, pages 498–504. 2019.

- Alex Wojaczek, Regina-Veronicka Kalaydina, Mohammed Gasmallah, Farhana Zulkernine, and Myron R. Szewczuk. Computer Vision for Detecting and Measuring Multicellular Tumor Spheroids of Prostate Cancer. *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019.

Nomenclature

$\hat{V}(\omega)$	The normalized amplitude value of a movement profile at frequency ω
ω	A frequency
ω_c	A frequency threshold
$\sigma(x)$	Sigmoid function
AP_c	The average precision of a particular class c
B_p	The set of predicted bounding boxes
b_p	The predicted bounding box
b_{gt}	The ground truth bounding box
b_{gt}	The set of ground truth boxes
b_{MAR}	The smallest convex bounding that encompasses all boxes.
I_h	The image height
I_w	The image width
p_i	The precision value at a particular iteration i

P_{kh}	The k^{th} anchor box's' height
P_{kw}	The k^{th} anchor box's width
r_i	The recall value at a particular iteration i
S_h	The number of sliding windows in the height dimension of the image
S_w	The number of sliding windows in the width dimension of the image
s_x	The sliding window x index
s_y	The sliding window y index
t_x	The regressed x offset of the region proposal
t_y	The regressed y offset of the region proposal
$V(\omega)$	The amplitude value of a movement profile at frequency ω
C	Set of total classes
c	Index of class
N	Total number of inference samples
n	Index of inference sample

Acronyms

ALDLJ Adapted Log Dimensionless Jerk.

ASAL Adapted Spectral Arc Length.

CNN Convolutional Neural Network.

DLJ Dimensionless Jerk.

GIOU Generalized Intersection over Union.

IOU Intersection over Union.

LDLJ Log Dimensionless Jerk.

LSTM Long Short Term Memory.

mAP Mean Average Precision.

MOT Multi-Object Tracking.

SAL Spectral Arc Length.

SSD Single Shot Detector.

YOLO You Only Look Once.

YOLOW You Only Look Once Windowed.

Glossary

Batch updates A batch update is a multi-example backpropagation weight update where the error is aggregated over multiple examples.

Centroid A centroid is a box which only has two values, $[w, h]$. It it is always presented as being in the centre of another box.

Convolutional neural network A convolutional neural network is typically a network that has a convolutional layer which performs the feedforward pass as a convolution between a kernel and the input set. The kernel acts as the weights which are to be optimized for the loss function.

Dense network A dense network is a network where each *unit* in a given layer is densely connected to every unit in the next layer. Typically this is represented as a matrix multiply between the inputs and the weights.

Dimensionless A dimensionless quantity is a quantity with no physical units and is thus a pure number. It is the product of a function which cancels all the physical units.

Recurrent network A recurrent network is a dense network which has, as input, a function of its output at a previous timestep.

Stochastic updates A stochastic update is a single example backpropagation weight update.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As we progress in the era of real-time information processing, video object detection and tracking are becoming increasingly more important areas of research in computer vision [23, 29, 39, 44]. Having the ability to detect objects and categorize them in a scene allows systems to be able to make complex and important decisions regarding real-world objects.

There have been many breakthroughs with deep learning approaches to object detection, in particular deep convolutional neural networks [29, 39, 41]. These models are often initially trained on large image recognition datasets such as the ImageNet Large Scale Visual Recognition Challenge [44] and then transfer learning is applied to learn object detection datasets (such as Common Objects in Context [29]). These object detection systems make their predictions by doing both bounding box regression and image classification. Typically, object detectors perform single frame object detections [23, 29, 38, 39].

Image object detection and object tracking are fundamentally important problems in computer vision [23, 38]. In recent years, the field has seen many interesting solutions involving deep neural networks [26, 38, 39, 41]. With the advent of newer more complete image and video datasets, interesting questions regarding the use and application of these object detectors and object trackers have arisen [29, 32]. For example, single image object detectors are often used in tracking by detection scenarios and yet unable to take advantage of temporal information [23]. More recently, since deep neural networks are being used for object detection and tracking, it has become more feasible to use these systems in industrial applications [46, 50]. Using object tracking by detection allows for video analysis to determine and track changes in paths that objects may take. For example, self-driving cars should track pedestrians, surveillance cameras should track people and all these systems should generate good paths along the object track [36].

Deep neural networks have been a sound alternative to the previous kernel methods and have produced amazing results by taking advantage of large repositories of labelled datasets. Many current object detection networks such as You-Only-Look-Once from Redmon et al. [38] or Faster R-CNN from Ren. et al. [41] are single image object detectors that perform very well. These single image object detectors lack temporal information in video object detection and do not maintain any visual memory of a particular video sequence. The ability to have a visual memory of a video sequence can aid in spatial and temporal based object recognition and segmentation [51]. Others have replaced or added memory modules to the convolutions aiding networks in detecting objects in video [5] or in predicting objects in video [13]. An example of the output of the predictions from video object detectors can be seen in Fig. 1.1. Bounding boxes are shown with the corresponding class prediction at the top and a network confidence score beside predicted class label. Note the blurry or partially obstructed pedestrians are not well classified. In video object detection, bounding boxes are predicted in each image frame of a video and an object path is defined by these boxes. Often this path is jerky as the boxes are not accurately detected or are not detected at all because of obstructions or occlusions.

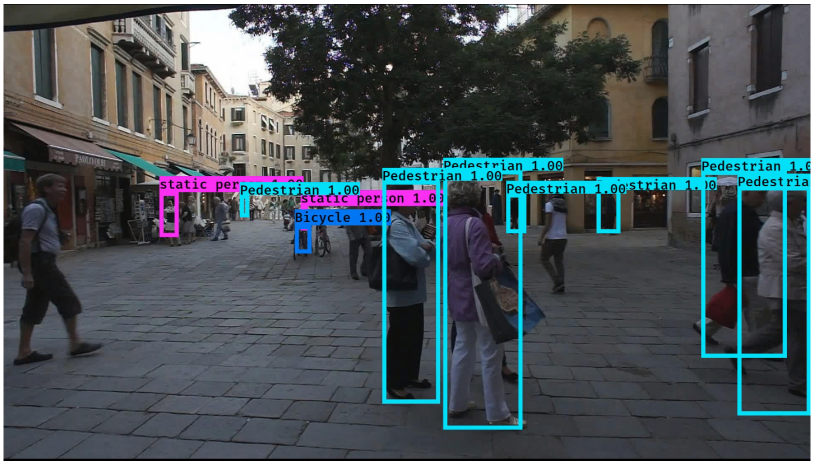


Figure 1.1: An example of object detection from video [13].

1.1 Motivation

Object detection and object tracking systems that have a temporal signal may allow for spatiotemporal pattern recognition and thus could be useful for generating complex temporal patterns to match object trajectories as they move in more complex ways. These complex trajectories and unforeseen occlusions are important to be able to mitigate and predict as they can severely affect object tracking quality. Timing critical problems such as self-driving cars [32], drone path projection [8], and pedestrian detection [50] require that the network be robust to complex object path noise. Leal-Taixé uses a smoothing term to overcome the unsmooth noise in object paths from the Multi-Level Hungarian multi-object tracker [22]. Since single image object detectors do not have a temporal signal, previously predicted objects can be dropped in a frame and quickly forgotten. This can result in jittering of object paths, leading to non-smooth object paths. This can be well demonstrated in videos where blurry or partially obstructed pedestrians can drop bounding boxes quickly, and bounding boxes must change aspect ratios to accommodate moving or turning pedestrian. Examples of some of these can be seen in the static image of Fig. 1.1.

Smooth paths are important as they allow human interpreters of the system to better understand and predict object paths [47]. In addition, smooth object paths allow for more robust and stable implementations of algorithms that use these object paths, such as those involved in collision detection in self-driving cars [22, 32] and those involved in Unmanned Arial Vehicles (UAV) [55].

1.2 Problem Definition

Since single image object detectors, video object detectors and object trackers are only trained on metrics that relate to ground truth paths [14, 38, 39, 40, 41, 49] there is no direct way for networks to learn to predict smoother boxes. Instead, the assumption is that ground truth bounding boxes are the smoothest possible paths and no other smoothness regularization is required. Smoothness is an intuitive concept in motion and can be seen in the examples from Fig. 1.2. This example comes from an object detector trained on a variety of classes. For Fig. 1.2 we focus on bounding boxes tracking people. The green bounding box (labelled A) changes in width and height, and jitters in its centroid location. The white bounding box (labelled B) changes in a temporal sense, in that it drops the bounding box for one frame, making it temporally unsmooth.

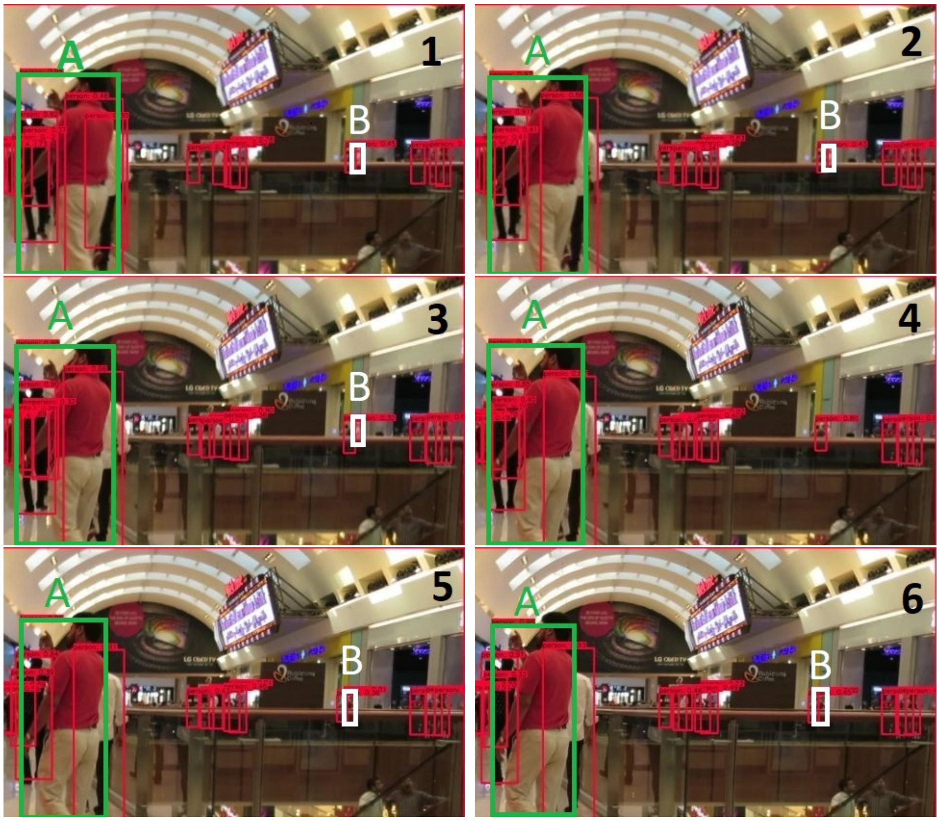


Figure 1.2: Consecutive object detections on 6 frames from the Multi-Object Tracking dataset. The object with the green bounding box (labelled A) demonstrates the smoothness problem on the spatial bounding box properties. The object with the white bounding box (labelled B) demonstrates the smoothness problem on the temporal bounding box properties. Note that the white bounding box disappears in frame 4.

Object path smoothness is vital in qualitative assessments of object predictions in video and could be useful in determining which object detection methods should be used in production systems. Cognitive science research in object recognition has found that smoothness constraints in object recognition is necessary for enhanced colour/shape recognition and binding [53]. Studies have found that infants familiarized with smooth objects are better at identifying those objects than infants who are familiarized with unsmooth or discontinuous objects [47]. Thus having some way of measuring and quantifying object path smoothness would be useful for the domain of object detection and object tracking.

In this thesis, we address a few important research questions regarding smoothness in object detections. They are:

- What is the definition of object path smoothness?
- How do the state-of-the-art single image object detectors differ with respect to object path smoothness?
- What is the effect of the use of object path smoothness as a regularization method for a temporal object detector?

By answering these questions, we explore the field of motion smoothness as it relates to bounding box paths in object detection systems.

1.3 Proposed Solutions

In approaching this thesis, we began by first finding suitable definitions of smoothness and ways to quantify smoothness. We tackle the idea that smoothness is inherent in ground truth boxes and formulate a hypothesis that we test. These smoothness metrics could be used to aid networks with a temporal signal to predict object paths which are more robust to noise in movement and inherent network noise in bounding box generation. We validate this hypothesis by including our proposed smoothness metrics in an optimization algorithm for a video object detection model to develop a smooth video object detector.

To define object path smoothness, we research smoothness literature in physics and biokinematics and find appropriate definitions which can be adapted. We devise experiments to evaluate if these mathematical models can be used for computer vision. We select a video object dataset and a few bounding box generation methods which are then evaluated using the adapted smoothness metrics. This experimentation provides us with two insights: it helps confirm the effectiveness of the proposed smoothness metrics and helps evaluate the smoothness of the state-of-the-art object detectors in video object detection scenarios against traditional well-known object detection metrics. Using these metrics, we compare the differences in object path smoothness amongst these single image object detectors. Finally, we train a windowed temporal object detector and regularize it with the adapted smoothness metric. We evaluate the differences in performance of this regularized windowed temporal object detector.

1.4 Contributions

The following are the key contributions from this thesis:

- We explore the field of single image and video object detection using convolutional neural networks and report that, to our knowledge, motion smoothness is not a field that is discussed in object detection. We research mathematical models which have been used in biokinematics for human motion and propose two new metrics for measuring bounding box path smoothness by adapting the models from biokinematics. An example bounding box path can be seen by following the green (A) or white (B) bounding box in Fig. 1.2.
- The mathematical models from motion smoothness in biokinematics are found to be either jerk-based or non-jerk-based models. In order to adapt these models to bounding box paths, we perform an extensive study to evaluate both the

adaptations made to these mathematical models and the effectiveness of these metrics.

- The proposed adapted mathematical models for motion smoothness in bounding box paths are used to evaluate You Only Look Once (YOLO)v3 and Retinanet, two state-of-the-art approaches to object detection in computer vision and validate that the metrics can be used to perform qualitative analysis to compare video object detection models. All comparisons are done against the baseline of human generated bounding box proposals from the ground truth.
- We develop methods to integrate the metrics in the training regimen to regularize the models to perform smooth object detection. Then we evaluate the effects that the metrics have on the final performance of these models and perform some qualitative analysis on these final predictions.

1.5 Organization of Thesis

This thesis is organized in the following way:

- Chapter 2 presents a background study of concepts regarding deep learning models for object detection, temporal object detection and tracking, and motion smoothness from biokinematics.
- Chapter 3 presents our study of the smoothness metrics for use in video object path smoothness including a validation of these metrics using the Multi-Object Tracking Dataset and two well known single image object detectors.
- Chapter 4 uses the proposed smoothness metric and adapts them as losses. These losses are used to regularize a network that has some temporal signal.

Then we evaluate the impact of using these adapted smoothness metrics as loss on the network performance.

- Chapter 5 concludes and summarizes all the findings of this thesis and suggests possible future works.

Chapter 2

Background

2.1 Introduction

This chapter presents the background concepts relevant to commonly used deep learning models for computer vision such as convolutional neural networks, object detection in single images and sequences of images, object tracking by detection, and motion smoothness. Convolutional neural networks are the building blocks for neural network-based object detection in both single images and videos. Motion smoothness is an important concept in biokinematics and is important to this thesis for determining how to quantify object path smoothness in video object detection scenarios.

Section 2.2 presents the background necessary for deep learning in computer vision, including backpropagation, convolutional neural networks, and recurrent neural networks. Section 2.3 presents the relevant background information for object detection in computer vision including the precise nature of the task, current convolutional architectural strategies and metrics for evaluation. Section 2.4 discusses the mathematical models found in biokinematics relating to motion smoothness. Finally, Section 2.5 summarises the previous sections.

2.2 Deep Learning for Computer Vision

In the past, pattern recognition based object detection in images was composed of two parts: a segmenter which extracted objects of interest and a feature extractor that extracted only the important patterns in an image [25]. The introduction of gradient-based learning from Rumelhart et al. [43] and the addition of the works of Lecun et al. [24] allowed for future work from Lecun et al. [25] to develop a gradient-based feature extraction method that proved extremely useful to the field of computer vision.

2.2.1 Neural Networks

Rumelhart et al. introduce the concept of a new learning procedure named back-propagation on neuron-like units [43]. The principal idea here is that the outputs of the units are linear functions of the input and the weights of the neuron-like units [43]. In essence, there are two phases of each learning step as described by Rumelhart et al. [43]. The first step is a forward pass which begins by taking input and determining the state of hidden units through some function which has parameters W, see equations (2.1)-(2.2) for a single layer example.

$$Y_{pred} = F(W(i), W_b(i), X(i)) \tag{2.1}$$

$$F(W(i), W_b(i), X(i)) = \sigma(W(i)X(i) + W_b(i)) \tag{2.2}$$

where Y_{pred} is the predicted matrix output, $X(i)$ is the input matrix at training index i, $W(i)$ is the set of trainable parameters for the function F at training index

i, $W_b(i)$ is a set of trainable parameters for the bias of function F at training index i, F is the function which details the forward pass, and σ is the element-wise activation function.

Rumelhart et al. began their formulation of these nBertasius2018eural networks and the backpropagation method with the simple concept of the feedforward neural network. This feedforward neural network is as described above, with an input that is modified by a function F and a set of weights W and some activation function σ, producing an output Y_{pred}. The formulation of the backpropagation algorithm and the chain rule partial derivation allows for these functions to be stacked on top of each other quite easily in what Lecun et al. call the simplest multilayer learning machine [24]. An example of a two-layer neural network can be described by equation (2.3). This type of network is often called a dense network, as every *unit* in a layer is fully connected to its inputs.

$$Y_{pred} = F_2(W_2(i), W_{b2(i)}, F_1(W_1(i), W_{b1(i)}, X(i))) \tag{2.3}$$

where Y_{pred} is the same as in Equation 2.1, and each F, W and W_b is subscripted to denote independent functions, weights and biases.

The performance of the model can be calculated using a cost function that evaluates the *error* of a model [24, 43]. Rumelhart et al. define the total error as in equation (2.4) [43],

$$E = \frac{1}{2} \sum_i \sum_j (y_{pred(j,i)} - y_{true(j,i)})^2 \tag{2.4}$$

where the i is the index of a particular set of input-output pairs during training,

j is the index over output units, $y_{pred(j,i)}$ is the predicted output from the forward pass, and $y_{true(j,i)}$ is the desired output [43]. Lecun et al. do not specify a specific cost function, and instead leave it as a generalized differentiable function [24].

Once outputs and error have been calculated, the second phase begins. The second phase is the backwards pass. The backwards pass starts with computing the partial derivative $\partial E/\partial Y$ [43]. These calculated values allow us to, through the chain rule, *backpropagate* this error and adjust the weights. The weights are adjusted using the following gradient descent equation [24, 43]:

$$W(i) = W(i-1) - \eta \frac{\partial E}{\partial W(i-1)} \tag{2.5}$$

where $W(i)$ are the weights at training index i, η is the learning rate, $\partial E/\partial W(i)$ is the gradient of the error with respect to the weights at training index i.

Certain formulations of the backpropagation algorithm use centroid, where weight updates are made per sample. Other formulations are batch updates, where weight updates are aggregated for a certain number of samples and updated all at once. Lecun et al. were one of the first to discuss the advantages and disadvantages of stochastic vs batch updates, and a variety of other modern tricks in neural networks such as example shuffling, input normalization, and nonlinear activation functions [24]. Many of these techniques have become standardized techniques in neural network algorithms and machine learning.

2.2.2 Convolutional Neural Networks

Lecun et al. observed that simple fully-connected feed-forward neural networks could not maintain the topology of the input [25]. As images have specific local structures and pixels have a high spatial correlation, Lecun et al. presented convolutional neural networks (CNN) to extract local features [25]. Fig. 2.1 represents the typical setup for a fully connected neural network with an image as input. For comparison, Fig. 2.2 represents a single layer convolutional neural network architecture. Convolutional neural networks differ from fully-connected feed-forward neural networks in that the parameters that are trained are the parameters for the kernel in the convolution. As a reminder, a convolution is an operation defined as in equation (2.6). In computer vision, convolutions were used with predefined filters to extract particular image features such as edges.

$$(f * g)(t) = \int_{m=0}^{t} f(t)g(t-\tau)d\tau \tag{2.6}$$

where f and g are the convolutions operands, t is typically the time-domain (though it is not necessary that it be time), and τ is a free variable.

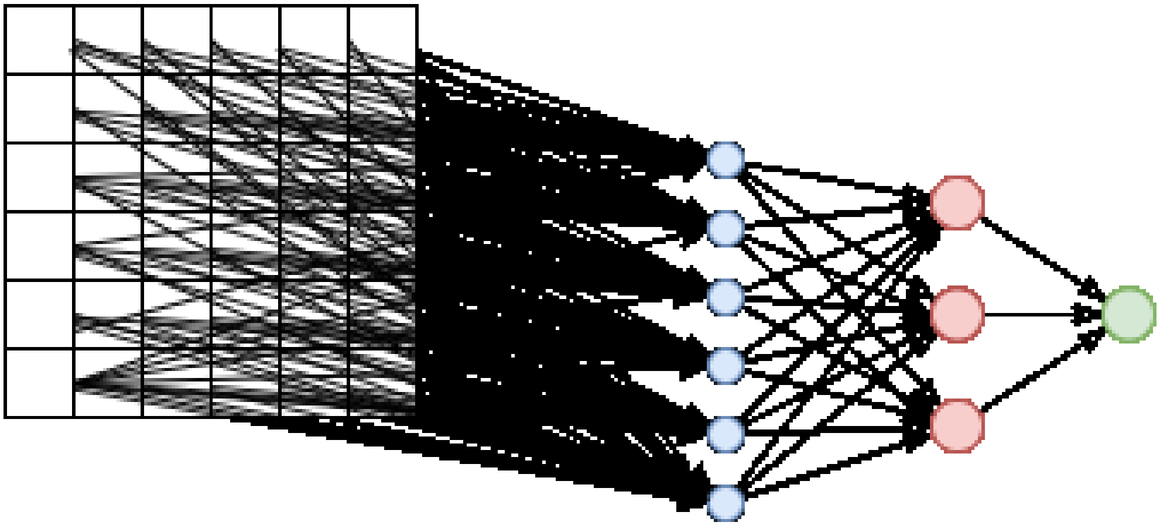


Figure 2.1: A fully connected neural network with an image as input. Note that each pixel-node pairing has it's own trainable weight.

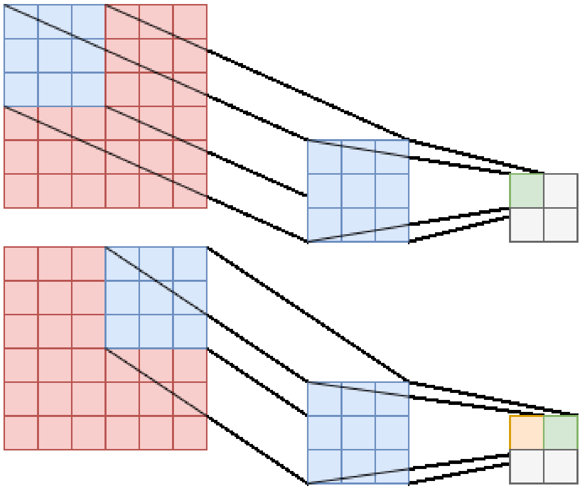


Figure 2.2: A single layer convolutional neural network with a 3x3 kernel and no padding. The convolution does the matrix multiply on a sliding window over the image. This is illustrated by the top figure and the bottom figure. Note that only the kernel has trainable weights. These are reused spatially.

The network presented by Lecun et al. paper dubbed LeNet-5 and uses two convolutional layers, each followed by subsampling layers which perform an average pool in a two by two area, and two fully connected layers to perform digit recognition [25]. This network with minor variations achieved an error rate of 0.8% in digit classification [25]. This CNN showed that in the field of computer vision there is no need to hand-craft feature extractors and that we can train feature extractors using gradient-based learning methods [25]. The major boom in computer vision and many other fields in computer science can be attributed to the rise of computing power and the availability of larger datasets [18, 26, 29, 32, 44].

2.2.3 Long Short Term Memory Networks

In an effort to use these neuron-like units to learn temporal patterns, Hochreiter and Schmidhuber proposed a novel, gradient-based method called Long Short Term

Memory (LSTM) [17]. The LSTM is a recurrent network which has become one of the standards in temporal learning [13, 33, 45]. Hochreiter and Schmidhuber designed the LSTM for learning with gradient-based methods and enforces a constant error flow through the LSTM internal states [17]. The LSTM became one of the few methods to learn long term and short term patterns in excess of 1000 steps [17].

Interestingly, these same LSTM units can be used in convolutional neural networks. Shi et al. proposed a convolutional LSTM in order to benefit from the spatial learning enabled by convolutional neural networks, and to learn the temporal patterns using the LSTM units [45]. The convolutional LSTM is created by using LSTM units for each of the units in the convolutional kernel. Shi et al. show that the convolutional LSTM is able to learn spatiotemporal patterns by demonstrating its efficacy in precipitation nowcasting [45].

2.3 Object Detection

In the field of computer vision, object detection is vital as it is one of the many tasks involved in understanding visual scenes [4, 26, 29, 35, 36, 37]. In the past decade, there have been many major improvements in object detection due to improvements in deep neural network algorithms, computer vision techniques, and access to large databases containing labelled images [9, 23, 28, 29, 38, 41, 44]. The object detection task is characterized by having a system that accomplishes two computer vision tasks simultaneously: object localization, and object classification. Object localization is a computer vision task in which an algorithm must, given an image, infer the smallest bounding box that contains the entirety of a desired object [41]. Object classification is a computer vision task in which a model must, given an image of an object, infer the class of that object [41]. The classes are often defined by the dataset used to train the model and can vary, for example, the pascal VOC class has 20 labelled object classes [9], and the OpenImagesV4 dataset has 600 classes labelled [21].

Object detectors can be placed into two categories: a two-stage object detector or one stage object detector [18, 28, 30, 38]. Two-stage object detectors perform object localization in the first stage and object classification in the second stage. One stage object detectors perform both object localization and object classification in a single processing stage.

2.3.1 Two Stage Detectors

Two-stage detectors are known for being accurate yet slow [18, 28]. For many years, the state-of-the-art object detectors were two-stage object detectors [28]. The two-stage object detection approach was initially proposed by Girshick et al. through the method they coined *R-CNN: Regions with Convolutional Neural Networks (CNN) features* [15]. This approach has two separate stages. The first stage is a category-independent object region proposal stage which provides a large number of possible regions containing objects [15, 41]. Girshick et al. adopt methods from external modules for proposing regions. The speed improvement gained by Ren et al. is due to their use of a region proposal network which they train for generating object proposals using prior anchor boxes [41]. These anchor box proposals are generated in a sliding window over the entire image providing a large variety of different region proposals (see Section 2.3.1 details anchor boxes in more detail).

The second stage is the feature extraction and object classification stage [15, 41]. Girshick et al. generate a 4096-dimensional feature vector using a convolutional feature extractor for each region proposal and train a linear SVM to classify the proposed regions [15]. To speed up the computation, Girshick R. use a convolutional feature extractor and then extract the regions from the feature extractor itself [14]. Ren et al. speed up this computation again by training a neural network classifier instead of using a linear support vector machine (SVM) for the classification of the proposed regions [41]. One of the novel ideas proposed by Ren et al. allows for feature maps used in the region proposal network to be passed forward to the classifier so that no feature information is lost [41]. The ideas from Girshick et al, Ren et al., and Dai et al. inspired the state-of-the-art for many years due to the robustness of the region proposals provided by two-stage object detection [6, 14, 15, 18, 41].

Anchors

Anchor boxes are useful prior template boxes with different scales and aspect ratios that allow object detectors to avoid directly regressing the bounding box values: $[b_x, b_y, b_w, b_h]$. An anchor box is composed of the scale/aspect ratio of the box: $[P_w, P_h]$. Anchor boxes allow region proposals to be generated using offset values that modify anchor boxes. As there are multiple anchor boxes in a sliding window, the region proposals must also include a confidence value for each anchor box [39, 41]. Anchor boxes with a variety of scales allow for object detectors to predict objects with a large variety of sizes.

Given the network output of $[t_x, t_y, t_w, t_h]$, the equation to compute the bounding box values in pixel space are given by equations (2.7)-(2.10).

$$b_x = \frac{\sigma(t_x) + s_x}{S_w} I_w \tag{2.7}$$

where b_x is the final box center x value, t_x is the tensor x output, s_x is the grid x index from which t_x was extracted, S_w is the output tensor grid width and I_w is the final image width.

$$b_y = \frac{\sigma(t_y) + s_y}{S_h} I_h \tag{2.8}$$

where b_y is the final box center y value, t_y is the tensor y output, s_y is the grid y index from which t_y was extracted, S_h is the output tensor grid height and I_h is the final image height.

$$b_w = \frac{P_{kw} e^{t_w}}{S_w} I_w \tag{2.9}$$

where b_w is the final box width value, t_w is the tensor width output, k is the anchor index from which t_w was extracted, P_{kw} is the k anchor box's template width, S_w is the output tensor grid width, and I_w is the final image width.

$$b_h = \frac{P_{kh} e^{t_h}}{S_h} I_h \tag{2.10}$$

where b_h is the final box width value, t_h is the tensor height output, k is the anchor index from which t_h was extracted, P_{kh} is the k anchor box's template height, S_h is the output tensor grid height, and I_h is the final image height.

Ren et al. originally proposed using template anchor boxes [41]. They used 9 template anchor boxes. The 9 anchor boxes were split into three different retinal scales and three different aspect ratios. The three retinal scales are $\{128^2, 256^2, 512^2\}$ and the three aspect ratios are $\{1:1, 2:1, 1:2\}$ [41]. Ren et al. choose the scale and aspect ratio independent of the distribution of bounding boxes in the dataset

[41]. The scales and aspect ratios are based on the intuition that three different scales will allow a large variation of bounding boxes to be predicted. Redmon, and Farhadi take a different approach to choose the optimal bounding boxes [39]. They pick better prior anchor boxes based on the distribution of the bounding box aspect ratios in the dataset [39, 40]. This is done by running a k-means clustering algorithm on the translation-invariant Intersection over Union (IOU) of the anchor box priors and bounding boxes in the training set (see equation (2.11)) [39].

$$d_{iou}(box, centroid) = 1 - IOU(box, centroid) \tag{2.11}$$

where d_{iou} is the distance which is minimized by k-means clustering between a box and a centroid, and IOU is defined in Equation (2.12).

This allowed Redmon, and Farhadi to use fewer anchor boxes in YOLOv2, $k = 5$ and yet achieve an mAP that greater than the one achieved by using the same aspect ratios as those presented in [41]. In YOLOv3, Redmon, and Farhadi found that using a combination of three different retinal scales with three data mined anchor priors led to the best results [40].

2.3.2 Single Stage Detectors

Single Stage Detectors are well known for being exceptionally fast object detectors with a trade-off in accuracy [18, 30, 39, 41, 49]. Single-stage detectors use a single CNN to predict both class and anchor box offset without a second feature extraction and classification stage [18]. Two of the first single-stage object detectors are the Single Shot Detector (SSD) [30] and You Only Look Once (YOLO) [38]. SSD uses a fully convolutional neural network that predicts the class and anchor box offsets while YOLOv1 reframes the whole problem as a regression problem that tries to immediately regress the bounding box coordinates and the class probabilities [38]. A newer version of YOLO regress anchor box offsets instead of directly regressing bounding box coordinates [39, 40]. The use of anchor box offsets led to an increase in mAP for YOLOv2 [39]. The release of SSD, YOLOv1 and YOLOv2 led to an increase in popularity of one-stage object detectors such as Retinanet, EfficientDet, and others [5, 10, 28, 33, 40, 49].

With this increase in popularity, many new object detectors attempt to use a variety of different techniques to overcome the accuracy trade-off of moving from two-stage detectors to single-stage detectors. One of the more successful breakthroughs for single stage object detection is the use of a novel loss function developed by Lin et al. [28]. They use the feature pyramid architecture from Lin et al. [27] and Liu et al. [30] as well as output predictions at three retinal scales. This inspired Redmon, and Farhadi to release YOLOv3 which also produced output predictions at the three different retinal scales [40]. Table 2.1 shows YOLOv3 as the fastest network and Retinanet as the best performing network.

Table 2.1: Table of the mAP and time taken for inference of different object detection methods [40].

Method	mAP	time in ms
YOLOv3-608	33.0	51
YOLOv3-418	31.0	29
YOLOv3-320	28.2	**22**
Retinanet-50-500	32.5	73
Retinanet-101-500	34.4	90
Retinanet-101-800	**37.8**	198
FPN-FRCN	36.2	172
R-FCN	29.9	85
SSD513	31.2	125
DSSD513	33.2	156

2.3.3 Validation Metrics

To assess the performance of object detectors, a metric must be used which combines the performance of both bounding box regression analysis and object classification. The most commonly used metric in object detection is the mean average precision metric [9, 18, 29]. Note that this metric must be applied to predictions after determining true or false positive rates based on the Intersection over Union (IOU) of the ground truth and the predicted bounding box [9]. Therefore, we will first cover Intersection over Union before discussing Mean Average Precision.

Intersection over Union

The Intersection over Union (IOU) metric is an evaluation criterion for bounding box regression and is used to determine the true-positive or false-positive rates of an object detector [9, 42]. The IOU is a ratio of the intersection area of the ground truth box with the predicted box to the union of the area of the ground truth box with the predicted box. IOU is a good evaluation criterion as it evaluates how good our prediction box is without having to recreate the exact ground truth x, y, w, h values. Additionally, the IOU is scale-invariant which means that it focuses on the area of the shapes regardless of the size of the bounding box [42]. This rewards boxes that heavily overlap with the ground truth. See equation (2.12) for the IOU equation.

$$IOU = \frac{area(b_p \cap b_{gt})}{area(b_p \cup b_{gt})} \tag{2.12}$$

where b_p is the predicted bounding box, and b_{gt} is the ground truth bounding box.

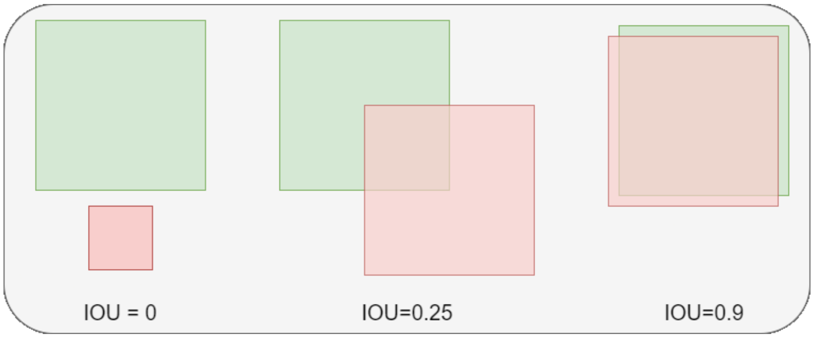


Figure 2.3: Example of two boxes and their respective IOU.

In the evaluation of true-positive or false-positive rates, ground truth matches are assigned to one of the predicted bounding boxes with the highest IOU. This is done by first sorting predictions in descending order of confidence. Predictions are matched to ground truth if their IOU is above a certain threshold and they have the same label. Everingham et al. use a threshold of 50%, whilst Lin et al. use the average over multiple IOU ranges of thresholds beginning at 50%, ending at 95% in increments of 5% [9, 29]. Beginning with the predicted bounding box having the highest IOU, we assign matches until either the ground truth has already been matched or the class has not been predicted correctly. For each positive match, the number of true positives is incremented. If the predicted box has a different class than a matched ground truth or has no matched ground truth, the number of false positives is incremented. Finally, if a ground truth has no matched predicted bounding boxes, the number of false negatives is incremented.

Generalized Intersection over Union

Although the IOU metric is the most popular evaluation metric for segmentation, object detection and tracking, it is not commonly used for bounding box loss [42]. This is mainly because the IOU metric is not differentiable everywhere and flattens to zero if predicted boxes do not overlap with ground truth boxes [42]. This means that the IOU metric would have no gradient after flattening to zero and so would not be able to suitably train a network [42]. Before the paper presented by Rezatofighi et al., networks trained for bounding box regression used an ℓ_2 or ℓ_1 norm (see equations for their bounding box regression [14, 38, 39, 40, 42]) and yet evaluated their networks using the IOU metric. Rezatofighi et al. have shown that although previous methods do train networks for bounding box regression, there are many cases where two boxes may have the same loss using ℓ_1 or ℓ_2 loss but these same boxes have different IOU metrics [42]. For this reason, they introduced the Generalized Intersection over Union

(GIOU) metric [42]. The GIOU metric is formulated as shown in equation (2.15).

$$\ell_1 = \sum_{i=1}^{N} |y_{true}(i) - y_{pred}(i)| \tag{2.13}$$

where i is an index of a bounding box set which has maximal index N. $y_{true}(i)$ is the ground truth of box index i and $y_{pred}(i)$ is the matched bounding box for index i.

$$\ell_2 = \sum_{i=1}^{N} (y_{true}(i) - y_{pred}(i))^2 \tag{2.14}$$

where i is an index of a bounding box set which has maximal index N. $y_{true}(i)$ is the ground truth of box index i and $y_{pred}(i)$ is the matched bounding box for index i.

$$GIOU = IOU - \frac{area(b_{MAR} \setminus (b_a \cap b_b))}{area(b_{MAR})} \tag{2.15}$$

where b_{MAR} is the smallest convex box that encompasses both bounding boxes b_a, and b_b, which is also known as the minimum area rectangle. Therefore, the GIOU varies in the range of [−1,1]. Negative values only occur when the enclosed bounding box b_{MAR} is greater than the IOU [42]. The IOU and GIOU can be used to define loss functions as in equations (2.16)-(2.17). Note that because of the particular ranges of both the IOU and GIOU, the ℓ_{GIOU} has a range of [0,2] and the ℓ_{IOU} has a range of [0,1].

$$\ell_{GIOU} = 1 - GIOU \tag{2.16}$$

$$\ell_{IOU} = 1 - IOU \tag{2.17}$$

Rezatofighi et al. show that the ℓ_{GIOU} is useful for training current object detectors such as YOLO V3 and Faster R-CNN, and provides an improvement of $4\% - 8\%$ in mAP over traditional ℓ_1 and ℓ_2 methods [42].

Mean Average Precision

To calculate the Mean Average Precision (mAP), one must calculate the average precision of each class in a set of classes. After calculating true positive, false positive and false negative rates using a particular thresholded IOU, one must calculate the precision and recall of each class using equations 2.18-2.19. Using these equations, the average precision can be calculated using equation (2.20).

$$p_i = \frac{tp_i}{tp_i + fp_i} \tag{2.18}$$

where i is an iteration, tp_i is a the true positive at iteration i, fp_i is the false positive at iteration i.

$$r_i = \frac{tp_i}{tp_i + fn_i} \tag{2.19}$$

where i is an iteration, tp_i is a the true positive at iteration i, fn_i is the false negative at iteration i.

$$AP = \sum_{n=1}^{N} p_n(r_n - r_{n-1}) \tag{2.20}$$

where n is an individual inference sample out of the set of all inference samples N.

The mAP is the mean of all of the average precisions of a set of classes as seen in equation (2.21). Although initially proposed by Everingham et al., the standard mAP calculation is the one that was proposed by Lin et al. [9, 29].

$$mAP = \frac{\sum_{c=1}^{C} AP_c}{|C|} \tag{2.21}$$

where c is a class in the set of all classes C.

2.3.4 Tracking-by-Detection

Visual object tracking is an important field in computer vision and, in prior years has been viewed as a separate task than object detection [37]. Object trackers must be robust to the complex motion characteristics that objects may undertake such as rotation, scaling and even partial occlusion [37]. Classical models require many specifications of the particular observation models as well as robust data association [36]. Many prior works use a variety of different techniques such as kernel tracking [37], using optical flow as an estimation of motion [7]. More recent work involves tracking in between detection frames, using deep learning methods to learn to track, and more recently, tracking-by-detection [23].

Held et al. train a CNN to learn the generic relationship between object motion and object appearance to track objects [16]. They require initial object proposals but can run quickly, nearing the 100 frames per second mark. Others perform fully end-to-end object tracking, leveraging the temporal abilities of recurrent neural networks to pass object information between frames such as [33, 36]. In particular, Ning et al. use a YOLO detector to perform detections at regular intervals and use a recurrent neural network to directly regress those object detections in between detection frames. Others frame the entire problem as a reinforcement learning problem. Zhang et al.

formulate the problem as a sequential decision-making problem where their network is trained on the reward of taking actions [54].

With the advent of better object detectors, new tracking systems often require complex interweaving of object tracking mechanisms and object detection mechanisms [10]. Feichtenhofer et al. propose a simpler method of tracking by detection through the use of multi-frame object detection and the use of feature correlation to inform the bounding box regressors and the classification proposals [10]. Leal-Taixé defines that the tracking-by-detection paradigm as being composed of a detector that performs detection on the entire scene and a tracker which performs the final data association step [22].

Object tracking is a difficult problem as the variety of scenes and camera angles are huge [22, 23]. Leal-Taixé differentiates two different scales of object tracking:

- *Microscopic* tracking which focuses on tracking individuals and specific objects.
- *Macroscopic* tracking which focuses on the density flow of crowds and typical motion tendencies.

In this thesis, we address microscopic tracking.

Smoothness

An interesting problem in object tracking is object motion estimation and object identification. Smoothness is a very important factor in preserving the main motion in a scene [7]. Even in humans, the principle of continuity and smoothness of motion is important for infants to develop object identity [47]. Spelke et al. define the smoothness principle as being “related to the principle of inertia in classical mechanics, whereby objects undergo linear motion at a constant speed in the absence of forces..."[47].

If we compare object tracking in computer vision systems and models of object recognition in vertebrate visual systems, there are clear elements in biological visual systems that leverage temporal smoothness [53] which are not modelled in any object detection or object tracking system.

Although single image and video-based object detection systems are impressive, the predicted object paths are jerky and inconsistent. Objects appear and disappear quickly in-between frames and boxes change rapidly. To our knowledge, none of the object detection or tracking systems include motion smoothness as part of their training schema.

2.4 Motion Smoothness

"Object Recognition is one of the most important functions of the vertebrate visual system" [53]. In biological visual systems, temporal smoothness has been theorized as one of the many features that may be heavily taken advantage of [53]. In the many studies on the mature visual system, there is evidence that a sequential view of an object helps in associating the object in a manner that helps recognition [53]. Spelke et al. suggest that the human visual system may be affected by the smoothness of object motion [47] and Wood J. suggests that a smoothness constraint in object recognition in the visual system aids in object recognition capabilities [53].

Smoothness, being an often qualitative metric, can be difficult to quantify. In the field of biokinematics motion smoothness is an important aspect to quantify. In this research paradigm, there are many smoothness measures, and Balasubramanian et al. clarify a few requirements for the smoothness measures to be useful [1]:

- It must be *dimensionless.* That is it must be a quantity with no physical units and is thus a pure number.
- It must have a *monotonic response* to motion, which in the case of smoothness measures, is to be entirely nonincreasing.
- It must be *sensitive to changes in movement characteristics.*
- It must be *computationally inexpensive* and *robust* to instrumentation noise.

Many of the previously used measures of motion smoothness in the field, do not satisfy these requirements and so are not useful for analyzing motion smoothness [1]. Balasubramanian et al. test a variety of smoothness measures and find that the Dimensionless Jerk (DLJ) and Log Dimensionless Jerk (LDLJ) are the only valid jerk-based measures of movement smoothness [2]. They modify the smoothness measure from their previous work [1] to make it valid as well, using inspiration from Beck et al. [3]. Thus there are two smoothness measurements that we will describe in detail in this section as they pertain the most to this thesis. They are the Log Dimensionless Jerk (LDLJ) and the Spectral Arc Length (SAL).

2.4.1 Log Dimensionless Jerk

The Dimensionless Jerk (DLJ) and the Log Dimensionless Jerk (LDLJ) are the only valid jerk based measures of movement smoothness [2]. The LDLJ and DLJ are defined as:

$$DLJ \triangleq -\frac{(t_2 - t_1)^5}{v_{peak}^2} \int_{t_1}^{t_2} |\frac{d^2 v(t)}{dt^2}|^2 dt \tag{2.22}$$

$$LDLJ \triangleq -ln(\frac{(t_2 - t_1)^5}{v_{peak}^2} \int_{t_1}^{t_2} |\frac{d^2 v(t)}{dt^2}|^2 dt) \tag{2.23}$$

where t is time, v_{peak} is the peak velocity between times t_1 and t_2, and $v(t)$ is the velocity at time t.

The LDLJ was developed as a way to mitigate some of the "ceiling effect" described by Balasubramanian et al. [1]. This metric is less robust at dealing with measurement noise as it is unable to differentiate between noise or signal [2].

2.4.2 Spectral Arc Length

This novel method for quantifying smoothness was pitched with the idea that we can picture smooth movements as being composed of low-frequency components and non-smooth movements as being composed of higher-frequency components. Balasubramanian et al. claim that instead of analyzing the frequency spectrum for quantifying smoothness, one can look at the complexity of the shape of the Fourier magnitude spectrum [1]. To measure the complexity of a curves' shape, they use the arc length (defined as the length along a curve). Using this basis, they define the Spectral Arc Length (SAL) as the "negative arc length of the amplitude and frequency-normalized Fourier magnitude spectrum of the speed profile with some frequency thresholds and amplitude thresholds" [1, 2] (see equation (2.24)).

$$\eta_{sal} \triangleq \int_0^{\omega_c} \sqrt{(\frac{1}{\omega_c})^2 + (\frac{d\hat{V}(\omega)}{d\omega})^2} d\omega \tag{2.24}$$

$$\hat{V}(\omega) \triangleq \frac{V(\omega)}{V(0)} \tag{2.25}$$

where ω is the frequency, ω_c is the frequency threshold, $V(\omega)$ is the amplitude of the movement profile at frequency ω, and $\hat{V}(\omega)$ is the normalized amplitude of the movement profile at frequency ω.

The SAL has been used in bio-kinematics as well as in assessing surgical skills with regards to smoothness [20]. Jantscher uses SAL as a real-time metric for assessing surgical performance and show that a sliding window of five seconds provided good feedback to the subjects of the trials [20]. The SAL is a promising measure as it is valid and particularly stable [1, 2]. The appropriate threshold values must be chosen. Otherwise, it is possible to get measures that do not appropriately relate to motion smoothness.

2.5 Summary

In this chapter, we have gone through the majority of the background information required to understand CNNs, single image object detection systems, video object tracking by detection, motion smoothness and a variety of metrics and terms important for the field. Although the field of video-based object detection has been advancing in terms of matching ground truth, no work has been done on analyzing motion path smoothness. We explore the usage of biokinematics based smoothness measurements for object motion path smoothness analysis in Chapter 3 and evaluate the adaptations made to these mathematical models for usage in computer vision.

Chapter 3

Measuring Smoothness in Video Object Detection

3.1 Introduction

The performance of object detection using neural networks is often evaluated using the Mean Average Precision (mAP) after Intersection over Union (IOU) thresholding [42]. The mAP metric is useful for comparing the ground truth boxes and predicted boxes. However, when applied to video, it does not provide any additional information about its temporal quality. In video, more information than just bounding box position and class is present and the quality of predictions can differ in a multitude of different ways. In particular, many object tracking by detection systems have run into the problem of bounding box path smoothness problems [31, 34, 35]. There is currently no agreed-upon metric to quantify motion smoothness or jerkiness of bounding box paths for tracking and detection systems. We propose two smoothness metrics for use with object tracking by detection scenarios.

To quantify object smoothness, we go to the field of biokinematics for inspiration. Since, smooth coordinated movements are often good characteristics for healthy human motor behaviour [1, 2], various smoothness metrics have been developed in the field of biokinematics to assess sensory-motor performance in patients [1]. We chose two valid, sensitive and practical metrics based on the findings of Balasubramanian et al. [2], namely Log Dimensionless Jerk (LDLJ) and Spectral Arc Length (SAL). We adapt these metrics for use in bounding box path scenarios and analyze the results of these metrics on the Multi-Object Tracking (MOT) Dataset [32].

The main contributions of this chapter are as follows:

- We propose applying the IOU between the boxes on two consecutive frames as a measure of speed of the bounding box path and transformation.
- We propose modifications to the LDLJ metric for usage with object detection and evaluate it using two object detection deep convolutional neural networks and ground truth.
- We propose modifications to the SAL metric for usage with object detection and evaluate it using two object detection deep convolutional neural networks and ground truth.
- We perform statistical tests on the results of applying both metrics to the entire MOT dataset to demonstrate that there are significant differences in the performance of YOLOv3 and Retinanet deep convolutional neural networks and the ground truth.

The rest of this chapter is organized as follows. Section 3.2 gives an overview of object detection networks and the two networks that we use in the experimentation. Section 3.3 is a reminder of motion smoothness metrics and how we define bounding box motion characteristics and explains the adaptations made to the metrics, the methods, and experiments that were performed to validate the adaptations. The results of the experimentation and testing of the metrics are reported in Section 3.4. We summarize the results of these experiments in Section 3.5.

3.2 Object Detection Networks

You Only Look Once is a state-of-the-art, real-time object detection system for use on standard object detection tasks [39]. YOLO is a single-frame, one-stage object detection system that prefers an accurate detector that is still fast and uses a 53 layer feature extractor known as Darknet-53 [38, 39]. The YOLO version used in this thesis is YOLOv3 which contains nine possible anchor boxes organized in groups of 3 [40]. Each of the groups corresponds to a different retinal scale, allowing a large variety of predictions in terms of bounding box scales [40]. This means that the final prediction tensor for YOLOv3 is of shape $[N, N, (3 * (4 + 1 + \text{num classes}))]$ where N is the number of grids the input image is divided by for each retinal scale. Note that N changes based on the retinal scale stride and the input image. In Redmon & Farhadi's work, they use strides of 8, 16 and 32, which leads to a possible N of 64, 32 and 16 respectively. The predicted boxes are then extracted from these tensors using equations (2.7)-(2.10) from chapter 2, and non-max suppression is used to get rid of duplicate boxes. This network is chosen as it is state-of-the-art in terms of speed of inference in object detection, making it a common choice for video-based object detectors. The architecture for YOLOv3 can be seen in Fig. 3.1.

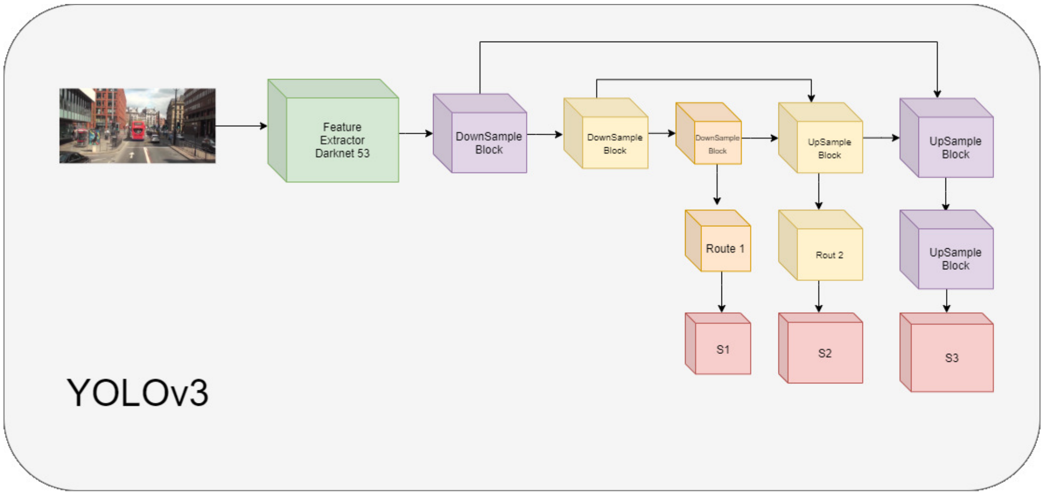


Figure 3.1: The YOLOv3 architecture in detail. Note that the dimensions of S1, S2 and S3 are based off the retinal strides and the shape of the input image.

Retinanet is a state-of-the-art, one-stage object detector that shares many similarities with previous dense, two-stage object detectors such as Region-Proposal Network and Fast-RCNN [28]. Retinanet focuses on using a feature pyramid network backbone and a novel focal loss to deal with class imbalance in object detection datasets [28]. The feature pyramid backbone constructs efficient multi-scale features from a single resolution image [28]. Retinanet uses 9 anchor boxes, grouped into 3 different retinal scales. This is comparable to the retinal scales and anchor boxes that YOLOv3 uses. We focus on Retinanet with Resnet-50 (a 50 layer version of Resnet) as the feature extractor. The architecture for Retinanet can be seen in Fig. 3.2.

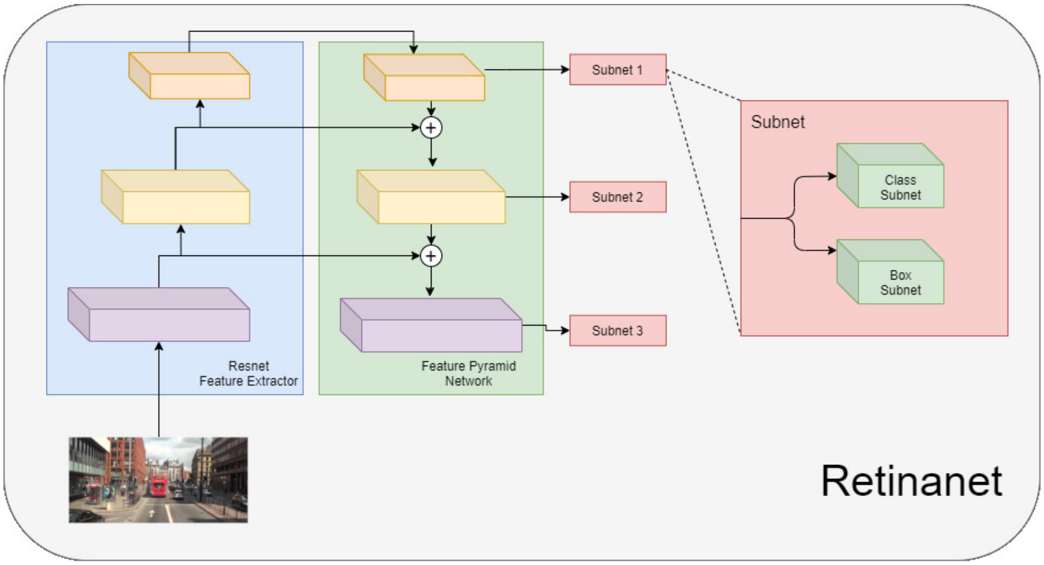


Figure 3.2: The Retinanet architecture in detail. Note that the dimensions of the outputs are based off the retinal strides and the shape of the input image.

3.2.1 Object Detection and Tracking Metrics

Object detection accuracy metric. The most commonly used evaluation metric in object detection is the Intersection over Union (IOU) metric [42]. This metric is a useful way to determine true positives and false positives when comparing predictions against ground truths and is a part of the process in determining mAP [29, 42]. The metric is often used to match predicted boxes with ground truth boxes based on IOU and threshold boxes which are not close to any ground truth. A Generalized Intersection over Union (GIOU) metric has been developed by Rezatofighi et al. in order to use an overlap metric as a regression loss [42]. This is particularly useful when bounding boxes are not overlapping in any way. This thesis will focus on the IOU metric.

Multi object tracking metrics. The Multi-Object Tracking Accuracy (MOTA) is the most widely used metric to evaluate the performance of an object tracking system [32]. MOTA is one of the many metrics used in the Multi-Object Tracking (MOT) dataset challenge, although they indicate that it may not serve as a single performance measure [32]. The MOTA was initially introduced by Stiefelhagen et al. [48] and is defined as:

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t} \tag{3.1}$$

where t is the frame index, GT is the number of ground truth objects, FN is the number of false negatives, FP is the number of false positives, and $IDSW$ is the number of mismatched errors. The IDSW can be calculated by counting the number of times an object path switches identity based on ground truth.

Additionally, Multiple Object Tracking Precision (MOTP) is commonly used in tracking challenges. The MOTP denotes the average dissimilarity between true positives and the corresponding ground truth [32]. For bounding boxes, it is defined as:

$$MOTP = \frac{\sum_{t,i} d_{t,i}}{\sum_t c_t} \tag{3.2}$$

where c_t is the number of matches in frame t, $d_{t,i}$ is the distance between the matched bounding box i with its assigned ground truth object.

Finally, in most object tracking scenarios, the only measure of a trajectory quality is known as the track quality [32]. Track quality is classified as either mostly tracked, partially tracked or mostly lost. This is done based on a percentage measure of successful object tracking [32].

3.3 Motion Smoothness

3.3.1 Bounding Box Speed Profiles

To measure bounding box smoothness over time, a speed profile using the proposed bounding boxes must be defined. We need a single metric that encompasses the smoothness of a box in terms of bounding box position (x and y) and bounding box scale change (w and h). Since IOU encodes the shape properties of the objects compared to a region and gives a normalized measure of their area [42], we can use the (1-IOU) between bounding boxes at times t and $t+1$ to encode the speed profile of a box at time t. Therefore, we formulate the speed profiles as a temporal IOU between two consecutive frames with the following equation:

$$\begin{aligned} v_{IOU}(t) &= 1 - IOU \\ v_{IOU}(t) &= 1 - \frac{|A_t \cap A_{t+1}|}{|A_t \cup A_{t+1}|} \end{aligned} \tag{3.3}$$

$$\begin{aligned} v_{GIOU}(t) &= 1 - GIOU \\ v_{GIOU}(t) &= 1 - \left(IOU - \frac{|C_t \setminus (A_t \cup A_{t+1})|}{|C_t|} \right) \end{aligned} \tag{3.4}$$

where $v_{IOU}(t)$ is the IOU speed at time t, $v_{GIOU}(t)$ is the GIOU speed at time t, A_t is the bounding box at time t and C_t is the smallest enclosing convex box for A_t and A_{t+1}.

If an object A is stationary and does not move between time t and $t+1$, we note that $v_{IOU}(t) = 0$ and $v_{GIOU}(t) = 0$. The maximal values for these metrics are based on the smallest possible overlap. These maximal values are $v_{IOU}(t) = 1$ and $v_{GIOU}(t) = 2$, this is because the IOU is bounded below by 0 and the GIOU allows for negative values up to -1 (where a value between 0 and -1 represents how far away the bounding boxes are from one another) [42]. Since it is unlikely for a bounding box to move beyond itself within one frame at 24fps, in this thesis we will focus on the IOU formulation only. We demonstrate the effectiveness of using the temporal IOU as a measure of smoothness in Subsection 3.4.2 by plotting the temporal IOU of an object path and its smoothed variants. Fig. 3.3 illustrates three examples of the temporal IOU at times $t - 1$ and time t.

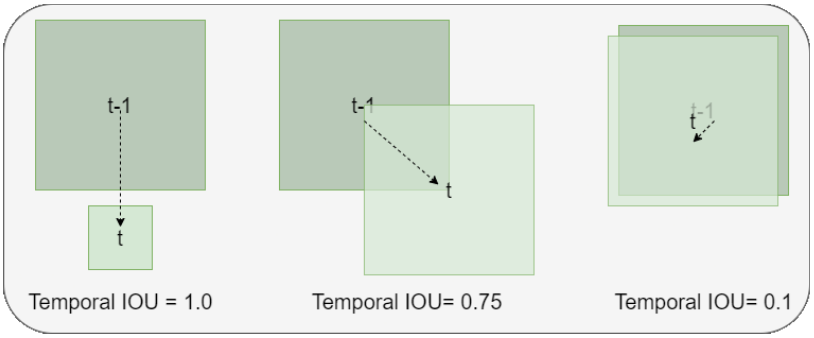


Figure 3.3: Example of the bounding box of an object at time $t-1$ and time t as well as their respective temporal IOU.

3.3.2 Smoothness Metrics

Balasubramanian et al. define motion smoothness as “a quality related to the continuality or non-intermittency of a movement, independent of its magnitude and duration”[2]. A smoothness measure is a metric that can be given a movement profile and should provide a valid, sensitive, reliable and practical measure [2]. In this thesis, we only focus on the Log Dimensionless Jerk (LDLJ) and the Spectral Arc Length

(SAL) as they are the only existing smoothness measures in biokinematic motor control literature that are sensitive, valid and practical [2]. It should be noted, however, that only SAL is reliable against measurement noise [2].

Log Dimensionless Jerk.

One of the older, most frequently used smoothness measures that is valid, sensitive and practical is the Log Dimensionless Jerk (LDLJ) [2]. The LDLJ is defined as below:

$$DLJ = -\frac{(t_2 - t_1)^5}{v_{peak}^2} \int_{t_1}^{t_2} |\frac{d^2 v(t)}{dt^2}|^2 dt \tag{3.5}$$

$$LDLJ = -ln|DLJ| \tag{3.6}$$

where t_1 is the start time, t_2 is the end time, v_{peak} is the peak velocity and $v(t)$ is the velocity at time t. The LDLJ is often used to quantify smoothness and coordination in biokinematics tasks to analyze sensorimotor differences in stroke patients [1, 2]. However, Balasubramanian et al. have found the LDLJ to be relatively non-robust to sensor noise [2].

Adapted Log Dimensionless Jerk

To use the IOU as a speed profile in LDLJ, some modifications are required. First, a non-moving object would have a $v_{IOU}(t) = 0$, which could lead the DLJ term in the ln of equation (3.6) to be zero. To fix this, we modify the LDLJ as follows:

$$ALDLJ = -ln|1 + DLJ| \tag{3.7}$$

This allows for $DLJ = 0$ and it does not greatly affect the LDLJ calculation. We name this adaptation as the ALDLJ. It should be noted that when comparing two objects, the object with the higher ALDLJ is smoother. As an additional modification, we note that v_{peak} cannot be set per trial and must be set for all trials. Since we are using the temporal IOU as a measure of speed, there is a theoretical peak of 1 and so we set $v_{peak} = 1.0$. Finally, we need to find an appropriate window length N, to perform the ALDLJ calculation. Although the entire scene may be used, using a rolling window for the ALDLJ calculation allows for an online measure of network performance in terms of bounding box prediction smoothness. This will be illustrated in Subsection 3.4.3.

Spectral Arc Length.

Spectral Arc Length (SAL) is a novel smoothness metric that is more reliable and robust than the other previously used smoothness metrics [2]. The intuition behind this metric is that movements can be thought of as being composed of numerous interfering low-frequency components and high-frequency components [1]. This means that if we analyze the complexity of the shape of the speed profiles' Fourier Magnitude spectrum, we will be able to quantify smoothness. Balasubramanian et al. define the SAL as the negative arc length (length along a curve) of the magnitude and frequency-normalized Fourier Magnitude of the speed profile [1, 2]. The SAL has been used in biokinematics as well as in assessing surgical skills with regards to the surgeons' smoothness [20]. The SAL is defined as below:

$$\eta_{sal} \triangleq -\int_0^{\omega_c} \sqrt{(\frac{1}{\omega_c})^2 + (\frac{d\hat{V}(\omega)}{d\omega})^2} d\omega \tag{3.8}$$

$$\hat{V}(\omega) \triangleq \frac{V(\omega)}{V(0)} \tag{3.9}$$

where ω_c is the frequency threshold, ω is the frequency, $\hat{V}(\omega)$ is the normalized magnitude of the speed profile at frequency ω, $V(\omega)$ is the magnitude of the speed profile at frequency ω, and thus $V(0)$ is the magnitude of the speed profile at frequency 0.

Note that SAL requires two hyper-parameters: a frequency threshold and a magnitude threshold. Balasubramanian et al. use a frequency threshold of $\omega_c = 40\pi rad/s$ and a magnitude threshold of 0.05 [1]. These values were tuned for patient trials in biokinematics and so may not work well for our purposes. A magnitude threshold of 0.05 only allowed for 1 frequency bin in our use case, which would make the spectral analysis useless as we require a curve from which we could extract arc length. We devise an experiment to find better values and present its results in Section 3.4.4.

3.3.3 Adapted Spectral Arc Length

To use the SAL with temporal IOU as speed profiles, we must be able to take the discrete Fourier transform of $v_{IOU}(t)$ profiles. We employ a sliding Discrete Fourier Transform (DFT) [19] to allow for an online calculation of the metric. The sliding DFT requires a minimum of N samples (where N is the window length) before the DFT is valid [19], so we do not calculate the Spectral Arc Length for the first N samples. The sliding discrete Fourier transform has a few nice properties. It requires a constant number of operations to compute a successive DFT [19] and it only requires

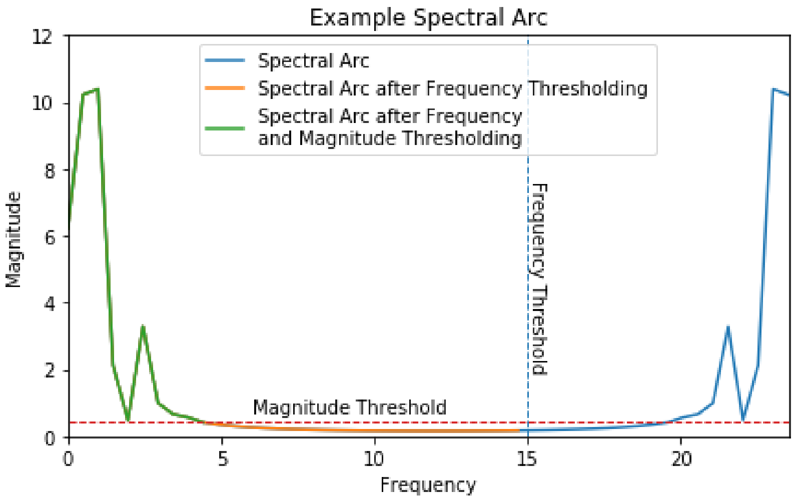


Figure 3.4: Example Spectral Arc after frequency thresholding and after magnitude thresholding. Note that the first threshold reached is the one used.

two real adds and one complex multiply new sample [19]. We refer to this adapted Spectral Arc Length as the ASAL.

We note that if we are to compare two objects, the object with the higher ASAL is smoother. Additionally, we do not normalize per trial, as this leads to an inability to compare intertrial results. To resolve this issue, we do not normalize by $V(0)$ in equation (3.9). Finally, as we are adopting the ASAL from another field, we perform some tests on the frequency and magnitude thresholds to find appropriate parameters by analyzing the effect the parameters have on the final ASAL. This is done in Subsection 3.4.4.

3.4 Evaluation of Smoothness Metrics

3.4.1 Data

To evaluate and analyze the ALDLJ and ASAL, we chose to use the Multi-Object Tracking (MOT) Dataset as it contains a variety of scenes with a large variety of object paths that are clearly labelled [32]. To map network detections with object paths, we use the IOU metric to find the closest match for each ground truth along the path and assign the predictions accordingly. This method of assigning ground truths to predictions is how networks train on object detection internally [28, 38, 39, 40]. Finally, as all evaluations are on a sliding window, we use a stride of 4 frames as this was empirically determined to be the minimum number of frames required for the jerk to be calculated.

In Subsection 3.4.2 we show that the temporal IOU (i.e. $v_{IOU}(t)$) is a suitable speed profile for a bounding box. In Subsection 3.4.3, we find the appropriate window length for LDLJ and SAL on bounding box smoothness calculations using a single object path to evaluate the hyperparameters. In Subsection 3.4.4, we find the best magnitude and frequency thresholds that allow the most information to be collected for calculating the SAL. In Subsection 3.4.5, we examine the intuition that the ground truth path is the smoothest. Finally, in Subsection 3.4.6, we analyze the performance of YOLOv3, Retinanet and ground truth object paths using the ALDLJ and ASAL.

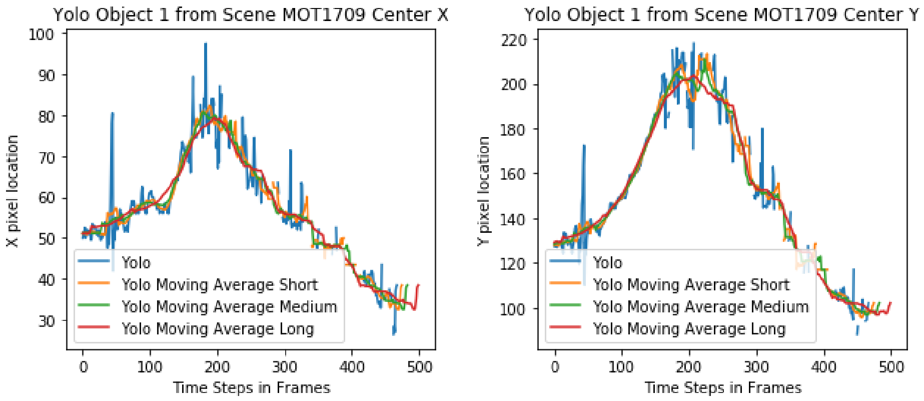
3.4.2 Validating IOU-based Speed Profile

Smoothness measures require a speed signal from which we can measure the smoothness of an object path. We propose using a temporal IOU to define this singular speed signal as explained in Section 3.3.1 and demonstrate its effectiveness by formulating an experiment using the object path predicted by YOLOv3 on MOT1709 matched against the ground truth of object with $ID = 1$. To validate the effectiveness of the temporal IOU, we generate 3 other object paths using different length moving averages. As a reminder, the moving average is a windowed average of the signal in a way that acts as a type of impulse filter.

We define a short, medium and long moving average as having window lengths of 8, 16 or 32 frames respectively. We define the moving average in equation (3.10). The x and y values of the centre point of the predicted bounding box of all of these paths are plotted in Fig. 3.5a and Fig. 3.5b respectively. We plot the temporal IOU of the bounding box paths in Fig.3.6a, and the IOU of the predicted bounding box paths against the ground truth are plotted in Fig. 3.6b. It is clear from these figures that the moving average of a path does smoothen the temporal IOU ($v_{IOU}(t)$). Additionally, from Fig. 3.6b, we see that the moving average of an object bounding box path can maintain or improve on the IOU against the ground truth. This demonstrates that sometimes, simply smoothing the path of an object can result in an object path that is closer to the ground truth.

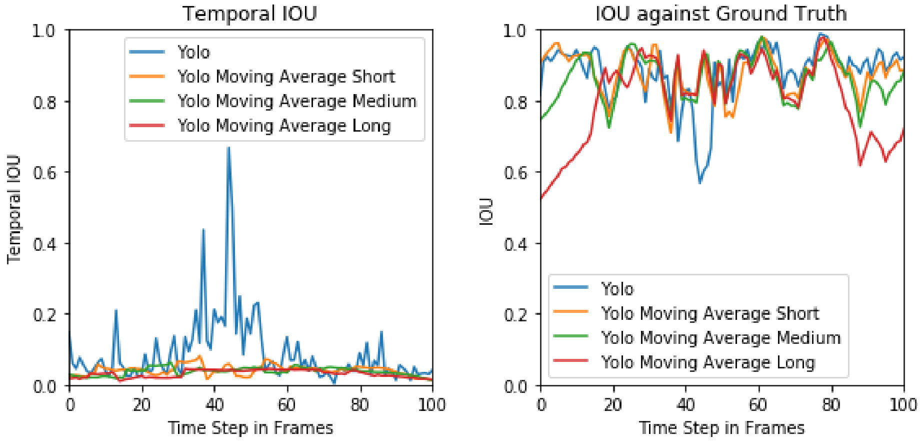
$$\overline{MA}_w = \frac{B_1 + B_2 + B_3 + ... + B_w}{w} \tag{3.10}$$

where $\overline{MA}_w$ is the moving average with window length w, B_i is the bounding box at timestep i, and w is the window length.



(a) (b)

Figure 3.5: Plot of YOLOv3's predicted box center coordinate x (left) and y (right) value, and their mobile average in Scene MOT1709 for object ID 1 over time.



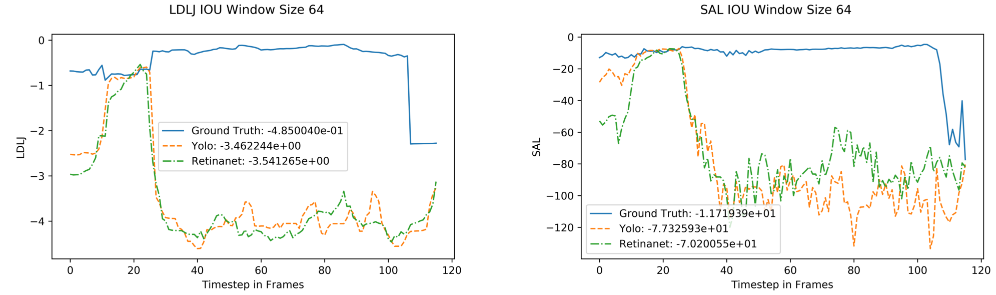
(a) (b)

Figure 3.6: Plots of result of the experiments with moving average on the effect of $v_{IOU}(t)$ (left) over timestep in frames. Plot of the result of the experiments with moving average on the IOU against ground truth bounding boxes (right) over timestep in frames.

3.4.3 Determining Window Length

To effectively use the LDLJ and SAL, we need to determine a suitable time window N that will encompass enough information about object paths. It should be noted that larger window lengths allow for more information of an object's motion characteristics and in both the case of SAL and LDLJ, To do this, we plot the LDLJ and SAL at the following range of window lengths in terms of frames: {8, 16, 32, 64, 96, 128}.

The plots for these experiments can be seen in Figs. A.1-A.2 in appendix A. We note that the decrease in smoothness of the last few collected points in the figures is due to ground truth moving in and out of frame. Based on these experiments we note that LDLJ is more sensitive to the window size than SAL. To choose a window size, we must balance local information with global information. A small window has much local information, but not enough global information about the movement profile of the bounding box. Similarly, a large window can often flatten out the local information in favour of global information. Considering this balance, the window length we choose is 64 frames as this window allows for the ground truth to have changes in smoothness (allowing intratrial comparisons). We plot the graph of the moving LDLJ and SAL with a window of 64 frames on object path ID 1 in MOT1709 in Figs. 3.7a-3.7b. The mean LDLJ and SAL values are shown in Tables 3.1-3.2.



(a) (b)

Figure 3.7: Plots for a window size of 64 on MOT17-02 object path 1 (Note higher is smoother). Results are shown for ground truth, YOLOv3 and Retinanet as a running plot.

Table 3.1: Mean LDLJ values at a variety of window lengths (higher is smoother) for MOT17-02 object path 1

Mean Log Dimensionless Jerk

Window Length	8 Frames	16 Frames	32 Frames	64 Frames	96 Frames	128 Frames
Ground Truth	$\boldsymbol{-1.63e^{-6}}$	$\boldsymbol{-2.20e^{-4}}$	$\boldsymbol{-1.83e^{-2}}$	$\boldsymbol{-4.85e^{-1}}$	**-1.82**	**-3.35**
YOLO	$-5.86e^{-5}$	$-6.88e^{-3}$	$-4.20e^{-1}$	-3.46	-6.01	-7.92
Retinanet	$-6.07e^{-5}$	$-7.16e^{-3}$	$-4.44e^{-1}$	-3.54	-6.10	-7.98

Table 3.2: Mean SAL values at a variety of window lengths (higher is smoother) for MOT17-02 object path 1

Mean Spectral Arc Length

Window Length	8 Frames	16 Frames	32 Frames	64 Frames	96 Frames	128 Frames
Ground Truth	**-1.28**	**-2.20**	**-5.97**	**-13.19**	**-20.04**	**-27.28**
YOLO	-2.08	-8.88	-28.86	-78.41	-151.01	-246.27
Retinanet	-2.93	-8.95	-26.88	-74.10	-139.63	-215.67

3.4.4 Determining Magnitude and Frequency Thresholds

SAL as defined in equations (3.8)-(3.9) requires two thresholding parameters [1]. Fig. 3.4 is an example of the SAL of a movement profile. These thresholds are useful in making SAL robust to noise [2]. We note, however, that the original parameters provided by Balasubramanian et al. [1] were not suitable for the object bounding box paths as they were initially found for patient sensory-motor trials.

We do an exhaustive grid search on frequency threshold and magnitude threshold at a variety of ranges. Balasubramanian et al. [1] use 5 as their frequency threshold which corresponds to their sampling frequency of 100Hz, we began with this value and incremented it by 5 up until 35 as our corresponding sampling frequency is 24Hz. Any frequency bin beyond the frequency threshold tested is ignored to calculate SAL. For the magnitude threshold, we begin with an exceptionally small value of $1e^{-5}$ and in log scale, we increase this threshold until we reach $1e^{-1}$. We begin with this small value to allow more information into the SAL calculation and try to find the effect

Table 3.3: Spectral Arc Length Threshold Experimentation on MOT17-02 object path 1 (higher is smoother) of the Ground Truth path. This is used to view the effect of parameters on the Spectral Arc Length in order to choose suitable parameters.

	Magnitude Threshold					
Frequency Threshold		$1e^{-5}$	$1e^{-4}$	$1e^{-3}$	$1e^{-2}$	$1e^{-1}$
	5	-7.72	-7.72	-7.72	-7.72	-7.21
	10	-8.67	-8.67	-8.67	-8.65	-8.59
	15	-9.59	-9.59	-9.59	-9.56	-8.59
	20	-10.60	-10.60	-10.60	-10.56	-9.22
	25	-13.19	-13.19	-13.19	-13.13	-11.55
	30	-13.19	-13.19	-13.19	-13.13	-11.55
	35	-13.19	-13.19	-13.19	-13.13	-11.55

that increasing this threshold may have. Once the spectral arc reaches either the frequency threshold or the magnitude threshold, all other frequency bins are ignored.

The results of our experimentation are presented in Table 3.3 and a few insights are readily apparent. Firstly, we note that changes in the magnitude threshold are minimal. We choose a magnitude threshold of $1e-5$ as a lower magnitude threshold is preferable in allowing more information to be included in the SAL calculation. Finally, the frequency threshold has a scaling effect on the Spectral Arc Length up until a threshold of 25. Frequency thresholds beyond 25 frequency bins do not affect the SAL calculation. Due to these findings, we choose to use a frequency threshold of 25 and a magnitude threshold of $1e^{-5}$ for object path analysis. The plot for this set of hyperparameters is presented in Fig. 3.8 and the remaining set of hyperparameters can be found in appendix B, Figs. B.1-B.4. These experiments show that a frequency threshold of 25 and a magnitude threshold of $1e^{-5}$ are useful for bounding box path smoothness evaluation.

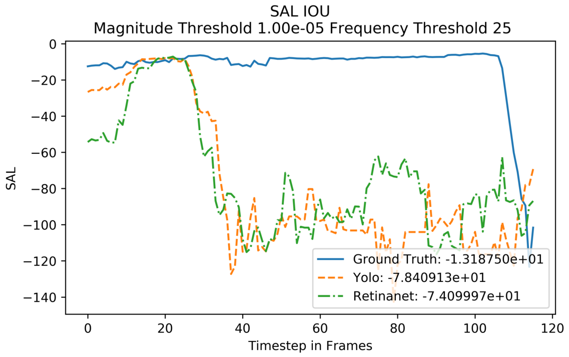


Figure 3.8: SAL plot for frequency threshold 25 and magnitude threshold $1e^{-5}$ of MOT17-02 object path 1.(Note higher is smoother)

3.4.5 Ground Truth Smoothness Analysis

It may seem intuitive to believe that ground truth is the smoothest path, however, we develop an experiment using the moving averages from Subsection 3.4.2 and the ground truth of that very same object path. We plot the LDLJ and the SAL of those object paths using the hyperparameters chosen after experiments from Subsections 3.4.3-3.4.4 in Fig. 3.9. This analysis of the moving average paths and the ground truth show that the ground truth is not the smoothest path and simply matching ground truth does not necessarily lead to a smooth bounding box motion characteristic.

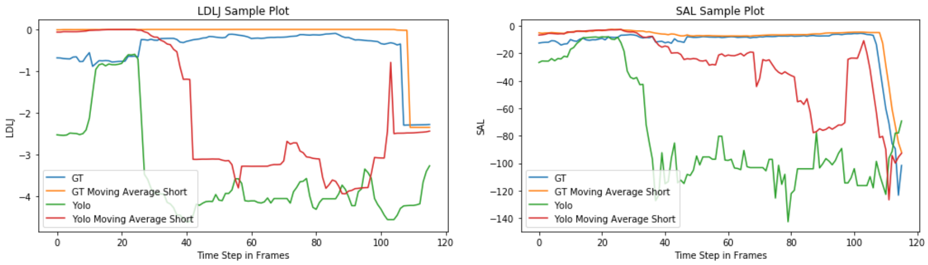


Figure 3.9: Left is the LDLJ plot of the YOLOv3 predicted path, moving average of the YOLOv3 predicted path and, the ground truth. Right is the SAL plot of the YOLO predicted path, moving average of the YOLOv3 predicted path, and the ground truth. GT is the ground truth.

3.4.6 Validating LDLJ and SAL using YOLO and Retinanet

Previous subsection experiments have been on object path 1 in MOT17-09, to make sure there is no bias in our experimentation of the LDLJ and the SAL, we report the LDLJ and SAL mean on all object ID's in all scenes in the Multi-Objective Tracking Dataset. We plot the box plot of the LDLJ and SAL values on all object IDs in all scenes in MOT in Figs. C.1-C.2 of appendix C. The histogram plots of the LDLJ and SAL means are all shown in appendix C Figs. C.3-C.4.

If these metrics are indicative of object path smoothness, we would expect that the ground truth would be the smoothest. In our experimentation in Subsection 3.4.6 with these metrics, this expectation holds (see table 3.4). Although LDLJ is known to be unreliable when affected by sensor noise [2], the object detection scenario has very little such noise. In Table 3.4 we see that the LDLJ claims Retinanet as being worse than YOLO for motion smoothness, and the SAL confirms this. Retinanet has many more object proposals than YOLO [28, 40] and thus may have more high-frequency noise in object paths for the long term and this may explain the reason for it's lower smoothness metrics.

Table 3.4: Mean LDLJ and mean SAL values for all networks as well as Ground Truth on all objects that are present in the ground truth, and predicted by YOLOv3 and Retinanet in all scenes of the Multi-Object Tracking Dataset. Note that for both LDLJ and SAL, higher is smoother.

Metrics	LDLJ	SAL
Ground Truth	$\mathbf{-0.209} \pm 0.18$	$\mathbf{-50.819} \pm 15.33$
YOLO	-0.652 ± 0.64	-57.848 ± 12.04
Retinanet	-0.742 ± 0.74	-65.687 ± 10.96

To show that the LDLJ and the SAL properly differentiate between the ground truth, YOLOv3, and Retinanet, we perform two one-way ANOVA tests on the mean SAL and the mean LDLJ of all object IDs that are present in the ground truth and predicted by YOLOv3 and Retinanet from the Multi-Object Tracking Dataset. To maintain comparability, if any object was not predicted by either YOLOv3 or Retinanet, it is not included in the one-way ANOVA tests. The results of these ANOVA tests can be seen in Table 3.5. With $p < 0.0001$ for both SAL and LDLJ, we decided to conduct a multi-comparison post-hoc test to determine which population means are significantly different from the others. As the population means were found to be normally distributed with a D'Agostino's K-squared Test, we conducted a Tukey's Honest Significant Difference (HSD) Test with $\alpha = 0.05$. The results of this test on the LDLJ mean can be seen in Table 3.6 and the results on the SAL means can be seen in Table 3.7. As the null hypothesis for the Tukey's HSD test is that all population means are the same, we note that both LDLJ and SAL can differentiate all population means. This supports that both the LDLJ and SAL can be used as reliable ways to determine the smoothness of object paths generated.

Table 3.5: Results of one-way ANOVA test on the mean SAL and mean LDLJ of all objects in the ground truth, and predicted by YOLOv3 and Retinanet in all scenes of the MOT Dataset.

Metric	F value	$P <$
SAL IOU	910.69	*0.0001*
LDLJ IOU	641.14	*0.0001*

Table 3.6: Multiple Comparison of LDLJ Means using Tukey HSD with α of 0.05. Reminder that H_0 is that all population means are the same.

Group 1	Group 2	Mean Diff	p adjusted	lower	upper	Reject H_0
Ground Truth	Retinanet	-0.53	0.001	-0.57	-0.50	**True**
Ground Truth	YOLOv3	-0.44	0.001	-0.48	-0.41	**True**
Retinanet	YOLOv3	-0.09	0.001	-0.05	0.13	**True**

Table 3.7: Multiple Comparison of SAL Means using Tukey HSD with α of 0.05. Reminder that H_0 is that all population means are the same.

Group 1	Group 2	Mean Diff	p adjusted	lower	upper	Reject H_0
Ground Truth	Retinanet	-14.87	0.001	-15.68	-14.05	**True**
Ground Truth	YOLOv3	-7.03	0.001	-7.84	-6.21	**True**
Retinanet	YOLOv3	7.84	0.001	7.02	8.66	**True**

3.5 Summary

In this section, to quantify bounding box path smoothness, we adapt two smoothness metrics from the field of biokinematics for use in object bounding box path analysis in object tracking by detection challenges. We show the process by which we adapt the smoothness metrics for bounding box path analysis and show that these metrics can quantify object path smoothness. Finally, we compare and analyze the results of using these metrics on a particular object in a particular scene in the Multi-Object Tracking dataset. As these metrics can quantify path smoothness of a particular object detection/tracking system, ALDLJ and ASAL are good for testing multi-object tracking systems for smoothness.

We provided implementation details for ALDLJ and ASAL on objects in video and we analyzed the window size for both metrics and found the best hyperparameters for SAL (Subsections 3.4.3-3.4.4).

In the next chapter, we investigate the differentiability of these metrics to use them for regularization in object tracking by detection systems such as recurrent video object detectors. This enables a system that not only detects objects but attempts to predict smooth object paths. Experimental results have found that animals have better object recognition with smoother input [53]. This suggests that learning from temporal input, instead of static frames can improve object recognition in these systems. Similarly, biasing the production of smooth predictions through smoothness regularization may improve the learning of object detection systems.

Chapter 4

Regularization with Smoothness Metrics for Improved Video Object Detection and Tracking

4.1 Introduction

In Chapter 3, we discuss a variety of motion smoothness metrics used in the field of biokinematics and adapt them for computer vision bounding box path usage. These adapted metrics are called Adapted Log Dimensionless Jerk (ALDLJ) and Adapted Spectral Arc Length (ASAL). As they have demonstrated their efficacy in quantifying motion smoothness amongst object bounding box proposal methods, we know that these metrics are useful in characterizing bounding box path smoothness.

Having the ability to separate population means based on motion smoothness metric calculations leads to the following important questions:

- Can ALDLJ and ASAL be adapted for use as loss functions?
- Can Video Object Detectors be made smoother through regularization of their outputs using these smoothness losses?
- What is the effect of regularization loss on the performance of a Video Object

Detector's Mean Average Precision?

First we need to convert the metrics from Chapter 3 into losses in Section 4.2. We detail the model used in this chapter in Section 4.3. Then we devise the methodology of the experiments in Section 4.4. Results of the methodology are provided in Section 4.5. A discussion of these results is presented in Section 4.6, and we summarize our findings in Section 4.7.

4.2 Converting Metrics to Loss

4.2.1 LDLJ Loss

Since the ALDLJ monotonically decreases as smoothness decreases, we modify the LDLJ to become a loss by formulating it as shown in the following equations:

$$\ell_{DLJ} = \frac{(t_2 - t_1)^5}{v_{peak}^2} \sum_{t=t_1}^{t_2} (\frac{d^2 v(t)}{dt^2})^2 dt \tag{4.1}$$

$$\ell_{ALDLJ} = ln(1 + \ell_{DLJ}) \tag{4.2}$$

where t_1 is the start time, t_2 is the end time, v_{peak} is the peak velocity and $v(t)$ is the velocity at time t. Note that as stated in Chapter 3, v_{peak} is kept constant and is set to the maximal possible IOU of 1. The change between ℓ_{ALDLJ} and ALDLJ is that the ℓ_{ALDLJ} is monotonically increasing because it has its sign flipped. An interesting property of using ℓ_{ALDLJ} as a loss is that the loss can easily be minimized by simply predicting either no object or by predicting non-moving objects. This is important because it may negatively impact object detection.

4.2.2 ASAL Loss

As the ASAL monotonically decreases as smoothness decreases, we modify the ASAL to become a loss by formulating it as follows:

$$\ell_{\eta_{ASAL}} \triangleq \sum_{\omega=0}^{\omega_c} \sqrt{(\frac{1}{\omega_c})^2 + (\frac{dV(\omega)}{d\omega})^2 d\omega} \tag{4.3}$$

where ω_c is the frequency threshold, ω is the frequency, and $V(\omega)$ is the magnitude of the speed profile at frequency ω. The main difference between the $\ell_{\eta_{ASAL}}$ and the ASAL is that the $\ell_{\eta_{ASAL}}$ is monotonically increasing because it has its sign flipped to make it the positive arc length of the Fourier of the speed profile. As noted in Chapter 3, we do not use the normalized Fourier spectrum for the ASAL calculation in order to maintain intertrial comparisons.

In calculating the ASAL, we use a sliding discrete Fourier transform as described in Chapter 3 Section 3.3. As the sliding discrete Fourier transform only needs two real additions and one complex multiply [19]. With N frames in a window, we require $2N$ real additions and N complex multiplications. This gradient can be approximated using automatic differentiation techniques.

4.3 Models

This experiment begins with a trained single image YOLOv3 network on the Multi-Object Tracking dataset. For all experiments, no training data is ever used to evaluate or test any network. This pretrained network performs single image object detection and has no temporal signal by which a temporal gradient could propagate. Since the smoothness regularization requires multiple consecutive frames, the gradient which it will propagate may be temporal. For example, it is possible for an object detected in frame 3 to affect the smoothness of the path of this object in frame 4. From Chapter 3 Sections 3.4.2-3.4.5, we know that local smoothness can be applied by using a moving average and that this can positively affect mAP. For this experiment, we propose using the Long Short Term Memory unit so that the network can learn long or short term weights to improve smoothness. To maintain spatial consistency and to ensure effective memory usage, these Long Short Term Memory units are embedded as part of the kernel for 2D convolutions.

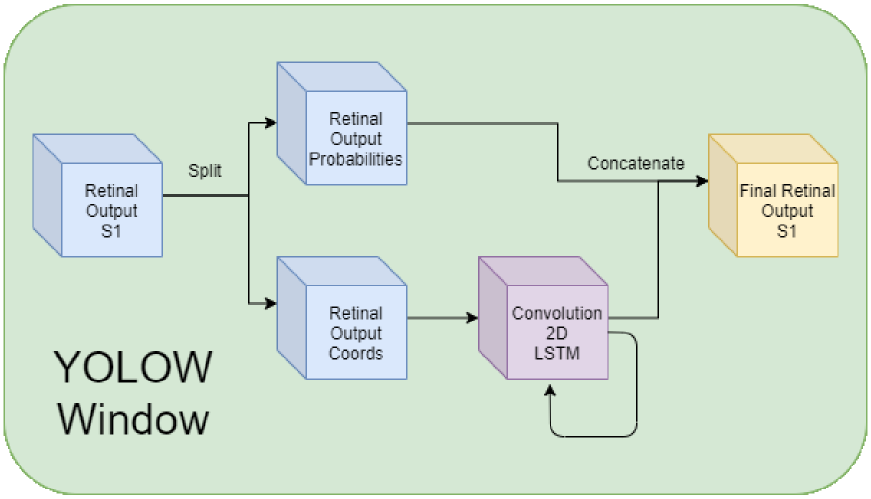


Figure 4.1: The architectural drawing of the window segment of the YOLOW network. Note that we split the probabilities and the coordinates from the final tensor output of YOLOv3.

To simplify this network, we only use one extra layer at the end of the original fully trained single image YOLOv3 network and we only operate on the part of the tensor which corresponds to the coordinates of bounding boxes (see Fig. 4.1), we refer to this network as You Only Look Once Windowed (YOLOW). We use a Convolutional 2D Long Short Term Memory (LSTM) layer as in the works of Shi et al. with a $[1, 1]$ kernel [45]. The principle being that these weights can be used to learn a kernel which will optimize for both bounding box proposals and for bounding box path smoothness. Note that since there are three retinal outputs for YOLOv3, this Convolutional 2D Long Short Term Memory layer is added to all three retinal outputs and weights are not shared between retinal outputs. The weights in the Convolutional 2D LSTM layer are initialized randomly and no activation function is used. The network architecture for YOLOW is shown in Fig. 4.2.

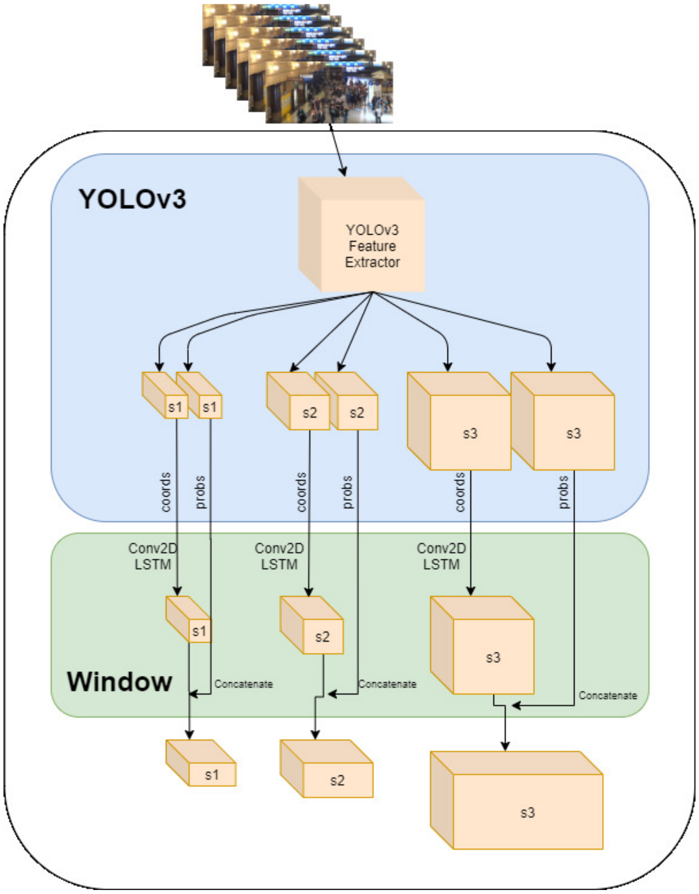


Figure 4.2: The network architecture for You Only Look Once Windowed. Note s_1, s_2, s_3 are the three retinal scales. Their dimensions are based off the retinal strides and the input image size. They are separated by whether they represent the coordinates (coords) or the probability (probs) of classes.

4.4 Methodology

Training is done using the Multi-Object Tracking dataset and all weights except for the Convolutional 2D LSTM layer are frozen. This means that the gradient will only modify the newly added layers. An example input for YOLOW can be seen in Fig. 4.3. YOLOv3 and other object detection networks require image augmentation for their inputs and YOLOW is no exception. The same augmentation strategies as those used in Redmon et al. [40] for YOLOv3 are used with the only modification being that the same augmentation is done for all images in a window. Using the same optimizer and learning rate from Redmon et al. for YOLOv3 [40], we train YOLOW in 7 different ways.

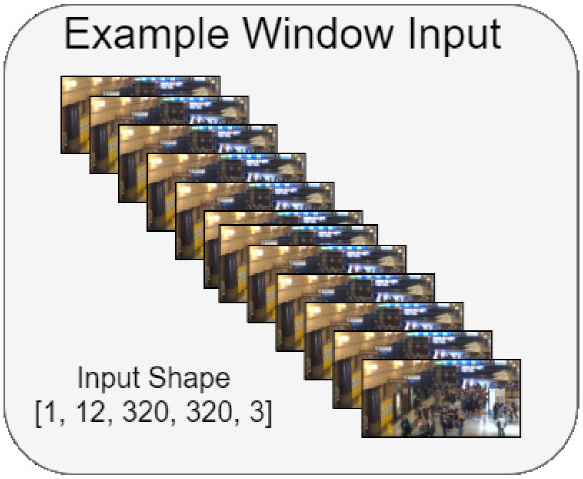


Figure 4.3: Example input for a training sample window.

These seven training regimens are used in this experiment to evaluate the effect that either ℓ_{ALDLJ} or ℓ_{ASAL} has on YOLOW. Training regimens using ALDLJ use the loss from equation (4.4) and training regimens using ASAL use the loss from equation (4.5).

$$\ell_{YOLOW} = \alpha \sum_{s=s_1}^{s_3} \ell_{yolov3}^{s} + \beta \ell_{ALDLJ}^{s} \tag{4.4}$$

where ℓ_{YOLOW} is the YOLOW loss, s is the retinal scale, s_1 is the smallest retinal scale, s_3 is the largest retinal scale, ℓ_{yolov3}^{s} is the YOLOv3 loss at retinal scale s, ℓ_{ALDLJ}^{s} is the ALDLJ regularization loss at retinal scale s, α is the learning rate and β is the relative learning rate for the smoothness loss.

$$\ell_{YOLOW} = \alpha \sum_{s=s_1}^{s_3} \ell_{yolov3}^{s} + \beta \ell_{\eta_{ASAL}}^{s} \tag{4.5}$$

where ℓ_{YOLOW} is the YOLOW loss, s is the retinal scale, s_1 is the smallest retinal scale, s_3 is the largest retinal scale, ℓ_{yolov3}^{s} is the YOLOv3 loss at retinal scale s, ℓ_{ASAL}^{s} is the ASAL regularization loss at retinal scale s, α is the learning rate and β is the relative learning rate for the smoothness loss.

The first training regimen does not use any smoothness regularization and is considered as the baseline network. The second, third and fourth training regimens use the loss from equation (4.4) with three different relative learning rate schedules. We refer to networks trained with these regimens as ALDLJ YOLOW. The fifth, sixth and seventh training regimens use the loss from equation (4.5) with the same three relative learning rate schedules. We refer to networks trained with these regimens as ASAL YOLOW.

The three relative learning rate schedules are used to evaluate the effect of the regularization loss on the metrics that have been discussed in Chapter 3 and mAP. The first relative learning rate schedule has a constant value of 1. The second relative learning rate schedule has a linearly increasing value from 0 to 1. The third relative learning rate schedule has a linearly decreasing learning rate schedule from 1 to 0.

An ANOVA test and a Tukey Honestly Significant Difference (HSD) test is performed on all the results for the ALDLJ metric, the ASAL metric and the mAP. This is done to view the effect that ℓ_{ALDLJ} and $\ell_{\eta_{ASAL}}$ have on the ALDLJ, ASAL and mAP metrics. As mAP is a single value, we divide the testing set into 10 sets of images for which the networks are assessed on each set individually. These 10 mAP values are used for the ANOVA and Tukey HSD tests.

4.5 Results

In this section, we present the results of the experiments described in Section 4.4. The seven different networks are trained on the MOT dataset training and evaluated using mAP, ALDLJ, and ASAL. Table 4.1 details the results of the mAP analysis of the network on the test sets. The network training regimen which produced the best mAP was the ASAL YOLOW with an increasing learning rate schedule. There is some effect on the mAP using these training regimens, however, based on the ANOVA tests of the mAP as shown in Table 4.2, we note that the increase in mAP is not statistically significant as the P-value is > 0.05.

Table 4.1: The mAP of all three training regimens, with all three relative learning rates, on the entirety of the test set. Reminder that β is the relative learning rate.

	YOLOW	ALDLJ YOLOW			ASAL YOLOW		
β	n/a	1.0	$[0., 1.]$	$[1., 0.]$	1.0	$[0., 1.]$	$[1., 0.]$
mAP	74.98	74.52	74.93	74.76	74.01	**75.38**	74.00

Table 4.2: The results of the ANOVA test on the mAP values of all three training regimens with all three relative learning rates on the test set.

mAP Analysis	
F Value	0.0083
P Value	1.0000

Table 4.3: The LDLJ and SAL values of all three training regimens with all three relative learning rates on the entirety of the test set (higher is better). β is the relative learning rate.

Regimen	β	ALDLJ Metric Value	ASAL Metric Value
YOLOW	n/a	-0.767 ± 0.715	-33.571 ± 22.646
ALDLJ YOLOW	1.0	-0.768 ± 0.711	-33.632 ± 22.653
	[0., 1.]	-0.769 ± 0.712	-33.409 ± 22.676
	[1., 0.]	-0.761 ± 0.705	$\mathbf{-33.236} \pm 22.611$
ASAL YOLOW	1.0	-0.767 ± 0.707	-33.680 ± 22.555
	[0., 1.]	$\mathbf{-0.760} \pm 0.703$	-33.644 ± 22.649
	[1., 0.]	-0.762 ± 0.708	-33.501 ± 22.687

The results from the ALDLJ and ASAL metric analysis are presented in Table 4.3. The ALDLJ metric analysis shows that the network trained with the ASAL increasing regimen produced the best results. The ANOVA tests for both these metrics are shown in Table 4.4 and in Table 4.5. Although it is frequent in the literature to look for 1% to 2% improvement, we decided to analyse our results using an ANOVA test to determine statistical significance. The ANOVA tests did not show that these training regimens have a statistically significant effect on the ALDLJ and ASAL metrics as their P-value is > 0.05.

Table 4.4: The results of the ANOVA test on the ALDLJ values of all three training regimens with all three relative learning rates on the test set.

ALDLJ Analysis	
F Value	0.0751
P Value	0.9984

Table 4.5: The results of the ANOVA test on the ASAL values of all three training regimens with all three relative learning rates on the test set.

ASAL Analysis	
F Value	0.1417
P Value	0.9906

4.6 Discussion

Although the results show a small impact on mAP, ALDLJ and ASAL using the smoothness loss, these results were not statistically significant. We note that the best mAP is achieved from an increasing relative learning rate schedule with an ASAL loss regularization. Additionally, we note that regularizing with ASAL loss leads to best results in ALDLJ metric, while regularizing with ALDLJ loss leads to best results in ASAL metric. This pattern was not expected and may be due to a few factors. Firstly, it is possible that since the ANOVA tests for the ALDLJ metric and the ASAL metric show that this performance increase is not statistically significant, this increase in performance could be training random chance. Alternatively, it could be that the effects of regularizing a network on smoothness affect the network in subtle ways such that the relative learning rates used here are far too small.

Secondly, in attempting to make the network architecture simple, it introduces a few limitations. For example, by using a kernel of $[1, 1]$, the information that the convolutional 2D LSTM can use is limited to the specific tensor grid it is looking at. This means that information from grids around the tensor grid we are convolving are not used in determining future states. For example, if an objects path involves traversing grids, it is impossible for information from those grids to be used in smoothing the path. Additionally, this network architecture is very heavy in memory usage and could only fit a window size of 12 frames. From Chapter 3, we know that a window length of around 64 frames would be better. Finally, no probability information is being used in the convolutional 2D LSTM, meaning that patterns that hold for certain classes, or information from previous object confidence scores are not being used. It should be noted however, that memory requirements become a limiting factor when learning over sequences of video frames and when using recurrent neural networks [13]. This is the main reason for the computational trade-offs made in these experiments.

4.7 Summary

In this chapter, we used the metrics from Chapter 3 to define new loss functions which are used to regularize recurrent object detection networks. We develop a methodology for training, validating and testing a recurrent version of YOLOv3 we call YOLOW. YOLOW is trained on three different training regimens at three different relative learning rate schedules. This gives a total of 6 different trained networks with one additional baseline network which has not been regularized. Using the test set from the Multi-Object Tracking dataset to analyze the 7 networks led to the finding that although training YOLOW with ASAL as regularization and an increasing relative learning rate led to the best mAP, these results were not statistically significant. Additional analysis led to an interesting finding that regularizing with ALDLJ led to the best ASAL and vice-versa. Again, these results were found not to be statistically significant, however, they raise interesting questions about the loss landscape of ℓ_{ALDLJ} and $\ell_{\eta_{ASAL}}$. We discuss some of these questions in Section 4.6 and possible reasons for the lack of statistical significance. In the future, work should be done on examining a variety of possible network architectures to explore the type of information needed for object smoothness to be optimized.

Chapter 5

Conclusions and Future Work

5.1 Conclusion

In this work, we explore the concept of motion smoothness and evaluate its usage as both a metric and a loss for bounding box proposal systems in neural network-based object detection. We propose two motion smoothness metrics and evaluate them with the Multi-Object Tracking (MOT) dataset.

We begin by conducting a literature study on neural networks, video object detection and tracking systems, and motion smoothness in Chapter 2. We also discuss the choice for motion smoothness equations that were used in this thesis and the reasoning for it. The motion smoothness metrics chosen for this thesis are the Log Dimensionless Jerk (LDLJ) and the Spectral Arc Length (SAL) from the field of biokinematics as, to our knowledge, this is new concept and has not been explored or applied to the paradigm of computer vision using deep learning networks.

We use two state-of-the-art object detection systems, YOLOv3 and Retinanet (Resnet50 feature extractor) to evaluate our proposed motion smoothness metrics. In Chapter 3, we adapt LDLJ and SAL and design specific experiments to determine the best hyperparameters for the motion smoothness metrics. Additionally, Chapter 3 presents the methodology and results of the evaluation of the metrics. The evaluation shows that YOLOv3 and Retinanet do not score the same on ALDLJ and ASAL. This leads to further evaluation of a simple way of increasing motion smoothness by using a moving average. The experiments show that although ground truth is the smoothest bounding box generation method, simply matching ground truth does not necessarily increase smoothness.

In Chapter 4, we attempt to use our proposed metrics to design a smooth object tracking network. We adapt the LDLJ and SAL as loss functions and describe the network we use for the experiments and the baseline training regimen. Additionally, we develop the methodology by which we test the effect that motion smoothness regularization has on object detection systems. These experiments show that although the motion smoothness regularization can lead to better mAP, LDLJ, and SAL, the improvements are not statistically significant. We discuss possible reasons for this and ways to increase the efficacy of the regularization losses in the experiments.

5.2 Future Work

There are a variety of different ways future work can proceed. More work can be done on the analysis of the regularization loss and the effects it has on object detection networks. In particular, how does minimizing LDLJ affect SAL and vice-versa? Finding this out can be done by using a variety of relative learning rates and analyzing the smoothness metrics at a few different points during training. Relating this information with the regularization loss would allow insight into the specific effects that the different smoothness metrics have on each other.

5.2.1 Additional Testing

More testing should be done with object tracking systems that do not do object detection in order to quantify these methods and their motion smoothness. Typical kernel-based methods, and object tracking networks should be a part of this testing. Perhaps training a subnetwork to perform object path smoothing while maintaining mAP when objects pass in between grids on the output tensor would allow for interesting dynamics between objects and the movement in the final tensor output. Finally, although only one of the metrics is jerk-based, it would be interesting to see if directly regularizing object speed would allow for smoother networks by increasing the "*viscosity*" of moving objects in the scene.

5.2.2 Loss Hyperparameter Searching

Possible future work should be done on finding the right relative learning rate schedule for both ℓ_{ALDLJ} and $\ell\eta_{ASAL}$ by thoroughly testing a large number of parameters. The learning rate schedules used may be a limiting factor for the network learning object path smoothness.

5.2.3 Regulatization

Although YOLOW begins with a pretrained YOLOv3 network, it does not perform as well as the single image version on its own. This is likely due to the network not learning a good mapping as it receives the exact tensor that the single image YOLOv3 produces. In the future, this can be mitigated by optimistically initializing YOLOWs LSTM convolutional 2D layer using an identity initialization. Alternatively, other network structures should be examined. Perhaps using the same convolutional 2D

LSTM but with a larger $[3, 3]$ kernel. In particular because it is possible for objects in a window to transition grids in a tensor, having information from the nearby grids may aid the performance of the network. Additionally, perhaps having connections between the retinal scales could be useful for this exact same reason. As objects come closer to the camera they become larger and change retinal scales, making it possible for information from other retinal scales to aid both object detection and object path smoothness. Moreover, although we hypothesized that long term memory may allow for better smoothing of object paths, is this really the case? Does including bounding box class information improve the smoothness of object paths?

These questions show that the field of object detection and motion smoothness is one with great potential for further scientific research and improvement.

Bibliography

[1] Sivakumar Balasubramanian, Alejandro Melendez-Calderon, and Etienne Burdet. A robust and sensitive metric for quantifying movement smoothness. *IEEE Transactions on Biomedical Engineering*, 59(8):2126–2136, 2012.

[2] Sivakumar Balasubramanian, Alejandro Melendez-Calderon, Agnes Roby-Brami, and Etienne Burdet. On the analysis of movement smoothness. *Journal of NeuroEngineering and Rehabilitation*, 12(1):1–11, 2015.

[3] Yoav Beck, Talia Herman, Marina Brozgol, Nir Giladi, Anat Mirelman, and Jeffrey M. Hausdorff. SPARC: A new approach to quantifying gait smoothness in patients with Parkinson's disease. *Journal of NeuroEngineering and Rehabilitation*, 15(1):1–9, 2018.

[4] Gedas Bertasius, Lorenzo Torresani, and Jianbo Shi. Object detection in video with spatiotemporal sampling networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11216 LNCS:342–357, 2018.

[5] Xingyu Chen, Junzhi Yu, and Zhengxing Wu. Temporally Identity-Aware SSD With Attentional LSTM. *IEEE Transactions on Cybernetics*, PP:1–13, 2019.

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: Object detection via region-based fully convolutional networks. *Advances in Neural Information Processing Systems*, pages 379–387, 2016.

[7] Ashish Doshi and Adrian G. Bors. Smoothing of optical flow using robustified diffusion kernels. *Image and Vision Computing*, 28(12):1575–1589, 2010.

[8] Dawei Du, Pengfei Zhu, Longyin Wen, Xiao Bian, Haibin Ling, and Qinghua Hu. VisDrone-DET2019 : The Vision Meets Drone Object Detection in Image Challenge Results. *International Conference on Computer Vision 2019*, 2019.

[9] Mark Everingham, Luc Van Gool, Christopher K I Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, jun 2010.

[10] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to Track and Track to Detect. In *2017 IEEE International Conference on Computer Vision (ICCV)*, volume 2017-Octob, pages 3057–3065. IEEE, oct 2017.

[11] Mohammed Gasmallah, Francois Rivest, and Farhana Zulkernine. Quantifying Path Smoothness in Video Object Tracking. *Submitted to ECCV2020*, 2020.

[12] Mohammed Gasmallah, Farhana Zulkernine, Francois Rivest, Parvin Mousavi, and Alireza Sedghi. Fully End-To-End Super-Resolved Bone Age Estimation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11489 LNAI, pages 498–504. 2019.

[13] Mohammed Hamada Gasmallah and Farhana Zulkernine. Video Predictive Object Detector. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 365–371. IEEE, nov 2018.

[14] Ross Girshick. Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:1440–1448, 2015.

[15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587. IEEE, jun 2014.

[16] David Held, Sebastian Thrun, and Silvio Savarese. *Learning to track at 100 FPS with deep regression networks*. PhD thesis, Stanford University, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[18] Jonathan Huang, Alireza Fathi, Vivek Rathod, Ian Fischer, Chen Sun, Zbigniew Wojna, Kevin Murphy, Menglong Zhu, Yang Song, Anoop Korattikara, and Sergio Guadarrama. Speed/Accuracy Trade-offs for modern convolutional object detectors. *Proceedings of the IEEE International Conference on Computer Vision*, 84(3-4):209–230, 2017.

[19] Eric Jacobsen and Richard Lyons. The sliding DFT. *IEEE Signal Processing Magazine*, 20(2):74–80, 2003.

[20] William H. Jantscher. *Using Real-Time Smoothness Metrics to Deliver Haptic Performance Cues for A Dexterous Task*. PhD thesis, Rice University, 2018.

[21] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. Technical report, 2018.

[22] Laura Leal-Taixé. *Multiple object tracking with context awareness*. PhD thesis, Gottfried Wilhelm Leibniz Universitat Hannover, 2014.

[23] Laura Leal-Taixé, Anton Milan, Konrad Schindler, Daniel Cremers, Ian Reid, and Stefan Roth. Tracking the Trackers: An Analysis of the State of the Art in Multiple Object Tracking. (March), 2017.

[24] Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus Robert Müller. Efficient BackProp. In *Lecture Notes in Computer Science*, volume 1524, pages 9–50. Springer Berlin Heidelberg, 1998.

[25] Yann Lecun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object Recognition with Gradient-Based Learning. *Shape, Contour and Grouping in Computer Vision*, 1681:319–345, 1999.

[26] Xi Li, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel. A Survey of Appearance Models in Visual Object Tracking. *ACM Trans. Intell. Syst. Technol.*, 4(4):58:1–58:48, 2013.

[27] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2017-Janua, pages 936–944. IEEE, jul 2017.

[28] Tsung Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal Loss for Dense Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-Octob, pages 2999–3007, 2017.

[29] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS(PART 5):740–755, 2014.

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9905 LNCS, pages 21–37. 2016.

[31] Ala Mhalla, Thierry Chateau, and Najoua Essoukri Ben Amara. Spatio-temporal object detection by deep learning: Video-interlacing to improve multi-object tracking. *Image and Vision Computing*, 88:120–131, 2019.

[32] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A Benchmark for Multi-Object Tracking. pages 1–12, mar 2016.

[33] Guanghan Ning, Zhi Zhang, Chen Huang, Xiaobo Ren, Haohong Wang, Canhui Cai, and Zhihai He. Spatially supervised recurrent convolutional neural networks for visual object tracking. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4. IEEE, may 2017.

[34] Zailiang Pan and Chong Wah Ngo. Moving-object detection, association, and selection in home videos. *IEEE Transactions on Multimedia*, 9(2):268–279, 2007.

[35] Dennis Park, C. Lawrence Zitnick, Deva Ramanan, and Piotr Dollar. Exploring weak stabilization for motion feature extraction. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1(c):2882–2889, 2013.

[36] Ingmar Posner and Peter Ondruska. *Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks.* PhD thesis, University of Oxford, 2016.

[37] Wang Qicong and Liu Jilin. Visual tracking using the kernel based particle filter and color distribution. *Proceedings of 2005 International Conference on Neural Networks and Brain Proceedings, ICNNB'05*, 3:1730–1733, 2005.

[38] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788. IEEE, jun 2016.

[39] Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 53, pages 6517–6525. IEEE, jul 2017.

[40] Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement. Technical report, University of Washington, Washington, D.C., 2018.

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[42] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[43] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, oct 1986.

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[45] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. pages 1–12, 2015.

[46] Bi Song, Ahmed T. Kamal, Cristian Soto, Chong Ding, Jay A. Farrell, and

Amit K. Roy-Chowdhury. Tracking and activity recognition through consensus in distributed camera networks. *IEEE Transactions on Image Processing*, 19(10):2564–2579, 2010.

[47] Elizabeth S. Spelke, Roberta Kestenbaum, Daniel J. Simons, and Debra Wein. Spatiotemporal continuity, smoothness of motion and object identity in infancy. *British Journal of Developmental Psychology*, 13(2):113–142, 1995.

[48] Rainer Stiefelhagen, Keni Bernardin, Rachel Bowers, R. Travis Rose, Martial Michel, and John Garofolo. The CLEAR 2007 Evaluation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 4625 LNCS, pages 3–34. 2008.

[49] Mingxing Tan, Ruoming Pang, and Quoc V. Le. EfficientDet: Scalable and Efficient Object Detection. Technical report, Google Brain Team, nov 2019.

[50] Yonglong Tian, Ping Luo, Xiaogang Wang, and Xiaoou Tang. *Deep learning strong parts for pedestrian detection.* PhD thesis, The Chinese University of Hong Kong, 2015.

[51] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning Video Object Segmentation with Visual Memory. *Iccv*, pages 4481–4490, 2017.

[52] Alex Wojaczek, Regina-Veronicka Kalaydina, Mohammed Gasmallah, Farhana Zulkernine, and Myron R. Szewczuk. Computer Vision for Detecting and Measuring Multicellular Tumor Spheroids of Prostate Cancer. *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019.

[53] Justin N. Wood. A smoothness constraint on the development of object recognition. *Cognition*, 153:140–145, aug 2016.

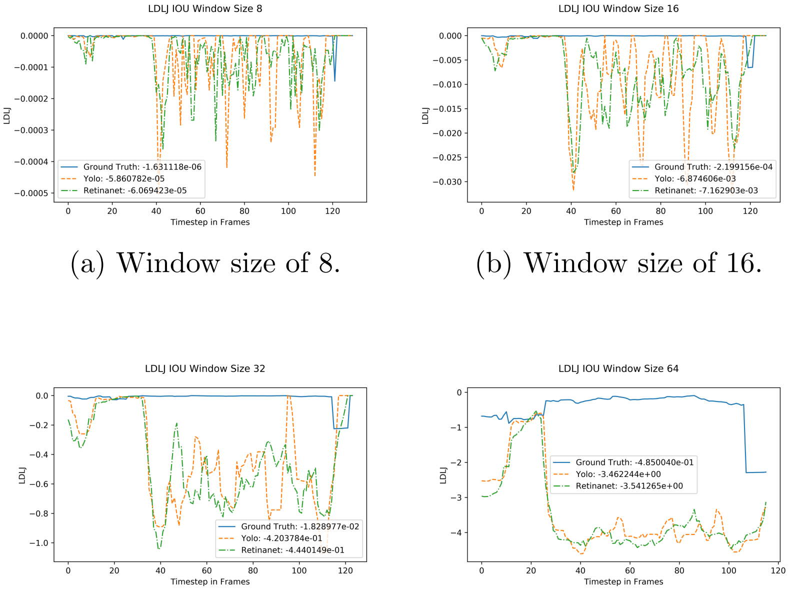
[54] Da Zhang, Hamid Maei, Xin Wang, and Yuan-Fang Wang. Deep Reinforcement Learning for Visual Object Tracking in Videos. Technical report, University of California at Santa Barbara, Santa Barbara, 2017.

[55] Pengyi Zhang, Yunxin Zhong, and Xiaoqiong Li. SlimYOLOv3: Narrower, faster and better for real-time UAV applications. *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, pages 37–45, 2019.

Appendix A

Determining Window Length Figures

Below are the figures for all the window size experiments on MOT1709 Object ID 1 for ALDLJ and ASAL from Chapter 3.

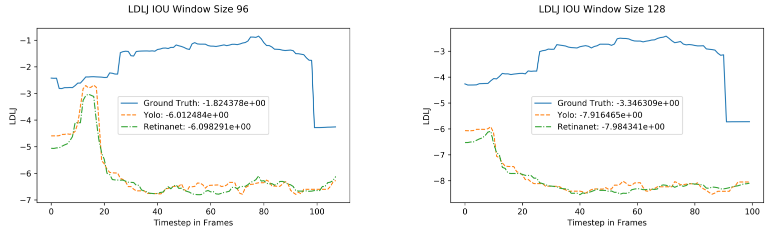


(a) Window size of 8.

(b) Window size of 16.

(c) Window size of 32

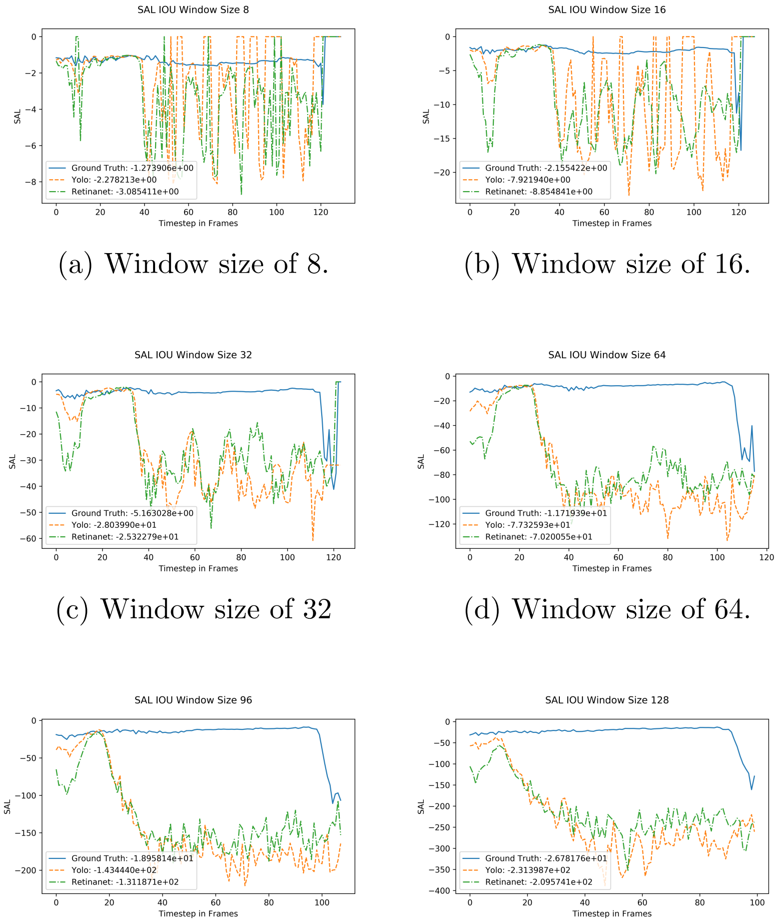
(d) Window size of 64.



(e) Window size of 96.

(f) Window size of 128.

Figure A.1: Window length experiment for ALDLJ against timestep in frames. (Note that a higher value is smoother).



(a) Window size of 8.

(b) Window size of 16.

(c) Window size of 32

(d) Window size of 64.

(e) Window size of 96.

(f) Window size of 128.

Figure A.2: Window length experiment for ASAL against timestep in frames. (Note that a higher value is smoother).

Appendix B

Determining Amplitude and Frequency Thresholds

Below are the plots for the amplitude and frequency thresholds experiment from Chapter 3. We plot only at the amplitude threshold of $1e^{-5}$ and plot all of the frequency thresholds.

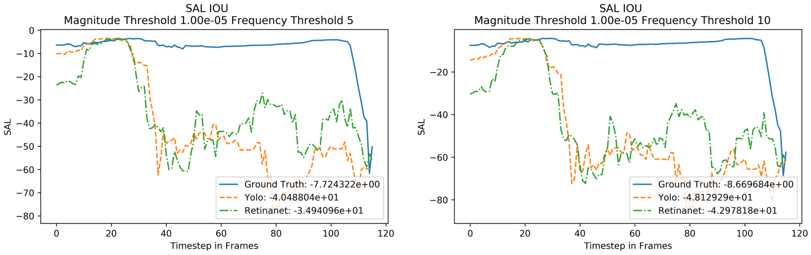


Figure B.1: Left is amplitude ($1e^{-5}$) and frequency(5) Threshold experiment for ASAL. Right is amplitude ($1e^{-5}$) and frequency (10) Threshold experiment for ASAL (Note that higher is smoother). All figures are against timestep in frames.

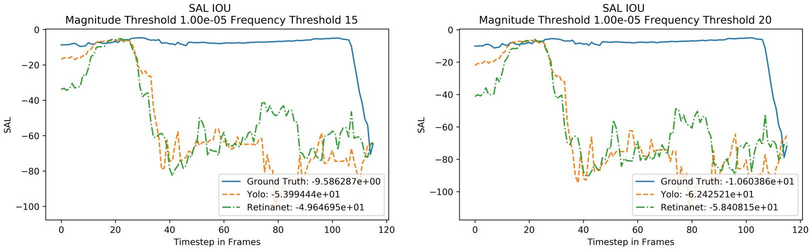


Figure B.2: Left is amplitude $(1e^{-5})$ and frequency(15) Threshold experiment for ASAL. Right is amplitude $(1e^{-5})$ and frequency (20) Threshold experiment for ASAL (Note that higher is smoother). All figures are against timestep in frames.

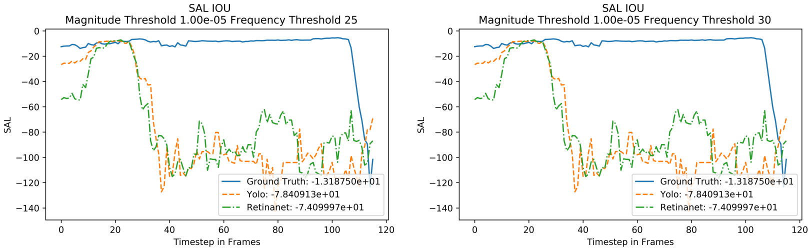


Figure B.3: On the left is amplitude $(1e^{-5})$ and frequency (25) threshold experiment for ASAL. On the right is amplitude $(1e^{-5}$ and frequency (30) threshold experiment for ASAL (Note that higher is smoother). All figures are against timestep in frames.

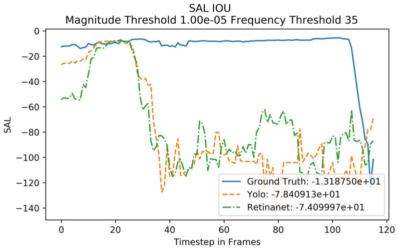


Figure B.4: Figure is amplitude $(1e^{-5})$ and frequency (35) threshold experiment for ASAL (Note that higher is smoother). All figures are against timestep in frames.

Appendix C

Validating ALDLJ and ASAL using YOLO and Retinanet

Below we plot the box plots of the mean ALDLJ and ASAL of the MOT dataset using the ground truth as method 1, YOLOv3 as method 2 and Retinanet as method 3 from Chapter 3. All thresholds and window sizes used are those detailed in the Subsections 3.4.3-3.4.4.

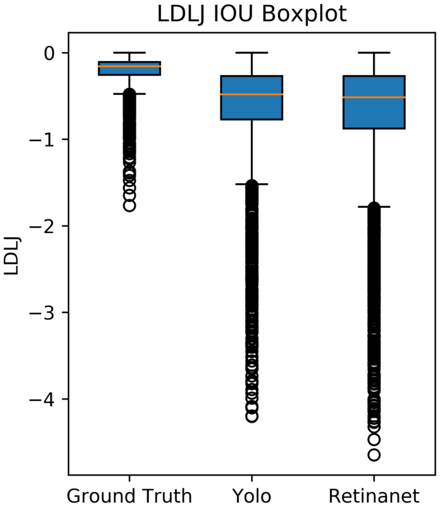


Figure C.1: Box plot of the ALDLJ values on entire MOT dataset.

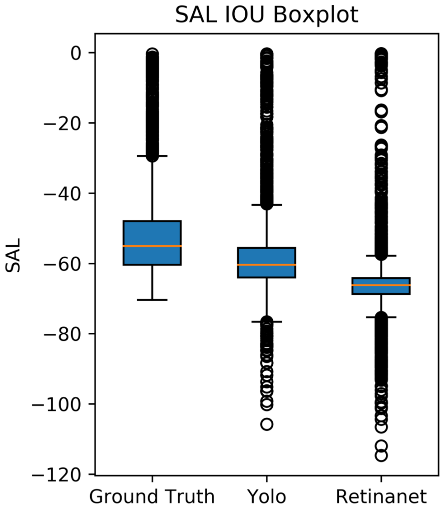


Figure C.2: Box plot of the ASAL values on entire MOT dataset.

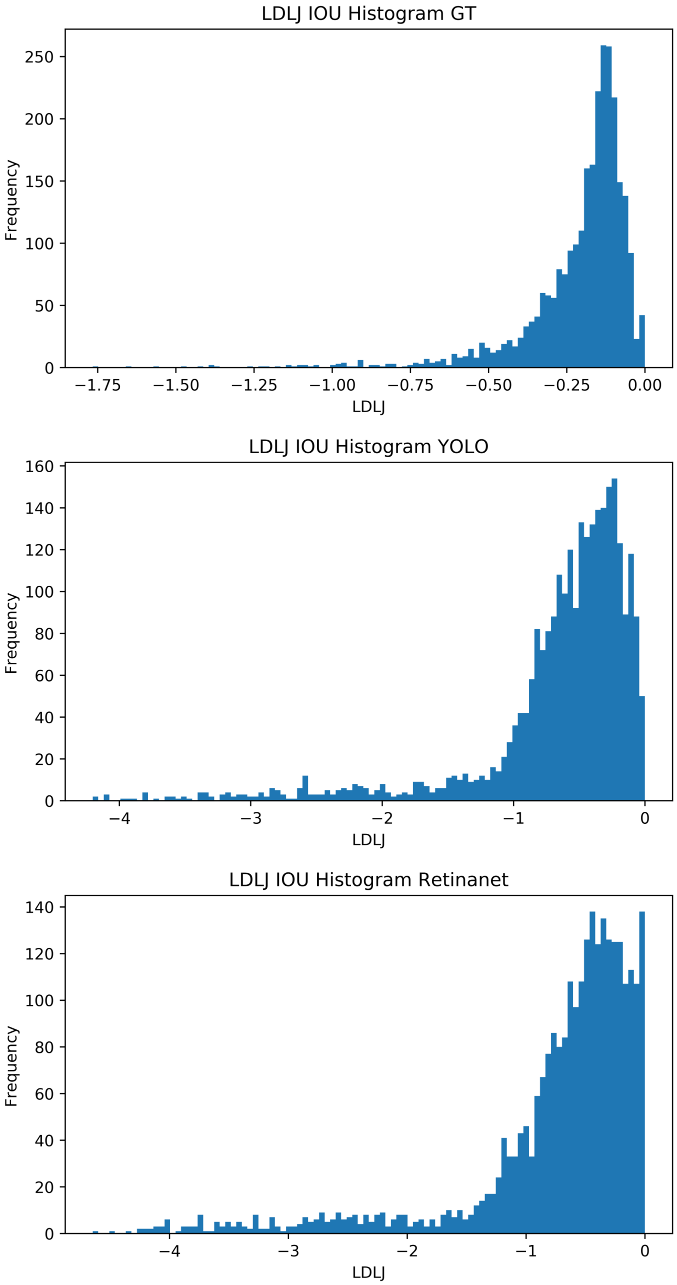


Figure C.3: Histogram plot of the ALDLJ mean values for Ground Truth (GT), YOLOv3 and Retinanet.

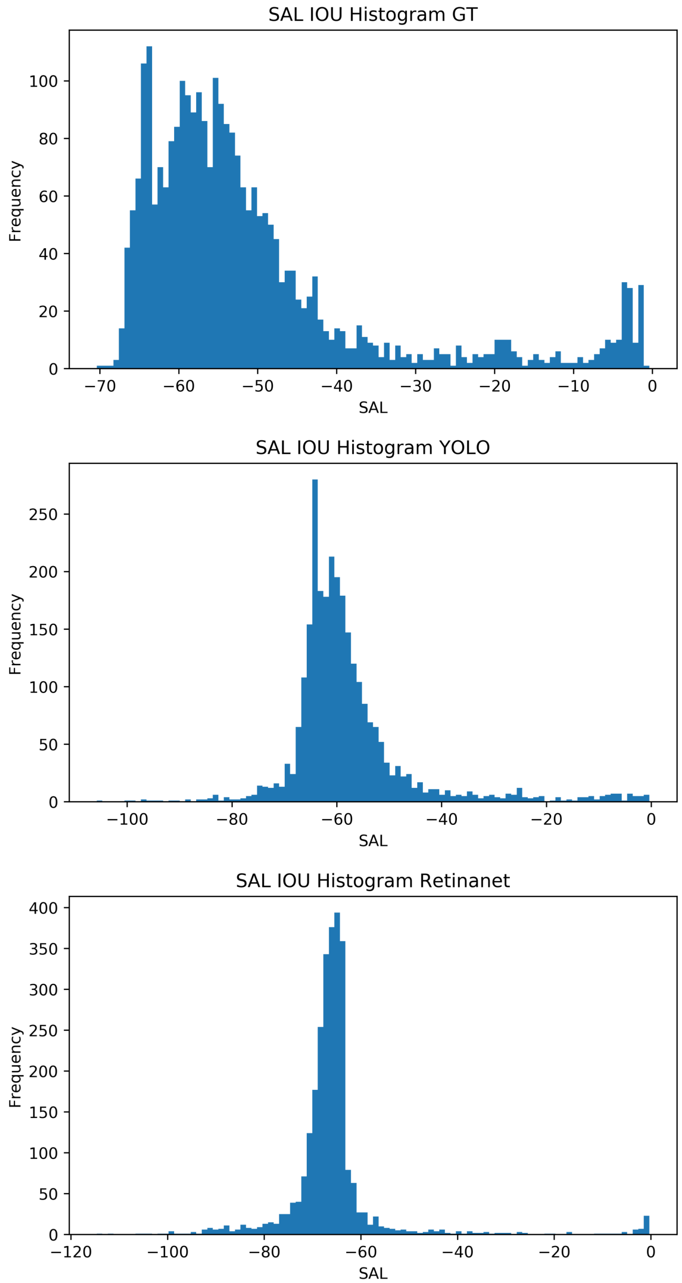


Figure C.4: Histogram plot of the ASAL mean values for Ground Truth (GT), YOLOv3 and Retinanet.