Gottfried Wilhelm Leibniz Universität Hannover
Institute of Photogrammetry and Geoinformation

Master Thesis

Deep learning-based multiple object tracking in the context of farm animal ethology

Rasho Ali, 3203570

Examiner:	Prof. Dr.-Ing. Christian Heipke
Second Examiner:	M. Sc. Mareike Dorozynski
Supervisor:	M. Sc. Mareike Dorozynski

Hannover, 2022-05-02

Statement

I declare that this Master Thesis is the result of independent research conducted by me under the guidance of my supervisor. It does not contain the results of any other scientific research that has been published or written by any other individuals or groups, except for those already cited in the thesis. Furthermore, I state that this work in the same or a similar form has not been submitted to an examination authority.

Hannover, 2022-05-02

Abstract

Automatic detection and tracking of individual animals is important to enhance their welfare and to improve our understanding of their behaviour. Due to methodological difficulties, especially in the context of poultry tracking, it is a challenging task to automatically recognise and track individual animals. Those difficulties can be, for example, the similarity of animals of the same species which makes distinguishing between them harder, or sudden changes in their body shape which may happen due to putting on or spreading out the wings in a very short period of time. In this thesis, an automatic poultry tracking algorithm is proposed. This algorithm is based on the well-known tracktor approach and tackles multi-object tracking by exploiting the regression head of the Faster R-CNN model to perform temporal realignment of object bounding boxes. Additionally, we use a multi-scale re-identification model to improve the re-association of the detected animals. For evaluating the performance of the proposed method in this study, a novel dataset consisting of seven image sequences that show chicks in an average pen farm in different stages of growth is used.

Contents

1 Introduction **1**

2 Theoretical Background **3**
2.1 Convolutional Neural Networks 3
2.2 ResNet 4
2.3 Faster R-CNN 5
2.4 Siamese Neural Network 6
2.5 MuDeep 6

3 Related Work **8**
3.1 Tracking-By-Detection 8
3.2 Re-Identification 9
3.3 Animal Tracking 9
3.4 Non-Motion-Model Based Tracker 10

4 Methodology **11**
4.1 Problem Statement 11
4.2 Tracktor 12
4.3 Animal Re-Identification 14

5 Dataset and Experiments **16**
5.1 Objectives 16
5.2 Dataset 17
5.3 Experiments 18
5.4 Metrics 21

6 Results **24**
6.1 Detection Results 24
6.2 Tracking Results 27

7 Conclusions and Future Work **32**

Bibliography **34**

1 Introduction

Analysing the behaviour of farm animals is a fundamental prerequisite for defining their needs and thus ensuring animal welfare. In this context, it is of great interest to determine the movement of animals in their habitat as a function of time, to analyse their behaviour in groups, during foraging and in relation to the use of space. In animal science, it is currently common practice to use video recordings of animal behaviour which are manually analysed. Since such manual procedures are extremely time-consuming, the goal of this work is to provide automatically extracted trajectories of all animals in an image sequence. To the best of our knowledge, there is only little research in the area of image-based tracking of farm animals (Zhang et al. 2019; Bergamini et al. 2021), which is particularly true for tracking of poultry (Li et al. 2020; Neethirajan 2022).

Up to now, there is no effective method to identify individual animals as a function of time. The research regarding animal tracking concentrates predominantly on animal movements. This can be done, for example, by analysing the optical flow patterns at flock level, to determine if animals are infected by a human pathogen (Colles et al. 2016), or by applying background subtraction to detect the lying-down behaviour of individual broiler chicken (Aydin 2017). On the other hand, many approaches have been developed to track pedestrians as illustrated by the extensive review on this topic by (Luo et al. 2022).

While pedestrian and animal tracking have similar goals, i.e., automatically detecting and associating each individual in an image sequence to a track, some differences arise due to animal anatomy and behaviour: The first difference concerns the similarity of individuals of the same species. While pedestrians can often be distinguished from each other by their appearance, size, as well as colour and shape of clothing, this is not always possible for animals of the same species. This is especially relevant when re-identification becomes necessary, for example, to connect two partial trajectories resulting from a temporary occlusion of an animal. Further challenges in tracking poultry arise with re-identification over longer time periods, as young animals grow much faster compared to humans resulting in relatively fast changes in appearance (Wurtz et al. 2019). Second, in poultry, sudden and significant changes in appearance can be observed when the wings are put on or spread out in a very short time. This makes it challenging to detect the animals in the image sequence and to re-identify all the detections of the individual animals in the course of the entire image sequence to determine those animal's trajectories (Kashiha et al. 2013). Third, in contrast to pedestrians who move in an almost straight path on sidewalks, or move in small groups from which some motion models can be derived by using a social force model (Helbing and Molnár 1995), it is harder to justify any such assumption to describe the motion of poultry as a function of time (Colantonio et al. 2007). As such animals, especially

young ones, are very energetic and playful in their movements, it is harder to model their motion. Fourth, the recording configuration often differs between pedestrian tracking and animal tracking. While the footage of pedestrians used for tracking is often captured by cameras at street level with the optical axis parallel to the ground level, cameras used to track livestock are usually mounted below the barn ceiling and result in nadir or oblique images.

In this thesis, we investigate the potential of animal tracking using the tracktor approach presented in (Bergmann, Meinhardt, and Leal-Taixé 2019). This approach is based on the Faster R-CNN model (Ren et al. 2015) and thus, it can detect objects that are small and close to each other well since it is a region proposal-based method (Zhao et al. 2019). Furthermore, tracking using the tracktor approach is achieved via bounding box regression, i.e., no explicit motion model is needed. To improve the re-identification of animals that have been occluded temporary, we use MuDeep (Qian et al. 2017). The MuDeep network architecture is based on a Siamese network. It learns features at different scales and evaluates their importance for object matching. The underlying assumption for the usage of this method is that MuDeep improves the re-association of the animals by focusing on the small differences in their appearance given that those animals are similar in many respects.

The main contribution of the present work is a method for the tracking of multiple animals in a confined space for research purposes, focusing on the tracking of poultry. For this intent, we use a tracking model that does not require a defined explicit motion model by exploiting the regression head of a detector to perform temporal realignment of object bounding boxes. Additionally, we use a multi-scale re-identification model to address problems that originate from occlusions.

2 Theoretical Background

In this chapter, we will discuss the theoretical background for the used tracking method. The chapter is split into four sections. In the first section, we present an overview of the Convolutional Neural Networks. In the next section will describe the Residual Neural Network architecture. In the next section we introduce the Faster-RCNN convolutional network which is used for animal detection, in the last section we describe the MuDeep re-identification network.

2.1 Convolutional Neural Networks

In image processing, the so-called Convolutional Neural Networks (CNNs) are considered to be one of the most popular types of neural network. The structure of these CNNs is modelled on the visual cortex of the human brain. Individual neurons fire in a specific local region belonging to a specific receptive field of view. In the next layer, other neurons respond to the previous neurons and thus cover a larger area by overlapping. This leads to a hierarchical structure in which neurons in the initial layers learn low-level features (e.g., edges and points) and neurons in the final layers learn high-level features (e.g., objects and shapes).

The convolutional layer corresponds to a 2-dimensional convolution and thus captures spatial context, which would be lost in a simple 1-dimensional representation (see Figure 2.1 left).

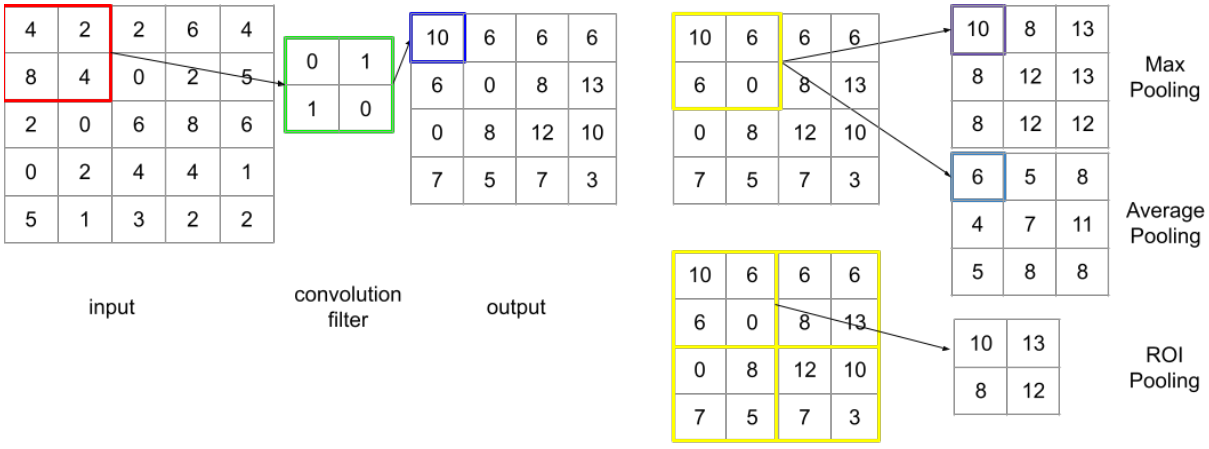

Figure 2.1: Convolutional and pooling layer

To that end, for each layer, several filters are used which are used to extract local features from each layer. Each filter is applied across the entire layer from top left to bottom right, this outputs a 2-dimensional map of activations which is called a feature map. By using this approach the

model has to learn only the parameter of the used filters, the number of those parameters is only dependent on the size and number of the used filters, this significantly lower the number of to be learned parameters compared to the one in a fully connected layer. The outputs of each filter are stacked and passed to the next layer where another set of filters is used to extract higher-level features. The output of the last layer is a stack of feature maps that indicates the locations and strength of a specific detected feature from the input.

The pooling layer helps to reduce the spatial dimension (see Figure 2.1 right). Instead of using a kernel, it simply determines the largest value (Max Pooling) of a given region or the average value (Average Pooling) of this region. In the case of Region of Interest pooling (ROI) (Girshick 2015) for given region of the feature map it scales it to some pre-defined size. The scaling is done by:

1. Dividing the region proposal into equal-sized sections (the number of which is the same as the dimension of the pre-defined output size)
2. Find the largest value in each section
3. Allocate tat value to the corresponding section in the output.

The fixed output size of the ROI Pooling makes it possible to use variable-sized input images and get the same sized feature-map in the output.

2.2 ResNet

ResNet or Residual neural network is a type of artificial neural network that was proposed by He et al. (He et al. 2016a) to address the problems of Deep networks in which the accuracy of the model rises at the beginning of training and then decreases, such problem can be for example the vanishing gradient problem. In the back-propagation step in which the neural network's weights are trained and updated using a gradient, which is based on the partial derivative of the used error function multiplied by the neural network's current weights, as the gradient is back-propagated to earlier layers, recurring multiplication may make the gradient infinitely small. The basic structure of a residual block is the so-called "identity shortcut connection" or "skip connection" that skips one or more layers, as shown in the Figure 2.2.

The main ideea of each of the ResNet blocks is to map $H(x) := F(x)+x$, which can be represented as a sum of a residual mapping $F(x)$ and an identity mapping x. The authors, He et al., argue that stacking layers shouldn't degrade the network performance, because by using the skip connection the error of the deeper model should not produce a training error higher than its shallower counterparts.

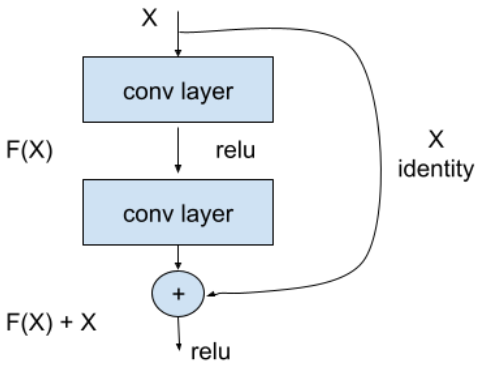

Figure 2.2: Residual learning: a building block.

2.3 Faster R-CNN

Faster R-CNN (short for Faster Region-based Convolutional Neural Network) is a deep convolutional network used for object detection, an overview of its architecture can bee seen in Figure 2.3.

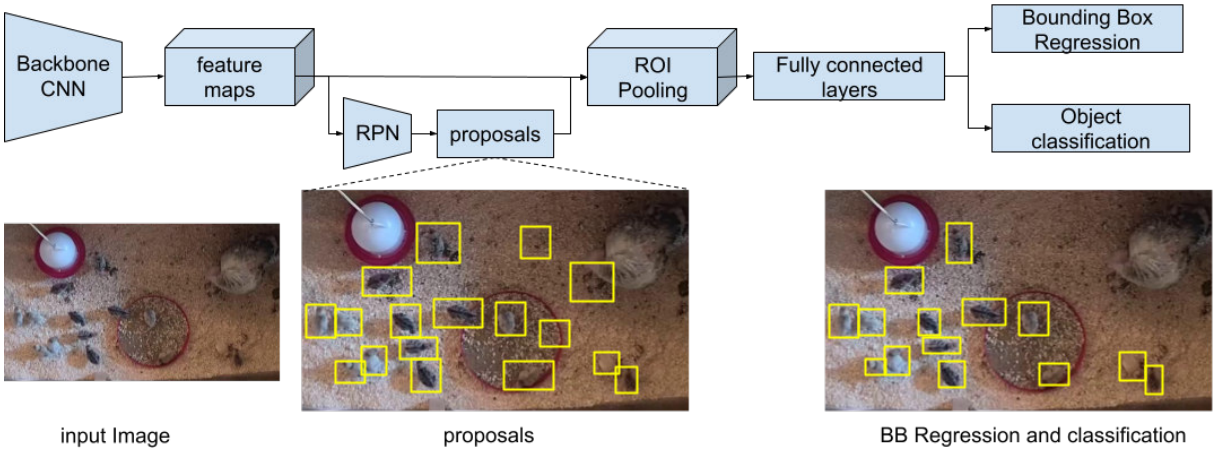

Figure 2.3: Overview of the Faster-RCNN architecture

Faster-RCNN architecture can be separated into three different steps. The first step is image feature-maps extraction in which a backbone CNN is employed. we used ResNet-50-FPN (He et al. 2016b) as backbone in the present work which is a convolutional neural network that is 50 layers deep.

In the next step, the network learns whether a pixel belongs to a certain object and estimates the size of that object. This is done by sliding a window over the feature maps and placing a set of "anchors" on corresponding positions between the input image and the window over the feature maps. By using fifteen anchors with three different aspect ratios and five different sizes we ensure that animals of different sizes are detected. This step generates a multitude of bounding box proposals for each potential animal.

In the third step, feature maps for each proposal are extracted via ROI Girshick 2015 and passes to a fully connected classification and regression heads. The classification head assigns an object

score s_k^t to each proposal. This score represents the probability s in frame t of a proposal k showing an object of interest, i.e., an animal. The regression head refines the coordinates of the proposals that contain animals. Next, non-maximum-suppression is applied to obtain the final set of detected animals.

2.4 Siamese Neural Network

A Siamese neural network is a type of neural network architecture that consists of two identical subnetworks. Siamese neural network is used to find the similarity of the inputs by comparing its feature vectors. In this thesis, the inputs in question are the images of the detected animals. The output of the Siamese neural network is a similarity score that describes how similar the two input images are thus if they belong to the same object. Siamese-ResNet-50 is a Siamese neural network in which each of the subnetworks consists of a ResNet-50 neural network. It shuld be noted that both the subnetworks in the Siamese-ResNet-50 are identical in ther configuration and with the same parameters and weights.

2.5 MuDeep

MuDeep (Qian et al. 2017) is a multi-scale deep learning model, its network architecture is based on a Siamese network which means it uses the same weights while working on two different input images to compute the similarity between the input images.

MuDeep is manly used to re-identify the tracked objects after occlusions. It is capable of detecting subtle differences between objects and thus, it is suitable to distinguish very similarly-looking animals. This model requires two object images as input and has two branches, one for processing each image. Each branch consists of the following components:

1. Tied convolutional layers: this layer is used to extract feature-maps from each input image.
2. multi-scale stream layers: the purpose of the multi-scale layers is to extract high-level features.
3. saliency-based learning fusion layers: this layer is combining the output of the multi-scale stream layer and emphasise the channels with highly discriminative high-level feature.
4. verification subnetwork: a fully connected layer that decides whether the input image-pair belongs to the same animal.

After each convolutional and fully connected layer, a batch normalisation and a Rectified Linear Unit activation (ReLu) (Agarap 2018) are used. The weights are shared between the two branches, resulting in the Siamese network structure. Figure 2.4 shows an overview of the MuDeep architecture.

The following steps are carried out by the MuDeep model to check whether two images belong to the same object.

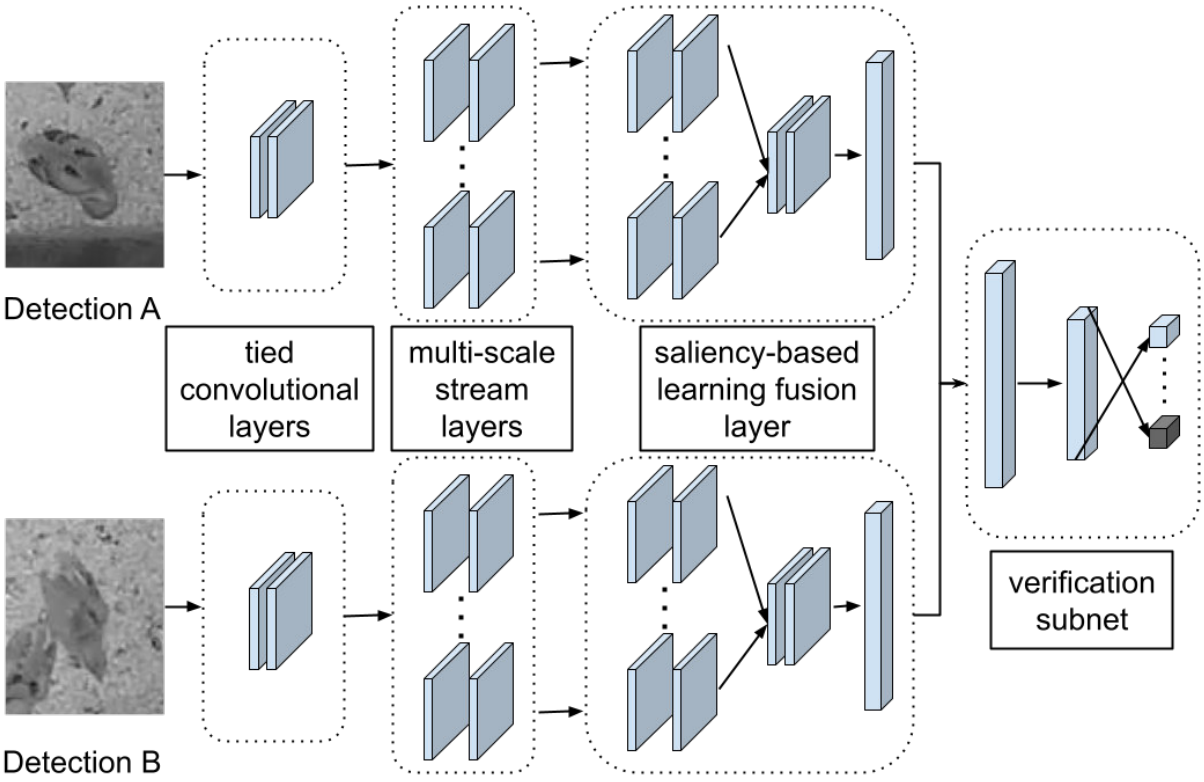

Figure 2.4: Overview of the MuDeep architecture.

In the first step, the tied convolutional layers extract feature-maps from each input image. The generated feature maps are passed to the second step, the multi-scale stream layers.
In these layers, the data stream is down sampled with four different sized convolution masks. This step is split into three layers:
i) Multi-scale-A layers, which serve as the first stage of the high-level feature extraction in which the data stream is down sampled with four different sized convolution masks. The four output streams from Multi-scale-A are passed to
ii) the Reduction layer. In this layer the widths and heights of the feature maps are halved. The outputs of the Reduction layer are passed to
iii) Multi-scale-B layers, that serves as the last stage of high-level features extraction.

In the third step, a saliency-based learning fusion layer is used to combine the output of the multi-scale stream layer and emphasise the channels with highly discriminative high-level features. Such features may, for example, be associated with the head of animals or feathers with a unique colour. This layer is connected to a fully connected layer that takes the high-level features and delivers a feature vector with a lower dimensionality being the outpt of the verification network.

We use the output of the verification subnetwork, the last step in the MuDeep model, to decide whether the input image-pair belongs to the same animal.

3 Related Work

In this chapter, the literature relevant to this thesis is reviewed and discussed. In the first section, we will introduce one of the currently most used approaches namely Tracking-by-detection, we will describe how it works and review literature that uses it that is relevant to this thesis. In the next section, we will discuss re-identification which includes non-CNN based and CNN based re-identification models. In the third section, we will discuss the used approaches for animal tracking. In the last section, we will introduce a non-motion-model based tracking algorithm namely tracktor.

3.1 Tracking-By-Detection

Much research has been carried out on pedestrian tracking during the last years (Chen et al. 2018; Ristani and Tomasi 2018; Tang et al. 2017; Klinger, Rottensteiner, and Heipke 2014); these algorithms are predominantly based on the tracking-by-detection paradigm (Chen et al. 2018; Bergmann, Meinhardt, and Leal-Taixé 2019). In tracking-by-detection the tracking problem is broken down into two steps: i) object detection in each frame, ii) object association between adjacent frames. It should be noted that the quality of the tracking algorithm is limited by the performance of the underlying detection method (Luo et al. 2022). Recently, neural network-based detectors have outperformed conventional methods for detection (Ren et al. 2015; Redmon and Farhadi 2018), making them the main choice for approaches for tracking-by-detection (Bergmann, Meinhardt, and Leal-Taixé 2019).

Many methods can be used for the second step of tracking-by-detection in the context of pedestrian tracking. One of those methods is motion modelling and trajectory prediction (Shafique, Lee, and Haering 2008). This method captures the dynamic behaviour of an object to estimate its potential position in future frames by expressing assumptions about the movements of the individuals to be tracked in form of a motion model that can be integrated into a filter approach (Luo et al. 2022). One could, for example, make a simple constant velocity assumption (CVA). Alternatively, the motion model can be made more complex by applying prior knowledge from a social force model (Helbing and Molnár 1995) being an example for interaction modeling (Luo et al. 2022). The social force model describes the behaviour of crowds as a result of the interactions of individuals. Concerning our task, it is hard to adapt a social force model from pedestrian tracking to poultry tracking due to difficulties in modelling animal motion and the limited understanding of the interaction of stock individuals (Colantonio et al. 2007). Additionally, the

fairly playful and energetic motion of the chicks in the scene makes it hard to implement a motion model that can estimate the potential position of the chicks in future frames.

3.2 Re-Identification

Another method is to use an appearance-based model to create links between the detected object in the individual frames this is called Re-Identification. Such information on the appearance can be particularly helpful in crowded scenes with many object-object occlusions where an ID-switch is probabe to happen. Such an appearance-based model can exploit optical flow (Ali and Shah 2008), point features (Ommer, Mader, and Buhmann 2009) or gradient-based features such as features of a histogram of oriented gradients (HOG) (Dalal and Triggs 2005). Due to the significant advances of machine learning approaches in recent years, many CNN-based re-identification models have been developed in the context of pedestrian tracking (Yu et al. 2018; Li, Zhu, and Gong 2018). However such methods are not necessarily reliable for animal tracking, since animals often have similar shapes, and colour statistics are often contaminated by background pixels and illumination changes, while the differences in appearance between the animals are often subtle and only detectable at particular locations and scales.

3.3 Animal Tracking

The proposed algorithms for animal tracking in the literature mainly tackle specific animal behaviour aspects, such as eating or drinking (Li et al. 2020). This is achieved by detecting the position of an animal relative to the position of the feeder, water or the nest (Li et al. 2020). While those systems could be used for tracking, they concentrate on detecting different behaviour of the animals and neglect tracking. By using wireless wearable sensors (e.g. accelerometer, RFID microchip) (Chien and Chen 2018) the position of the animal can be detected if the animal is near the feeder, water or the nest. Due to the expense and invasive nature of wireless wearable sensors, visual-based monitoring systems have been used more and more in recent years (Li et al. 2020).

Bergamini et al.(Bergamini et al. 2021) use an image-based tracking algorithm to extract long-term behavior changes of pigs. They use the YOLO v3 (Redmon and Farhadi 2018) model to detect pigs in each frame independently. In the first frame, a new tracklet is created and initialized for each detection. The tracker is based on a Minimum Output Sum of Squared Error (MOSSE) (Bolme et al. 2010), this tracker uses an adaptive correlation for object tracking which produces stable correlation filters when initialized. In the following frames, updated trackers and single-frame detections are matched together comparing the Intersection over Union and their appearance and finding the best assignments with the Hungarian algorithm (Kuhn 1955). If no match is found the track is removed. On the other hand, if a detection is not matched to any track, a new track is created and initialized. The algorithm in (Bolme et al. 2010) is easy to calculate and can quickly track objects, but it does not guarantee accurate results when the

object's appearance changes which is a concern for animal tracking since, as discussed previously, Body shape can change in poultry by putting on or spreading out the wings in a very short period of time.

Neethirajan (Neethirajan 2022) uses the YOLO v5 model, which is based on YOLO v3 (Redmon and Farhadi 2018), to detect chickens in each frame. To track each individual chicken, a Kalman filter is applied. The downside of this approach is that the accuracy of the Kalman filter depends on the assumption of linear motion for any chick to be tracked. If a chick takes some abrupt turns, whcih often happens, the nonlinear movement can not be well handled by the Kalman filter framework.

3.4 Non-Motion-Model Based Tracker

Tracktor (Bergmann, Meinhardt, and Leal-Taixé 2019) is a possibility that can help achieving the stated goals. Traktor is a non-motion-model based tracking approach, simplifying tracking by eliminating the need for any knowledge or assumptions about animal motion behaviour. Tracktor requires an image sequence of the animals to track as input. It tackles multi-object tracking by exploiting the regression head of a detector to perform temporal realignment of object bounding boxes; objects are detected and classified using Faster R-CNN (Ren et al. 2015). The data association step in the tracktor method is done by combining object classification scores from the detection step and intersection over union (IoU) between bounding boxes of two subsequent time steps. Tracktor generates a region suggestion for each animal in each frame and combines the suggestions of different frames to form a trajectory for the observed animal over the entire image sequence.

4 Methodology

In this chapter, we will introduce the used Method for poultry tracking. The chapter is split into two sections, Tracktor and Animal Re-identification. We describe the used tracking algorithm in Tracktor section and propose MuDeep for the occlusions problem in the Animal Re-identification section.

4.1 Problem Statement

In the context of this work, it is investigated how existing tracking algorithms can be transferred to poultry. The developed methodology aims to generate a region suggestion for each animal in each frame and to combine the suggestions of different frames to form a trajectory for the observed animal over the entire image sequence. As input for the developed methodology, we use the frames from mono-image sequences which can have grey or RGB frames. Each of those sequences shows a group of chicks with a Turkay. all the poultry are located in a barn which is 2.60 x 1 Meter. The output of the developed methodology is a list of Bounding Boxes for each chick in the input sequence. We assume that the recordings are done from a stationary camera to eliminate the need for any camera motion compensation model. Additionally, we assume that all the recordings have been taken from a nadir view which allows us to use additional types of augmentation namely horizontal flip.

Our methodology is based on the tracktor approach which tackles multi-object tracking by exploiting the regression head of the Faster R-CNN model. We assume that the animals in the video sequences move only slightly between frames, which is ensured by the high frame rate of the used video sequences. This assumption is important for the Tracktor approach since it doesn't use any motion model to track the animal. The tracktor approach has a problem tracking an object if there is a discontinuity in that object's detection which can happen if the object is for example occluded which can happen when the chicks cluster together or get under the turkey, another example for the discontinuity in an object's detection is when the Faster R-CNN model has a problem finding it in one frame. To deal with the detection discontinuity problem we use the MuDeep re-identification model in addition to the tracktor approach. For which we assume that the chicks in the sequence seen are distinguishable from each other by their shapes or the colour of their feathers.

4.2 Tracktor

As previously mentioned different challenges arise in the context of animal tracking compared to that of pedestrians, like the sudden and significant change in the poultry shape when moving or flying and the many occlusions that happen since the animals tend to cluster together, this requires a robust detection model. Additionally, the non-existing motion mode that could be used for poultry tracking makes it necessary to use a non-motion-model based tracking approach. The tracktor achieves that without using any motion-model and just by assuming that the tracked objects in the scene have moved slightly between frames, which can be achieved by a high frame rate. The tracktor is using the detections of the objects from the previous frame, which still covers a big part of the objects in the current frame since those objects moved just slightly between frames, and uses the regression head of the Faster R-CNN to regress the position of the object's old detection to the new object position thus crating a track. Additionally, by using Faster R-CNN for object detection, tracktor can detect small and clustered objects well since it uses fifteen anchors in a single grid in the RPN step thus each of those anchors can detect chicks of different size. The Loss function of the Faster R-CNN model is the combination of two different loss functions by adding them together. The first loss function is the log loss over two classes (There is an object or not). The second term is the regression loss of bounding boxes namely the robust loss function (smooth L1) as used in (Ren et al. 2015), according to (Ren et al. 2015) this loss is less sensitive to outliers, than other regression loss. Smooth L1 is calculated as flow $L_{1smooth} = |x| \; if \; |x| > 1 \; else \; x^2$ where x is the L1 distance (Manhattan distance) between 2 vectors, the first vector contains the coordinate of the detected BB and the second vector contains the coordinate of the gt BB. The only part of the tracktor approach that needs training is the detection part namely the Faster R-CNN model thus the tracktor does not need any training after the Faster R-CNN model has been trained. Figure 4.1 shows an overview of the tracktor architecture.

Tracktor extracts the trajectories of objects in a video sequence. Each trajectory is given as a list of ordered object bounding boxes $T_k = \{b_0^k, b_1^k, ...\}$, where b_t^k is a bounding box b of object k in frame $t \in \{0, 1, ...\}$. In each frame t, the list of detected objects assigned to a trajectory is defined as $B_t = \{b_t^0, b_t^1, ..., b_t^{K_t-1}\}$ listing the bounding boxes b of all K_t objects $k \in \{0, 1, ..., K_t - 1\}$ in frame t. At $t = 0$ the tracker initialises tracks from the first set of detections D_0 as $B_0 := D_0 = \{d_0^0, d_0^1, ...\}$, where d_0^j is the j^{th} bounding box d delivered by the detector in frame $t = 0$ that is not yet assigned to a trajectory and has an object score s_j^0 bigger than a threshold $\lambda_{detect} = 0.5$, as defined in the original approach of (Bergmann, Meinhardt, and Leal-Taixé 2019).

For all frames $t > 0$ the following two steps are carried out to determine B_t:

- Bounding box regression (red arrows in Figure 4.1): given the assumption that an object k moves only slightly between two subsequent frames, its trajectory list $T_k = \{b_0^k, ..., b_{t-1}^k\}$ is extended from the preceding frame $t - 1$ to the current frame t, leading to $T_k = \{b_0^k, ..., b_{t-1}^k, b_t^k\}$. This is achieved by exploiting bounding box regression, i.e. by regressing

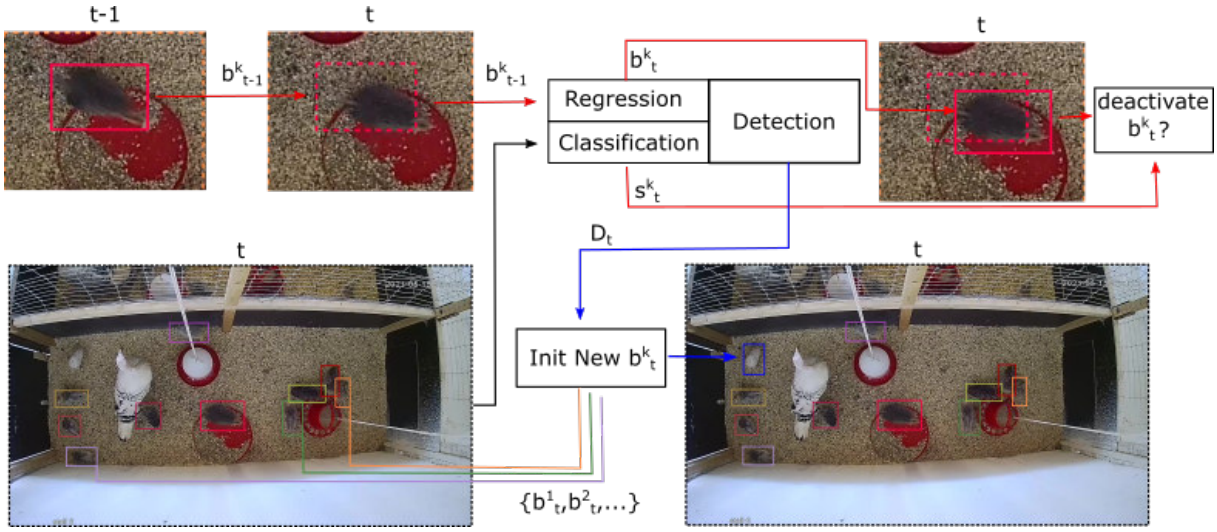

Figure 4.1: The proposed methodology performs simultaneous tracking of multiple animals using the tracktor approach. The two basic processing steps, bounding box regression and initialisation, are shown by red and blue arrows, respectively, for a single image at time t using an example from our poultry dataset. In the bounding box regression step, the object detector regression head adjusts the existing bounding boxes b^k_{t-1} of frame $t-1$ to the new positions of the objects at frame t. The corresponding object classification scores s^k_t of the new bounding boxes are used to deactivate potentially occluded tracks. To determine whether a new bounding box has to be initialised, the Intersection over Union (IoU) is calculated between each element of the set of detections D_t of frame t and the elements of the active tracks $B_t = \{b^0_t, b^1_t, ...\}$. If the IoU between a detection and the bounding boxes of all active tracks is smaller than a threshold, a new track is initialised for this detection and is added to the active tracks. (Figure adapted from Bergmann et al., 2019). ©IPI, TiHo Hannover.

the bounding box b^k_{t-1} in frame $t-1$ to the object's new bounding box position b^k_t at frame t. This is conducted for all K_{t-1} objects, where K_{t-1} is the number of objects in frame $t-1$, leading to the list of bounding boxes $\{b^0_t, ..., b^{K_{t-1}-1}_t\}$ for the current frame t referred to as active trajectories.

- Bounding box initialisation (blue arrows in Figure 4.1): a new trajectory of an object i that is not yet contained in the list of objects $\{b^0_t, ..., b^{K_{t-1}-1}_t\}$ resulting from bounding box regression can be initialised using the list of detections D_t for the current frame t, assuming that there is a detection $d_i \in D_t$ representing object i. A new trajectory is initialised for that object if the IoU of d_i with any of the $\{b^0_t, ..., b^{K_{t-1}-1}_t\}$ is smaller than a threshold $\lambda_{new} = 0.3$, leading to $B_t = \{b^0_t, ..., b^{K_{t-1}-1}_t, d_i\} =: \{b^0_t, ..., b^{K_t-1}_t\}$.

A trajectory is deactivated, if the IoU between two objects in B_t is larger than a threshold $\lambda_{active} = 0.6$. which means that the object with the smaller classification score is occluded by the other object. Alternatively, a trajectory is also deactivated if the classification score s^t_k of any object k in frame t resulting from Faster R-CNN is below a threshold $\lambda_{score} = 0.5$, which means that the object has left the frame or is occluded by another part of the scene.

4.3 Animal Re-Identification

In tracking, especially in the context of poultry tracking, occlusions are a common problem. This problem can, for example, be seen when all the animals group together for food or sleeping as depicted in Figure 4.2.

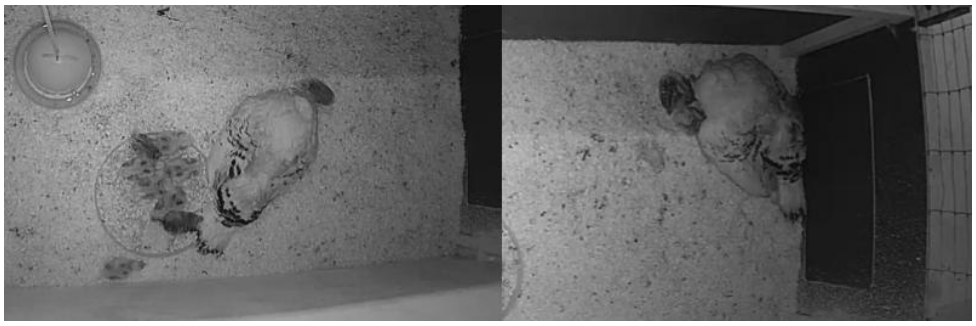

Figure 4.2: An example of chick-to-chick occlusion while feeding on the left and while sleeping on the right. ©TiHo Hannover

While the tracktor approach works well for tracking objects frame by frame, re-identification makes it possible to find objects that were no longer visible on several consecutive frames. To that extent, we assume that the chicks in the sequence are distinguishable from each other by their body shape, the colour of their feathers and the distribution of the dark to light feathers on their body. The limits of this assumption are that those differences are very subtle and hard to differentiate from each other.

In the original Tracktor approach, a Siamese-ResNet-50 neural network is used to aid with the re-identification of the pedestrians which works well since pedestrians can often be distinguished from each other by their appearance, and size, as well as colour and shape of clothing. This is not always possible in the case of poultry tracking since the differences between the individual chicks are very subtle and that is why we propose the use of the MuDeep re-identification model. MuDeep is capable of detecting subtle differences between objects and thus, it is suitable to distinguish very similarly-looking animals. This is due to the use of multi-scale stream layers and saliency-based learning fusion layers. In the multi-scale stream layers, high-level features are extracted in different scales, this is done by using fore parallel convolution layers each with a different convolutional mask size. This is done to ensure that bigger features like the body shape and smaller features like the feathers or the head shape are extracted, ensuring that all the subtle features are extracted. Not all the extracted features from the multi-scale stream layers are important, for example the background context, thus MuDeep utilize saliency-based learning fusion layers to automatically discover and emphasize the channels that had extracted highly discriminative patterns, such as the previously mentioned shape of the body or feathers with a unique colour.

To allow the re-identification of animals that have not been seen by the detector for multiple frames, we save deactivated tracks for 50 frames such as in the original approach of (Bergmann, Meinhardt, and Leal-Taixé 2019). Then, the MuDeep model is used to compare the deactivated

with the newly initialised tracks and connects them. This is done by calculating their similarity score, in our case by calculating triplet loss, if the score is smaller than 200 then the two images belong to the same object. If the associated animal can be re-identified, meaning that both tracks belong to the same animal.

5 Dataset and Experiments

This chapter is split into two sections namely Dataset and Experiments. In the Dataset section, we describe the different video sequences that are used in this work. In the Experiments section, we evaluate the method presented in this work using the dataset from the Dataset section. Additionally, we describe the used Augmentation and the different used Experiments to evaluate the model.

5.1 Objectives

The overall objective of this work is to investigate how existing tracking algorithms can be transferred to poultry tracking, which is achieved by an algorithm based on the Tracktor approach in combination with the MuDeep re-identification model. The approach presented in this work is evaluated by using hand-annotated reference data. To better understand the presented approach the detection and re-identification parts are split and examined separately. The aim is to investigate the following questions:

How well does the detector model used in the tracktor paper detects chicks and how much does using targeted augmentations improve the detection model?

This question investigates the influence of using targeted augmentation on improving the detection model, like brightness augmentation to simulate dark and well light areas in the barn or the Gaussian augmentation to simulate fast moving chicks. To address this question we trained two different Faster R-CNN models, one with the same augmentation that has been used in the tracktor paper and one with extended augmentation that targets specific detection difficulties like uneven brightness in the barn and the fact that many chicks tend to run in the barn and some tend to sit still thus their shapes looks different in the image.

does using a re-identification model improves the tracking results, and does using Mudeep improves the re-identification in comparison with Saimin ResNet-50 which was used in the Tracktor paper?

This question aims to better understand whether the tracktor approach can better track the chicks in the video sequence by just itself, with the aid of Saimin ResNet-50 and with the aid of MuDeep.

5.2 Dataset

For the evaluation of the method presented in this work, a novel dataset was captured. The dataset was recorded by TiHo Hannover (Tierärztliche Hochschule Hannover) over a period of 48 days. This dataset is composed of a total of seven videos that were acquired during different times of the day and with varying lighting conditions. The dataset comprises grey and RGB video sequences. Each video sequence is recorded via a stationary BERGHOCH 8MP Aussen I Basic camera, mounted over a pen observing the scene from a nadir view. The pen dimensions are 2.6 x 1.0 meters. In each sequence, one pen is fully visible, containing various chicks and an adult hen. Note that in this work, we are focusing on tracking the chicks only. The chicks in the different recordings are of different ages and thus of different sizes. We split the dataset accordingly into three sections: small, medium and big. In addition to the fully visible pen, several partially visible pens can be seen at the boundary of the images (see Figure 5.1).

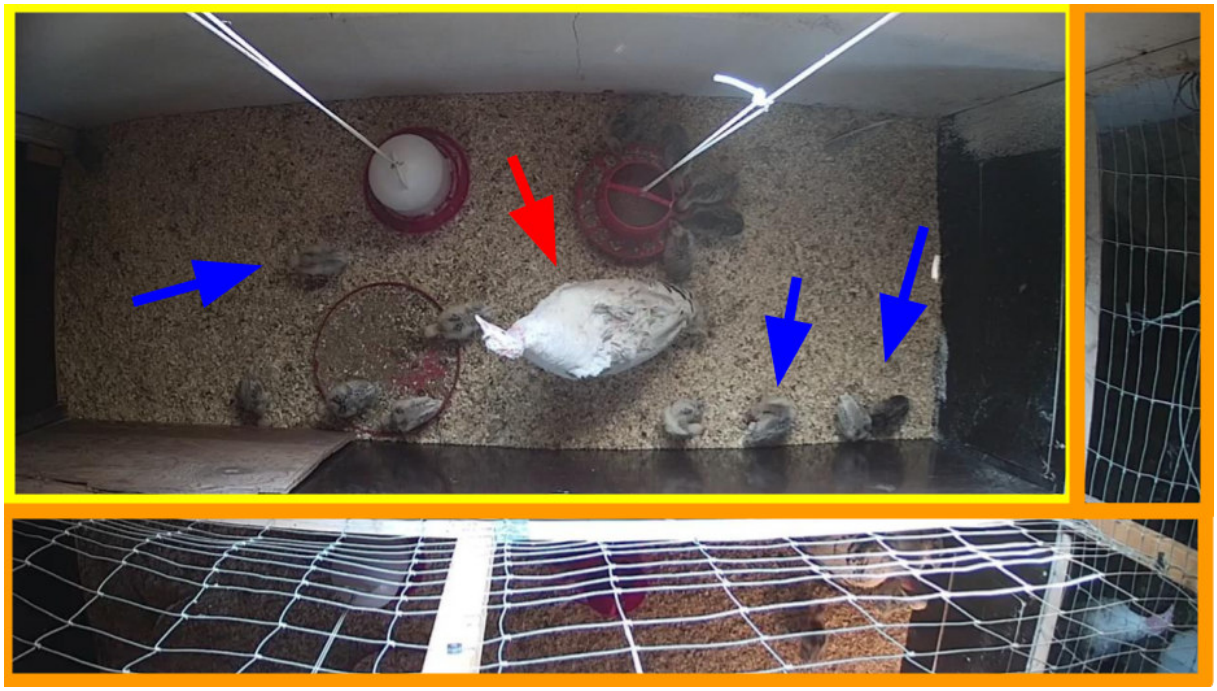

Figure 5.1: An example of the scene recorded by the used video sequences. The red arrow shows the adult turkey, blue arrows show the chicks. The yellow rectangle shows the main pen, the orange rectangles show the adjacent pens. ©IPI, TiHo Hannover

To provide a reference, all the data was manually annotated, using axis-aligned bounding boxes for all chicks in all frames. Those boxes are set such that they are the smallest possible boxes containing the chicks completely. The bounding boxes are defined by the coordinates of the top left and bottom right corners.

The video recordings were acquired at 30 frames per second with a resolution of 1280 x 720 pixels. The recordings are split into three different chick sizes and two camera colour settings (see Table 5.1). While Figure 5.2 shows examples of RGB and grey value images, Figure 5.3 shows the chicks of different sizes in the different stages of their growth. In total, the dataset consists of seven annotated video sequences, of which five have a length between 1800 and 1850 frames,

one video sequence has 2000 frames and one has 4000 frames.

ID	#Frames	Chick size	Colour setting	Use
1a	1000	Small	RGB	train
1b	1000	Small	RGB	**val**
1c	2000	Small	RGB	**test**
2	1850	Small	RGB	train
3	1815	Small	RGB and Grey	**test**
4a	900	Medium	Grey	train
4b	900	Medium	Grey	**val**
5	1808	Medium	RGB	train
6a	1000	Big	Grey	train
6b	1000	Big	Grey	**test**
7	1819	Big	RGB	train

Table 5.1: Description of our novel dataset. ID: the number indicates the ID of a video sequence, while the letters denote that a sequence has been split. For example, the subsets 1a and 1b are both part of video sequence 1, where 1a contains frames 0 to 999 and 1b the frames 1000 to 1999. #Frames: number of frames in the video sequence. Size: the growth status of the chicks in the video sequence. Colour setting: colour settings used while recording (RGB or grey). Use: indicate whether a video sequence is used for training, validation or testing.

Figure 5.2: Example of the different colour settings used. RGB video sequence on the left, grey video sequence on the right. ©TiHo Hannover

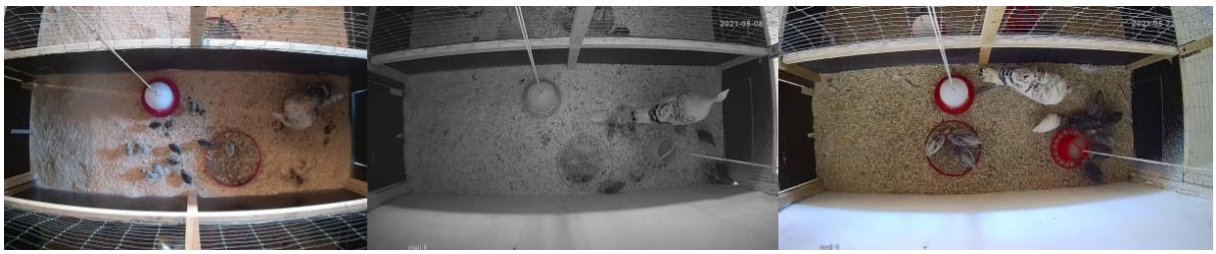

Figure 5.3: Different stages of chick growth. From left to right: small, medium, big. ©TiHo Hannover

5.3 Experiments

In this section, we evaluate the method presented in this work using the dataset introduced in Section 5.2. To that extent, we will use the split shown in Table 5.1 to create the training, validation and test sets.

5.3.1 Training and Test Strategy

For the training of the Faster R-CNN model, we use a maximum of 30 epochs with early stopping to avoid overfitting. The model is trained with a batch size of 4 and a learning rate of 0.001. we used the Stochastic gradient descent optimization with the following parameters: momentum=0.9, weight-decay=0.0005. Additionally, we used a learning rate scheduler which decreases the learning rate by 0.1 every 10 epochs. The Loss function of the Faster R-CNN model is the combination of two different loss functions by adding them together. The first loss function is the log loss over two classes (There is an object or not). The second term is the regression loss of bounding boxes namely the robust loss function (smooth L1). This function takes the coordinate of the detected BB and gt BB as input then it calculates x which is the L1 distance (Manhattan distance) between the two BBs, at the end x is used to calculate $L_{1smooth} = |x| \; if \; |x| > 1 \; else \; x^2$.

The MuDeep model is trained with a maximum of 40 epochs, a batch size of 32. We used the adam optimization with the following parameters: lr=0.0001. For the loss function, we used the triplet loss in which the reference input is compared to a matching input and a non-matching input. The distance from the reference to the matching input is minimized, and its distance to the non-matching input is maximized.

5.3.2 Augmentations

While the footage of pedestrians used for tracking is often captured by cameras at street level with the optical axis parallel to the ground level, cameras used to track the chicks are mounted below the barn ceiling and result in nadir or oblique images. This gives us the possibility to extend the used augmentation setting for pedestrian tracking in Bergmann, Meinhardt, and Leal-Taixé 2019, where only random flipping along the vertical axis is applied. In the case of poultry tracking in a barn, some difficulties have to be addressed like the uneven brightness in the barn and the fact that many chicks tend to run in the barn and some tend to sit still thus their shapes look different in the image. In this work we suggest adding Gaussian noise, to augment the brightness and to randomly flip along the horizontal axis.

The use of the added Gaussian noise is motivated by the assumption that very fast motions of the chicks result in motion blur. We apply Gaussian noise with a zero mean and a standard deviation of $\sigma = 1.7$. The brightness augmentations are motivated by the observation that the pen is unevenly illuminated. Accordingly, brightness augmentations are supposed to simulate the situations in which the chicks are in shadow. For this purpose, a brightness factor is randomly drawn from the interval $[0.5, 1.5]$ and each channel of the image is multiplied by this factor. If a colour value exceeds the upper or lower bound of possible values after applying the brightness factor, this value is set to the respective bound. By randomly applying flipping along the vertical and horizontal axis, we ensure that the model observes the chicks from many angles. An example of the augmentation strategies applied can be seen in Figure 5.4.

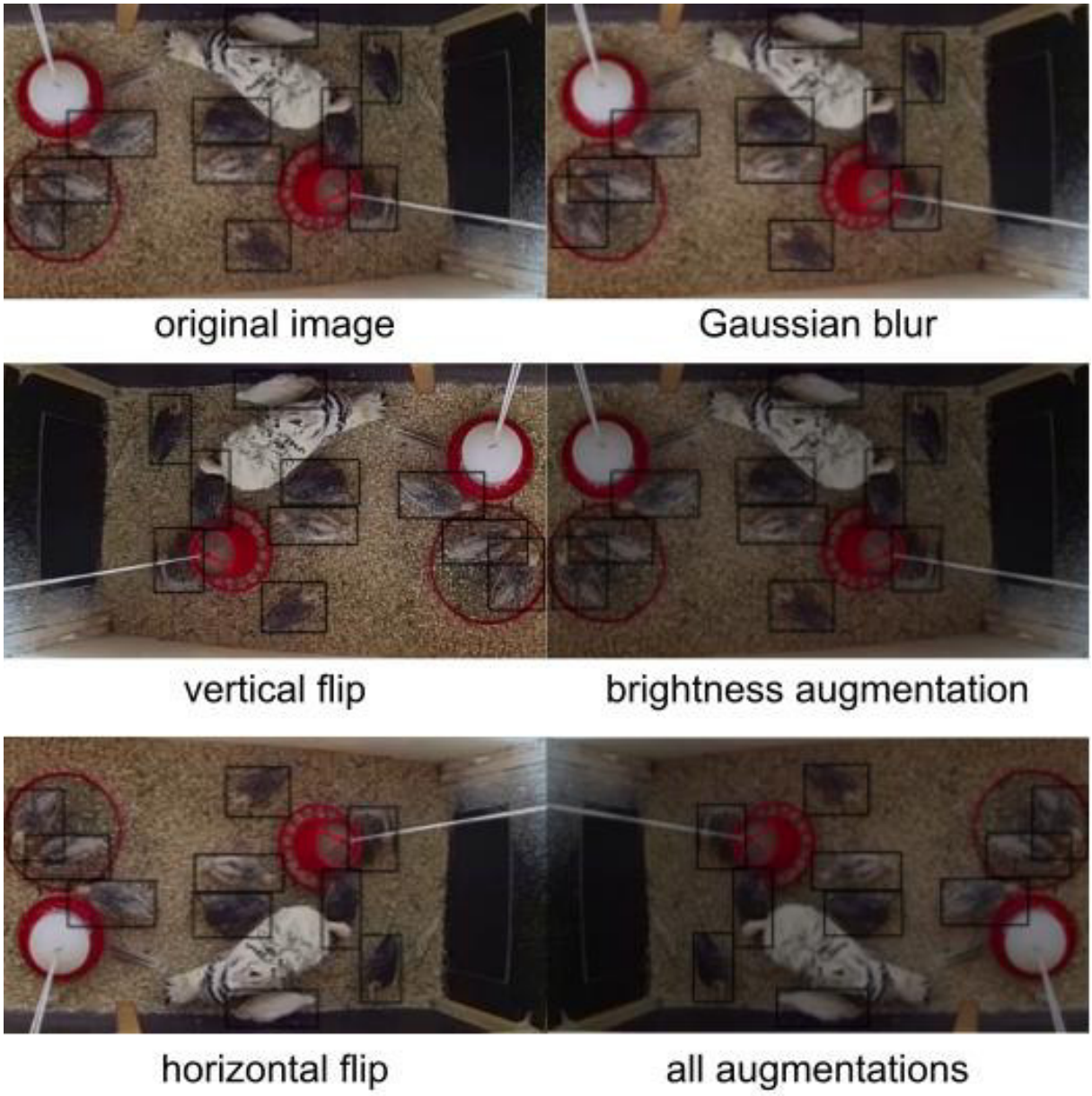

Figure 5.4: Example of the different augmentation strategies used.

5.3.3 Experiments

To define a baseline for tracking the chicks and to obtain a better understanding of the influence of the MuDeep re-identification model on the tracking results in the context of poultry tracking, we carry out multiple experiments. Those experiments can be split into two groups: detection experiments and tracking experiments. In the first group of experiments, i.e., the detection experiments, we train the Faster R-CNN model with two different data augmentation methods. For the first experiment, we used the same augmentation that has been used in the tracktor approach for pedestrian tracking, namely random vertical flip. We call this experiment the baseline-detection test. In the second experiment, we apply the extended augmentation strategy mentioned above to the dataset. We call this experiment the extended-detection test. In the

second group of experiments, i.e., the re-identification experiments, we apply both the baseline-detector and the extended-detector to obtain detections in the context of Tracktor. Furthermore, Tracktor is used with different re-identification models, i.e., without any re-identification model (denoted as none), with the Siamese re-identification model based on ResNet-50 (denoted as ResNet-50), which has also been used in the tracktor approach for pedestrian tracking, and with MuDeep re-identification (denoted as MuDeep). We combine each detector with each of the re-identification models in our experiments, whereas an overview of the experiments is given in Table 5.2.

Augmentation	Re-identification model	Experiment ID
baseline	none	baseline-none
	ResNet-50	baseline-ResNet-50
	MuDeep	baseline-MuDeep
extended	none	extended-none
	ResNet-50	extended-ResNet-50
	MuDeep	extended-MuDeep

Table 5.2: Overview of the experiments carried out (for details see text).

5.4 Metrics

We use Intersection over Union (IoU) to determine if a detection is correct or not. IoU is a metric to evaluate object detection accuracy. It is calculated between the ground truth and predicted bounding boxes overlap area with union area, see figure 5.5. If IoU is larger than 0.5 the detection is considered correct.

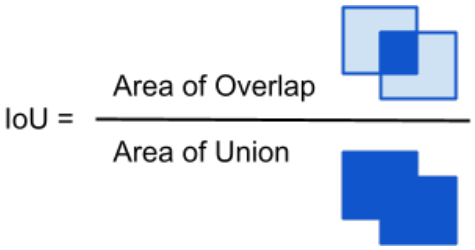

Figure 5.5: IoU

Three metrics are used for evaluating the detection model, namely Precision, Recall and Average Precision (AP). Precision is the number of true positives divided by the number of true positives plus the number of false positives $P = TP/(TP+FP)$, while Recall is the number of true positives divided by the number of true positives plus the number of false negatives $R = TP/(TP + FN)$. Average Precision is the harmonic mean of Precision and Recall.

For the quantitative evaluation of the tracking approach, the following metrics, which are common in the tracking domain, are used: IDF1 Ristani et al. 2016, MOTA and MOTP Bernardin and Stiefelhagen 2008. The combination of these three metrics allows, on the one hand, to evaluate the results in terms of the correctness of the object IDs and thus the consistency of the trajectories

(IDF1), as well as the accuracy of the determined position (MOTP). On the other hand, this combination allows us to put the tracking results obtained into the context of other work in terms of classification accuracy (MOTA).

IDF1 combines ID precision (IDP) and ID recall (IDR) into a single value by using the harmonic mean, using the definitions of precision and recall given above.

$$IDF1 = \frac{2 \cdot IDP \cdot IDR}{IDP + IDR} \tag{5.1}$$

MOTP represents the average of all IoU errors of the chick detections 5.2.

$$MOTP = \frac{\sum_{i,t} IoU_t^i}{\sum_t c_t} \tag{5.2}$$

where:
IoU_t^i = Intersection over union between object i in the ground truth and the detection output at frame t and c_t = total matches made between ground truth and the detection output at frame t

MOTA, on the other hand, combines three different error metrics, namely the number of ID switches, false positives and false negatives in one score. Summing up these three metrics and dividing the sum by the total number of objects present in all frames, gives us the total error rate E_{tot}. MOTA is then defined as $MOTA = 1 - E_{tot}$.

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDS_t)}{\sum_t GT_t} \tag{5.3}$$

where:
FN_t = False negative at frame t and c_t = total matches made between ground truth and the detection output at frame t

IDF1 better measures the consistency of ID matching in comparison with MOTA. Figure 5.6 illustrates the influence of ID switches on the MOTA and IDF1 metrics scores.

It can be seen in Figure 5.6 that MOTA scores for Tracker 1 and 2 are calculated based on: FN = 0, FP = 0, ID switches = 4 (indicated with red link) thus MOTA = 0.6 for both of them. On the other hand IDF1 score for Tracker 1 is $IDF1_1 = (2 \cdot 2)/(2 \cdot 2 + 8 + 8) = 0.2$ and IDF1 score for Tracker 2 is $IDF1_2 = (2 \cdot 8)/(2 \cdot 8 + 2 + 2) = 0.8$

Tracker 1 and Tacker 2 start their tracking with target A which lasts 10 frames. Tracker 1, correctly tracks target A for 2 frames and the remaining frames are assigned to other wrong targets (B, C, D, E) while Tracker 2 tracks target A for 8 frames with just 2 ID switches.
This example shows that, on the contrary to MOTA, IDF1 measures how long the identification is correct and gives different scores, which can better reflect trackers' performance.

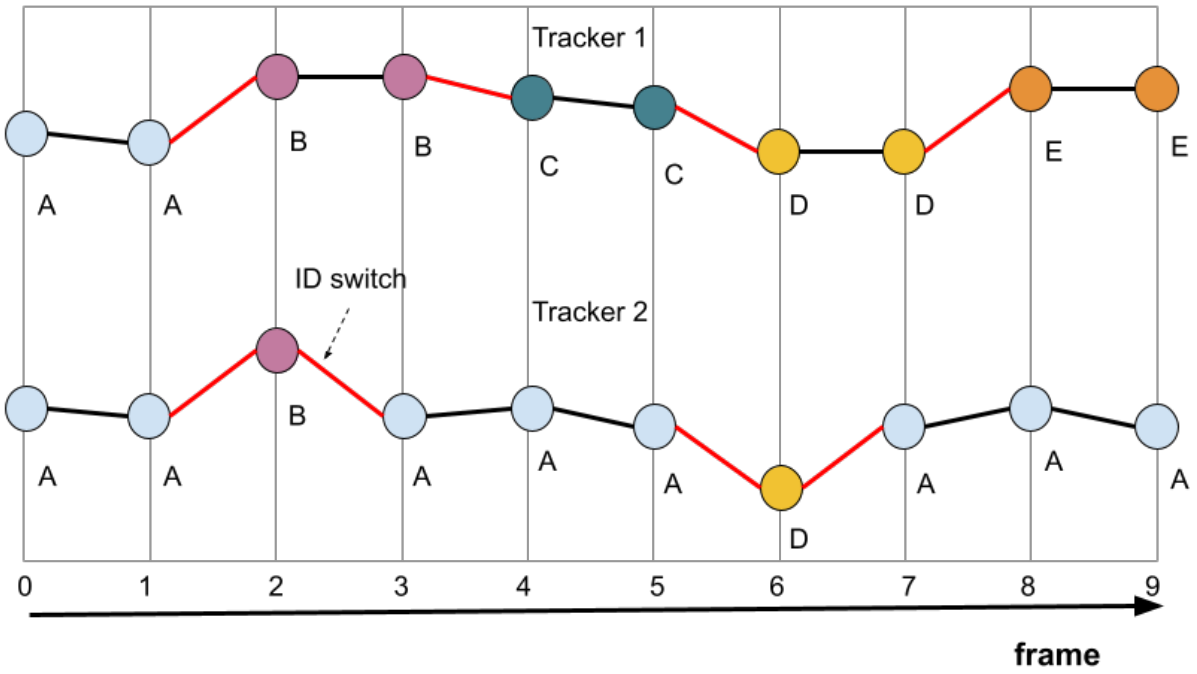

Figure 5.6: comparison of MOTA and IFD1. The comparison shows two Trackers (Tracker 1 and Tracker 2). The X-Achses shows frame number. Coloured circles represent the added object to each Tracker, the Color represents which object is added (i.e. yellow circles represent object D). The lines between the objects show if an object is correctly tracked (Black lines) or an ID switch has acquired (Red lines).

6 Results

For testing, we use the three different video sequences 1c, 3 and 6b. The chicks and the barn in sequences 1c and 6b have been seen by the model in the training and validation phase, respectively, where different frames have been used there, i.e., the disjoint sequences 1a, 1b and 6a. On the other hand, the chicks and the barn in sequence 3 have neither been seen during training nor while validating the detection model.

6.1 Detection Results

Comparing the results of the baseline detector and the extended detector shown in Table 6.1, it can be seen that the Average Precision (AP) of the extended model is slightly higher for both sequences 1c and 6b, which can be explained by the fact that similar sequences have been used in the training and validation of the detector. In contrast, the results for sequence 3 are slightly worse for the extended version, which is probably caused by its differences to the sequences used for training and validation. In sequence 1c the very small change in the Recall score in comparison to that of the Precision suggests that the extended model is not detecting any new objects in the scene which have not been detected by the baseline model, rather it getting less FP detection. Additionally, it can be seen that the Recall of sequence 3 has increased by 3% while the Precision decreased this means that the model has increased its TP detection but at the cost of increasing its FP by more than 10%. Overall the extended detection model shows an improvement in comparison with the baseline model this can be seen in the last row of Table 6.1. The increase in the mean number of FP detection is due to the very high number of FP detections in sequence 3.

ID	Augmentation	AP	Precision	Recall	FP
1c	baseline	78.2%	85.5%	91.4%	2321
	extended	79%	86.3%	91.5%	2178
3	baseline	29.1%	55.4%	49.6%	10846
	extended	28.8%	54.6%	52.5%	11866
6b	baseline	93.9%	95.7%	97.8%	463
	extended	95.5%	96.4%	98.9%	411
mean	baseline	67.1%	78.9%	79.6%	4543
	extended	67.8%	79.1%	81%	4818

Table 6.1: Detection results. ID: the ID of the used video sequence. Augmentation: the type of the used augmentation. AP: average precision score. FP: number of false positive detections.

6.1.1 Examples of Detections Difficulties

An example of detections in sequence 1c can be seen in Figure 6.1. It shows the detection output of the same image from both detection models i.e. extended Augmentation (top image) and baseline Augmentation (bottom image). This Figure shows a decrease in the number of FP detection in the dark area of the scene (Orange arrows) and the brighter area of the scene (Magenta arrows). The green arrow shows a mutual FP detection between both modes, in which the feather and teal shape of the turkey cause FP detections, similar FP detections can be seen in sequence 3 detections.

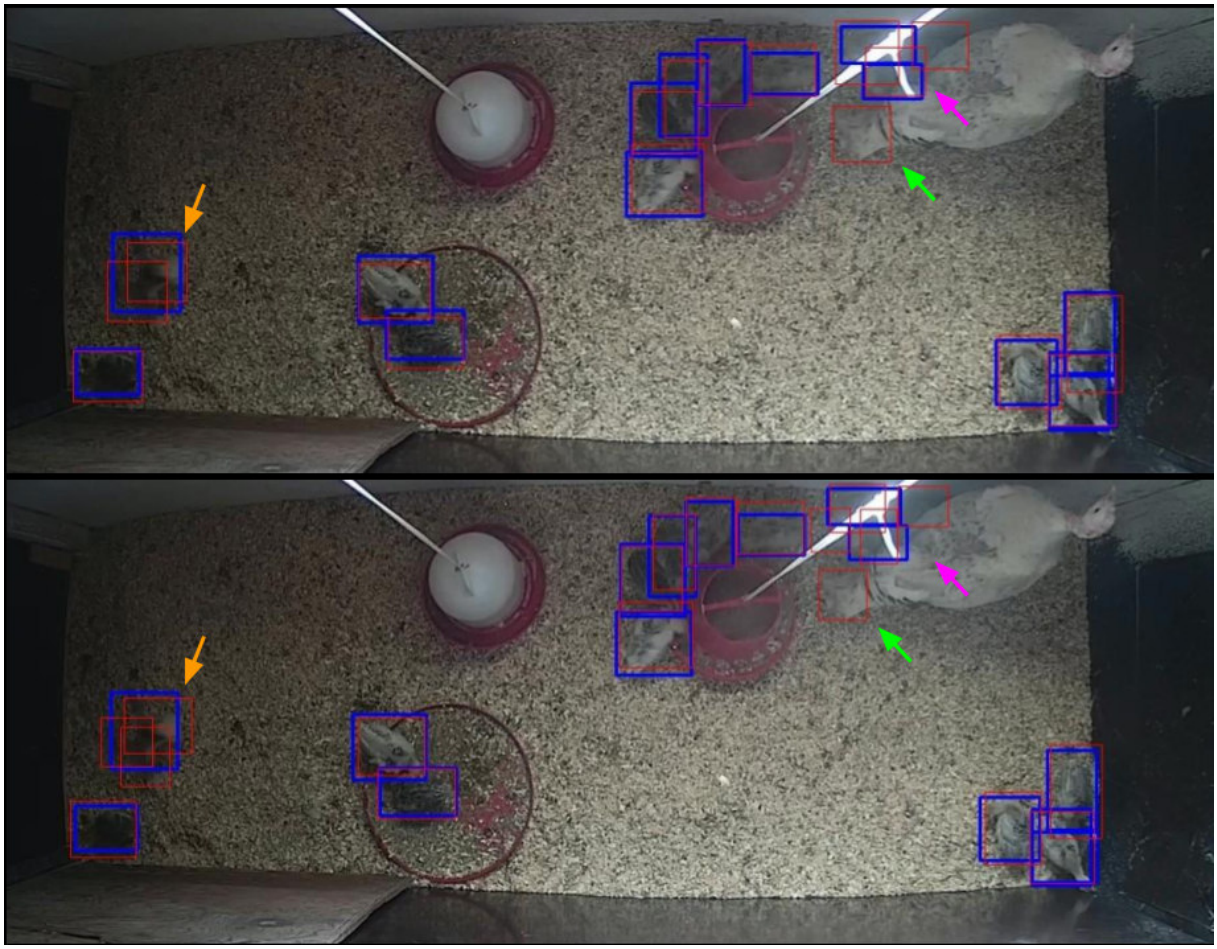

Figure 6.1: An example of the detections in sequence 1c. Top image show detection using extended Augmentation, bottom image show detection using baseline Augmentation. Blue rectangles are ground truth. Red rectangles are model detection's. Orange arrows show the location of chicks in the shadow. Magenta arrows show the location of partially occluded chicks. Green arrows show the location of a mutual FP detection between both models. ©IPI, TiHo Hannover

An example of detections in sequence 3 can be seen in Figure 6.2. It shows that the condition of the barn namely the dirt on the floor causes false-positive detections. The adult turkey causes false-positive detections too, due to its black tail feathers. In the training sequences, the adult turkey has fewer of these black feathers in the tail area and the barn is cleaner with no visible dirt on the floor.

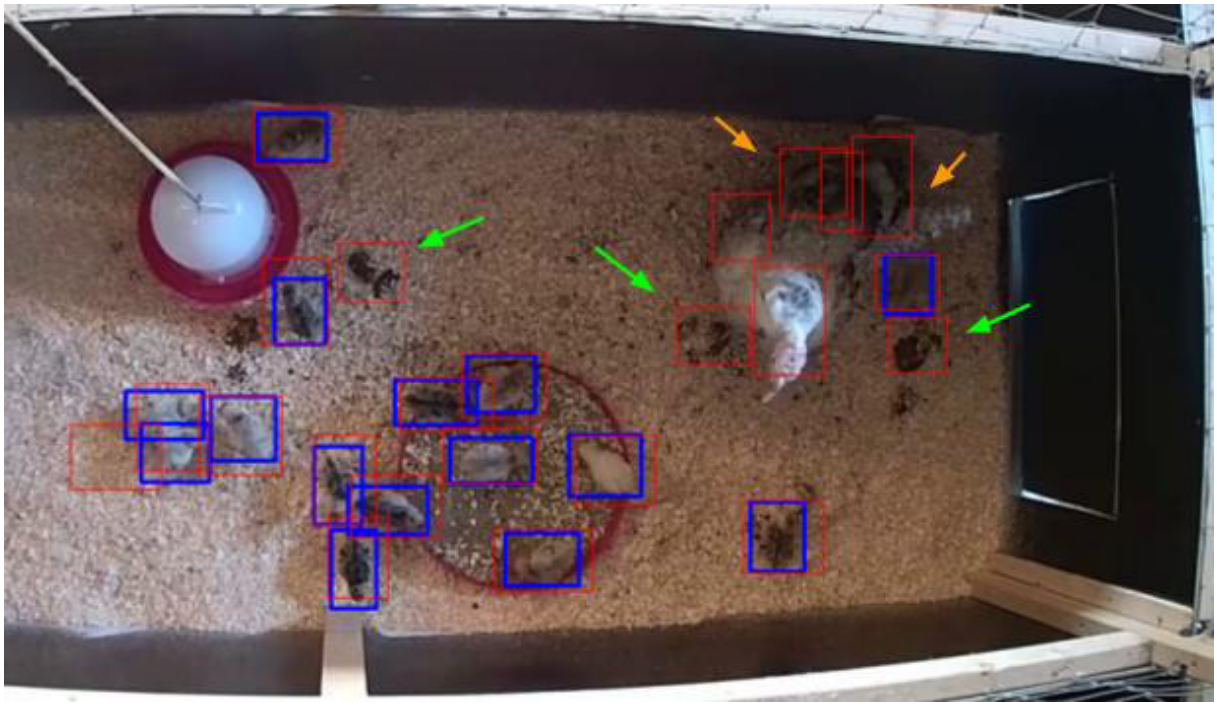

Figure 6.2: An example of the detections in sequence 3. Green arrows indicate dirt on the floor. Orange arrows indicate black feathers on the tail of the chicken. ©IPI, TiHo Hannover

Sequence 6b has shown the best results, an example of the detection in this sequence can be seen in Figure 6.3. The Figure shows how the size of chicks makes it easier to detect them even when they cluster together. Additionally, the colour of their feather makes it easier to distinguish them from each other.

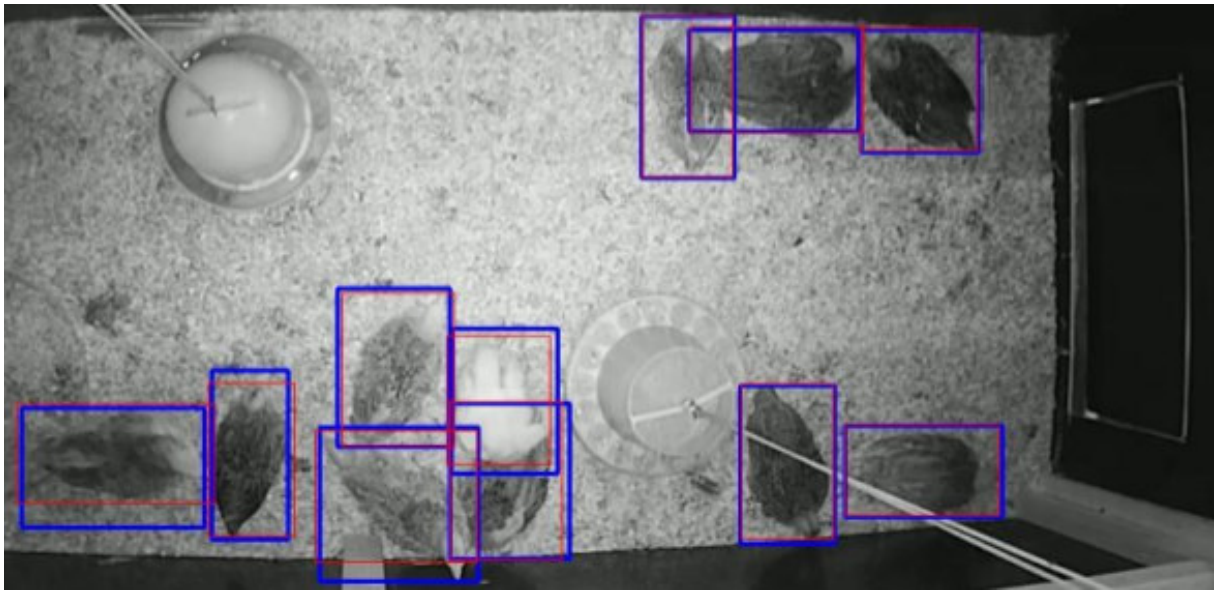

Figure 6.3: An example of the detections in sequence 6b. ©IPI, TiHo Hannover

Overall, the detection model with the extended augmentations shows a slight improvement compared to the detection model with the baseline augmentation. The fact that sequences 1c

and 6b have much better results compared to sequences 3 shows that increasing the variety of the training and validation data can improve the detection model. Additionally, since the detection model shows better results while detecting bigger chicks we believe that better adjusting the set of "anchors" that are used in the Faster R-CNN detection model can improve the detection of the chicks.

6.2 Tracking Results

For the six different tracking experiments, the results corresponding to the IDF1, MOTA and MOTP metrics are given for each of the test sequences 1c, 3 and 6b in Table 6.2. The results show that for sequences 1c and 6b the extended Augmentations model increased MOTA and IDF1 scores. In sequence 3 the MOTA and IDF1 scores have decreased or just did not change significantly like in the case of extended-MuDeep experiment, this decrease can be explained by the decrease in the AP score in the detection model for sequence 3 since the detections from the detection model are used in the tracking step. Additionally, the results show that using a re-identification model increase the MOTA and IDF1 scores, this increase is bigger in the case of MuDeep in competition to the Saimin ResNet-50 re-identification model.

Experiment ID	Sequence ID	IDF1↑	MOTA↑	MOTP↓
baseline-None	1c	58.5%	78%	0.23
	3	26.5%	9%	0.36
	6b	88.1%	95.5%	0.14
	mean	57.7%	60.8%	**0.2**
extended-None	1c	63.7%	81.2%	0.24
	3	23.8%	8.9%	0.37
	6b	92.8%	97.3%	0.14
	mean	60.1%	62.5%	0.21
baseline-ResNet-50	1c	64.3%	78.2%	0.23
	3	27.7%	9.4%	0.36
	6b	94.7%	95.7%	0.14
	mean	62.2%	61.1%	**0.2**
extended-ResNet-50	1c	69.6%	81.5%	0.24
	3	26.8%	9.3%	0.37
	6b	98%	97.4%	0.14
	mean	64.8%	**62.7**%	0.21
baseline-MuDeep	1c	69%	78.2%	0.23
	3	28.3%	9.4%	0.36
	6b	96%	95.7%	0.14
	mean	64.4%	61.1%	**0.2**
extended-MuDeep	1c	69.2%	81.5%	0.24
	3	28.4%	9.3%	0.37
	6b	98%	97.4%	0.14
	mean	**65.2**%	**62.7**%	0.21

Table 6.2: Results of all the tracking experiments. For each experiment, we report the scores of each video sequence. The last row of each experiment shows the average scores of all the three video sequences.

It can be seen that the MOTP score does not change by using a re-identification model, this is understandable since the MOTP score is an error metric that evaluates the accuracy of the placed BB around the tracked object, thus it is directly influenced by the detection model and not by the tracking model. MOTP score shows a slight increase for sequences 1c and 3, which means that the precision of the detection decreased by using the extended augmentation model. The reason why MOTP score of sequence 1c has increased yet its AP score has improved while using the extended augmentation model is the way how AP and MOTP are calculated. To calculate the AP score of the model the TP detections have to be counted. It should be noted that for a detection to be considered TP it has to have IoU score higher than a specific threshold which means it is just a binary decision, the BB is a correct detection or an incorrect detection, while in the case of MOTP metric the IoU metric is used to calculate the distance between the detection and gt. This means that BBs with an IoU slightly higher than the threshold are considered TP at the same time they increase the overall distance error thus increasing MOTP score.

The MOTA score, similarly to IDF1, shows an improvement when using the detection model with the extended augmentation. Unlike IDF1, there are no differences between Saimin ResNet-50 and MuDeep for re-identification; both extended-ResNet-50 and extended-MuDeep achieve an average MOTA of 62.7%. Compared to MOTA, IDF1 is better at expressing the consistency of ID matching, by measuring how long the identification is correct (Huang et al. 2020), which means that the use of MuDeep improves the ID matching of the tracked chicks. The reason why the tracking results of Sequence 3 are much worse than the results of the other two test sequences, i.e., around 30% worse in IDF1 and around 70% worse in MOTA compared to 1c, is the high number of false-positive detections. Those false-positive detections create false-positive tracks, which decreases the IDF1 and MOTA score of Sequence 3. It can be seen that the IDF1 score of sequence 3 is decreasing when using the extended augmentation model with no re-identification model or Siamin ResNet-50 re-identification model. Still, there is no noticeable decrease in the IDF1 score when using the extended augmentation model with MuDeep, this means that even with the decrease in the detection quality the MuDeep re-identification model is capable of re-identifying the detected chicks in the scene.

6.2.1 Examples of Tracking Difficulties

An ID switch (IDS) can acquire if two chicks are near each other or occluded by each other or by the turkey. Figure 6.4 shows an example of IDS's in three frames from sequence 1c. It should be mentioned that for visualisation purposes each tracked object is denoted with a BB with a unique colour when the BB of an object changes colour this means that an IDS has occurred. Figure 6.4 is split into two rows, the top row shows the extended-ResNet-50 model tracking a chick (denoted in the red rectangle in the top-left image). In the next image (top-middle image) the colour of the rectangle changes from red to green which means the model has lost the track of the chick from the first image and has assigned a new track to the same chick, this is due to the chick getting partially occluded by the turkey. In the last image (top-right) the same chick gets out from underneath the turkey but again gets assigned a new track. A similar problem

can be seen in the bottom row, which shows the extended-MuDeep model tracking the same chick. This model manages to track the chick correctly in the first two images (bottom-left and bottom-middle), an IDS acquires after the chick gets out from underneath the turkey. This shows that MuDeep improves the re-identification of the tracked chicks.

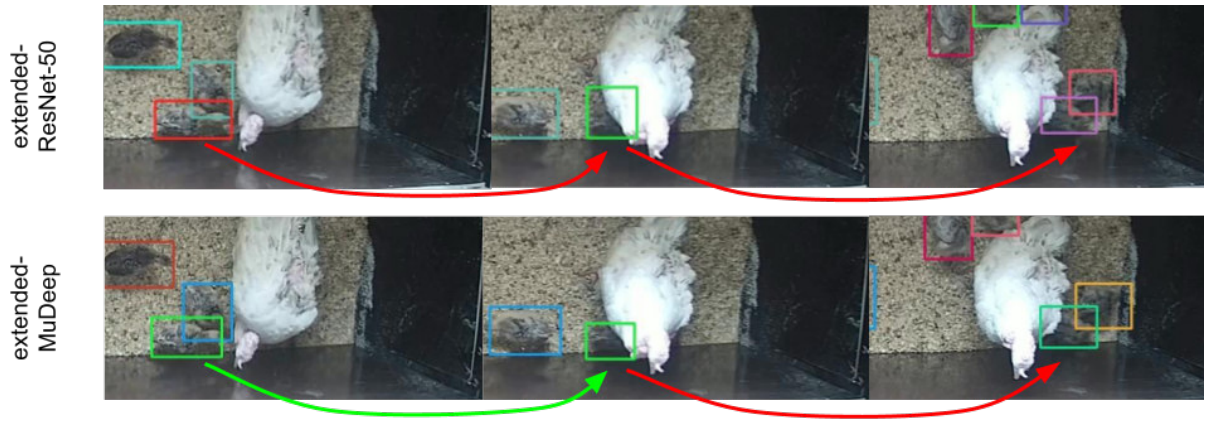

Figure 6.4: An example of IDS situation that can accrue. The top row shows the extended-ResNet-50 Model output, and the bottom row shows the extended-MuDeep Model output. The red arrows between the images of each row denote an IDS, and the green arrow denotes a correct re-identification of the chick. ©IPI, TiHo Hannover

Another example of an IDS can be seen in Figure 6.5. The Figure shows the influence of extended Augmentation in improving the tracking results in comparison to baseline Augmentation. Figure 6.4 is split into two rows, the top row shows the baseline-MuDeep model tracking two chicks (denoted in the red and blue rectangles in the top-left image), those two chicks get near each other in the next image, this causes their BB's to overlap, in the next image an IDS happens (denoted by the yellow arrow). The sequence of images can be seen in the bottom row, this row shows the extended-MuDeep model tracking the same two chicks but no IDS happen, this is due to the BB's in the extended model being more confined than the one outputted by the baseline model.

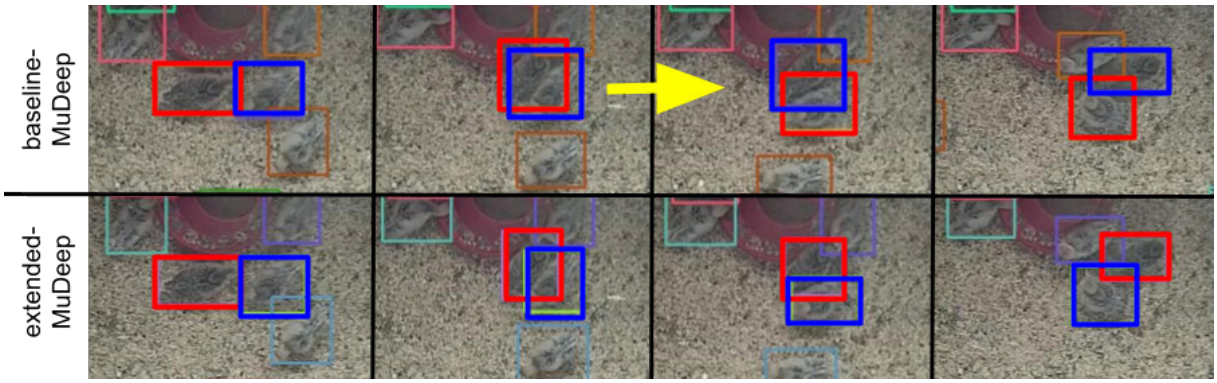

Figure 6.5: An example of the decrease in the number of IDS by improving the detection model.©IPI, TiHo Hannover

Overall the tracking results show that by using a re-identification model the IDF1 and MOTA scores increases furthermore the results show that the MuDeep improves the re-identification

of the tracked chicks. The example in Figure 6.5 shows that ID-switch can happen even while using a re-identification model, the main reason behind it is that the model does not use the re-identification model in that situation since no occlusion has happened. We believe that using the re-identification model in combination with a simple motion model that detects when the object has moved unexpectedly, like a sudden change in the direction can improve the track. In the example from Figure 6.5 this will be when the blue BB moved downwards to the right and then suddenly changed its direction and started moving upwards to the left since an ID-switch has occurred this could be detected by a motion model so that a re-identification model makes sure that the correct object is still getting tracked. Another way to improve the tracking results can be by exploiting the constant number of tracked chicks in the barn. In each video sequence, there is a constant number of chicks in the scene, thus if an occlusion occurs the re-identification model should compare the newly detected chick with a set of chicks that are not seen in the current frame. Additionally, a feature vector of each chick can be learned while tracking, this can improve the re-identification since each newly detected chick after occlusion has already been seen by the model and has a learned feature vector that identifies it.

6.2.2 The Influence of Tracktor on Detection

Table 6.3 shows how by using the tracktor approach the number of FP detection decreases. It shows a comparison between the number of PF detection while using the detection model i.e. Faster R-CNN and the number of PF detection while using the tracktor approach.

ID	Augmentation	#FP detection with the detection model	#FP detection with tracktor
1c	baseline	2321	1880
	extended	2178	1326
3	baseline	10846	10167
	extended	11866	10952
6b	baseline	463	298
	extended	411	219
mean	baseline	4543	4115
	extended	4818	4165

Table 6.3: Comparison between the number of PF detection while using the detection model and while using tracktor. ID: the ID of the used video sequence. #FP detection with the detection model: number of FP detection while using Faster R-CNN. #FP detection with tracktor: number of FP detection while using tracktor.

Table 6.3 shows a strong decrease in the number of FP detection for sequences 1c and 6b, this is understandable since the majority of the FP detection in those two sequences appear in the general area of the where the chicks are located like in Figure Figure 6.6. Those detections are ignored by the trackor due to the nature of how it initialises new bounding boxes (BB's). This improvement is because tracktor does not initialise new BB's if there IoU with any already existing BB's bigger than 0.3. The Figure shows the decrease in the number of FP detection between the Faster R-CNN outputs and Tracktor outputs. This is spatially noticeable in the models that use the extended Augmentation. In the right column, the number of BB's decreases

from two down to just one (Magenta arrow), this is due to the IoU between both the BB's is bigger than 0.3 thus just the BB with the highest classification score is used i.e. BB-1. In the left column, the baseline detection model is used thus there are more FP detections. The IoU between BB-4 and BB-5 is bigger than 0.3 and BB-5 has a higher classification score s at the same time IoU between BB-3 and BB-4 and the IoU between BB-3 and BB-5 are smaller than 0.3 thus the Tracktor tracks two separate BB's i.e. BB-3 and BB-5.

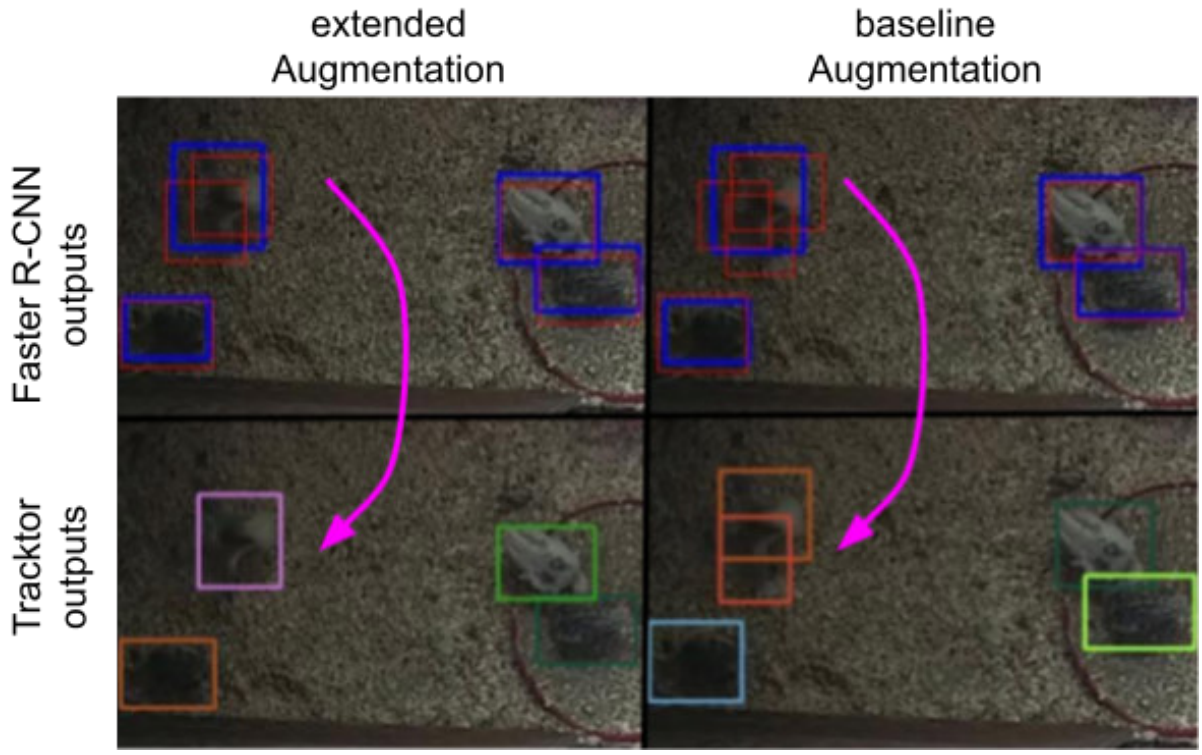

Figure 6.6: An example of the decrease in the number of FP detection in sequence 1c. The top row is Faster R-CNN outputs, and the bottom row is Tracktor outputs. In the right column, the baseline detection models are used, and in the left column, the extended detection models are used. The Magenta arrow shows the decrease in the number of FP detection between the Faster R-CNN outputs and Tracktor outputs. In the right column, just BB-1 is used for tracking and in the left column, BB-3 and BB-5 are used for tracking. ©IPI, TiHo Hannover

In the case of sequence 3 the consistent and the high number of false-positive detections in the turkey area create false-positive tracks, this is why the tracktor can not distinguish between FP detection and TP detection.

7 Conclusions and Future Work

It is of significant interest to track animals to analyse their behaviour and thus, improve their welfare. In this thesis, we presented an approach based on Tracktor to track poultry; Tracktor does not use a motion-model for tracking which makes it suitable for poultry tracking where a motion-model is not present. To improve the detection step we extended the used augmentation strategy by introducing Gaussian blur, brightness augmentation and horizontal flip to the used augmentation. Additionally, we use a multi-scale model to improve the re-identification of detected chicks that have been temporarily occluded. We evaluate the network and the designed methods on a novel dataset. The dataset was recorded by TiHo Hannover (Tierärztliche Hochschule Hannover) over a period of 48 days. This dataset is composed of a total of seven videos that were acquired during different times of the day and with varying lighting conditions. The dataset comprises grey and RGB video sequences.

The results showed improvements in the IDF1 and MOTA metrics while using the detection model with the extended augmentation in combination with the MuDeep re-identification model compared to the results delivered by the original tracktor approach without any tracking extensions. Overall, the detection model with the extended augmentations shows a slight improvement compared to the detection model with the baseline augmentation. The results show that the model has difficulties detecting the chicks when the barn condition is unfamiliar, like in the case of having multiple dirt spots on the floor, this is why we believe by increasing the variety of the used data can improve the detection model. Additionally, since the detection model shows better results while detecting bigger chicks we believe that better adjusting the set of "anchors" that are used in the Faster R-CNN detection model can improve the detection of the chicks. Currently, the biggest anchor can detect objects that are half the size of the whole frame, this is important in the case of pedestrian tracking since the body size of the tracked pedestrian changes proportionally to their distance from the recording camera. in the case of poultry tracing since the camera recording is mounted below the barn ceiling and results in nadir or oblique images the size of the chicks in the scene does not change drastically while moving thus the anchors-sizes can adjust more precisely to the size of the tracked chicks. Additionally, it is interesting to investigate the influence of different augmentation methods on the detection and tracking models in more detail, for example, in form of a more differentiated analysis of the improvements resulting from each individual augmentation method, and based on that, the introduction of new augmentation methods.

Tracking results show that by using a re-identification model the IDF1 and MOTA scores increases furthermore the results show that the MuDeep improves the re-identification of the tracked

chicks. One of the difficulties for the tracking model is that the re-identification model is used just to deal with occlusions, we believe that by using the re-identification model in combination with a simple motion model that detects when the object has moved unexpectedly, like a sudden change in the direction can improve the track. Another way to improve the tracking results can be by exploiting the constant number of tracked chicks in the barn. In each video sequence, there is a constant number of chicks in the scene, thus if an occlusion occurs the re-identification model should compare the newly detected chick with a set of chicks that are not seen in the current frame. Additionally, a feature vector of each chick can be learned while tracking, this can improve the re-identification since each newly detected chick after occlusion has already been seen by the model and has a learned feature vector that identifies it.

Bibliography

Agarap, Abien Fred (2018). "Deep Learning Using Rectified Linear Units (ReLU)". In: *CoRR* abs/1803.08375. arXiv: 1803.08375. URL: http://arxiv.org/abs/1803.08375.

Ali, Saad and Mubarak Shah (2008). *Floor Fields for Tracking in High Density Crowd Scenes*. Springer Berlin Heidelberg, 1–14. DOI: 10.1007/978-3-540-88688-4_1. URL: http://dx.doi.org/10.1007/978-3-540-88688-4_1.

Aydin, A. (2017). "Using 3D vision camera system to automatically assess the level of inactivity in broiler chickens". In: *Computers and Electronics in Agriculture* 135, pp. 4–10. ISSN: 0168-1699. DOI: https://doi.org/10.1016/j.compag.2017.01.024.

Bergamini, Luca et al. (2021). "Extracting Accurate Long-Term Behavior Changes From A Large Pig Dataset". In: *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. Vol. 4. SciTePress, pp. 524–533.

Bergmann, Philipp, Tim Meinhardt, and Laura Leal-Taixé (2019). "Tracking Without Bells And Whistles". In: *CoRR* abs/1903.05625. arXiv: 1903.05625. URL: http://arxiv.org/abs/1903.05625.

Bernardin, Keni and Rainer Stiefelhagen (2008). "Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics". In: *EURASIP Journal on Image and Video Processing* 2008.1, pp. 1–10. DOI: 10.1155/2008/246309.

Bolme, David S. et al. (2010). "Visual Object Tracking Using Adaptive Correlation Filters". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2544–2550. DOI: 10.1109/CVPR.2010.5539960.

Chen, Long et al. (2018). "Real-Time Multiple People Tracking With Deeply Learned Candidate Selection And Person Re-Identification". In: *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6.

Chien, Ying-Ren and Yu-Xain Chen (2018). "An RFID-Based Smart Nest Box: An Experimental Study of Laying Performance and Behavior of Individual Hens". In: *Sensors (Basel, Switzerland)* 18.

Colantonio, S. et al. (2007). "Object tracking in a stereo and infrared vision system". In: *Infrared Physics & Technology* 49.3, pp. 266–271. ISSN: 1350-4495. DOI: https://doi.org/10.1016/j.infrared.2006.06.028.

Colles, Frances M. et al. (2016). "Monitoring Chicken Flock Behaviour Provides Early Warning Of Infection By Human Pathogen Campylobacter". In: *Proceedings of the Royal Society B: Biological Sciences* 283.1822, p. 20152323. DOI: 10.1098/rspb.2015.2323. URL: http://dx.doi.org/10.1098/rspb.2015.2323.

Dalal, N. and B. Triggs (2005). "Histograms Of Oriented Gradients For Human Detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 886–893 vol. 1. DOI: 10.1109/cvpr.2005.177.

Girshick, Ross (2015). "Fast R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448.

He, Kaiming et al. (2016a). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

– (2016b). "Deep Residual Learning For Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

Helbing, Dirk and Péter Molnár (1995). "Social Force Model For Pedestrian Dynamics". In: *Phys. Rev. E* 51 (5), pp. 4282–4286. DOI: 10.1103/PhysRevE.51.4282. URL: https://link.aps.org/doi/10.1103/PhysRevE.51.4282.

Huang, Yanru et al. (2020). "SQE: A Self Quality Evaluation Metric For Parameters Optimization In Multi-Object Tracking". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8303–8311. DOI: 10.1109/CVPR42600.2020.00833.

Kashiha, Mohammadamin et al. (2013). "Automatic Identification Of Marked Pigs In A Pen Using Image Pattern Recognition". In: *Computers and Electronics in Agriculture* 93, 111–120.

Klinger, T., F. Rottensteiner, and C. Heipke (2014). "A Dynamic Bayes Network for visual Pedestrian Tracking". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XL-3, pp. 145–150. DOI: 10.5194/isprsarchives-XL-3-145-2014. URL: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XL-3/145/2014/.

Kuhn, H. W. (1955). "The Hungarian Method For The Assignment Problem". In: *Naval Research Logistics Quarterly* 2.1-2, pp. 83–97. DOI: https://doi.org/10.1002/nav.3800020109. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800020109. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109.

Li, N. et al. (2020). "Review: Automated Techniques For Monitoring The Behaviour And Welfare Of Broilers And Laying Hens: Towards The Goal Of Precision Livestock Farming". In: *animal* 14.3, 617–625. DOI: 10.1017/S1751731119002155.

Li, Wei, Xiatian Zhu, and Shaogang Gong (2018). "Harmonious Attention Network For Person Re-Identification". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2285–2294. DOI: 10.1109/CVPR.2018.00243.

Luo, Wenhan et al. (2022). "Multiple Object Tracking: A Literature Review". In: *CoRR*. arXiv: 1409.7618 [cs.CV].

Neethirajan, Suresh (2022). "ChickTrack – A Quantitative Tracking Tool For Measuring Chicken Activity". In: *Measurement* 191, p. 110819. ISSN: 0263-2241. DOI: https://doi.org/10.1016/j.measurement.2022.110819.

Ommer, Björn, Theodor Mader, and Joachim M. Buhmann (2009). "Seeing The Objects Behind The Dots: Recognition In Videos From A Moving Camera". In: *International Journal of*

Computer Vision 83.1, 57–71. DOI: 10.1007/s11263-009-0211-7. URL: http://dx.doi.org/10.1007/s11263-009-0211-7.

Qian, Xuelin et al. (2017). "Multi-scale Deep Learning Architectures for Person Re-identification". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5409–5418.

Redmon, Joseph and Ali Farhadi (2018). "YOLOv3: An Incremental Improvement". In: *CoRR* abs/1804.02767.

Ren, Shaoqing et al. (2015). "Faster R-CNN: Towards Real-Time Object Detection With Region Proposal Networks". In: *CoRR* abs/1506.01497. arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

Ristani, Ergys and Carlo Tomasi (2018). "Features for Multi-target Multi-camera Tracking and Re-identification". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6036–6046. DOI: 10.1109/CVPR.2018.00632.

Ristani, Ergys et al. (2016). "Performance Measures And A Data Set For Multi-Target, Multi-Camera Tracking". In: *CoRR*. arXiv: 1609.01775 [cs.CV].

Shafique, Khurram, Mun Wai Lee, and Niels Haering (2008). "A Rank Constrained Continuous Formulation Of Multi-Frame Multi-Target Tracking Problem". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. DOI: 10.1109/CVPR.2008.4587577.

Tang, Siyu et al. (2017). "Multiple People Tracking by Lifted Multicut and Person Re-identification". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3701–3710. DOI: 10.1109/cvpr.2017.394.

Wurtz, Kaitlin et al. (2019). "Recording Behaviour Of Indoor-Housed Farm Animals Automatically Using Machine Vision Technology: A Systematic Review". In: *PLOS ONE* 14.12. Ed. by Didier Raboisson.

Yu, Qian et al. (2018). "The Devil Is In The Middle: Exploiting Mid-Level Representations For Cross-Domain Instance Matching". In: *CoRR*. arXiv: 1711.08106 [cs.CV].

Zhang, Lei et al. (2019). "Automatic Individual Pig Detection and Tracking in Pig Farms". In: *Sensors* 19.5, p. 1188. DOI: 10.3390/s19051188. URL: http://dx.doi.org/10.3390/s19051188.

Zhao, Zhong-Qiu et al. (2019). "Object Detection With Deep Learning: A Review". In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11, pp. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.