POLITECNICO DI TORINO

Master's Degree Course in Communications and Computer Networks Engineering

Master's Degree Thesis

DEEP LEARNING MODEL FOR 2D TRACKING AND 3D POSE TRACKING OF FOOTBALL PLAYERS

Supervisor:

Prof. Andrea Giuseppe Bottino

Candidate:

Seyedamirreza Hesamian

July 2020

ACKNOWLEDGEMENTS

I would like to sincerely express my gratitude to my advisor Dr. ANDREA GIUSEPPE BOTTINO, for his persistent and valuable feedback. His guidance has always made me ask the right questions about each aspect of my research and steered me into the right direction whenever required.

CONTENTS

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF TABLES

1 ABSTRACT

2D and 3D multiple object tracking, is an open problem inside the computer vision community with multiple applications in the industry such as in the autonomous vehicles or in the sport field. Many works have been conducted in the past to solve and improve this task, especially for person tracking due to its greater interest.

Recently, the deep learning techniques have been able to beat the state-of-the-art in tasks such as image classification, object detection or 3D pose estimation. Thus, this work has made use of deep learning methods to build a 2D and 3D tracking applications. These techniques are combined with a tracking by detection scheme to perform the tracking and achieve a good result. Contribution of work proposed in this thesis would be two-fold. First, it is implemented multi-person object tracker in 2D which specialized for sport like soccer. Second, a 3D multi person tracker is designed, which inputs single RGB image and outputs the 3D poses with IDs.

2 INTRODUCTION

Computer vision seeks to build automated systems capable do the tasks which the human visual system can do, in some cases perform better than it and generally the human visual system is the motivation for the designers of the computer vision system (Huang, 1997).In specific, computer vision aims to take out useful data from images. This has demonstrated a notably difficult task; For the last four decades many brilliants and knowledgeable people get involved in it, even though we still could not build an unlimited “seeing machine”(Prince, 2012). Recently computer vision faces rapid progresses. Deep learning has helped to the massive progress of the computer vision field, giving the possibility to introduce a huge number of applications using computer vision techniques. Modern day computer vision depends heavily on deep learning techniques. From crucial tasks like object recognition, detection and tracking to high level semantic dilemmas like traffic scene understanding, the community has witnessed substantial performance boost with these algorithms. Latest GPU hardware provide enormous computational capabilities, which enable rapid prototyping and deployment of new ideas. Deep Learning methods have proven themselves to generalize much better than many hand-engineered efforts, but only as good as generalizability of the training data itself.

2.1 MULTIPLE OBJECT TRACKING

One significant area of computer vision is the Multiple Object Tracking or MOT which is interesting because of its potential in both the academic and commercial spheres. Moreover, the trajectories prediction of multiple targets in video sequences is one of the principles aims of MOT, in order to improve applications such as smart video analysis and autonomous driving (Wang et al., 2019). The real-world applications of the multi-object tracking are numerous including human-computer interaction, autonomous vehicles, robotics, video indexing, surveillance or security, sports, among others. The computer vision community have been making big efforts in the past few decades to solve the MOT problem, but the task is still open for improvement.

Video based information about team sports can be used in different ways to increase the performance of teams and athletes. Determining positions during interesting game situations, ratio of strain and relax time or physically exhausting actions like sprints or jumps would require an annotation of nearly every frame of the video sequence (Figure 1). For that reason, computer vision and in particular tracking is of increasing importance for digital game analysis. Many different games e.g. soccer, hockey and other sports have used tracking in the past. In these kinds of tasks, the algorithms have to deal with complex occlusion situations and difficult object matching (Mauthner & Bischof, 2007).

One of the most studied tracking areas is the pedestrian tracking, mainly because this particular kind of object can be seen in a large number of applications with commercial potential. As some studies indicate (Voigtlaender et al., 2019), about the 70% of the current research done in MOT is dedicated to pedestrians. The difficulty of MOT lies in various challenging situations that can occur such as variation of the illumination, variation of scale, target deformation or fast motion. Most of these challenges are common to Single Object Tracking (SOT) but MOT also needs to solve two main tasks: determining the number of objects and maintaining its identities over the time.

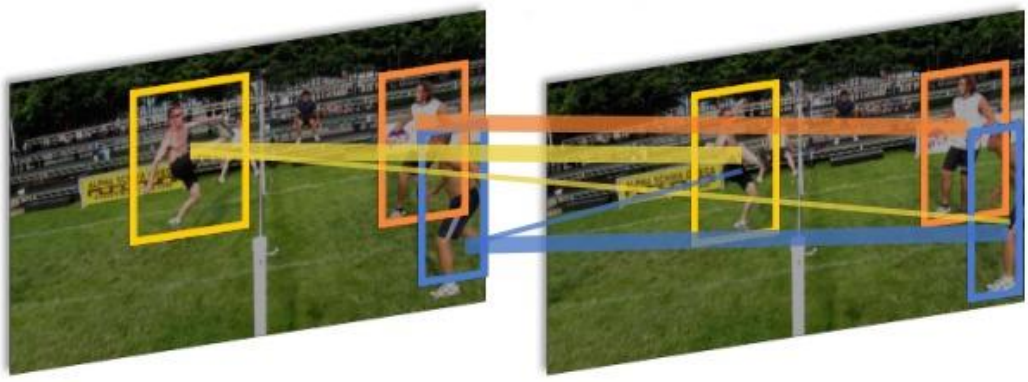


Figure 1: Tracking in sport (Girdhar et al., 2018).

2.2 MULTI-PERSON 3D POSE ESTIMATION AND TRACKING

Human pose estimation draws a considerable attention in computer vision community because of an array of applications in augmented reality, activity recognition, trajectory prediction, and marker-less tele-operation to name a few. Depending the application requirement for pose estimation, three categories can be modeled: 2D pose estimation (in pixel coordinates) from static monocular images, 3D pose estimation (in world/camera coordinates) from depth/range images, and 3D pose estimation from monocular/2D images directly (Figure 2) (Moon et al., 2019). Depth sensors have the ability to provide rich information about human postures in indoor settings, and are used in multiple setups, like gaming in Microsoft's Xbox with Kinect, etc. However, the human body flexibility along with multiple degrees of freedom leading to self-occlusion has kept humans pose estimation far from be solved (Bridgeman et al., 2019).

The estimation of 3D poses using videos is a well-explored problem. A quite large number of researches are focus on 3D poses calculation from multi view videos as well as monocular ones. However, few methods have been designed specifically for sport matters (Bridgeman et al., 2019).

Datasets with sport information are a challenge for the algorithms used in computer vision, due to fast motion tasks and contact between players, similarities in the appearance of the players, huge occlusion, low resolution and wide baseline cameras, moving and poor calibration. Nevertheless, the estimation of the players 3D poses performing different sports has potential applications such as analysis of the performance, motion capture, and others (Bridgeman et al., 2019).

The challenges of tracking human poses are many. Some of them are occlusion, pose changes and multiple overlapping instances. Imaging a tracker that works ideally, it would need to predict in an accurate way every instance of human poses at each time step, considering the appearance and the pose changes over time. Thus, the state of art in pose predictions should be followed closely, combining the usage of tools to be able to join time information in a specific instance level in a successful way (Girdhar et al., 2018).

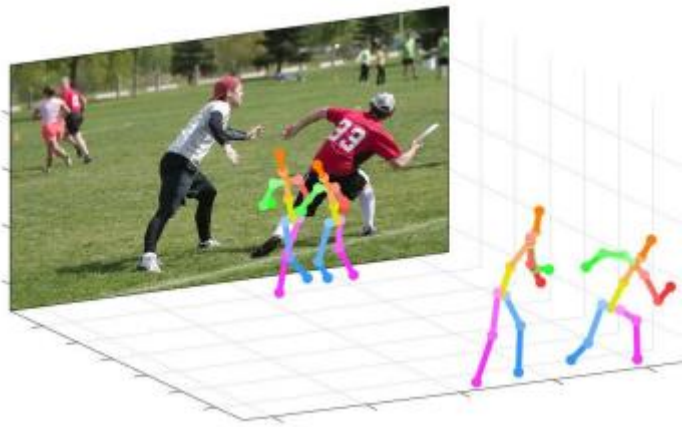


Figure 2: 3D Pose estimation of multi-person (Moon et al., 2019).

2.3 OBJECTIVES

Objectives of this master thesis are to build a multi-object tracking application which makes use of two techniques: object detection and 2D-tracking and separately add multi human 3D pose estimation algorithm to the current pipeline and finally track poses. In this work will be studied how to use these techniques to build a robust multi-object tracker and as well as built a 3D multi-person poses tracker. This task can be divided into different sub-objectives:

1. Development of an object detector using deep learning.
2. Development of a deep tracking module.
3. Tracking of an object in a single pipeline and combination of object detection
4. Development of a 3D multi-person pose tracker.
5. Analyze and Discuss the results.

3 STATE OF THE ART

The state of the art and the background related to the topic of the thesis will cover in this chapter. First, the object tracking (including algorithms), datasets and metrics used for a good development of a system for multi object tracking will be introduced. Second, the object detection will be discussed with goals of understanding what is happening under the hood. Third, 3D multi-person pose estimation will be explained following the same scheme. Other necessary tools and interesting subjects from the literature will be also briefly commented.

3.1 OBJECT TRACKING ALGORITHMS

The main aim for the object tracking is to estimate the target over the time in a frame sequence (images). This state can be defined by different features such as shape, appearance, position or speed. It is a difficult field since one or more difficulties must be solved by the algorithm. Among them the management of variations in lighting and in the point of view of the object that can lead to changes in its appearance. Likewise, the occlusions that occur when objects are mixed with other elements of the scene or the quality of the image itself may be a problem. To confront these problems the following paradigms have been followed (Smeulders et al., 2014):

- **Tracking using matching:** these algorithms match the model representation of the object created from the previous frame and the possible player in the next frame. These methods rely on the correct representation of the match and the similarity measurement used to perform the matching. The most outstanding methods are Normalized Cross-Correlation (Briechle & Hanebeck, 2001), Lucas-Kanade Tracker (Baker & Matthews, 2004), Kalman Appearance Tracker (Intelligence & Intelligence, 2004) and Mean Shift Tracking (Comaniciu & Ramesh, 2000). Most of them use the intensity values in the images to build the algorithm, for example, Lucas-Kanade performs spatiotemporal derivatives on these values.
- **Tracking by detection:** in order to differentiate the background with the object, a model was built (Harris & Stephens, 1988). Once you have one detection it is associated with the previous detections. Currently, the community is turning to neural networks to compute detections.

- **Tracking, learning and detection**: it is an extension of the previous group that includes a mechanism to update the model that is learned during execution. For example, you can use the results of an optical flow tracker for this update. This makes sure that the algorithm does not get altered with the object variations (Kalal, 2010).

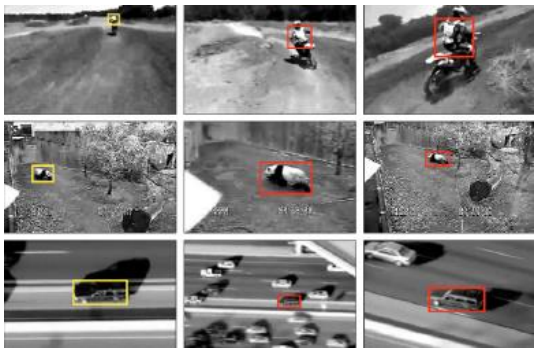


Figure 3: P-N learning mechanism (Kalal, 2010).

Prior to the modern techniques to be discussed here there are more "classic ways" of tracking objects that can be useful in problems that require real time, for example. One of the most well-known is feature tracking. This technique uses characteristic points that can be found in images and that allow to estimate the movement. These points must meet some requirements to be able to be characteristic of the image such as repeatability (the characteristic can be found in the images even if they have undergone some transformation), compatibility (each characteristic must be descriptive and easy to find) or efficiency (the representation of the information characteristic of the image must be done with as few characteristics as possible). The characteristic points most commonly used are corners. They are characterized by gradients with higher values in them in two or more directions. These techniques can be seen in Harris (Harris & Stephens, 1988) and Shi-Tomasi corner detectors (Shi & Tomasi, 1993).

There are tracking systems that take advantage of the feature tracking speed and the neural networks accuracy to create a "hybrid tracking". In this type of tracking the detections are done each N frames using some type of neural network and the intermediate tracking is done through feature tracking (Held et al., 2016).

With the arrival of neural networks this way of grouping the tracking methods changes to adapt to them:

- **Tracking by detection:** these are intended to follow a specific type of object (model-based) and to obtain a specific classifier. In practice, the detections are obtained with neural networks and they are linked in tracking using temporal information. They are limited to a single class of objects (Held et al., 2016).
- **Tracking, learning and detection:** they are characterized by being fully trained online. A typical tracker example of this group samples zones closes to the object and considers them foreground, the same happens with the distant zones that would be assigned to the background. With this a classifier can be built that differentiates them and estimates what is the new position of the object in the following frame (Babenko et al., 2009) .It has been tried to introduce neural networks in environments with online training but due to the slowness of the networks when training the results are slow in practice (Held et al., 2016).
- **Siamese-based tracking:** multiple patch candidates from the new frame are received and the one that has the higher matching score comparing to the previous frame is chosen as the best candidate, that is, the most similar according to the matching function. In the figure below, one of the last Siamese network-based tracker called SiamRPN++ (Held et al., 2016).

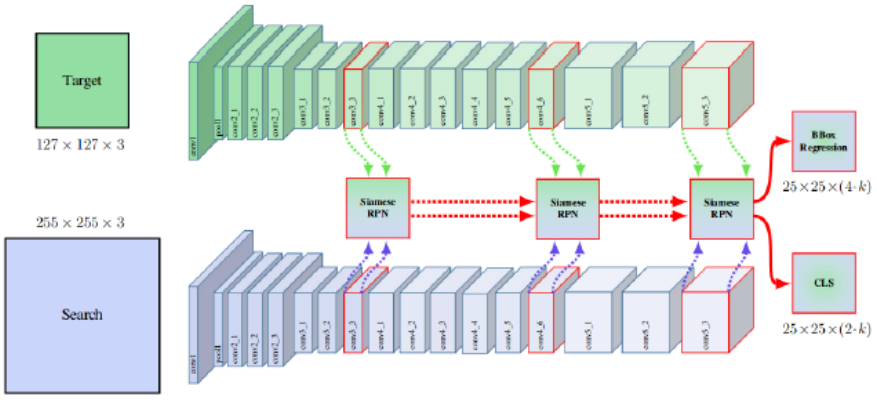


Figure 4: SiamRPN++ (B. Li & Zhang, 2018)

- **Tracking as regression:** in this group, on the other hand, network return location of the intended object by receiving only previous and current frames .It is able to model changes in scale and aspect of the tracked template, since this tracker predicts a bounding box instead of just the position. However, it only can process a single target and it needs from data augmentation techniques to learn all possible transformations of the targets (Held et al., 2016).
- **Tracking with RNN:** this type of algorithms uses Recurrent Neural Networks to model the sequence of movement of objects from the detection obtained. Thus improves the response to prolonged occlusions in time, for example(Sadeghian et al., 2017). They have good accuracy, but they usually do not perform well in real-time.

In addition, Multiple Object Tracking (MOT) algorithms are divided into two approach which are online and batch. In Batch tracking algorithms, for identifies object location, it can be used future frames information. For better result in tracking quality, they often exploit global information. The majority of MOT algorithms share the following steps (Ciaparrone et al., 2019):

- **Detection Step**
- **Feature extraction and Motion Prediction Step**
- **Affinity Step**
- **Association Step**

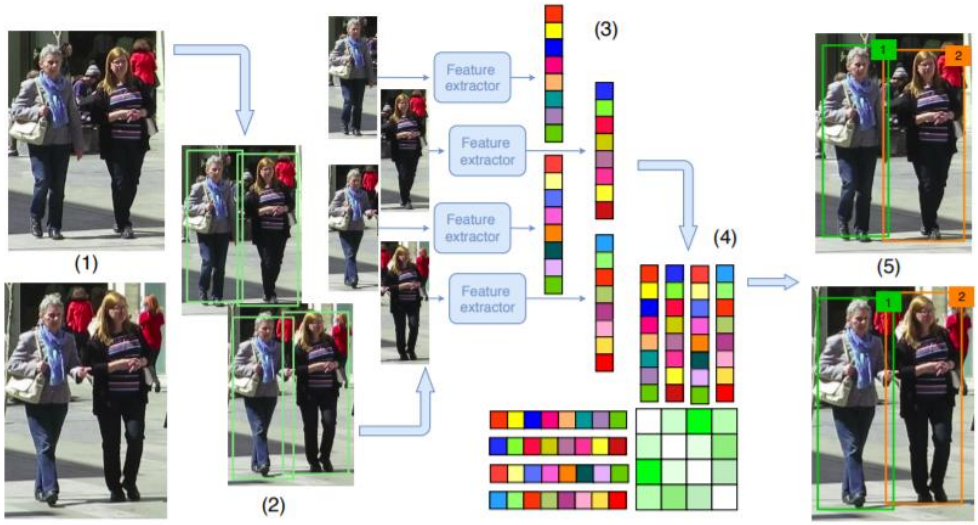


Figure 5: Multi Object Tracking algorithm workflow, it begins by input video frame (1), then bounding boxes are generated by detector algorithm (2). Following, calculated features are extracted (3). Next, by affinity stage, likelihood of objects belonging to same target is recognized (4) to in the next step assign IDs to objects (5) (Ciaparrone et al., 2019).

3.2 DATASETS FOR OBJECT TRACKING

The visual object tracking is a important task in computer vision which has many applications. This task, the same way as others in the field, needs datasets from which create and evaluate the algorithms. The datasets are also commonly associated with competitions that allow the benchmarking of the developed algorithms. These benchmarks often provide the most objective measure of performance and, for this reason, they are important guides for research in the area of study. The visual tracking datasets can be divided according to their tracking target, that is, if they are focused on the tracking of a single object (SOT) or on the tracking of multiple objects (MOT).

3.2.1 Multiple Object Tracking Datasets

- **MOT**:

  This dataset arises from the need to provide a general and standardized way to create multi-object tracking algorithms, evaluate the results and present them. Recently, the computer vision community has promoted several benchmarks for the evaluation of numerous tasks like object detection, optical flow or stereo estimation that have

advanced the state of the art in these areas. However, not so much effort has been made in the standardization of the evaluation of multiple target tracking (Milan et al., 2016).

As many other datasets it is associated with a challenge, the MOTChallenge. With this challenge the organizers try to create a unified framework for the evaluation of multi-target tracking. The dataset provides a collection of datasets, some of them coming from datasets already in use and some from new challenging data. The given data are video sequences (Milan et al., 2016).

The first release of the dataset named MOT15 was focused on multiple people tracking, following the trend of other datasets. The pedestrian tracking is by far the most studied case in the tracking context. In the next releases, more significant classes generally seen in urban scenarios were added, like vehicles, bicycles or motorbikes. The challenge has had three editions: MOT15, MOT16, MOT17. In each of them the sequences were more challenging than the edition before. This can include different camera viewpoints and positions, more challenging weather conditions (cloudy, night, sunny). For example, the mean crowd density in MOT16 is three times higher when compared to the first benchmark release (Milan et al., 2016).

Figure 6: An overview of the MOT16 dataset. Top: train sequences. Bottom: test (Milan et al., 2016).

- **ALOV:**

  The Amsterdam Library of Ordinary Videos for tracking is another well-known visual object tracking dataset in the field. It covers different situations including illuminations, transparency, zoom or low contrast, for example. There are 315 videos in 64 different targets that gathered from YouTube (*Dataset Resources*, 2020).

- **CAVIAR**:

  The CAVIAR is a project from INRIA labs was gathered to the development of algorithms that can describe and understand video scenes. The scenes were associated with surveillance scenarios where people performed some different activities related with the surveillance area. Those activities included walking, browsing, resting, leaving bags behind or two people fighting. The annotations contain, apart from the bounding boxes locations, the head and feet positions, the body direction, among others. Referring to the tracking task, the challenging problems include occlusions, appearance/disappearance, appearance changing or similar object tracking, for example. In terms of data size, the first set contains 28 video sequences and the second set contains 44 video sequences (Figure 7). It is a well-known dataset and is commonly used for development and testing of tracking algorithms (Dubuisson & Gonzales, 2016).

Figure 7: CAVIAR (Dubuisson & Gonzales, 2016)

- **TrackingNet**:

  Most of the commented datasets are small regarding to the size. Nowadays, this tracker relies on datasets that have the object detection information because absence of dedicated comprehensive tracking datasets. For this reason, TrackingNet one of the first large-scale object tracking data set in wild created. TrackingNet provides a total of 30643 video segments “with more than 14 million dense bounding box annotations” (Müller et al., 2018) (Figure 8). The contributions of this work include diverse methods that produce annotations that are dense from the ones with coarse and a baseline that is prolonged for the trackers’ benchmarked state of the art on TrackingNet. Referring to the latter, the authors affirm that doing a pretraining of the deep models on this dataset could “improve

their performance on other datasets by increasing their metrics by up to 1.7%" (Müller et al., 2018).

Datasets	Nb Videos	Nb Annot.	Frame per Video	Nb Classes
VIVID [38]	9	16274	1808.2	-
TC128 [37]	129	55652	431.4	-
OTB50 [4]	51	29491	578.3	-
OTB100 [5]	98	58610	598.1	-
VOT16 [10]	60	21455	357.6	-
VOT17 [11]	60	21356	355.9	-
UAV20L [40]	20	58670	2933.5	-
UAV123 [40]	91	113476	1247.0	-
NUS PRO [39]	365	135305	370.7	-
ALOV300 [17]	314	151657	483.0	-
NfS [36]	100	383000	3830.0	-
MOT16 [16]	7	182326	845.6	-
MOT17 [16]	21	564228	845.6	-
TrackingNet (Train)	**30132**	**14205677**	**471.4**	**27**
TrackingNet (Test)	**511**	**225589**	**441.5**	**27**

Figure 8: TrackingNet: difference of current datasets size for object tracking (Müller et al., 2018).

3.2.2 Single Object Tracking Datasets

- **OTB**:

  For surveillance tracking scenarios there are some data sets, however, objects that are human or cars are in small size with static background. Also, some of the scenes are sometimes not annotated with bounding boxes which makes them not very useful for the comparison of tracking algorithms. This data set is built with 50 fully annotated sequences in the first release OTB50 to ease the evaluation task. Later, the dataset was extended with another 50 sequences (OTB100) (Wu et al., 2013).

  Many factors can affect the tracking performance such as illumination variation or occlusion, for this reason the authors categorized the sequences with 11 attributes according to the occurrence of any of the selected factors (Figure 9). Apart from the data side, to simplify large scale performance evaluation, the authors integrated most of the publicly available trackers at the time to create a code library with uniform input and output formats (Wu et al., 2013). Including TLD (Kalal, 2010), MIL (Babenko et al., 2009) or CPF (Pérez et al., 2002) making a total of 29 tracking algorithms.

Attr	Description
IV	Illumination Variation - the illumination in the target region is significantly changed.
SV	Scale Variation - the ratio of the bounding boxes of the first frame and the current frame is out of the range $[1/t_s, t_s]$, $t_s > 1$ (t_s=2).
OCC	Occlusion - the target is partially or fully occluded.
DEF	Deformation - non-rigid object deformation.
MB	Motion Blur - the target region is blurred due to the motion of target or camera.
FM	Fast Motion - the motion of the ground truth is larger than t_m pixels (t_m=20).
IPR	In-Plane Rotation - the target rotates in the image plane.
OPR	Out-of-Plane Rotation - the target rotates out of the image plane.
OV	Out-of-View - some portion of the target leaves the view.
BC	Background Clutters - the background near the target has the similar color or texture as the target.
LR	Low Resolution - the number of pixels inside the ground-truth bounding box is less than t_r (t_r=400).

Figure 9: OTB: List of the annotated to test sequences (Wu et al., 2013).

- **VOT**:

  The visual object tracking creativity started in 2013 to address performance evaluation of short-term visual object trackers. By short term tracking means that assumed after target is lost, trackers cannot re-detect successfully that target, and they reset after that situation. In all the previous editions the challenge considers single camera, single target, model free, causal trackers applied to short-term tracking. The main goal of VOT is establishing datasets, evaluation measures and toolkits for visual object tracking (as many other initiatives). The successive editions were made in conjunction with Computer Vision Conferences like ICCV or ECCV. In 2015, a sub challenge focused on tracking in thermal infrared (TIR) was made due to the growing interest in this kind of imaging. Referring to the data itself, the VOT datasets try to pay more attention to the diversity of the data and the quality of the content and annotation with respect to the quantity. For example, some datasets assign a global attribute to the entire sequence when it is happening in a fragment of it. VOT dataset tries to avoid the assumption that the quality of the data is correlated with its size (Kristan et al., 2017). “The VOT Challenge has focused on developing a methodology for automatic construction and annotation of moderately large datasets from a large pool of sequences” (Kristan et al., 2017) (Figure 10).

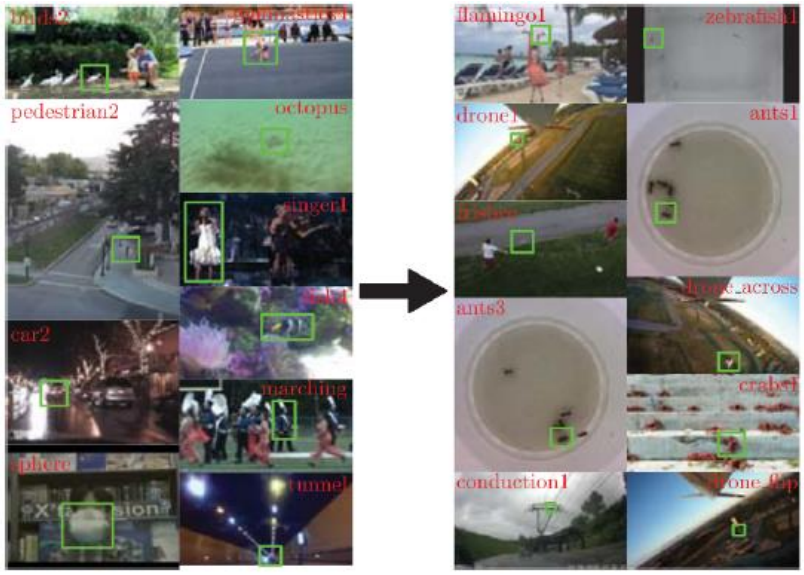


Figure 10: VOT: VOT2016 sequences (left) replaced by VOT2017 with new sequences (right).(Kristan et al., 2017)

3.3 METRICS FOR MULTI-OBJECT TRACKING EVALUATION

Apart from the datasets and algorithms used to solve a given task or problem it is necessary to use a measure or set of measurements that provide an evaluation of the performance of the obtained solution. In this section, the most important metrics used for evaluating multiple object tracking are commented.

For the metrics used in this evaluation, the classification from MOT (Milan et al., 2016) is used as reference. As it will be seen, the performance evaluation for MOT algorithms is not so easy as the one presented for object detection. Tracing metrics can according attributes can be classified into four subsets:

- **Accuracy**: this type of metrics tries to measure how accurately a tracking algorithm tracks targets. From this type of metric, the following two are briefly commented: IDs (Yamaguchi et al., 2011) and MOTA (Bernardin & Stiefelhagen, 2008). The IDs metric measures the ID switches, i.e. given an id for an object it measures how many times the MOT algorithm changes this id. The Multiple Object Tracking Accuracy (MOTA) is calculated as follows:

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t}$$

Equation 1: Where m_t , fp_t , mme_t, and g_t are number of misdetections , number of false positives, number of mismatches and sum of true positive and fp_t, respectively (Bernardin & Stiefelhagen, 2008).

Computing single number by combination of FP rate, FN rate and mismatch rate is, as the authors indicate, "by far the most widely accepted evaluation measure for MOT" (Milan et al., 2016) and it gives an intuitive measure on the tracker's performance at detection and trajectory. It does not consider the precision of that detections' location.

- **Precision**: in this metrics group the key factor is the description of the precision that the tracked objects have using criteria such as bounding box overlap or distance. The most important are MOTP (Bernardin & Stiefelhagen, 2008), TDE (Kratz & Nishino, 2010) and OSPA (Ristic et al., 2011). MOTP, for example, uses a ratio with the distance between the ground-truth detections locations d and the associated detected locations c.

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}$$

Equation 2: MOTP definition (Bernardin & Stiefelhagen, 2008).

- **Completeness**: it refers to ground truth trajectories that how completely they tracked. This set includes the results from Mostly Tracked (MT), Partly Tracked (PT), Mostly Lost (ML) and Fragmentation (FM) (Yuan Li et al., 2009).
- **Robustness**: this last type of metrics is linked to the recovering from occlusion. Examples of this group are Recover from Short-term occlusion (RS) and Recover from Long-term occlusion (Song et al., 2010).

3.4 OBJECT CLASSIFICATION AND DETECTION USING NEURAL NETWORKS

Many of the progress made in recent years on the classification field of computer vision can be directly associated with the use of neural network architectures. The first big step forward came in 2012 when AlexNet (Krizhevsky & Hinton, 2012) beat all the proposals of the state-of-the-art at that time in the ImageNet challenge, ILSVRC. This competition of classification in images is a reference in the computer vision community. Test error rate of AlexNet (15.3%) was lower than

the previous year's winner error which was 26.2%. This network follows the basic design archetype of convolutional neural networks: a sequences of convolution layers, tailed by max-pooling and activation layers before the final classification layers (fully connected). Typically, object detector can be classified into two categories, two stage like Faster R-CNN (Ren et al., 2017), and one stage like YOLO (Redmon et al., 2016), SSD (Liu et al., 2016). The one stage detectors can achieve higher speed, while two stage detectors are better in localization and recognition accuracy. In two stage detectors, RoI (Region of Interest) pooling layer can divide the two stages. For example, in Faster R-CNN, the first stage, propose candidate object bounding boxes that called RPN (Region Proposal Network). The second stage, operate bounding box regression and classification after extracted features from RoI Pooling operation (He, Gkioxari, Dollar, et al., 2017). Figure 11 displays One and Two stage detectors. *(Jiao et al., 2019)*.

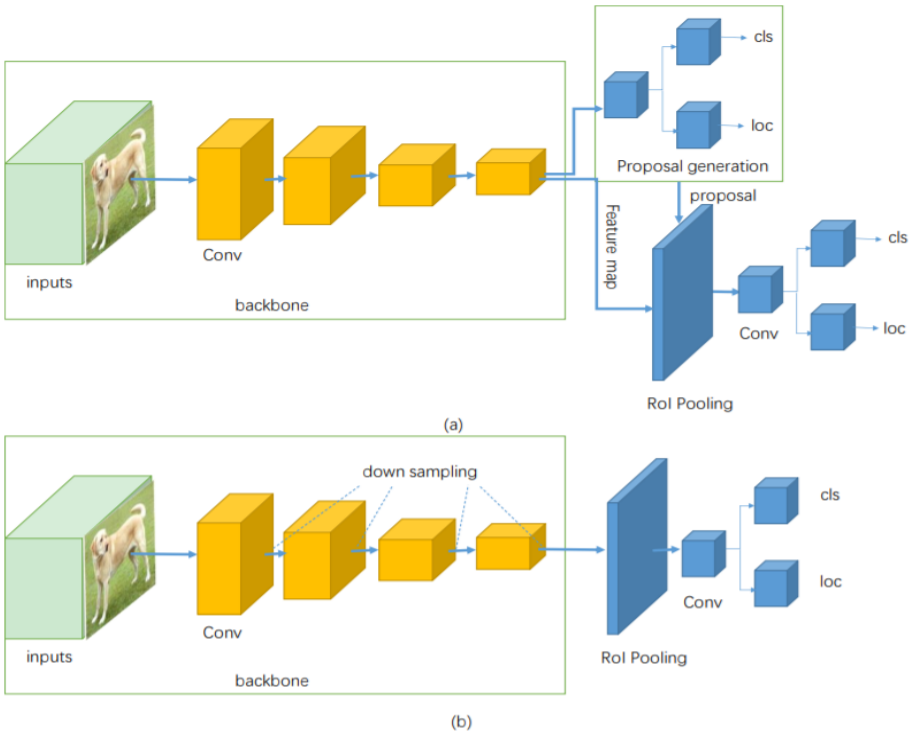


Figure 11: Two stage (a) and One stage (b) Detectors' Architecture (Jiao et al., 2019).

The next architectures are being used as blocks that serve as the basis for numerous subsequent works (commonly known as backbone networks) in computer vision and are briefly commented below:

- **VGG**: this architecture from the VGG Group (University of Oxford) makes the improvement over AlexNet by replacing larger kernel-sized filters of size 11 and 5 in the first layers with multiple 3x3 kernel filters one on top of each other. With multiple stacked smaller kernels, the depth of the network increases allowing it to learn more complex features at a lower cost (Simonyan & Zisserman, 2014).
- **MobileNet**: is a simplified version of Xception (Chollet, 2017) for mobile applications that is currently behind the computer vision applications used on Google mobile devices. A year after MobileNet v1, MobileNet v2 was introduced with a great improvement respect to the previous version. For example, the new models used two times fewer operations (Sandler et al., 2018). In terms of architecture, the main changes are the residual connections and the expand/projection layers in the main building block, the bottleneck residual block see Figure 12 (Sandler et al., 2018).

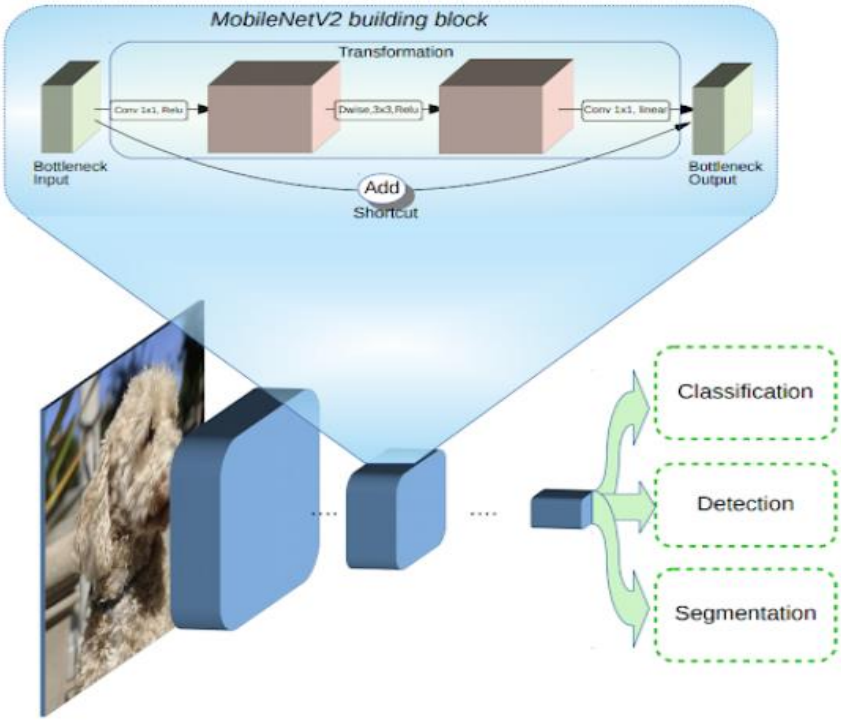


Figure 12: MobileNet v2 (Sandler et al., 2018).

- **Inception**: this family of networks looks for wider networks, that is, with more intermediate operations between layers. The authors try to increase neural networks, in terms of operations, without an increase in computational cost. They try to reduce the

still huge computational requirements of VGG specially in terms of reducing the number of calculations done due to large width of convolutional layers. Introducing different parallel convolution operations, the density of extracted information increases but also the computational costs. To solve the problem, they use 1x1 convolutions to reduce dimensionality while performing different transformations in parallel. The resulting networks are simultaneously deep and wide. The first version of Inception, known as GoogLeNet, was the winner of the ILSVRC in 2014 (Szegedy et al., 2015). It was improved later with Inception v2 and v3. The last Inception v4 creates a hybrid with ResNet, known as Inception-ResNet (Szegedy et al., 2017).

- **Res-Net**: this network tries to solve the problem that seems to appear when adding layers to a network which is that it generally behaves worse. For this reason, the authors propose that instead of trying to learn the hidden mapping of the input x to the function H(x), learn the difference between the two, that is, the residue (residual net). The original mapping is recanted into F(x) + x. This is a big change at the time as it solves the problem of the vanishing gradients that the neural networks have suffered until the date. In addition, it allows to create much deeper networks with more layers, that will allow better results Figure 13 (He et al., 2016).

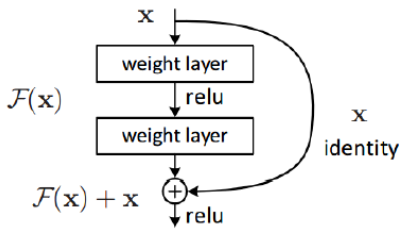


Figure 13: Residual learning block (He et al., 2016).

By increasing demand of the faster and more accurate object detection systems based on the emerge of new object detection applications. This added new tasks to object detector like obtaining the location with its corresponding bounding box. This makes object detection significantly more complicated than before that was image classification.

However, the most successful object detection algorithms today are extensions of image classification models. Usually, network architectures such as VGG or ResNet are used as

backbone networks as they perform the feature extraction. After the backbone, the head of the network is stacked. The following object detection models follow the commented scheme (Fu et al., 2017).

- **Faster R-CNN**: is one of the current reference models and one of the last detectors known as region-based from Girshick et al. This model basically works in the following way: it uses some mechanism to draw out regions from an image that are probably an object and then classifies those proposed regions with a CNN. The father of this model is the R-CNN and it was the real driver of this type of techniques (Girshick et al., 2014). In the proposed regions obtained through an algorithm called Selective Search the characteristics are extracted through a CNN by region and then those regions are classified based on the characteristics. But its performance was slow. This performance improves with Fast R-CNN (Girshick, 2015) for two main reasons. First, the CNN is applied over the whole image instead of over each region and then the regions are obtained from the last map of characteristics of the network. Second, the introduction of a Softmax activation layer simplifies classification. This mechanism was faster and easier to train than R-CNN but there was still a bottleneck in the generation of regions (Ren et al., 2017).

  To solve it the RPN (Region Proposal Network) is introduced and added to the Fast R-CNN to create Faster R-CNN. The RPN returns proposed regions based on a score that refers to the probability that the bounding box is an object, the abjectness (Figure 14). And these regions are passed directly to the Fast R-CNN to perform the classification.

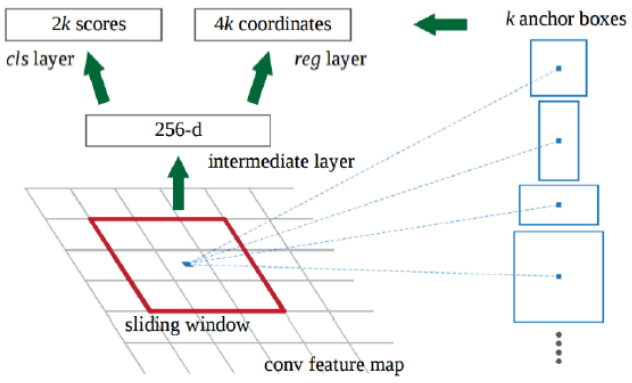


Figure 14: Region Proposal Network (Ren et al., 2017).

- **Overfeat**: winner of the ILSVRC 2013 in detection and location of objects, this work showed that training a convolutional network to instantaneously locate, classify and detect objects in images can enhance the success both in classification, detection and location. Subsequently, it has been replaced by SSD and YOLO for tasks that require better performance in real time (Sermanet et al., 2014).
- **SSD**: it provides great speed gains over Faster R-CNN by performing the phases of generating regions of interest and subsequent classification jointly (Single Shot MultiBox Detector). As a result, you get a lot of bounding boxes which most of them are not useful. By applying the techniques known as non-maximum suppression and hard-negative mining the final detections are achieved. In the MobileNet v2 paper (Sandler et al., 2018) SSDLite is proposed which reduces parameter count and computational cost with respect to regular SSD. To do so, the authors replace all the regular convolutions with separable convolutions (Liu et al., 2016).
- **R-FCN**: there are faster models than Faster R-CNN such as the Region-based fully convolutional network or R-FCN. The authors try to solve the problems of SSD for detecting small objects because the detection in SSD was done on the feature map when features have low spatial resolution. This network tries to improve system speed by maximizing shared computing and provides a good balance between accuracy and speed (Dai, 2016).
- **YOLO**: this model, also from the "single-shot networks family", uses a different approach with respect to the above. This network splits the image into regions and predicts the bounding box and likelihoods of each region. These are then weighted with the probabilities to obtain the definitive detections (Figure 15). This performs, as the authors indicate, a hundred times faster than Fast R-CNN maintaining a similar accuracy (Redmon et al., 2016).

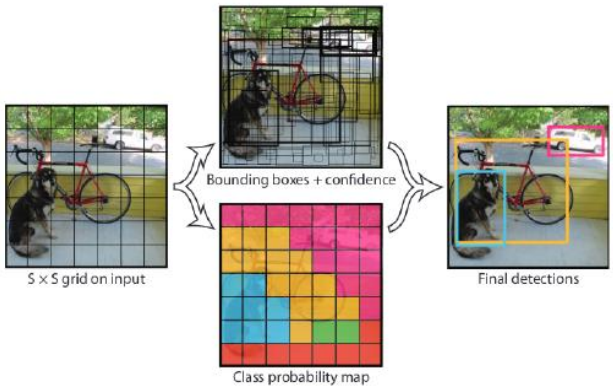


Figure 15: The YOLO v1 model (Redmon et al., 2016).

The YOLO v2 model introduced big improvements: removed all fully connected layers and used anchor boxes to predict bounding boxes, used batch normalization on all convolutional layers and allowed for multi-scale training, among others. In the next table, it can be seen how YOLO v2 is almost on a par with methods like SSD or Faster R-CNN. However, it has a better balance among accuracy and speed since it manages to work in some cases at 91 FPS (frames per second) when Faster R-CNN barely reaches 10 FPS, see Table 1 (Redmon & Farhadi, 2017).

Table 1: Accuracy comparison in test on PASCAL VOC 2012 (Redmon & Farhadi, 2017)

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast R-CNN [5]	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
Faster R-CNN [15]	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
YOLO [14]	07++12	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
SSD300 [11]	07++12	72.4	85.6	80.1	70.5	57.6	46.2	79.4	76.1	89.2	53.0	77.0	60.8	87.0	83.1	82.3	79.4	45.9	75.9	69.5	81.9	67.5
SSD512 [11]	07++12	74.9	87.4	82.3	75.8	59.0	52.6	81.7	81.5	90.0	55.4	79.0	59.8	88.4	84.3	84.7	83.3	50.2	78.0	66.3	86.3	72.0
ResNet [6]	07++12	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
YOLOv2 544	07++12	73.4	86.3	82.0	74.8	59.2	51.8	79.8	76.5	90.6	52.1	78.2	58.5	89.3	82.5	83.4	81.3	49.1	77.2	62.4	83.8	68.7

The latest version of YOLO, called YOLOv3, achieves a 57,9 mAP on COCO testdev. The frame rate is lower than the obtained with YOLOv2 (with the same image input size) but it still performs as a state-of-the-art real time object detection system, according to the author (Redmon, 2018).

3.5 INSTANCE SEGMENTATION USING NEURAL NETWORKS

The machine vision community has improved the results obtained in object instance segmentation and detection in a short time thanks, in large part, to powerful base systems such as Faster R-CNN. This project will use the detections coming from instance segmentation networks, so this type of segmentation is going to be introduced, including some recent instance

segmentation models such as Mask R-CNN. The instance segmentation requires the correct detection of all objects in the image along with the precise segmentation of each instance. Thus, each pixel belongs to one of the different categories without differentiating whether it is in a particular object. Semantic segmentation differs from the instance segmentation in that in the first the labels are class-aware whereas in the second the labels are instance-aware. Driven by the effectiveness of the R-CNN family many of the methods proposed for instance segmentation is based on segment proposals where segmentation precedes object type recognition (Pinheiro et al., 2015). This has proved to be slower and more inaccurate than if the prediction of object masks and class labels were done in parallel and separately. Li et al. proposes a system known as FCIS (Fully Convolutional Instance Segmentation) (Yi Li et al., 2017) that tries to predict the output of a set of position-sensitive channels in a completely convolutional way. These channels perform the tasks of class, bounding box and masks calculations simultaneously which makes them faster. But it shows errors in instances that overlap creating spurious edges systematically (Figure 16). Recently, Mask R-CNN (He, Gkioxari, Dollár, et al., 2017) arose to solve many of these problems and it has situated itself as a State of the Art technique in segmentation of instances see Figure 16.

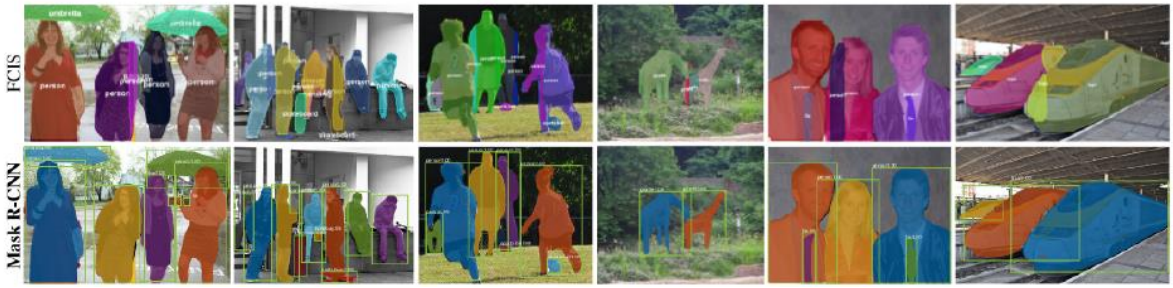


Figure 16: Results obtained by FCIS and Mask R-CNN in test images in COCO Dataset (He, Gkioxari, Dollár, et al., 2017).

Conceptually, Mask R-CNN adds a third stage to Faster R-CNN in which it obtains the mask of the object. The first stage of RPN coincides with that of Faster R-CNN while in the second stage it calculates, in parallel with the prediction of the class and the bounding box, a binary mask for each region of interest (RoI). The generation of masks for each class is done without the classes competing with each other, which allows to separate the mask and class predictions from the

object prediction. According to the authors, this proves to be the key to obtain good results in the final segmentation.

Another key factor in the proper functioning of this method is the correct alignment between the RoI and the extracted characteristics. This is usually done using RoIPool in Fast R-CNN but introduces misalignments if the purpose is to segment rather than classify. Therefore, Mask R-CNN authors create RoIAlign. To demonstrate the generality of the proposed method the authors introduce the mask prediction branch on several existing neural network architectures like Faster R-CNN with ResNet, for example, and they manage to surpass the winners of the 2015 and 2016 COCO Challenge segmentation, MNC (Dai et al., 2015) and FCIS (Yi Li et al., 2017).

3.6 DATASETS FOR OBJECT DETECTION

The datasets used when implementing or testing a certain system are a key factor, since they influence the performance that the system can achieve. They also permit for an assessment of the solution found regarding others that are part of the state-of-the-art in the task that is carried out, since they are usually associated with some type of competition. Therefore, it is necessary to correctly choose the dataset or datasets used in a computer vision problem. Here are some of the most well-known datasets used in many object detection applications:

- **COCO (Common Objects in Context)**: it is a comprehensive data set for segmentation and detection of objects mainly. It contains 80 categories of objects and 330000 images of which more than 200000 are labeled. It is a dataset widely used between the community and in congresses such as the ICCV7 (International Conference on Computer Vision).
- **PASCAL VOC**: this dataset is linked with another challenge, the Pascal VOC Challenges. The organizers ran this competition from 2005 to 2012. This project provides standardized image datasets for object class recognition, segmentation or action classification tasks.
- **ImageNet**: it consists of 14 million images approximately and an average of 500 images for each category. It organizes the well-known ILSVRC10 competition of location and detection of objects in images and videos. It is one of the reference datasets in this area.
- **KITTI**: centered in the autonomous driving field, this vision benchmark suite introduces itself as a novel challenging real-world computer vision benchmark. The main areas of interest include 3D/2D object detection, 3D tracking or stereo vision. The type of objects

for object detection available are focused in the ADAS field such as car, van, truck, pedestrian or cyclist.

- **Cityscapes**: this dataset focuses on semantic segmentation in urban scenes. It contains 30 kinds of objects, 5000 images labeled with a fine label (more precise) and 20000 labeled with a coarse label in 50 different cities.
- **OpenImages**: it is a data set of around 9 million images. This makes it the "largest existing dataset with object location annotations". It also has a bigger number of classes than other challenges as the previously cited COCO and PASCAL VOC, exactly 600 object classes. It must be mentioned that the label distributions are usually skewed and with OpenImages it occurred too. This means that there are many more objects of some kinds than others.

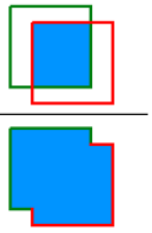
There are many other datasets such as those from research centers like INRIA, MIT or Caltech that contribute to the continuous improvement of the available data.

3.7 METRICS FOR OBJECT DETECTION EVALUATION

Before going deeper with the most common metrics in the evaluation of object detection, the basic concepts need to be mentioned. When talking about object detection, the following definitions usually appear:

- **Intersection over Union (IoU)**: recognized as Jaccard index, this measure evaluates the intersection between two bounding boxes, a ground truth bounding box and the predicted bounding box. With this definition a prediction can be classified into valid (TP) or invalid (FP) (see *Equation 3*).

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\quad}{\quad}$$



Equation 3: IoU definition.

- **True Positive (TP)**: is a correct detection. The condition is that the IoU must be above or equal to a given threshold. This threshold is usually defined in percentage to 50%, 75% or 95%. The results obtained by a system with these three thresholds can define its behavior. For example, a given object detector can easily have good results at a 0,5 IoU but not so easily at a 0,95 IoU.
- **False Positive (FP)**: is a false detection. The IoU of the detection must be below the threshold.
- **False Negative (FN)**: is a detection not detected.
- **True Negative (TN)**: it is defined as all the probable bounding boxes that were not detected correctly. It is not used in metrics.

It is very common to see that the metrics are established by a given challenge or associated with it. It is the case of the Pascal VOC challenge that uses the precision/recall curve and the average precision. These terms are now defined:

- **Precision**: this is the proportion of predictions that are correct positive (*Equation 4*).

$$Precision = \frac{TP}{TP + FP}$$

Equation 4: Proportion of predictions

- **Recall**: this is the proportion of positive predictions with respect to all positives (*Equation 5*).

$$Recall = \frac{TP}{TP + FN}$$

Equation 5:Recall

- **Precision/Recall curve**: this curve plots the performance of an object detector as the confidence is changed for each object class. A good precision/recall curve has a high precision while recall increases, i.e. if the confidence threshold varies, the precision and recall stay high.
- **Average precision (AP)**: the AP summarizes the shape of the previous curve allowing to obtain the Area Under the Curve (AUC). This is done because of the nature of the precision/recall curve in form of “zigzags” that does not permit an easy comparative

between different curves (detectors). This metric is the precision averaged all recall values that are between 0 and 1 (Figure 17).

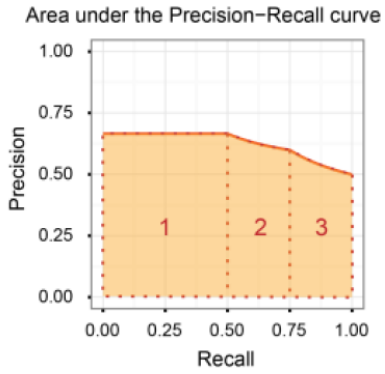


Figure 17: AUC example: the areas from the trapezoids are 0,335, 0,15875 and 0,1375.

This average can be done in two main ways: 11-point interpolation or interpolating all points.

1. **11-point interpolation**: it is defined as the mean precision at a group of eleven equally spaced recall values ranging from 0 to 1. At each recall value precision is obtained by taking the maximum precision measured value for a technique for which the corresponding recall is above r (Equation 6). This was the method used in Pascal VOC 2008.

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} \rho_{interp(r)}$$

$$\rho_{interp(r)} = \max_{\tilde{r}:\tilde{r} \geq r} \rho(\tilde{r})$$

Equation 6: 11-point interpolation

2. **All points interpolation**: in this case the mean precision is done interpolating through all recall points (Equation 7). The precision at each level r is obtained taking the maximum precision which has a recall value equal or greater than the recall value at the level r + 1 (Equation 7). This method of interpolation is used in Pascal VOC metrics from the year 2010 onwards.

$$\sum_{r=0}^{1}(r_{n+1}-r_n)\rho_{interp(r_{n+1})}$$

$$\rho_{interp(r_{n+1})}=\max_{\tilde{r}:\tilde{r}\geq r_{n+1}}\rho(\tilde{r})$$

Equation 7: All points interpolation

In the next image (Figure 18), these calculations are presented in a graphical way. With this interpolation the AUC obtained is exact.

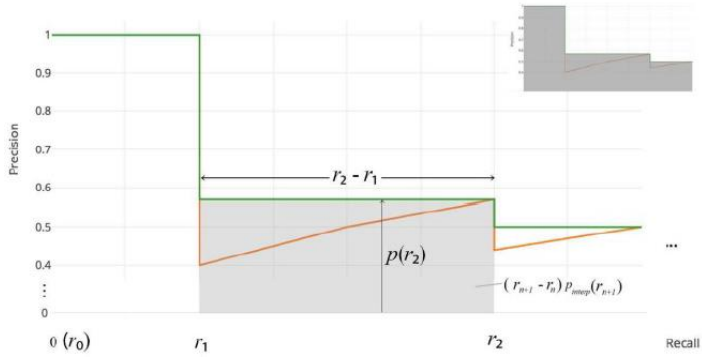


Figure 18: All points interpolation from (Hui, 2018).

3.8 2D MULTI-HUMAN POSE ESTIMATION

The Pose Human Estimation methods show person skeleton in graphical format, that is group of point in specific coordinate system and each of this point called joint or key point. Each two of these points with correct connection called pair (Raj, 2019). An example of human poses (Figure 19).

Figure 19: Human Pose Skeletons, in format of COCO data sets (left). Sample Human Pose Skeletons on image (Right) (Hidalgo, 2018).

Over the years, many methods to human pose estimation and generally it can be divided into two approaches:

- **Top-Down:** it is simple and work with person detector that for each detected person calculate the joints and key points (Raj, 2019).
- **Bottom-Up:** in this method first detect all joint of all the people in given image, following by grouping parts belonging to distinct people (Raj, 2019).

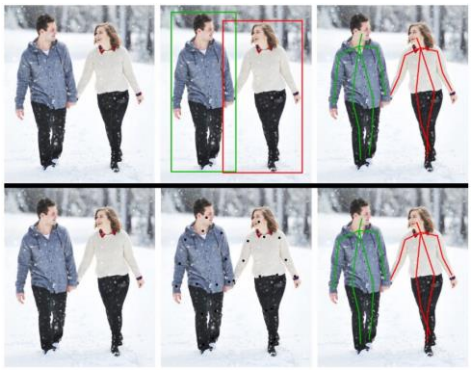


Figure 20: Top: Top Down method. Bottom: Bottom Up method (Raj, 2019).

Following explain some algorithms related to the top down and bottom up approaches:

- **OpenPose**: one of the famous bottom up approaches. OpenPose detect all joints related to the all person in the image, then organized detected joints by grouping joints to distinct person. In Figure 21 the OpenPose's architecture can be seen (Cao et al., 2018).

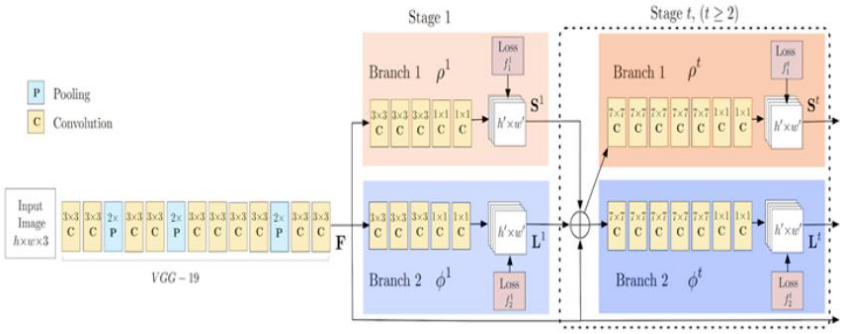


Figure 21: OpenPose architecture (Cao et al., 2016).

First few layers in the OpenPose model's architecture gather features from an input image. Then, these extracted features go to two divided and parallel conv layers. One of the conv layer branch responsivity is to characterizes the degree of grouping between parts by predicts a group of 38 Part Affinity Fields (PAFs). The other one, predicts a group of 18 maps that present human skeleton. Then, pairs of joints formed by Bipartite graphs step. All the steps can be seen in Figure 22 (Cao et al., 2018).

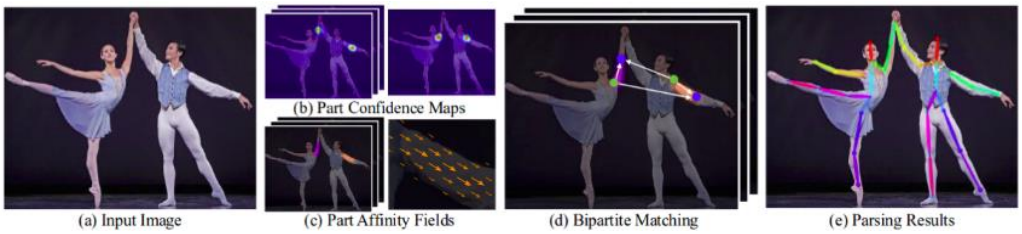


Figure 22: human pose estimation steps in OpenPose (Cao et al., 2018).

- **RMPE:** it is a top down approaches of human pose estimation. Pose estimation applied on the region and location where detector suggested that person can be there and the performance of this type of estimator totally rely on the accuracy of the human detectors (Fang et al., 2016).

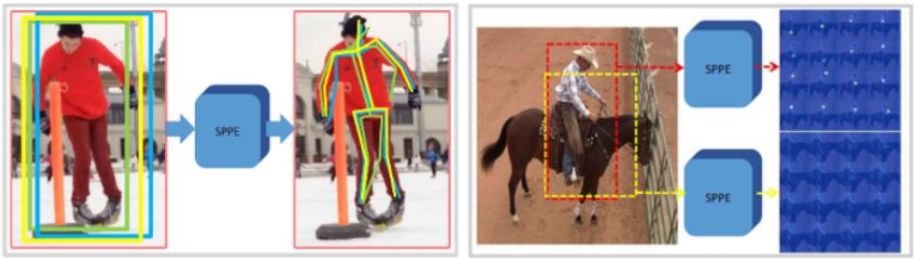


Figure 23: (left) Shows downside effets of more than one prediction for same object. (right) low score bounding boxes. (Fang et al., 2016)

To fix this problem, the paper proposed the usage of a network called Symmetric Spatial Transformer Network (SSTN) to draw out a good single person region from a wrong bounding box (Figure 23) (Fang et al., 2016).

3.9 3D HUMAN POSE ESTIMATION

Research on 3D Human Pose is less mature in comparison with the 2D case. There are two main methods: first to estimate a 2D pose and then reconstruct a 3D pose or to regress a 3D pose directly. Research on multi person 3D pose estimation is presently slightly restricted, mainly due to lack of good data sets. Most works report results on the Human3.6 dataset, which is the main data set for single person 3D pose performance comparison. It consists of multi-view videos of a single person in a room, whose pose is captured with the OptiTrack motion capture system (Ionescu et al., 2014).

Most researchers concentrate on reconstructing a 3D pose from a single image and few of them look at multi-view. Some consider depth in addition to an RGB image. Most works consider a

single frame, and few take time continuity constraints into account. Despite that 3D pose estimation is, in general, more computationally intensive, many 3D pose models are real-time. Most models use supervision, but there are few models, which are semi-supervised or fully self-unsupervised. Several models with the best performance will be described below in more details.

3.9.1 Single-person 3D Pose

Most works for estimating human pose for a single person use a single image/video. Despite of the ambiguity in the depth dimension, models trained on 3D ground truth (GT) show pretty good performance for the case of a single person without occlusions. Similarly to humans, a neural network can learn to predict depth from a mono image in case it has already encountered similar scenes, which is demonstrated in the recent research on depth estimation.

- **A Simple Effective Baseline for 3D Human Body Pose Estimation**: baseline model, which is a fully-connected network with residual connections, takes as an input 2D pose from an off the shelf stat-of-the-art a 2D detector and predicts a 3D pose with a regression loss from a single image. Interestingly, this simple model has a pretty good performance on Human 3.6: Mean per Joint Position Error (MPJPE) (protocol 1) is *63 mm*. Since the network is lightweight, if the 2D pose estimation is real-time, the full model is *real-time* as well (Figure 24) (Martinez et al., 2017).

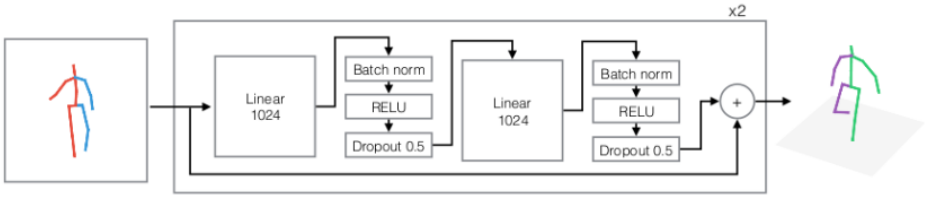


Figure 24: Baseline model for 3d human body pose estimation (Martinez et al., 2017).

- **To 3D Human Body Pose Estimation in the Wild: A Weakly Supervised Approach**: on the other hand, regress both 3D and 2D poses simultaneously and reaches similar performance to Martinez (Martinez et al., 2017): MPJPE *65 mm* on Human3.6. Zhou (Zhou et al., 2017) adopts a common approach for 2D pose estimation- an HourGlass network (Newell et al., 2016), which outputs heatmaps for every joint. The loss is an L2 distance between the predicted heatmaps and the GT, rendered through a Gaussian kernel. Depth is regressed directly. In addition, there is a 3D geometric constraint loss, since relations between bone lengths remain relatively fixed in a human skeleton,

allowing to extend the 3D pose estimations to images in the wild. The model is real-time (25 FPS on a laptop with Nvidia GTX 960) (Figure 25) (Zhou et al., 2017).

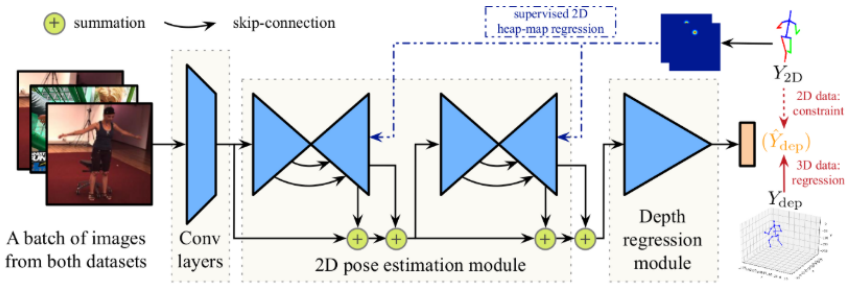


Figure 25: Network overview (Zhou et al., 2017).

- **Integral Person Pose Regression:** use integral regression, instead of an L2 regression, for joint 2D/3D pose estimation, which is the combination of all locations in the heat map weighted by their likelihoods. It is effective, and well-matched with any heat map-based approaches. Since integral regression is simple and non-parametric, it adds a negligible overhead in computation and memory, leaving a real-time base model to stay real-time but improving performance. The authors experiment with several backbones: HourGlass (Newell et al., 2016) and Resnet18, 50 and 101. Integral regression with Resnet50 achieves the best performance on Human3.6: MPJPE 41 mm (Figure 26) (Sun et al., 2017).

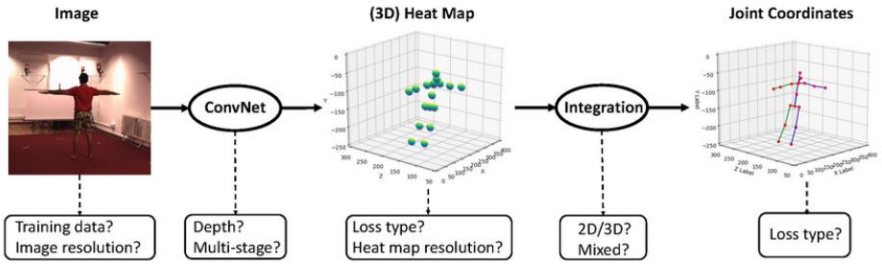


Figure 26: Integral regression model overview (Sun et al., 2017).

- **3D human pose estimation in video with temporal convolutions and semi supervised training:** start with predicted 2D poses from HourGlass (Newell et al., 2016), Mask R-CNN (He, Gkioxari, Dollár, et al., 2017) or CPN (Chen et al., 2017) from *N* frames (and *M* views during training only) as an input to a small network with 1D dilated *temporal* convolutions and predicts 3D pose for either the middle (symmetric

convolutions) or the next frame (causal convolutions). A model with CPN backbone achieves a very good performance of *47 mm* MPJPE on Human3.6. The 1D convolutional network is tiny and runs very fast, achieving *real-time* performance given the 2D pose backbone is real-time (Pavllo et al., 2018).

- **EpipolarPose:** infers 3D posse from single images using a fully *self-supervised* approach, but it is trained with multi-view images. The network with a ResNet50 backbone, pretrained on the MPII dataset, outputs volumetric heatmaps, from which a 2D pose for two or more views is inferred. A 3D pose pseudo-GT is obtained with the help of polynomial triangulation, which is used as a supervision signal in a smooth L1 loss. The model uses body joints as calibration targets when a camera's extrinsic parameters are unknown, which is often the case. EpipolarPose (with refinement) achieves 61 mm MPJPE on Human3.6, which is an excellent result for an unsupervised model (Figure 27) (Kocabas et al., 2019).

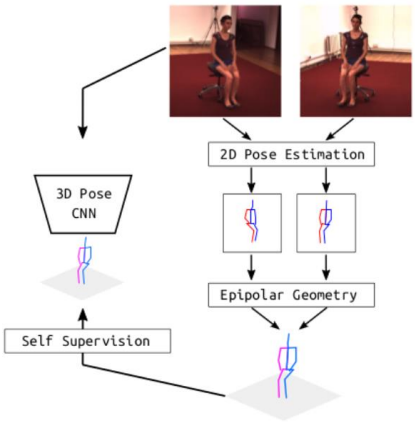


Figure 27: EpipolarPose overview (Kocabas et al., 2019).

- **3D Human Pose Estimation in RGBD Images for Robotic Task Learning:** use as an input mono images *with depth* from Kinnect and directly regresses 3D pose with a *3D convolutional* network. Since there is no depth dimension in the Human3.6 dataset, it is hard to compare the results of the network. For training, the authors use a Multi View Kinnect Dataset (Figure 28) (Zimmermann et al., 2018).

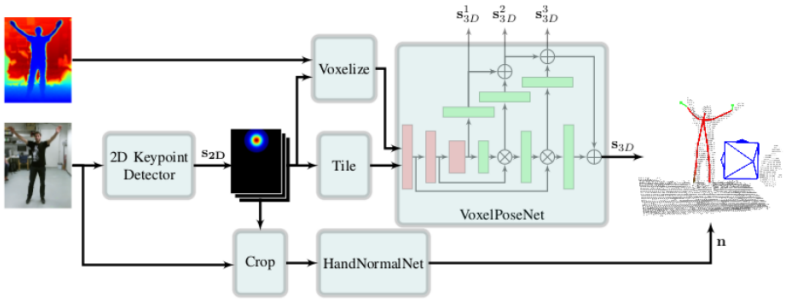


Figure 28: Network overview (Zimmermann et al., 2018)

3.9.2 Multi-person 3D Pose

The main challenge in multi-person 3D pose estimation is occlusions. In addition, unfortunately, there are almost no annotated multi-person 3D pose datasets like the Human3.6 dataset. Most multi-person datasets either do not have good GT or are not realistic:

- **MuPoTS-3D Dataset**: test set with GT from marker less motion capture (Mehta et al., 2017).
- **MuCo-3DHP Dataset**: synthesized dataset (Mehta et al., 2017).
- **Panoptic Dataset:** 480 calibrated cameras, no GT (*CMU Panoptic Dataset*, 2020).
- **Single Shot Multi Person 3D Pose Estimation from Monocular RGB:** introduce a single shot multi person 3D pose estimation model from a single image. Occlusion Robust Pose Maps (ORPM) is used by the authors that allows whole body pose estimation even when there are partial occlusions. "ORPM outputs a fixed number of maps, which encode the 3D joint locations of all people and 3D pose for an arbitrary number of people is inferred using body part associations" (Mehta et al., 2017). The backbone is Resnet50. MuCo-3DHP dataset is used for training and MuPoTs-3D for evaluation with 70% detection accuracy (3DPCK within a 15 cm ball) reported. Performance on Human3.6 is reported as well: 70 mm MPJPE (Figure 29) (Mehta et al., 2017).

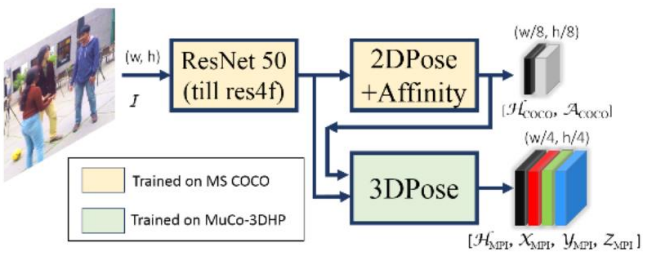


Figure 29: Network overview by (Mehta et al., 2017).

- **LCR-Net++:** Form single images, 2D and 3D human pose joints estimated simultaneously by this model which has Localization Classification Regression architecture (LCR-Net). “The main component is the pose proposal generator that proposes candidate poses at different locations in the image out of 100 anchor poses and a classifier scores the different pose proposals” (Rogez et al., 2018). As results the model gives poses, mixing hypotheses neighboring poses. The model is evaluated on the MuPoTs-3D dataset and reports 74% accuracy. Performance on Human3.6 is reported as well: 54 mm MPJPE (Figure 30) (Rogez et al., 2018).

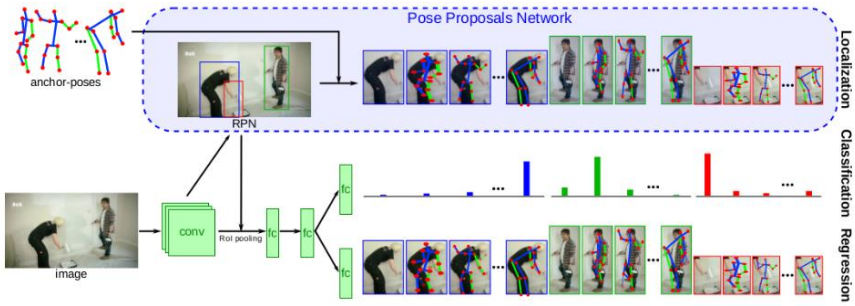


Figure 30: LCR-Net++ (Rogez et al., 2018).

- **Fast and Robust Multi-Person 3D Pose Estimation from Multiple Views:** use as an input *multi-view* images and estimates first multi-person 2D poses from CPN (Chen et al., 2017) in every view. Matching detected persons across multiple views is done by calculating affinity scores by using appearance similarity (Euclidean distance between descriptors from a pretrained re-ID network) and geometric compatibility (point-to-line distance between a joint and a corresponding epipolar line). 3D poses can be achieved by triangulation poses of the same person from different views, however, 2D poses are not errorless. In order to fix this issue, 3D Pictorial Structure (3DPS) model is used. The reported performance on the Campus dataset is 96 percent of correctly estimated parts (PCP) and on the Shelf dataset 97 PCP. Panoptic dataset is used for qualitative evaluation. It is real-time (without 3DPS) on GTX 1080Ti (Figure 31) (Dong et al., 2019).

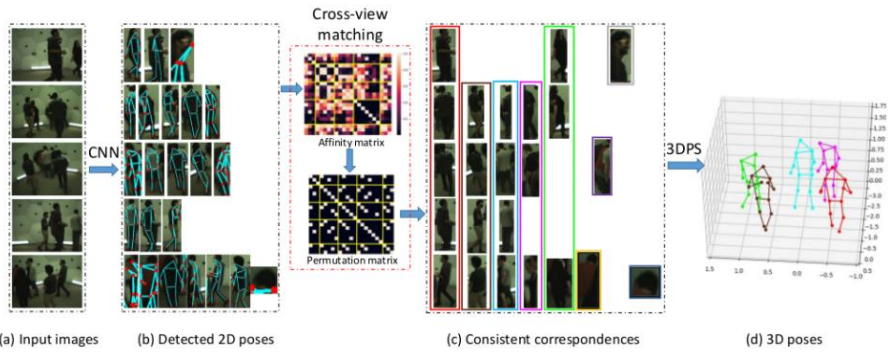


Figure 31: Network overview by (Dong et al., 2019).

- **Real time marker less multi human 3D pose estimation in RGB Depth camera networks:** estimate 3D poses from *multi view* images with *depth* from calibrated Kinnect v2 cameras. An input to the network is a multi-person 2D poses from OpenPose (Cao et al., 2018) for every view, which is lifted to 3D by incorporating the depth information. Views from multiple cameras are then fused together through multiple Neutral Kalman Filters. The authors recorded and annotated their own dataset. The model runs in real-time (Figure 32) (Carraro et al., 2017).

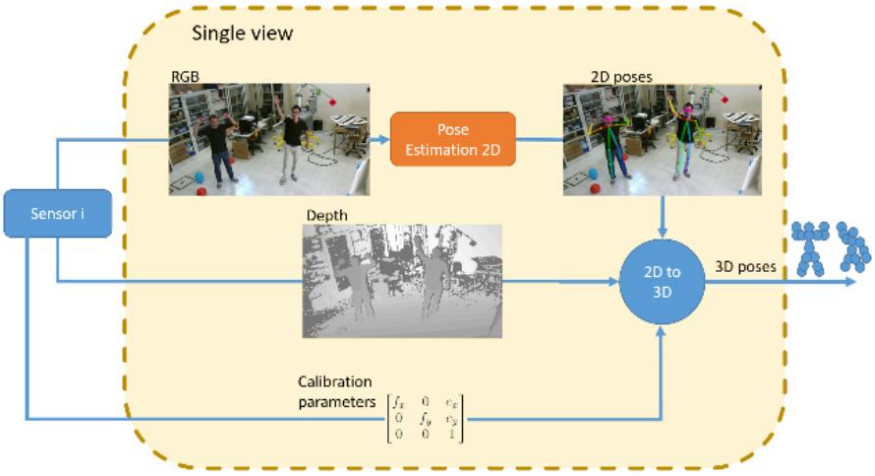


Figure 32: Network overview by (Carraro et al., 2017).

4 PROPOSED METHODOLOGY

This chapter is divided in two part. First, development of a visual tracking algorithm capable of tracking multi-person using deep learning techniques. Second, a solution developed for solving the 3D multi-person tracking is explained.

4.1 MULTI-PERSON TRACKING MODULE

To achieve this task the selected tracking algorithm is DeepSORT (Wojke et al., 2018) which the standard method used in it is tracking by detection. DeepSORT is an online tracking algorithm that means it can employ past and present data to make prediction related to the current frame. it is one of popular and widely used, object tracking framework and it is derived from SORT (Bewley et al., 2016). In following going through different parts of the DeepSORT.

4.1.1 Detections

4.1.1.1 RefineDet

In original implementation of DeepSORT, for detection part of input frames it used Faster R-CNN network. However, in this work RefineDet (Zhang et al., 2018) is used as object detector algorithm. It inherits One stage and Two stage approaches and overcome their limitations. The whole network of RefineDet contains two interconnected modules see Figure 33, the anchors refinement module (ARM) and the object detection module (ODM). Mentioned modules are connected by a transfer connection block (TCB) to transfer and enhance features from the former module in order to better predict objects in the latter module. RefineDet like SSD (Liu et al., 2016), is generate constant number of bounding boxes and for each bounding box is with score to shows likelihood of occurrence of that target class. The ODM goals to regress correct object positions and based on the refined anchors, predict multiple class labels. Whereas, The ARM aims are including decrease search space for the classifier and offer better initialization for the regressor by remove negative anchors and adjust the sizes and locations of anchors, respectively.

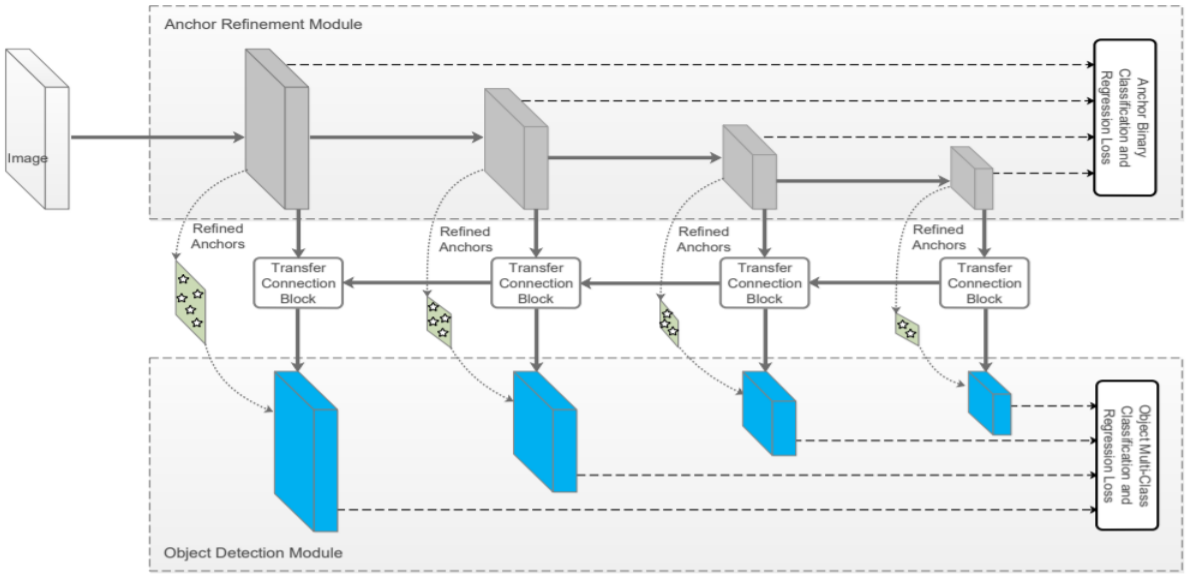


Figure 33: RefineDet architecture (Zhang et al., 2018).

4.1.1.2 Fine-tuned RefineDet

One way to increase performance of DeepSORT or in general object trackers is to improve Detector part by using Detector algorithms with higher accuracy or fine-tune it. For that reason this work RefineDet is fine-tuned with custom dataset. Custom dataset contains around 1000 soccer images that for each image, players and soccer ball(s) are labeled by hand by using labelImage (*LabelImg*, 2017) see Figure 34 and Figure 35 , dataset collected from websites like Google and YouTube .In custom dataset is tried to collect images with different viewpoints. In other words, images that captured by different distances from pitch. Majority of soccer images inherently includes occlusion which fine tuning helps the Detector to detect better players and balls in this situation.

For transfer learning in RefineDet VOC0712Plus_refinedet_vgg16_512x512 pre-trained model (*RefineDet*, 2018) is used with input size 512 * 512 that trained with PASCAL VOC 2007 (Everingham et al., 2010) and PASCAL VOC 2012 (Everingham et al., 2015) datasets and with VGG_16 as backbone. Also, coco_refinedet_resnet101_512x512 pre-trained model that like previous model with input size 512 * 512 is used. In this model used RESNET_101 as backbone network and trained with MS COCO data set (Lin et al., 2014). In addition, The solver.prototxt file which is a configuration file used to tell caffe framework to how the network trained, configured

as Figure 36 and parameters are same as original training parameters of RefineDet implementation.

Figure 34: Proposed dataset.

Figure 35: Proposed dataset.

```
train_net: "models/VGGNet/VOC0712Plus/refinedet_vgg16_512x512_ft/train.prototxt"
base_lr: 5e-05
display: 10
max_iter: 160000
lr_policy: "multistep"
gamma: 0.2
momentum: 0.9
weight_decay: 0.0005
snapshot: 10000
snapshot_prefix: "models/VGGNet/VOC0712Plus/refinedet_vgg16_512x512_ft/VOC0712Plus_refinedet_vgg16_512x512_ft"
solver_mode: GPU
device_id: 0
debug_info: false
snapshot_after_train: true
average_loss: 10
stepvalue: 120000
stepvalue: 160000
iter_size: 1
type: "SGD"
```

Figure 36: solver.prototxt file configuration

4.1.2 Kalman Filter

The idea behind of Kalman filter is to arrive at a best guess of the current state by using the available detections and last predictions, while taking account the likelihood of errors in the process.

Kalman filter is an important component in DeepSORT. On each bounding box, Kalman filter is applied. Predict and Update functions are called after the association step is done which is assign IDs to bounding boxes. These functions do the math which calculating state mean and covariance. Each of this state contains 8 variables related to center of bounding boxes, height of image, aspect ratio and velocities. The prior state used by Kalman filter to predict a good fit for bounding boxes.

4.1.3 Assignment

The Hungarian algorithm is a conventional method to solve the association between the predicted Kalman states and newly arrived objects. Into this problem formulation DeepSORT mix motion and information related to appearance over combination of two proper metrics. To integrate motion info, it used the Mahalanobis distance between newly arrived measurements and predicted Kalman states.

4.1.4 Deep Appearance Descriptor

A custom residual CNN extract combined visual information. As shown in (Figure 37) there are red, yellow, blue and green blocks. The first one is simple convolutional layers, the second one is the max pooling layer, the third ones are residual blocks with 3 convolutional layers each one and

the last one is a fully connected layer with batch and L2 normalization. Each block output size can be seen in parenthesis (Ciaparrone et al., 2019). “The green block offers vector of appearance feature for the given detected bounding box. After it trained, just needed to pass all detected bounding box from the image to this network and get the “128 X 1” dimensional feature vector” (Ciaparrone et al., 2019).

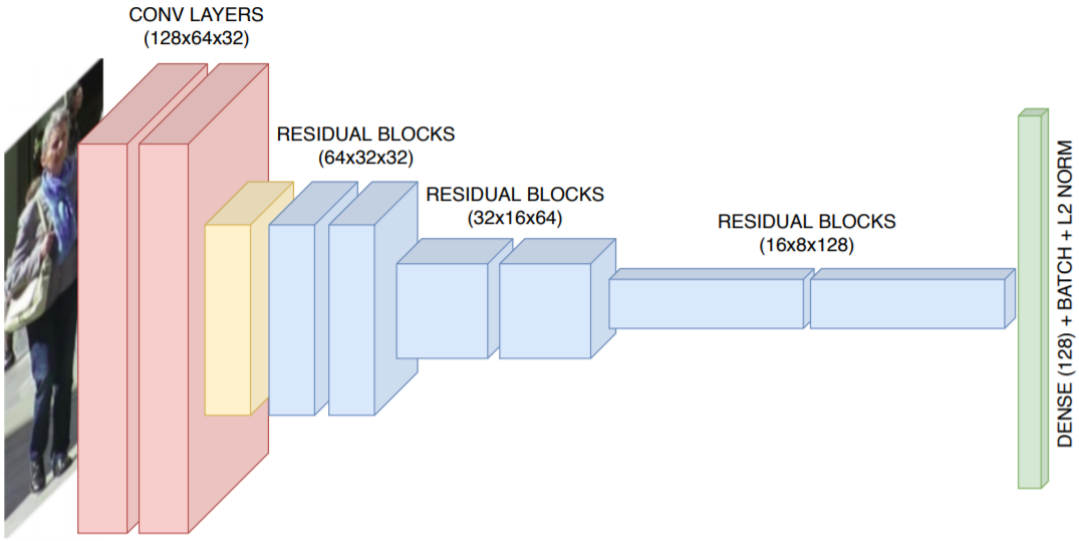


Figure 37: Deep Appearance Descriptor network (Ciaparrone et al., 2019).

4.2 3D MULTI-PERSON TRACKING

To implement 3D tracker, as it can be seen its diagram in Figure 38. Explained 2D tracker without change that introduced in last section is combined with a 3D multi human pose estimator from a single RGB image. Bounding boxes which generated by Detector for each frame, plus camera intrinsic parameters, i.e. Optical center and Focal length are sent to 3D multi-person pose estimator. Then generated 3D poses first convert again to 2D bounding boxes in order to send to the 2D tracker to have IDs, After that each ID with bounding box with Intersection over union technique re-assign to its 3D pose that in the end for each frames there are tracked 3D poses.

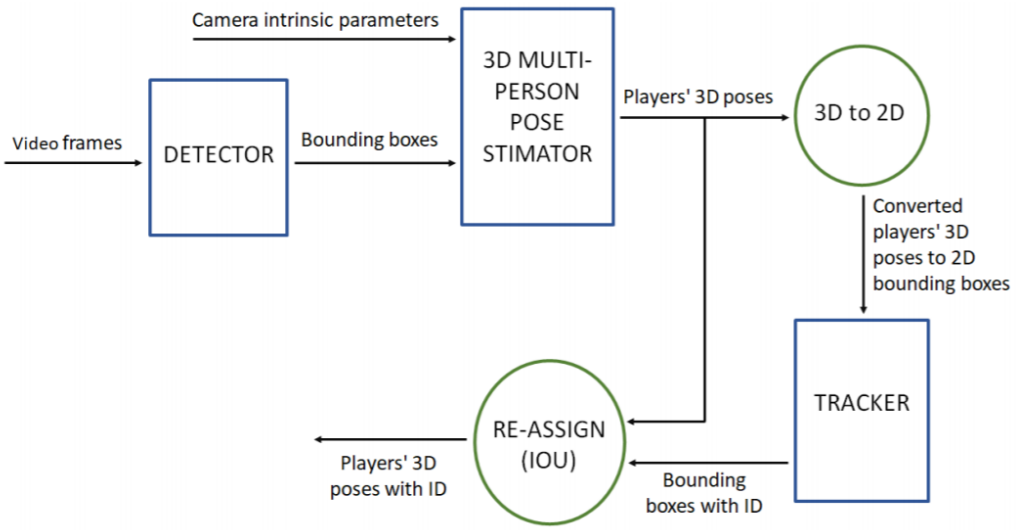


Figure 38: Proposed 3D Multi-person Tracker.

4.2.1 3D Multi-person Pose Estimator

To accomplish this Camera Distance Approach for 3D Human Pose Estimation (Moon et al., 2019) is used. Moon's approaches compute joints of multiple person $\{P^{abs}_j\}^J_j$ =1 that are absolute distance camera coordinates. It is a top down approach and contains two main components, DetectNet, RootNet and PoseNet.

The DetectNet is a human detector which in his work RefineDet is used. Human detector's results fed to the RootNet, then it job is to find location of the human roots R = (x_R, y_R, Z_R), which x_R and y_R are pixel coordinates, and Z_R is an absolute camera distance to object. PoseNet also, received same detector results as RootNet received, that estimate relative 3D poses of each detected bounding box P^{rel}_j = (x_j, y_j, Z^{rel}_j), it should be consider that depth value here is relative.

The same cropped human image is fed to the PoseNet, which estimates the root-relative 3D pose Finally, it converts Z^{rel}_j into Z^{abs}_j by adding Z_R and transform x_j and y_j to the original input image space see Figure 39.

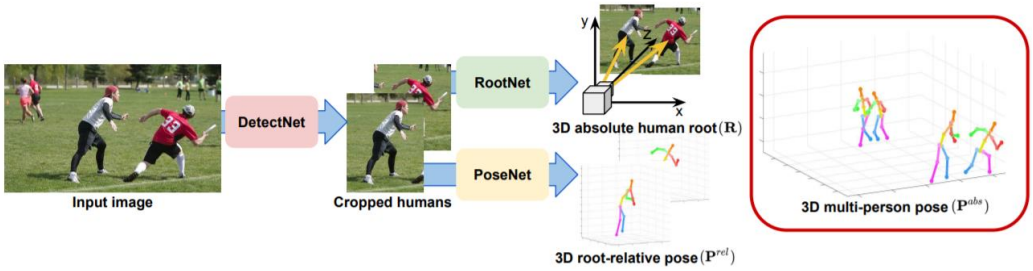


Figure 39: Moon's methods workflow (Moon et al., 2019).

4.2.1.1 RootNet

RootNet's architecture includes three parts as it can be seen in Figure 40. ResNet (He et al., 2016) as backbone network received cropped image from detector and extract useful features. Then for the second part, "the 2D estimation part receives a feature map from the backbone and up samples it uses three consecutive deconvolutional layers with batch normalization layers and ReLU activation function" (Moon et al., 2019). Next, to produce roots' 2D heatmap, 1-by-1 convolution is applied. Using the 2D heatmap, 2D images coordinates (x_R, y_R) can be extracted by soft-argmax. The last part is the depth estimation, "it takes a feature map from the backbone part and applies global average pooling. Then, the pooled feature map goes through a 1-by-1 convolution, which outputs a single scalar value ϒ" (Moon et al., 2019).

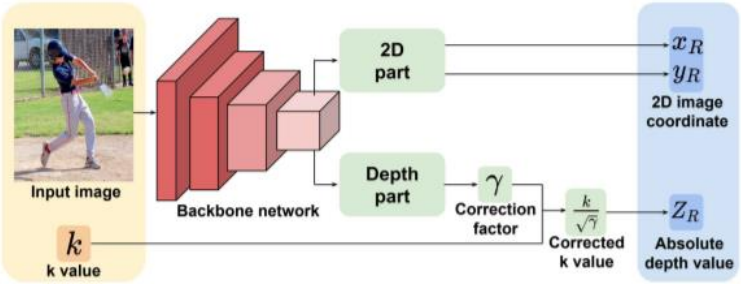


Figure 40: Network architecture of the RootNet (Moon et al., 2019).

4.2.1.2 PoseNet

The PoseNet job is to estimate the root-relative 3D pose $P^{rel}_j = (x_j, y_j, Z^{rel}_j)$ from a detector's results. To doing so Sun (Sun et al., 2017) model is used. Sun's model has two part. Backbone, ResNet (He et al., 2016) extract useful features. Second part, the pose estimation part received a feature map and up samples it with three consecutive deconvolutional layers by batch normalization layers and ReLU activation function. To produce the 3D heatmaps, for each joint A 1-by-1 convolution is applied to the up sampled feature map. The soft-argmax operation is used to extract the 2D image coordinates (x_j, y_j), and the root-relative depth values Z^{rel}_j (Moon et al., 2019).

5 RESULTS AND DISCUSSION

In this chapter, the quality of the 2D and 3D multi-person tracker modules is characterized. Because lack of proper benchmark dataset for sport for both Detector and 2D Tracker modules evaluated by custom dataset. Fine-tuned Detector and 2D tracker will be evaluated with manually labeled dataset and visual evaluation. Then, 3D tracker will evaluate with observation results.

5.1 DETECTOR

To evaluate fine-tuned detector, common metrics in the evaluation of object detection is used. Due to lack of proper available sport datasets with annotation for evaluation, evaluation performed on dataset as mentioned before that manually labeled. Test set ground truth consists of 80 soccer images similar to train set which distance of the camera from the field is not too close, as it can be perceived in Figure 41 with total 1233 bounding boxes as players(person) and 100 sport balls. Fine-tuned detector achieved overall 64.94% in mean Average Precision (mAP), however, just for person this metric is 91% see Figure 42.

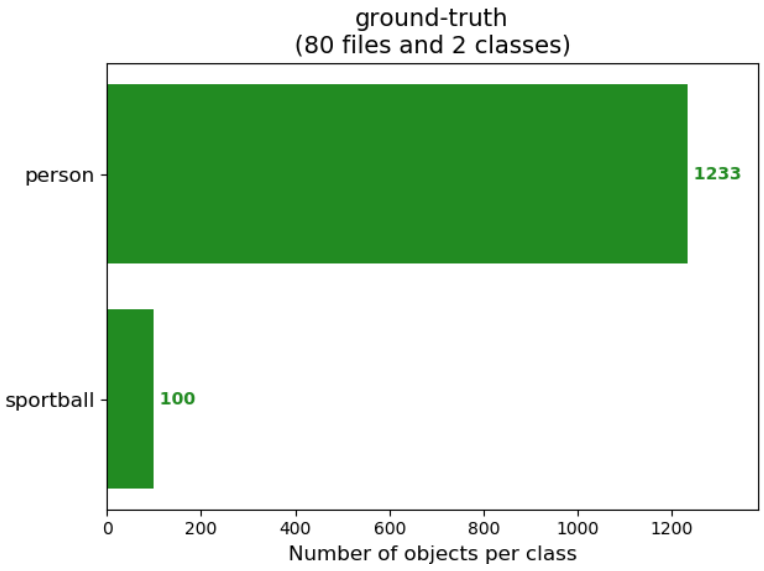


Figure 41: Test set information.

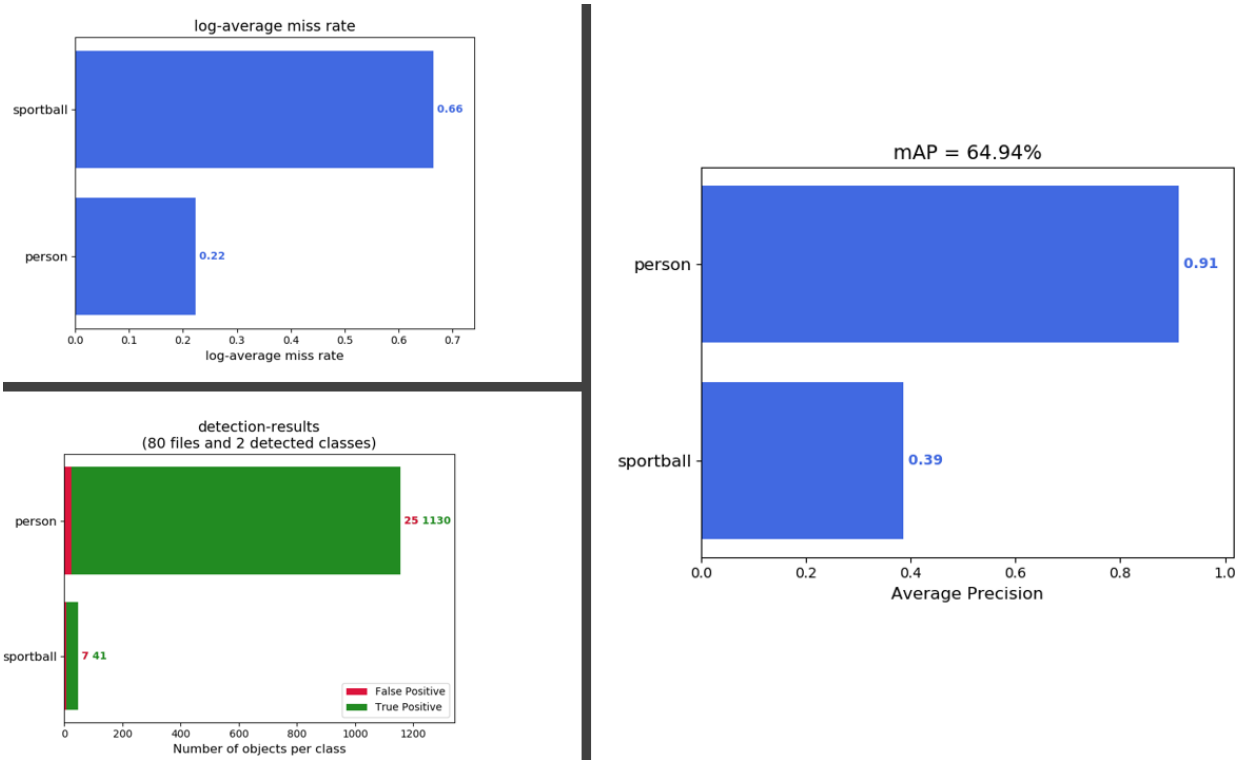


Figure 42: Evaluation results.

For analyzing results of fine-tuned detector, also its results with pre-trained VOC0712Plus_refinedet_vgg16_512x512 is compared which is as follows for different scenario.

As it shown in Figure 43 some cases of fine-tuned network has slightly higher detection score than pre-trained.

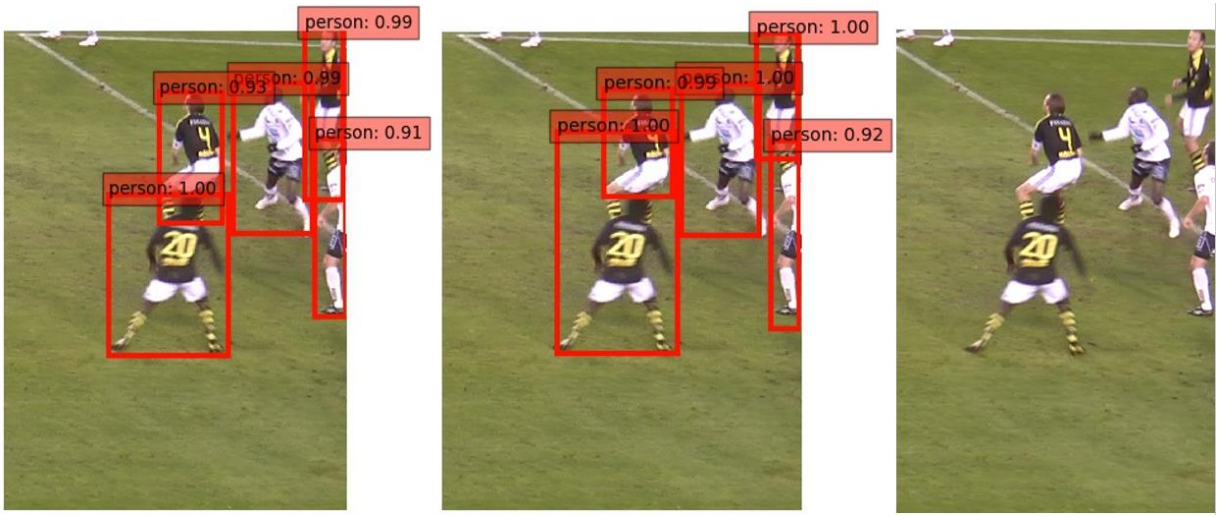


Figure 43: pre-trained model (Left), Fine-tuned model (Middle).

In some other cases fine-tuned detected a greater number of players than pre-trained model see (Figure 44) and vice versa see Figure 45.

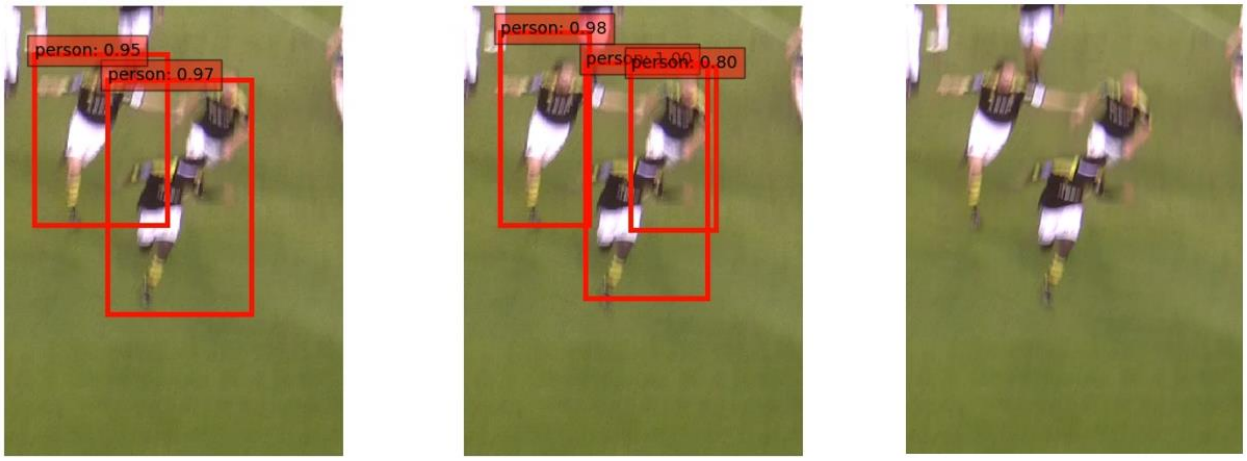


Figure 44: pre-trained model (Left), Fine-tuned model (Middle).

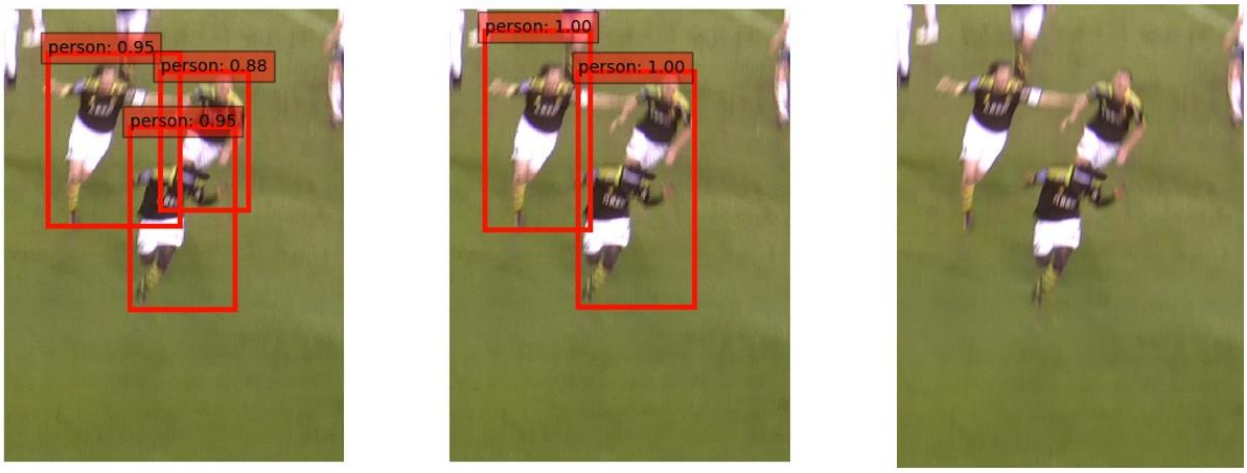


Figure 45: pre-trained model (Left), Fine-tuned model (Middle).

In addition, in some cases shown in Figure 46 fine-tuned model predict some incorrect detected bounding boxes in compare with pre-trained model. However, by increasing threshold to select detections with high score solve it in majority of cases.

One of the main problems of fine-tuned model is that performance of the prediction decreases dramatically when background of the image is not green. In other words, for image with non-green background it does not work properly see Figure 47.

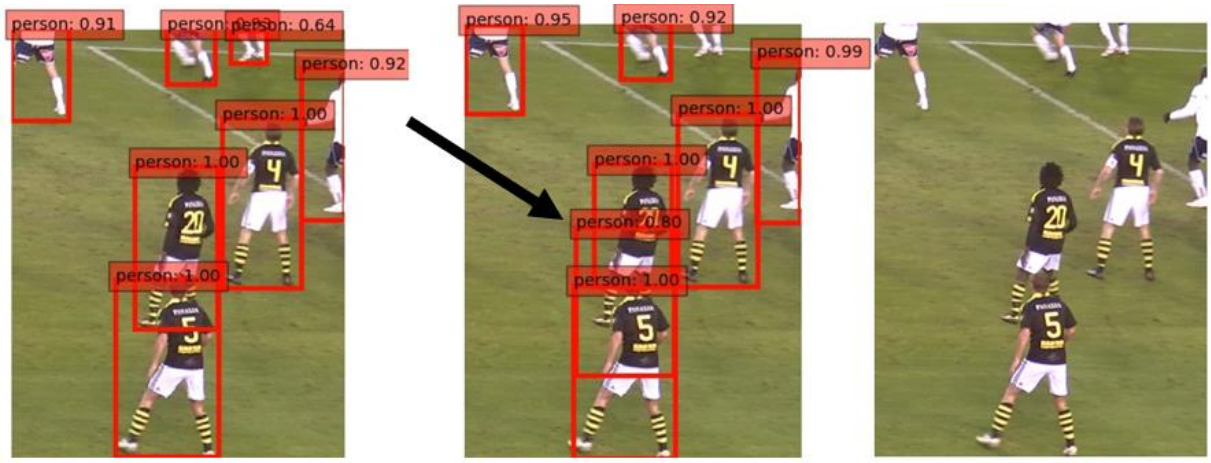


Figure 46: per-trained model (Left), fine-tuned model with incorrect detection (Middle).

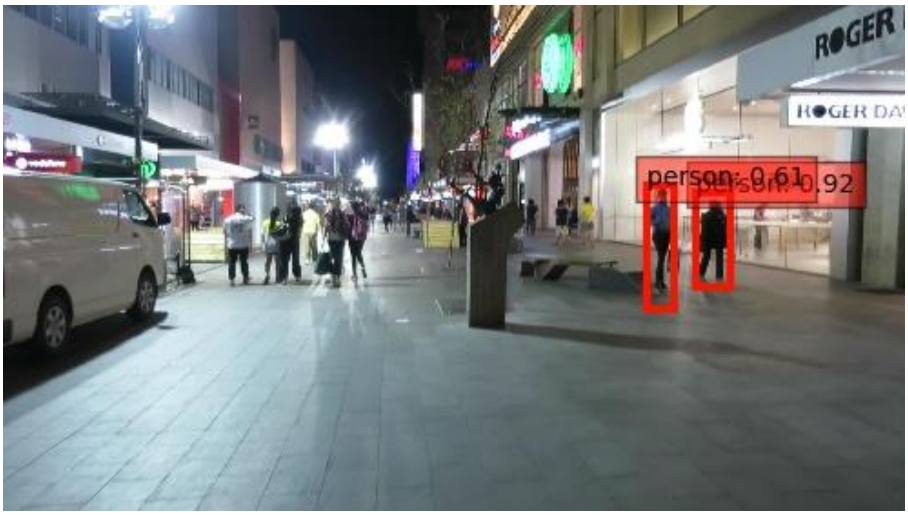


Figure 47: fine-tuned model result image from (Milan et al., 2016).

5.2 2D TRACKER

This section 2D Tracker will be evaluated based on different situation, however as mentioned before this will be evaluated by custom dataset due to lack of standard soccer benchmark for 2D multi-person tracking.

For evaluate 2D Tracker, 5 seconds of a soccer dataset (*Fujii Lab's Datasets*, 2020) which include 150 frames and labeled with CVAT annotation tool (Nikita Manovich, 2020) in MOT benchmark format and performed by using py-motmetrics library (Valmadre, 2020) . As mentioned in section 2, the performance evaluation for Multi-Object Tracking algorithms is not so simple as the one presented for object detection. In order to have a better vision regards to the performance of

the Tracker, as it can be seen in Figure 48 DeepSORT and SORT evaluation results on soccer dataset (*Fujii Lab's Datasets*, 2020) with same detections results are presented. To compare SORT and DeepSORT evaluation results different class of metrics can be considered like accuracy, precision and completeness, accuracy as it mentioned before this type of metrics tries to measure how accurately a tracking algorithm tracks targets. The IDs metric measures the ID switches, i.e. given an ID for an object it measures how many times the MOT algorithm changes this ID. Here number of ID switches in SORT (17) is more than two times than DeepSORT (7) see Figure 48, it can say one of the main disadvantage of the SORT is large number in amount of the IDs metric. The other metric for measure accuracy is the Multiple Object Tracking Accuracy or MOTA (Equation 1) that SORT (0.88) reached slightly higher than DeepSORT (0.83), however, with lager amount of dataset this results likely would change to more accurate one. In precision metrics group the key factor is the description of the precision that the tracked objects have using criteria such as bounding box overlap or distance. The most important is MOTP (Equation 2) uses a ratio with the distance among the ground-truth detections locations and the associated detected locations, as it can be understood in Figure 48 MOTP for DeepSORT and SORT are 0.31 and 0.23, respectively. For completeness that refers to how completely the ground truth trajectories are tracked. This set includes the results from Mostly Tracked (MT), Partly Tracked (PT), Mostly Lost (ML) and Fragmentation (FM), that for both methods numbers almost same (Figure 48) except Fragmentation, which following in this section it will be discussed. In addition, in order to receive more accurate evaluation, it is need larger chunk of data.

	idf1	idp	idr	Recall	Precision	GT	MT	PT	ML	FP	FN	IDs	FM	MOTA	MOTP	IDt	IDa	Idm
DeepSORT	0.826	0.832	0.820	0.911	0.924	20	17	3	0	169	201	7	49	0.833	0.314	7	0	0
SORT	0.788	0.817	0.761	0.911	0.978	20	18	2	0	46	201	17	33	0.883	0.233	4	13	0

Figure 48: 2D Tracker Evaluation Result in different Methods.

One of the main problems of object trackers is identity switches, that mainly happened because of occlusion and DeepSORT by using deep appearance descriptor approach extract person information, however in sport because appearance of players is same, descriptor cannot work perfectly. Introduced DeepSORT implementation generally works well in occlusion condition, as it can be seen in (Figure 49) players with IDs 18 and 21 in right image after occlusion with player

with ID 5, still have same IDs in left image, or in other case seen in the plyers with IDs 11 and 16 even with same appearance after occlusion have same IDs (Figure 50). But in some cases, like in Figure 51 players even with different appearance, in this case white and blue t-shirt identity switches happened.

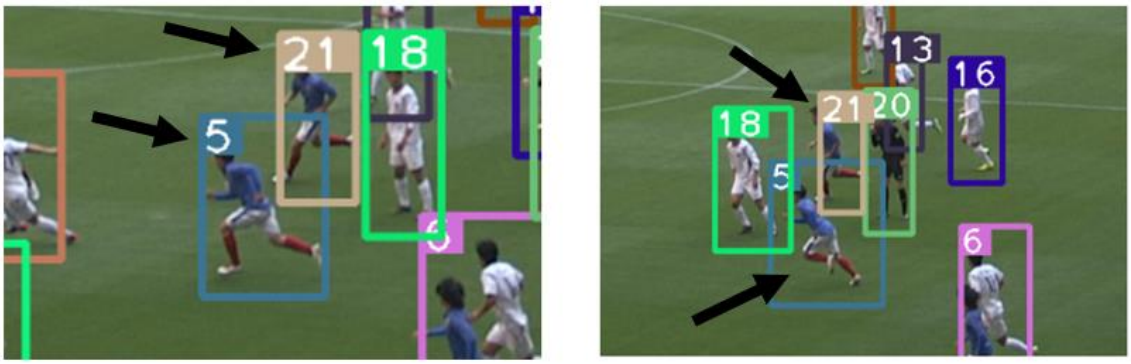


Figure 49: After Occlusion (Left), Before Occlusion (Right) images from (*Fujii Lab's Datasets*, 2020).

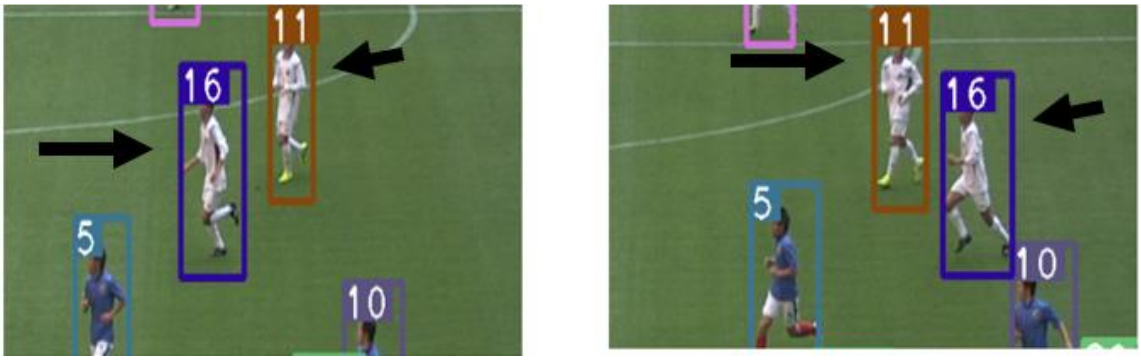


Figure 50: After Occlusion no identity switches (Left), Before Occlusion (Right) images from(*Fujii Lab's Datasets*, 2020).

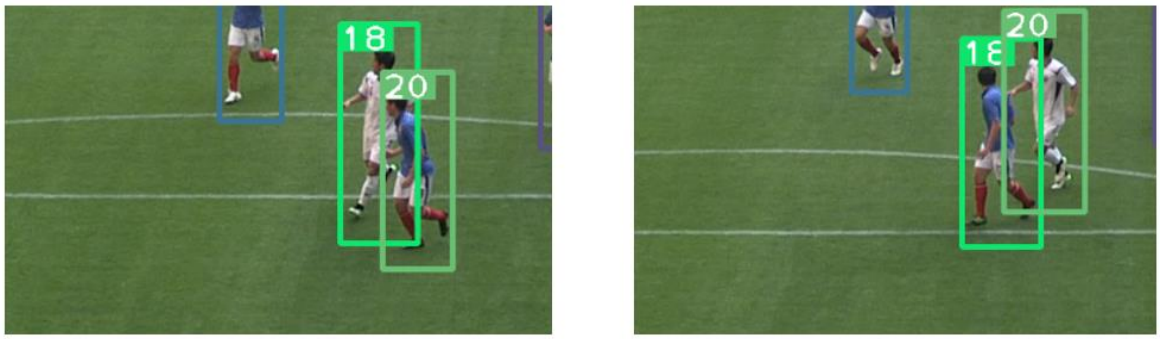


Figure 51: Identity Switches After (Left), Before (Right) images (*Fujii Lab's Datasets*, 2020).

One usual problem between online algorithm like DeepSORT is the higher number of fragmentations. This occurs because, when occlusions happen or detections are missing, online

methods cannot look forward in the video, re-identify the lost targets and interpolate the missing part of the trajectories. For an example, in Figure 52 as it shows, while DeepSORT is capable to re-identify lost target after occlusion, it is incapable to track it while the target is not visible, and this results in a fragmentation.

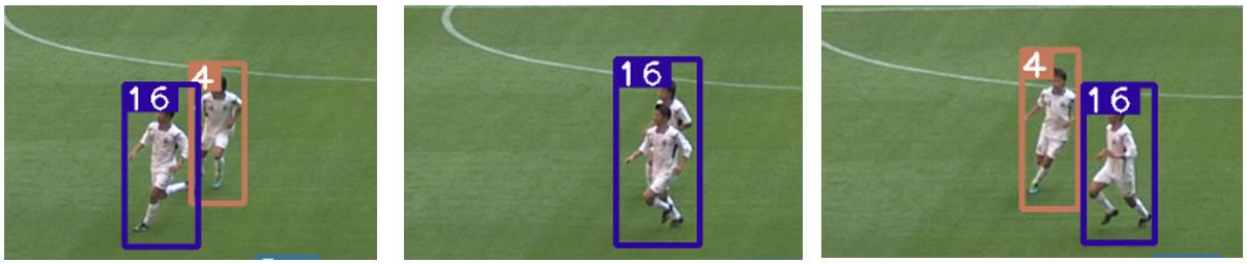


Figure 52: (left) after occlusion, (middle) during occlusion, (right) before occlusion images from (*Fujii Lab's Datasets*, 2020).

Number of Tracks in whole video frame is still high. For an example, DeepSORT and SORT (Bewley et al., 2016) algorithms are run in whole frames of one camera view from (*Fujii Lab's Datasets*, 2020) dataset that has 300 frames to compare number of final tracks. And as it can be seen in (Figure 53) for SORT right image number of labels reach to 68 which is quietly high, however in DeepSORT left image, highest label is 29 in last frame, almost less than half of the SORT algorithm. Eventually, the final performance of the algorithm is affected by the accuracy of bounding boxes and by using more accurate Detector can boost the performance 2D Tracker.

Figure 53: (left) DeepSORT output, (right) SORT output images from (*Fujii Lab's Datasets*, 2020).

5.3 3D TRACKER

Proposed 3D multiple-person tracker, as mentioned in last chapter, its pipeline contains Detector, RootNet, PoseNet and Tracker. RootNet play important role in this chain to estimate absolute depth of each person form single image and its accuracy affects the pipeline's performance. After implementation and visualization 3D and 2D results of 3D Tracker it was realized, suggested RootNet module is not accurate enough and there is depth estimating error between frames which position player (person) in 3D are erratic for different frames. As it can be seen in Figure 54 the visualized 3D tracks results by Unity that one player in continues frames as it marked by red circle is appeared as frozen and the result are not continuous and not fluid. This happen because it is quite hard for RootNet to predict robust absolute depth from a single RGB image. And by extent current RootNet module using multi-view or implement different algorithms in order to estimate absolute depth would be options to fix this error.

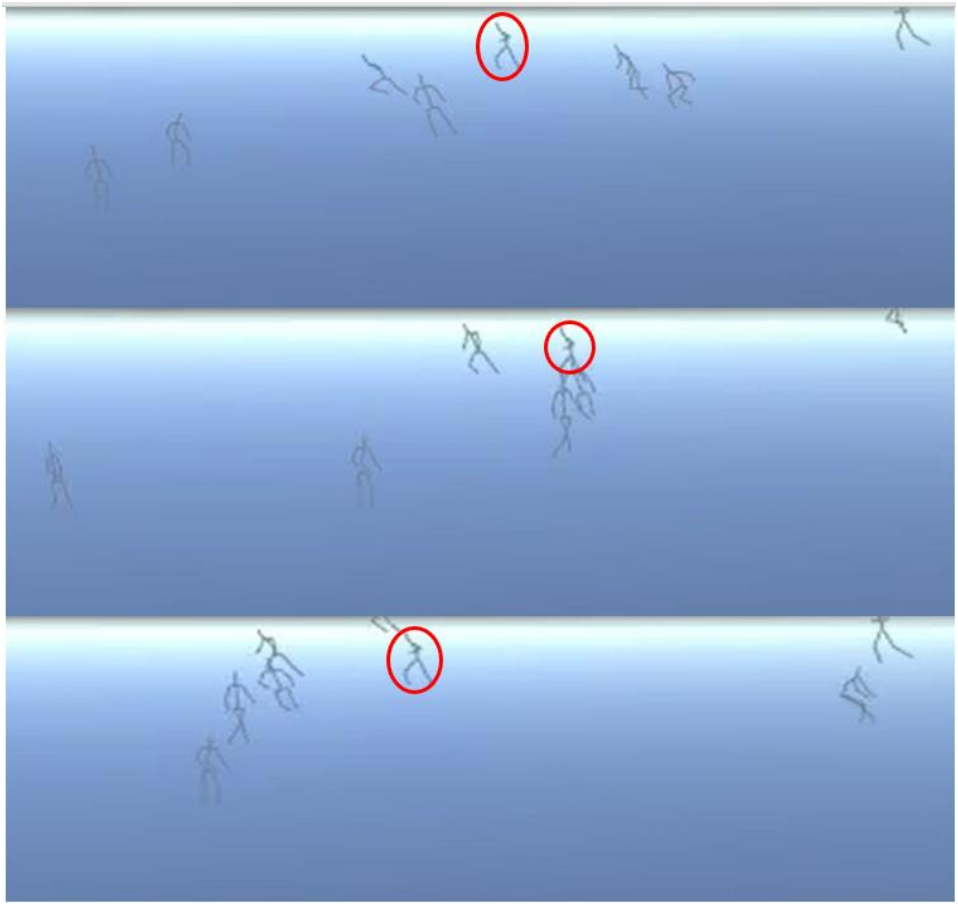


Figure 54: RootNet's error for contioues frames.

However, the 2D projection of the 3D poses are looks promising see Figure 55, but in some occlusion condition 3D pose estimator cannot properly estimates poses of players which results are incorrect see Figure 56. Again, as most of the problem, this one also can be partially fixed by using more accurate Detector. In addition, estimated root-relative 3D human poses, extremely depends on the accuracy of the PoseNet. Which means, by implementing more accurate 3D pose estimator it can be gained more accurate results.

Figure 55: 2D projection of 3D poses images from (*Fujii Lab's Datasets*, 2020).

Figure 56: 3D Pose Estimator Module Error images from (*Fujii Lab's Datasets*, 2020).

6 CONCLUSIONS

This chapter summarizes the main contributions of this work. Possible lines of future work are also outlined.

This master thesis studied the use of deep learning techniques to build a multi-object tracking system using the tracking-by-detection scheme. To solve this task, it was implemented a modular application composed of a neural network module as Detector and a Tracker module. The first module provides object detections using neural network models. These detections are handled by the Tracker module to track objects and increase the accuracy by using neural network Deep Appearance Descriptor. It may also help to adapt the tracking processing speed to the hardware on which it is being run. Next, in another work it is targeted, 3D multi-person poses tracking problem from single view by designing a pipeline includes different modules as follows, Detector modules, RootNet module, PoseNet module and Tracker module.

In future, this work can be extended in multiple directions. One, this 2D Tracker can be employed in multi-view object tracking in order to track multi-object from different cameras. Second, employing 3D Tracker and add multi-view factors to it, in order to achieve more accurate result which then can be used in many different fields like sport analysis. Third, by extending current introduced soccer dataset, to train or fine-tuning other object detector algorithms.

7 REFERENCES

Babenko, B., Yang, M.-H., & Belongie, S. (2009). *Visual Tracking with Online Multiple Instance Learning*.

Baker, S., & Matthews, I. (2004). *Lucas-Kanade 20 Years On : A Unifying Framework*. *56*(3), 221–255.

Bernardin, K., & Stiefelhagen, R. (2008). Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *EURASIP Journal on Image and Video Processing 2008 2008:1*, *2008*(1), 1–10. https://doi.org/10.1155/2008/246309

Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. *Proceedings - International Conference on Image Processing, ICIP*, *2016-Augus*, 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003

Bridgeman, L., Volino, M., Guillemaut, J.-Y., & Hilton, A. (2019). Multi-person 3D Pose Estimation and Tracking in Sports. *Computer Vision and Pattern Recognition*, 0–0.

Briechle, K., & Hanebeck, U. D. (2001). *<title>Template matching using fast normalized cross correlation</title>* (D. P. Casasent & T.-H. Chao (Eds.); pp. 95–102). https://doi.org/10.1117/12.421129

Cao, Z., Hidalgo, G., Simon, T., Wei, S., & Sheikh, Y. (2018). *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. *XXX*(Xxx). http://arxiv.org/abs/1812.08008

Cao, Z., Simon, T., Wei, S., & Sheikh, Y. (2016). *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. http://arxiv.org/abs/1611.08050

Carraro, M., Munaro, M., Burke, J., & Menegatti, E. (2017). *Real-time marker-less multi-person 3D pose estimation in RGB-Depth camera networks*. http://arxiv.org/abs/1710.06235

Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., & Sun, J. (2017). *Cascaded Pyramid Network for Multi-Person Pose Estimation*. http://arxiv.org/abs/1711.07319

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 1800–1807. https://doi.org/10.1109/CVPR.2017.195

Ciaparrone, G., Sánchez, F. L., Tabik, S., Troiano, L., Tagliaferri, R., & Herrera, F. (2019). *Deep Learning in Video Multi-Object Tracking: A Survey*. 1–42. https://doi.org/10.1016/j.neucom.2019.11.023

CMU Panoptic Dataset. (2020). http://domedb.perception.cs.cmu.edu/

Comaniciu, D., & Ramesh, V. (2000). Mean shift and optimal prediction for efficient object tracking. *Proceedings 2000 International Conference on Image Processing (Cat. No.00CH37101)*, 70–73. https://doi.org/10.1109/ICIP.2000.899297

Dai, J. (2016). *R-FCN : Object Detection via Region-based Fully Convolutional Networks*. *Nips*.

Dai, J., He, K., & Sun, J. (2015). *Instance-aware Semantic Segmentation via Multi-task Network Cascades*. http://arxiv.org/abs/1512.04412

Dataset Resources. (2020). http://alov300pp.joomlafree.it/dataset-resources.html

Dong, J., Jiang, W., Huang, Q., Bao, H., & Zhou, X. (2019). *Fast and Robust Multi-Person 3D Pose Estimation from Multiple Views*. http://arxiv.org/abs/1901.04111

Dubuisson, S., & Gonzales, C. (2016). A survey of datasets for visual tracking. *Machine Vision and Applications*, *27*(1), 23–52. https://doi.org/10.1007/s00138-015-0713-y

Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2015). The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, *111*(1), 98–136. https://doi.org/10.1007/s11263-014-0733-5

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, *88*(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4

Fang, H., Xie, S., Tai, Y., & Lu, C. (2016). *RMPE: Regional Multi-person Pose Estimation*. http://arxiv.org/abs/1612.00137

Fu, C.-Y., Liu, W., Ranga, A., Tyagi, A., & Berg, A. C. (2017). *DSSD : Deconvolutional Single Shot Detector*. http://arxiv.org/abs/1701.06659

Fujii Lab's Datasets. (2020). https://www.fujii.nuee.nagoya-u.ac.jp/multiview-data/

Girdhar, R., Gkioxari, G., Torresani, L., Paluri, M., & Tran, D. (2018). Detect-and-Track: Efficient Pose Estimation in Videos. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 350–359. https://doi.org/10.1109/CVPR.2018.00044

Girshick, R. (2015). Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 1440–1448. https://doi.org/10.1109/ICCV.2015.169

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 580–587. https://doi.org/10.1109/CVPR.2014.81

Harris, C., & Stephens, M. (1988). *A COMBINED CORNER AND EDGE DETECTOR*. 147–152. https://doi.org/10.5244/C.2.23

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-October*, 2980–2988. https://doi.org/10.1109/ICCV.2017.322

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(2), 386–397. https://doi.org/10.1109/TPAMI.2018.2844175

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Held, D., Thrun, S., & Savarese, S. (2016). *Learning to Track at 100 FPS with Deep Regression Networks*.

Hidalgo, G. (2018). *OpenPose*. Github. https://github.com/CMU-Perceptual-Computing-Lab/openpose

Huang, T. S. (1997). Computer Vision: Evolution and Promise. *Report*. https://doi.org/10.5170/CERN-1996-008.21

Hui, J. (2018). *Mean Average Precision for Object Detection*. https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

Intelligence, M., & Intelligence, M. (2004). *UvA-DARE (Digital Academic Repository) Fast Occluded Object Tracking by a Robust Appearance Filter IEEE Transactions on Pattern Analysis and Machine Intelligence Fast Occluded Object Tracking by a Robust Appearance Filter*. https://doi.org/10.1109/TPAMI.2004.45

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3. 6M. *Ieee Transactions on Pattern Analysis and Machine In TELLigence*, 1. https://doi.org/10.1109/TPAMI.2013.248

Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. (2019). A Survey of Deep Learning-based Object Detection. *IEEE Access*, *7*(3), 128837–128868. https://doi.org/10.1109/ACCESS.2019.2939201

Kalal, Z. (2010). *P-N Learning : Bootstrapping Binary Classifiers by Structural Constraints*.

Kocabas, M., Karagoz, S., & Akbas, E. (2019). *Self-Supervised Learning of 3D Human Pose using Multi-view Geometry*. http://arxiv.org/abs/1903.02330

Kratz, L., & Nishino, K. (2010). Tracking with local spatio-temporal motion patterns in extremely crowded scenes. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 693–700. https://doi.org/10.1109/CVPR.2010.5540149

Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Zajc, L. Č., Vojír, T., Häger, G., Lukežič, A., Eldesokey, A., Fernández, G., García-Martín, Á., Muhic, A., Petrosino, A., Memarmoghadam, A., Vedaldi, A., Manzanera, A., Tran, A., Alatan, A., … He, Z. (2017). The Visual Object Tracking VOT2017 Challenge Results. *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, *2018-January*, 1949–1972. https://doi.org/10.1109/ICCVW.2017.230

Krizhevsky, A., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *NIPS'12: Proceedings of the 25th International Conference on Neural Information Processing*

Systems - Volume 1, 1–9.

labelImg: LabelImg is a graphical image annotation tool and label object bounding boxes in images. (2017). Github. https://github.com/tzutalin/labelImg

Li, B., & Zhang, F. (2018). *SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks*.

Li, Yi, Qi, H., Dai, J., Ji, X., & Wei, Y. (2017). Fully convolutional instance-aware semantic segmentation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017-Janua*, 4438–4446. https://doi.org/10.1109/CVPR.2017.472

Li, Yuan, Huang, C., & Nevatia, R. (2009). *Learning to Associate : HybridBoosted Multi-Target Tracker for Crowded Scene*. 2953–2960.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014). 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. https://doi.org/10.1007/978-3-319-10602-1_48

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9905 LNCS*, 21–37. https://doi.org/10.1007/978-3-319-46448-0_2

Martinez, J., Hossain, R., Romero, J., & Little, J. J. (2017). A simple yet effective baseline for 3d human pose estimation. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 2659–2668. https://doi.org/10.1109/ICCV.2017.288

Mauthner, T., & Bischof, H. (2007). A Robust Multiple Object Tracking for Sport Applications. *Performance Evaluation for Computer Vision*.

Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., & Theobalt, C. (2017). *Single-Shot Multi-Person 3D Pose Estimation From Monocular RGB*. http://arxiv.org/abs/1712.03453

Milan, A., Leal-Taixe, L., Reid, I., Roth, S., & Schindler, K. (2016). *MOT16: A Benchmark for Multi-Object Tracking*. 1–12. http://arxiv.org/abs/1603.00831

Moon, G., Chang, J. Y., & Lee, K. M. (2019). Camera distance-aware top-down approach for 3D multi-person pose estimation from a single RGB image. *Proceedings of the IEEE International Conference on Computer Vision*, *2019-Octob*, 10132–10141. https://doi.org/10.1109/ICCV.2019.01023

Müller, M., Bibi, A., Giancola, S., Al-Subaihi, S., & Ghanem, B. (2018). TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11205 LNCS*, 310–327. http://arxiv.org/abs/1803.10794

Newell, A., Yang, K., & Deng, J. (2016). *Stacked Hourglass Networks for Human Pose Estimation*. http://arxiv.org/abs/1603.06937

Nikita Manovich. (2020). *Computer Vision Annotation Tool (CVAT)*. https://github.com/opencv/cvat

Pavllo, D., Feichtenhofer, C., Grangier, D., & Auli, M. (2018). *3D human pose estimation in video with temporal convolutions and semi-supervised training*. http://arxiv.org/abs/1811.11742

Pérez, P., Hue, C., Vermaak, J., & Gangnet, M. (2002). Color-based probabilistic tracking. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *2350*, 661–675. https://doi.org/10.1007/3-540-47969-4_44

Pinheiro, P. O., Collobert, R., & Dollar, P. (2015). *Learning to Segment Object Candidates*. 1–10. http://arxiv.org/abs/1506.06204

Prince, S. J. D. (Simon J. D. (2012). *Computer vision : models, learning, and inference*. Cambridge University Press.

Raj, B. (2019). *An Overview of Human Pose Estimation with Deep Learning*. https://medium.com/beyondminds/an-overview-of-human-pose-estimation-with-deep-learning-d49eb656739b

Redmon, J. (2018). *YOLOv3: An Incremental Improvement*.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*.

Redmon, J., & Farhadi, A. (2017). *YOLO9000: Better, Faster, Stronger*.

RefineDet. (2018). Github. https://github.com/sfzhang15/RefineDet

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

Ristic, B., Vo, B. N., Clark, D., & Vo, B. T. (2011). A metric for performance evaluation of multi-target tracking algorithms. *IEEE Transactions on Signal Processing*, *59*(7), 3452–3457. https://doi.org/10.1109/TSP.2011.2140111

Rogez, G., Weinzaepfel, P., & Schmid, C. (2018). *LCR-Net++: Multi-person 2D and 3D Pose Detection in Natural Images*. 1–15. https://doi.org/10.1109/TPAMI.2019.2892985

Sadeghian, A., Alahi, A., & Savarese, S. (2017). *Tracking The Untrackable : Learning to Track Multiple Cues with Long-Term Dependencies*.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2014). Overfeat: Integrated recognition, localization and detection using convolutional networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.

Shi, J., & Tomasi, C. (1993). *Good Features to Track*. *December*.

Simonyan, K., & Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 1–14. http://arxiv.org/abs/1409.1556

Smeulders, A. W. M., Member, S., Chu, D. M., Member, S., Cucchiara, R., Member, S., Calderara, S., & Member, S. (2014). Visual Tracking : An Experimental Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(7), 1442–1468. https://doi.org/10.1109/TPAMI.2013.230

Song, B., Jeng, T.-Y., Staudt, E., & Roy-Chowdhury, A. K. (2010). A Stochastic Graph Evolution Framework for Robust Multi-target Tracking. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 6311 LNCS* (Issue PART 1, pp. 605–619). Springer Verlag. https://doi.org/10.1007/978-3-642-15549-9_44

Sun, X., Xiao, B., Wei, F., Liang, S., & Wei, Y. (2017). *Integral Human Pose Regression*. http://arxiv.org/abs/1711.08229

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-ResNet and the impact of residual connections on learning. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 4278–4284.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June*, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Valmadre, J. (2020). *Metrics for Benchmark multiple object trackers (MOT) in Python*. https://github.com/cheind/py-motmetrics

Voigtlaender, P., Krause, M., Osep, A., Luiten, J., Sekar, B. B. G., Geiger, A., & Leibe, B. (2019). Mots: Multi-object tracking and segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*, 7934–7943. https://doi.org/10.1109/CVPR.2019.00813

Wang, Z., Zheng, L., Liu, Y., & Wang, S. (2019). *Towards Real-Time Multi-Object Tracking*. http://arxiv.org/abs/1909.12605

Wojke, N., Bewley, A., & Paulus, D. (2018). Simple online and realtime tracking with a deep association metric. *Proceedings - International Conference on Image Processing, ICIP*, *2017-Septe*, 3645–3649. https://doi.org/10.1109/ICIP.2017.8296962

Wu, Y., Lim, J., & Yang, M. H. (2013). Online object tracking: A benchmark. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2411–2418. https://doi.org/10.1109/CVPR.2013.312

Yamaguchi, K., Berg, A. C., Ortiz, L. E., & Berg, T. L. (2011). Who are you with and where are you going? *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1345–1352. https://doi.org/10.1109/CVPR.2011.5995468

Zhang, S., Wen, L., Bian, X., Lei, Z., & Li, S. Z. (2018). Single-Shot Refinement Neural Network for Object Detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4203–4212. https://doi.org/10.1109/CVPR.2018.00442

Zhou, X., Huang, Q., Sun, X., Xue, X., & Wei, Y. (2017). *Towards 3D Human Pose Estimation in the Wild: a Weakly-supervised Approach*. http://arxiv.org/abs/1704.02447

Zimmermann, C., Welschehold, T., Dornhege, C., Burgard, W., & Brox, T. (2018). *3D Human Pose Estimation in RGBD Images for Robotic Task Learning*. http://arxiv.org/abs/1803.02622