FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Analyse des Balztanzes des Golden-Collared Manakin von Videos

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Visual Computing

eingereicht von

Anna Gostler, BSc
Matrikelnummer 01129406

an der Fakultät für Informatik
der Technischen Universität Wien
Betreuung: O.Univ.Prof. Dipl.Ing. Dr.techn. Walter Kropatsch

Wien, 9. Oktober 2019

Anna Gostler	Walter Kropatsch

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Analyzing the Courtship Dance of the Golden-Collared Manakin from Videos

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Visual Computing

by

Anna Gostler, BSc
Registration Number 01129406

to the Faculty of Informatics

at the TU Wien

Advisor: O.Univ.Prof. Dipl.Ing. Dr.techn. Walter Kropatsch

Vienna, 9th October, 2019

Anna Gostler	Walter Kropatsch

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.ac.at

Erklärung zur Verfassung der Arbeit

Anna Gostler, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 9. Oktober 2019

Anna Gostler

Acknowledgements

We would like to thank Prof. Dr. Leonida Fusani and his team at COGBIO for providing the annotated manakin videos and this interesting problem.

Kurzfassung

Die Goldbandpipra (Manacus vitellinus, engl. golden-collared manakin) ist eine tropische Vogelart, bei der die Männchen akrobatische Balztänze aufführen. Um verschiedene Balztänze zu vergleichen, filmte ein Team von Biologen die Vögel in freier Wildbahn mit Hochgeschwindigkeitskameras. Um den Balztanz zu analysieren müssen die Vögel zuerst getrackt werden, um anschließend das Verhalten zu klassifizieren und schließlich zu visualisieren. Das manuelle Labeln jedes Einzelbilds in studenlangem Videomaterial ist ein sehr zeitintensiver Vorgang. Automatisches Tracking und Verhaltenserkennung ermöglichen eine schnellere Analyse von Videos, die monatelanges Annotieren ersparen kann. In dieser Arbeit präsentieren wir einen umfassenden State-of-the-Art Report, analysieren die Herausforderungen der Manakin-Videos und präsentieren einen Tracker (ManakinTracker), der die spezifischen Herausforderungen der Manakin-Videos bewältigen kann. Basierend auf der Trajektorie erkennen und visualisieren wir das Verhalten des Vogels während des Balztanzes. Die Manakin-Videos stellen verschiedene Herausforderungen an visuelles Tracking und automatische Verhaltenserkennung. Die schnelle und abrupte Bewegung des Vogels führt zu starken Bewegungsunschärfen und ist schwer vorherzusagen. Das Aussehen des Vogels ändert sich stark. Zusätzlich ähnelt der Hintergrund optisch dem Vogel und verdeckt ihn ganz oder teilweise. Der ManakinTracker findet potenzielle Boundingboxen mittels Hintergrundsubtraktion, modelliert das Erscheinungsbild des Vogels mit einem neuronalen Netzwerk und lernt ein Bewegungsmodell. Der ManakinTracker kann erkennen, wenn der Vogel das Bild verlässt und ihn wieder erkennen sobald er wieder im Bild erscheint. Basierend auf der Trajektorie identifizieren wir typische Verhaltensweisen des Vogels: Hocken, Springen, “Beard-up”-Pose und Flügelschlag. Das Verhalten wird auf zwei Arten visualisiert. Wir vergleichen unseren Tracker mit 11 state-of-the-art Trackern in Bezug auf Robustheit und Genauigkeit und analysieren Fälle in denen das Tracking fehlschlägt.

Abstract

The golden-collared manakin (*Manacus vitellinus*) is a tropical bird species, in which the male performs an acrobatic displays to court mates. To be able to compare different courtship displays and better understand the courtship dance, biologists recorded the birds in the wild with high-speed cameras. To analyze the courtship dance the birds need to be first tracked, so that the behavior can be classified, and finally visualized. Manually labeling every frame in hours of video material is a time-consuming process. Automatic tracking and behavior recognition enables faster analysis of videos, which would save human annotators months of work. In this thesis, we present a thorough state-of-the-art review and highlight the challenges of the manakin videos. The manakin videos present several challenges for visual tracking and behavior recognition. The bird's rapid and abrupt movement causes strong motion blur and is hard to predict. The bird's appearance changes strongly. Additionally, background clutter visually resembles and occludes the bird. The ManakinTracker is a visual long-term tracker designed to handle the challenges of the manakin videos. The ManakinTracker finds potential bounding boxes with background subtraction, models the bird's appearance with a convolutional neural network and learns a motion model. It is able to detect the bird moving out of the frame and re-detect it. Based on the trajectory obtained through the ManakinTracker, we identify the bird's typical courtship behaviors: perching, jumping, beard-up posture, and wing-snap. The behavior is then visualized by plotting the trajectory and in a sequence plot. We compare our tracker to 11 state-of-the-art trackers in terms of robustness and accuracy and perform an analysis of tracking failures.

Contents

Kurzfassung **ix**

Abstract **xi**

Contents **xiii**

1 Introduction **1**

2 Datasets: Properties and Tasks **5**
2.1 Manakin Datasets 5
2.2 Groundtruth annotations 7

3 Related Work **13**
3.1 Video Tracking 13
3.2 Traditional Trackers 15
3.3 Deep trackers 22
3.4 Behavior Recognition and Representation 30

4 Tracking and Recognizing the Behavior of the Manakin **37**
4.1 Visual Tracker 37
4.2 Automatic Behavior Recognition 43
4.3 Visualization of Trajectory and Behavior 44

5 Evaluation **47**
5.1 Visual Tracking 47
5.2 Behavior Recognition and Representation 52

6 Conclusion **57**

List of Figures **59**

List of Tables **63**

Bibliography **65**

CHAPTER 1

Introduction

The golden-collared manakin (*Manacus vitellinus*) (see Figure 1.1) is a small tropical bird, which lives in the Panama forest. The males perform elaborate, acrobatic displays to court mates [FFG+17]. During its courtship dance the male demonstrates its physical strength by jumping (not flying) between saplings. Mating success seems to be related to superior motor skills [BSWF11], which allow the male to execute its dance faster and more precisely. However, it is not fully clear yet how exactly the courtship dance has to be performed to impress a female. To be able to compare different courtship displays, in order to better understand the courtship dance, biologists recorded the birds in the wild with high-speed cameras. The biologists need to analyze a large number of high-speed recordings of the bird's courtship dance. To achieve this the birds need to be first tracked, so that the behavior can be classified, and finally visualized.

Figure 1.1: Close-ups of male (from left: first, second) and female (from left: third, fourth) golden-collared manakins

Manually labeling every frame in hours of video material, however, is a time-consuming and tedious process. When the person who labels the videos becomes inattentive errors can occur.

Automatic tracking and behavior recognition enables faster analysis of videos, which would save human annotators months of work. It also enables researchers to work with much larger datasets, which makes their research findings more robust and thus more statistically significant.

The contributions of this thesis are as follows:

(1) We present a thorough state-of-the-art review and highlight the challenges of the manakin videos.

(2) We propose a novel tracker (ManakinTracker) that can handle the specific challenges of the manakin videos.

(3) Based on the trajectory we recognize and visualize the bird's behavior during the courtship display.

The manakin videos present several challenges for visual tracking and behavior recognition. The main challenge is the **fast movement** of the bird which causes **strong motion blur** (see Figure 1.2). Additionally, the bird's movement is hard to predict because of **sudden stopping and starting**. The bird's **appearance changes** abruptly when it starts or stops moving or when it changes its orientation. While animals are typically recorded in laboratory conditions with little to no background clutter, the manakin was **recorded in the wild**. The bird moves inside the so-called arena, which consists of a set of saplings. These saplings often occlude the bird while it sits on a sapling or jumps behind one. Leaves can also occlude the bird or confuse the tracker as some of the yellow leaves visually resemble the bird's yellow neck.

Figure 1.2: Images of birds affected by strong motion blur.

To track the bird in the manakin videos, we present the ManakinTracker: a visual long-term tracker designed to handle the challenges of the manakin videos. The tracker is initialized with a bounding box in the first frame marking the bird's location. For every following frame, the tracker outputs a bounding box. The ManakinTracker consists

of three main components: candidate generation, an appearance model and a motion model.

Candidate generation finds potential bounding boxes in the current frame. We use a background subtraction method (Mixture of Gaussians) to find moving blobs in the frame, as well as a search window around the previous position.

The appearance model is used to assign a classification score to the candidate bounding boxes. It is implemented as a convolutional neural network (CNN) pre-trained on ImageNet and image patches cropped from manakin videos with ground truth bounding boxes (see Section 2). The CNN classifies an image patch as either bird or background.

We use the motion model to predict the bird's movement independent of visual cues. The motion model is used to track the bird through occlusions and to decide between multiple candidates that are classified as bird. The motion model is implemented as a Kalman filter [WB95].

The ManakinTracker detects that the bird has left the frame if the predicted bounding box is partially outside of the frame. The bird is re-detected by scoring image patches along the frame's border and blobs that occur inside the border region of the frame.

Based on the trajectory obtained through the ManakinTracker, we recognize the bird's behavior. We focus on a pose called "beard-up", in which the male presents its yellow neck while perching. The biologists aim to find out if the male points its neck towards the female bird. We perform behavior recognition based on different information extracted from the videos: from image patches we find the largest yellow region in order to detect the yellow neck; based on the trajectory we determine if the bird is jumping or perching and based on the change of the bounding box area we determine if the bird does a wingsnap. The behavior is then visualized by plotting the trajectory enhanced with symbols indicating the length of perching, the presence of a wingsnap and the view to which the yellow neck points. Additionally we plot the recognized behaviors in a sequence plot.

We compare our tracker to 11 state-of-the-art trackers in terms of robustness and accuracy. To gain a better understanding of the weak points of the ManakinTracker, we perform an analysis of tracking failures.

CHAPTER 2

Datasets: Properties and Tasks

2.1 Manakin Datasets

We use two datasets that contain 71 and 12 color video sequences, respectively. The videos in both datasets show male golden-collared manakins performing their courtship dance. All videos were recorded by a team of biologists with three synchronized, stationary high-speed cameras at 60 fps in the Panama forest and have per-frame groundtruth annotations.

	Dataset-2017	Dataset-2018
number of sequences	71	12
camera	Basler acA1920-155uc	Basler acA1920-155uc
recording time	March – April 2017	February 2018
frames per second	60	60
exposure time (in seconds)	1/10000	1/10000
frame size (W x H)	1928 x 1208 px	1936 x 1216 px
avg. groundtruth boundingbox (W x H)	113 x 95	135 x 114
number of frames with groundtruth	15599	16329
average image brightness	79	51
average image saturation	0.38	0.41
average image sharpness	0.0058	0.0041
avg. # of bird leaving frame per seq.	0.25	1.5

Table 2.1: Manakin Datasets

Dataset-2017

	average	variance	minimum	maximum
width	112.5722	1528.0501	38	274
height	94.9747	658.8979	37	242
area	10894.2804	33037442.1328	2236	55826
change of area between two frames	1.006	0.006173	0.54677	2.0142
distance of centers* between two frames	27.952	729.7115	0.25	158.6295
overlap ratio (IoU**) between two frames	0.8245	0.078581	0	1

Table 2.2: Properties of groundtruth bounding boxes (in dataset-2017).
*Only distances > 0 were used. **IoU: Intersection over Union [KLM+18]

Dataset-2018

	average	variance	minimum	maximum
width	135.2468	2343.4884	36	384
height	113.6805	1326.8796	32	409
area	16132.5685	89197562.9657	2438	131698
change of area between two frames	1.0361	0.12163	0.084038	8.9182
distance of centers* between two frames	31.1915	1107.7757	0.16346	223.8666
overlap ratio (IoU**) between two frames	0.828	0.07253	0	1

Table 2.3: Properties of groundtruth bounding boxes (in dataset-2018).
*Only distances > 0 were used. **IoU: Intersection over Union [KLM+18]

2.1.1 Comparison of Manakin Datasets

We compare our two datasets in tables 2.1, 2.2, and 2.3. The videos in both datasets have about the same frame size and were recorded at the same frame rate (60 fps) and exposure time (1/10000 s) with the same high-speed camera model. Dataset-2017 was recorded the year before dataset-2018.

The pixels in dataset-2017 are on average $79/51 \simeq 1.5$ times brighter and slightly less saturated than the videos in dataset-2018. Image saturation was calculated by transforming the images to the HSV (which stands for hue, saturation, color) color space and averaging the saturation component per pixel. We calculated image quality in terms of average sharpness using the image quality measure described in [DM13] for the patches cropped from the groundtruth bounding boxes: dataset-2017 reaches a higher value, i.e. the bird is imaged sharper than in dataset-2018. This is most likely caused by the

brighter lighting conditions under which dataset-2017 was recorded.

The average area of the groundtruth bounding boxes is 1.4 larger in dataset-2018, because – in contrast to dataset-2017 – the bounding boxes in dataset-2018 also include the bird's wings. The average groundtruth bounding box covers about 0.46% and 0.65% of the frame area in dataset-2017 and dataset-2018, respectively.

2.2 Groundtruth annotations

The videos of both datasets have groundtruth annotations in the form of axis-aligned bounding boxes. The annotations were manually created with a tool that allows the user to place a rectangle over the bird. The annotations of dataset-2017 were generated by a team of biologists on every second frame using *loopy*[1]. The annotations on dataset-2018 were created by myself on every frame.

Even for human annotators it is difficult to accurately draw bounding boxes around the birds: the strong motion blur smears the bird into an elongated shape, which does not represent the bird's true shape. Even for humans it can be hard to distinguish the bird against the background – under very strong motion blur the bird can even be barely visible.

There is some uncertainty concerning which part of the bird's body to include: wings when extended and feet are not always included. Wings often not visible. While the biologists did not include the wings, I did because it is useful to recognize some of the bird's behaviors (wingsnap, flip). When a part of the bird is occluded, the annotator has to guess the bird's shape behind the occluding object. If an annotator is inattentive even for only one frame and incorrectly selects the bounding box of the previous frame: if the bird is moving there might be little to no overlap with a correct prediction leading to tracking failure. Especially during periods where the bird is sitting, annotators had a tendency of keeping same location even if small movements occur. Every video was annotated only once, otherwise it would have been possible to find errors by comparing different annotations for the same frame.

2.2.1 Challenges of the Manakin Dataset

The following properties of the videos make tracking the birds and identifying their behavior challenging:

Ch1: No Markers: In order to not interfere with the manakin's behavior the biologists did not place markers on the birds.
Ch2: Speed: While jumping, the bird moves very quickly.
Ch3: Motion blur: Motion blur is an artifact where objects in an image appear streak-like because the camera integrates all positions of the object along its trajectory during exposure time. These streaks are characterized by their length and angle. Motion

[1] http://loopbio.com/loopy

Figure 2.1: Example images from dataset-2017. Left: perching bird; right: jumping bird

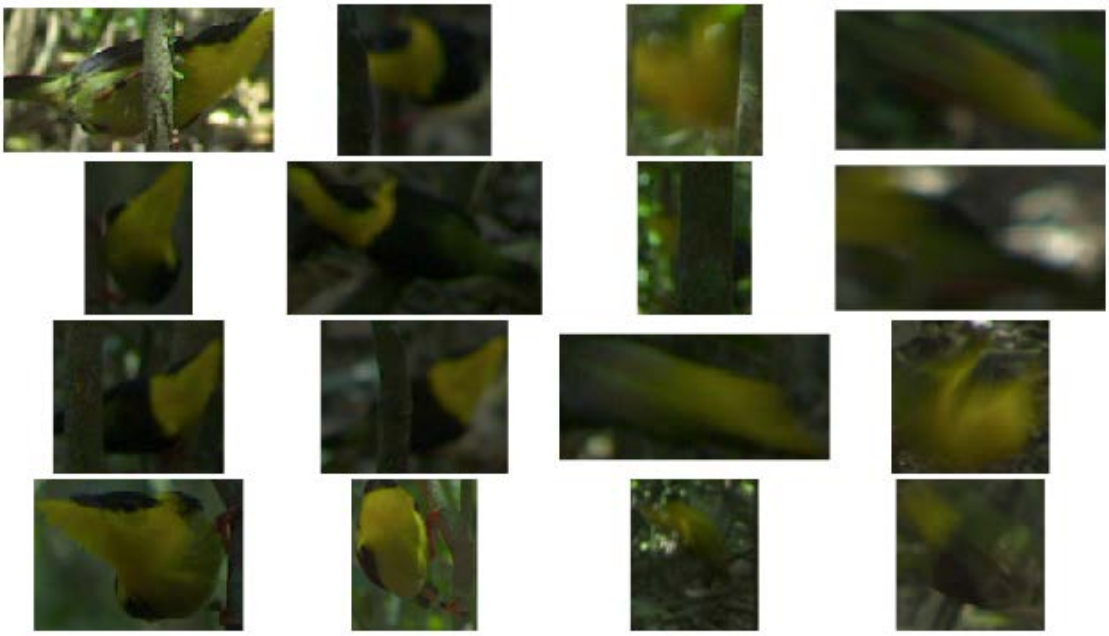

Figure 2.2: Example images from dataset-2018. Left: perching bird; right: jumping bird

blur is caused by the relative movement of the camera and objects in the recorded scene. In our case, the camera is static and the motion blur results from the objects moving in the scene. Figure 1.2 shows cases where the ManakinTracker fails due to strong motion blur. Strong motion blur can make the bird hard to recognize as it loses most of its local features.
Ch4: Size and shape change: The bird's bounding box changes in size and shape, especially when the bird opens its wings or turns.
Ch5: Appearance change: Appearance changes occur due to motion blur, when the bird jumps, does out-of-plane rotations or because of illumination changes and shadows.
Ch6: Occlusion: The bird can be partly or fully occluded by saplings and leaves.
Ch7: Out of frame: The bird frequently leaves the camera's field of view. Thus, the tracker has to deal with a target that disappears and should not erroneously track a different object while the bird is out of frame. When the bird re-appears at an unknown location the tracker should detect it again and resume tracking.
Ch8: Trajectory: The bird starts and stops abruptly and typically changes direction when starting a new jump. During a jump, however, the bird's trajectory is smooth.
Ch9: Background color: The forest is similar in color to both the male and female bird.
Ch10: Background motion: Leaves and branches move in the background. Saplings often move when the bird lands on them.

2.2.2 The Golden-collared Manakin

The golden-collared manakin (*Manacus vitellinus*) is a small Passerine bird that is about 10–11 cm long and weighs around 20 grams. It lives in the tropical forest of Panama. The males perform an acrobatic courtship dance [OFE13]. The golden-collared manakin is a sexually dimorphic species: the males are bright and colourful; while the female is well camouflaged with an olive-green body, darker wings and a black beak [C$^+$35]. The male also has an olive-green rump and black beak but the wings and the top of the head is black. Its most striking feature is the golden collar that is colored in a shade of yellow. Part of the golden collar is the so-called "beard": long feathers (about 20 mm) attached to the base of the beak. The bird can extend the beard using muscles [C$^+$35, FGDS07, SDF08]. Although the male is brightly colored, it is also camouflaged by a disruptive pattern in the feathers [C$^+$35]. The golden-collared manakin is a model species for studying sexual selection, as this bird uses both of the main mechanisms of sexual selection: male-male competition and female choice.

2.2.3 The Golden-Collared Manakin's Courtship Dance

The male golden-collared manakin performs a spectacular courtship dance to court females. The breeding season lasts from January until July or August [C$^+$35, SDF08]. The display takes place in the "arena": a roughly elliptical region on the forest ground delimited by vertical saplings. One of the saplings, the "mating sapling", is used for copulation at the end of the display if the female is willing to mate [C$^+$35, FGDS07].

The male cleans the floor of the arena, removing branches and leaves. The clean floor makes the male's golden beard more visible during the display [Per17].

The courtship display consists of a sequence of jumps and wingsnaps. The display starts with the male entering the arena from the top. He sits on a sapling then jumps (not flies) very fast to another sapling. Mid-jump he does the wingsnap by lifting the wings. While landing, he opens the wings to turn his body.

As soon as he has landed on the sapling, he assumes a rigid posture – similar to a gymnast landing after a jump – where he presents his beard ("beard up"), staying in this position until he does the next jump. In total he jumps between 1 and 20 times, depending on whether a female is watching. The last step of the courtship display is the ("grunt-jump"): the male does a jump from the "mating sapling" to the ground including a wingsnap. He lands on the ground doing a cartwheel ("flip"), then goes into the "beard up" posture. After holding the position for a short while, he returns to the "mating sapling" making a "grunt sound" by flapping the wings. On the sapling some males slide a few centimetres downwards, flapping the wings. Each male develops and practices his own version of the courtship display [Per17].

In the beginning of the displays, the female bird watches the males, sitting on a tree above the arena. In the course of the displays, she sometimes joins one of the males. This behaviour is called "duo dance" and serves the purpose of observing the male's performance more closely. The female does not display herself, i.e. she neither jumps, instead she flies between the saplings, nor makes wingsnaps. She always lands on the sapling opposite the male. The female usually flies first, setting the pace, and the male follows. In a "duo dance" the male jumps faster than in a display without a female and shows more movements in order to impress the female by demonstrating his motor skills. At the end of the display, she decides whether she wants to mate. If she does, she stays on the "mating sapling" and the male slides on top of her to mate; otherwise she leaves [C+35, FGDS07, BSF15].

2.2.4 Behaviors During Courtship Dance

The biologists have observed distinct behaviors that the male golden-collared manakin shows during his courtship display. The behaviors can also coincide: for example, the bird performs a wingsnap while doing a flip or a jump.

B1: On perch. The time the bird spends perching (sitting) on a sapling between jumps indicates to the female how much rest he needs before being able to do another jump. A short time on perch correlates highly with mating success [BSWF11].

B2: Jump. The male bird does not fly but jumps between the saplings. Mid-jump it performs the wingsnap. Biologists suspect that the speed of the jump might be related to mating success [Per17].

B3: Wingsnap. The wingsnap is a sonation, i.e. a sound that is not produced by the vocal chords but mechanically with other body parts. To make the wingsnap, the

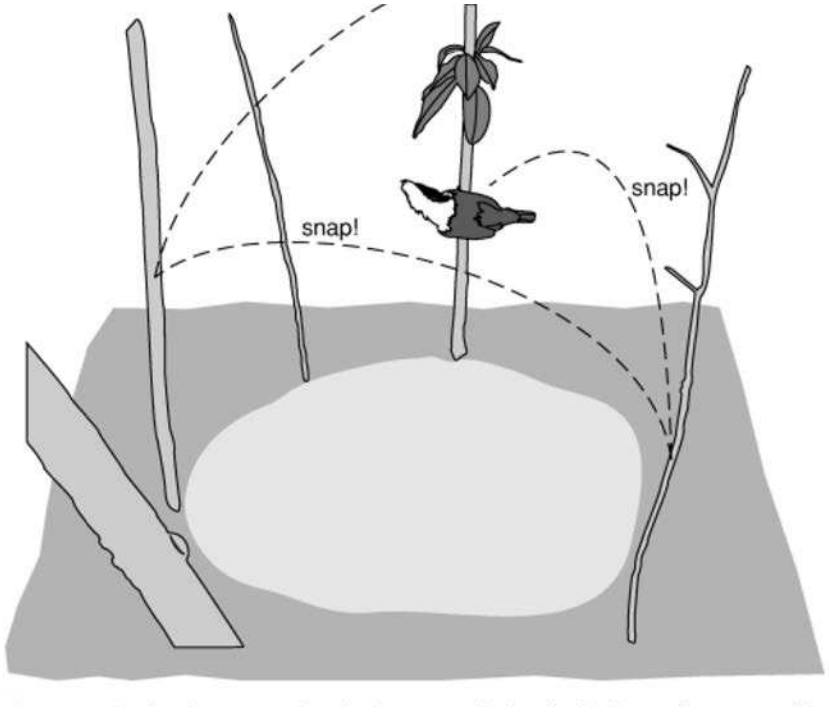

Figure 2.3: This drawing shows the golden-collared manakin's courtship dance (image from [FGDS07])

golden-collared manakin lifts his wings over his back, making a loud sound when they touch [SBD+13]. The wingsnap is done in the middle of every jump and sometimes while the male is sitting [Per17]. Biologists are interested in the wingsnap rate (number of wingsnaps per display) and the moment in which the bird does the wingsnap during the jump [Per17].

B4: Beard-up. After landing the bird assumes an upright posture where he presents his beard. The challenge for him is to skillfully slow down after landing a jump so that he can assume this position quickly. This movement requires neuromuscular coordination. The males vary in the time needed to assume the statuary, beard up posture – males with better coordination can do this movement faster [Per17]. Besides the time needed to assume this posture, the biologists are interested in whether the male points the beard directly at the female.

B5: Flip. While the bird jumps down from the mating sapling it does an unusual movement called the "flip": he rotates in a back-flip that requires high neuromuscular control similar to the beard up posture, since the bird needs to stop abruptly as he lands [FGDS07]. The flip also includes a wingsnap making it even more difficult. Perinot et al. [Per17] suspect that the flip plays a role in the female's decision making because it is such a challenging move to perform, but did not find much variance between birds concerning the duration of the flip.

CHAPTER 3

Related Work

3.1 Video Tracking

Video tracking is used for autonomous driving, surveillance or sports analysis. Animals are tracked for behavioral studies, pharmaceutical research, and wildlife counting [NNK+17].

A video tracker has five main parts [MC11]:

- **Feature Extractor:** extracts information from an area containing the target
- **Target Appearance Model:** representation of the target appearance that allows the tracker to distinguish between the target and the background
- **State Propagation:** method to use information obtained in previous frames
- **Target Appearance and Disappearance Management:** deals with target leaving the frame and discovering the target when it enters the frame
- **Meta-data Extractor:** extracts information to be used in the application for which the tracker is employed

3.1.1 Tracking: Relevant Terms

In this section we define terms that are relevant for video tracking.

Tracking Task: In this work we only consider visual tracking, i.e. we only use visual information provided by videos. The task of tracking is to locate an object (also called target or target object) in all the frames of a video. The tracker is initialized with the object's position in the first frame, usually a rectangular bounding box. For all subsequent frames, the tracker outputs the location of the target object [KLM+18].

Location: We use the term location to refer to a bounding box that encloses the target object in a given frame. This bounding box is defined by the x and y coordinate of its upper left corner and its width and height in pixels.

Detection: is the task of locating an object in an image, given only a single image and no information from prior frames. If the target object leaves the field of view, it also has to be detected when it re-appears.

Target (Object): The object that is being tracked.

Search Window: A region inside the current frame where the tracker looks for the target object.

Drift: occurs when the tracker slowly loses the target caused by the tracker erroneously learning the background's appearance as the target appearance [BBDL10a].

Single-object Tracker / Multi-object Tracker: A single-object tracker tracks only one object, while a multi-object tracker tracks multiple objects in the same video. Simple multi-object tracking can be realized by initializing multiple single object trackers on different objects in the same video.

Off-line Learning: Training the tracker before tracking starts.

On-line Learning: During tracking the tracker is updated with information (about background and/or target) of previously seen frames.

Real-time Tracking: processes frames at the same rate that they are produced by the camera (about 60 frames per second). Real-time tracking is necessary for real-time applications such as surveillance.

Long-term Tracker / Short-term Tracker: A long-term tracker can re-detect a target and deal with occlusion, while a short-term tracker is not required to handle these challenges. A long-term tracker also needs to be able to switch between tracking and detection [KLM+18]. Long-term trackers are aimed at tracking longer videos than short-term trackers: the videos used in the VOT2018 [KLM+18] long-term challenge have 420 frames on average.

Object-specific Tracker: tracker developed to track a specific object e.g. the golden-collared manakin.

Generic Tracker: tracker developed to track arbitrary objects. The target object might not have been in the training set. Therefore the tracker's model might be unfamiliar with the appearance of the object that is being tracked. If the tracker learns online it only gets the frames of successful tracking to learn about the appearance of the target object in this video and – if the tracker makes use of background information – the background. If the generic tracker allows for off-line training, it can learn about the appearance of that object before and thus be made into an object-specific tracker.

3.1.2 Tracking Birds in Videos

Most computer vision methods to track birds work with videos recorded inside custom-built arenas [KvBL15, EGK+08, AQRSM12, RBB+11, OGS15] (see Figure 3.1) or on videos of birds flying against the sky or distant landscapes [SX08, LS12] (see Figure 3.2). Therefore, background segmentation and tracking is not a particular concern for these studies. A method to get more precise information about the bird's position is to equip them with body markers [RBB+11] (see Figure 3.3). To prevent a behavior change in the birds, however, the biologists refrained from applying such tracking markers onto the birds as it could potentially stress the birds or impact their attractiveness and thus change the female's response to the display [Per17].

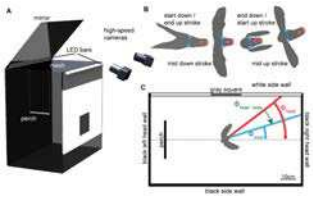

Figure 3.1: Birds recorded in a custom-built arena [KvBL15].

Figure 3.2: Birds recorded against a distant background [SX08].

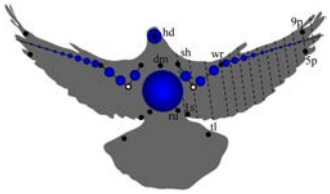

Figure 3.3: Pigeon with body markers [RBB+11].

3.1.3 Visual Tracking for Arbitrary Videos

The trackers presented above, which were developed specifically for birds, are not suited for the challenging conditions present in the manakin dataset. Much more research has been done to propose generic trackers which can be applied to any target object. Trackers can be placed into two main categories: traditional trackers based on hand-crafted features and deep trackers based on deep learning.

3.2 Traditional Trackers

Traditional trackers use hand-crafted feature extractors and traditional machine learning methods. By using features that are fast to compute, traditional trackers can achieve faster run-times than deep-learning based trackers. Also, they need less space as they do not need to store neural networks, which can be very large files. This typically makes traditional trackers better suited for a resource-constrained environment.

3.2.1 Boosting

The boosting tracker's [GGB06] main component is a binary classifier based on the AdaBoost algorithm [OR01] that assigns a confidence score to an image patch classifying it as either object or background.

This classifier consists of a pool of weak classifiers. These weak classifiers are based on features that can be computed efficiently (Haar-like features [24], orientation histograms [16,21,9], local binary patterns [20]), which allows the tracker to run in real-time even though the search region is evaluated at multiple positions using a sliding window approach.

The classifier is trained online on every frame with a positive sample (the tracked object in the current frame) and negative samples (surrounding background). This enables the tracker to adapt to a changing target appearance and background, and to select the most suitable features under changing conditions.

Advantages of Boosting:

- adapts choice of features to changing tracking conditions
- real-time

Disadvantages of Boosting:

- no offline learning

3.2.2 Multiple Instance Learning (MIL)

The MIL tracker [BYB09] uses Multiple Instance Learning. Other trackers update their model using the new target position as a single, positive training sample. However, if the target location is inaccurate the classifier will be trained with an incorrectly labeled sample which might result in drifting.

The MIL tracker, on the other hand, crops a set of positive training samples around the new, estimated target position, where only some might have the correct label and updates a boosting classifier similar to AdaBoost [OR01] that can handle ambiguous training data. Using Haar-like features [VJ+01] allows the MIL tracker to run in real-time.

Advantages of MIL:

- can handle inaccurately labeled training data
- real-time

3.2.3 Median Flow

The Median Flow tracker [KMM10] is based on the Forward-Backward error and aims to find reliable trajectories. The Forward-Backward is determined by measuring the difference between the trajectory that results from tracking forward to the trajectory that results from tracking the same frames backwards. This method is also able to detect tracking failures.

The Median Flow tracker receives two consecutive frames and the bounding box for the first frame. Inside this bounding box points are sampled in a grid (see Figure 3.4). These points are tracked using the Lucas-Kanade Tracker [LK81] to obtain the sparse optical flow between the two frames. In this way, each point on the grid receives a potential trajectory, which is evaluated by applying the Forward-Backward error measure. In order to keep only reliable point trajectories, the 50% which received high error scores are removed. From the predicted points which are thus assumed to be reliable, the new bounding box is found by calculating the median of those points' coordinates.

Tracking points on an object assumes that these points stay at the same distance relative to each other, i.e. that the object is rigid. The Median Flow tracker can deal with points on non-rigid parts of the object by rejecting them with the error measure. A sufficient amount of rigid object parts, however, is still required for tracking. Additionally, a change in the object's size can also cause this tracker to fail.

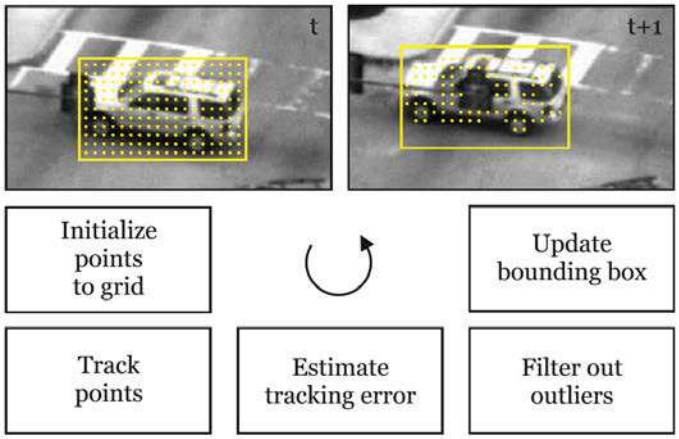

Figure 3.4: Block diagram of the Median Flow tracker. (image from [KMM10])

Advantages of Median Flow:

- can detect tracking failure

Disadvantages of Median Flow:

- relies on points on the target that stay consistent in brightness
- tracking performance suffers if target is not rigid
- cannot handle change in object's size

3.2.4 Kernelized Correlation Filter (KCF)

The kernelized correlation filter tracker [HCMB14] calculates the correlations between the target in the previous frame at the same image region in the current frame shifted vertically

and horizontally. Very fast run times (hundreds of FPS) are possible by expressing the shifts as cyclic shifts and calculating the correlations in the Fourier domain. The tracker is trained at each frame using the location that achieved the strongest correlation response. A disadvantage of using cyclical shifts is that they produce wrapped-around edges (see Figure 3.5).

Advantages of KCF:

- very fast
- calculates all possible correlations between two image patches

Disadvantages of KCF:

- does not keep a model of past appearances in frames before the previous frame; calculates correlation only between current and previous frame
- cyclical shifts create artifacts

Figure 3.5: Cyclical shifts produce erroneous wrapped-around edges. Note, for example, the ground that is below the cyclist in the middle frame was shifted above him in the left-most image. (image from [HCMB14])

3.2.5 Tracking, Learning and Detection (TLD)

The TLD tracker [KMM11] is aimed at long-term tracking, where an object that moves out of the field of view of the camera needs to be detected when it re-appears. Besides a tracking component, their tracking framework contains a detection and a learning component.

The tracking component, implemented as a Median Flow tracker, finds the object's location from frame to frame but fails when the object disappears, and is susceptible to drift. The detector scans every frame to localize previously observed objects. It detects the object when it re-enters the camera's field of view and can also correct the tracker.

The learning component monitors the tracking and detection components. It consists of two types of so-called "experts" that notice and correct errors in the detector's classifications: the P-expert and the N-expert.

The P-expert tries to find occurrences of the object that have been missed by the detector by using the trajectory provided by the tracker and turns them into positive training samples.

The N-expert tries to find examples of background that were misclassified by the detector. It takes all the patches suggested by the detector and assumes that only the most confident one depicts the tracked object, as the object can only be in one location at a time. The other suggestions, in case they do not overlap with the assumed correct location, are used as negative examples. With these positive and negative samples the detector is updated.

The TLD tracker runs in real-time.

Advantages of TLD:

- real-time
- long-term tracker: can handle occlusion and out-of-frame
- corrects itself

3.2.6 Minimum Output Sum of Squared Error (MOSSE)

The MOSSE tracker is based on the MOSSE filter [BBDL10b], which is a correlation filter. The MOSSE filter is learned from training images and corresponding outputs. These outputs are 2D Gaussians with the peak over the target's center in the respective training image. The filter is used in a tracker by setting the target location in the new frame to the position of the peak in the correlation filter output.

The filter is learned by minimizing the sum of squared error between the outputs produced by convolving the filter with the training images and the expected output for these training images:

$$\min_{filter} \sum_{i} |(filter \circledast training\ image_i) - training\ output_i|^2$$

This is realized by first computing the exact filter for each training image, i.e. the filter that produces exactly the training correlation output when applied to the training image. During tracking, the filter is updated by computing the moving average of the exact filters that were determined for the previously tracked target objects. This approach gives more weight to target appearances in more recent frames. For efficiency, this computation is done in the Fourier domain.

The MOSSE tracker can detect occlusion and tracking failure by measuring how pronounced the peak of the correlation output is in relation to the correlation output for the other pixels. This is achieved my computing the Peak-to-Sidelobe-Ratio. If the peak is not strong enough the tracker concludes that the target object is occluded or tracking has failed. In that case the filter is not updated until the object re-appears.

The MOSSE tracker runs faster than real-time.

Advantages of MOSSE:

- very fast
- can handle occlusion (if target re-appears in the same location)

Disadvantages of MOSSE:

- tends to drift when central point moves away from target's center (caused by out-of-plane rotation of the target)
- no background information used
- update using running average leads to overfitting to more recent target appearances. This is a disadvantage as the manakin has several distinct appearances which can quickly change.
- Gaussians have a round base and are thus better suited for compact objects than for elongated objects such as the golden-collared manakin.

3.2.7 Channel and Spatial Reliability (CSRT)

The CSRT tracker [LVČZ+17] extends discriminative correlation filters (DCF) with spatial reliability and channel reliability.

Spatial reliability is realized through a spatial reliability map which is an approximate segmentation of the target object: it is 1 at pixels that cover the target and 0 at pixels that cover the background. The spatial reliability map is constructed based on the probability of colors occurring in certain locations using color histograms of the foreground and background. The purpose of the spatial reliability is to prevent background pixels from contributing to the filter response. As a result, larger search region sizes are possible and non-rectangular objects can be handled better without the filter learning the parts of the background inside of the object's bounding box that are not covered by the object.

At every frame, the CSRT tracker first localizes the target using the spatial reliability map and reliability scores of the previous frame (see Figure 3.6). Then, using the localized target as a training example, the spatial reliability map, filters and reliability weights are updated.

In the localization step, the CSRT tracker combines multiple feature channels through weights, called *channel reliability scores*, giving more weight to feature channels that show better discriminative power which is measured in two ways:

1. The highest filter response value per-channel, as a higher peak in a filter response is assumed to indicate a more reliable filter.

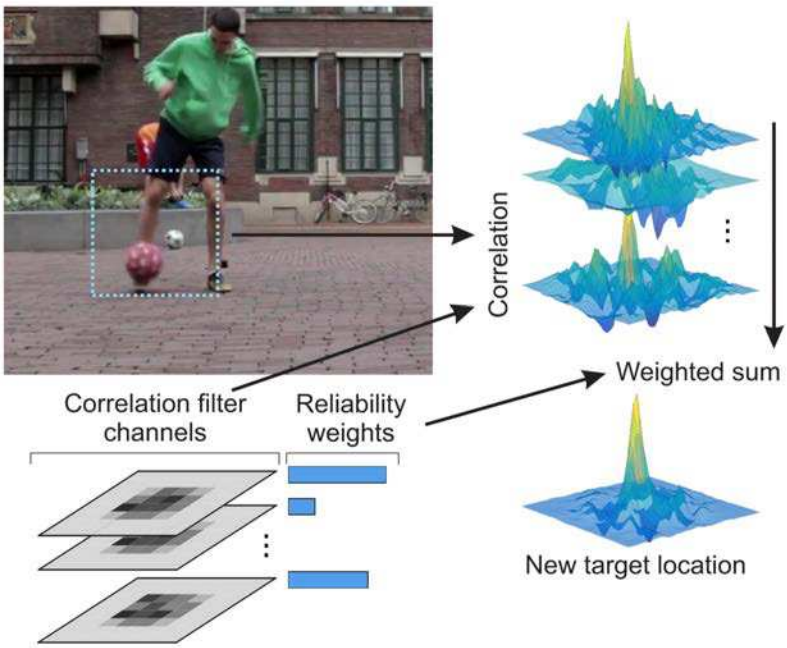

Figure 3.6: To localize the target an image region is convolved with correlation filter channels. The resulting correlation responses are weighted by channel reliability weights and summed up. The peak in the summed correlation responses indicates the new target location. (image from [LVČZ+17])

2. The relation of the highest and second highest peak, as one unique peak would indicate a clearer "decision" of the filter. This measure is not ideal, however, if a similar object occurs close to the target.

For the feature channels, standard features are used: histogram of oriented gradients (HoG) [DT05] and Colornames [VDWSVL09] as well as a gray-scale channel. On a CPU, the CSRT tracker achieves run times close to real-time.

Advantages of CSRT:

- can adapt to changing tracking conditions by varying the weights given to different features
- can handle search regions in which large parts of the region is covered with background (search region with large padding or non-rectangular object) by using spatial reliability map. The manakin often does not cover its bounding box, e.g. when it moves diagonally.
- Colornames might be a useful feature to detect the male manakin's conspicuous yellow neck.

Disadvantages of CSRT:

- Histogram of oriented gradients might not be a useful feature as it relies on consistent lines in the object, which are not present in the manakin due to blur.

3.3 Deep trackers

Deep trackers are based on deep learning. They use deep features, extracted from a deep neural network. In the long-term tracking challenge VOT2018 [KLM+18], 10 out of 11 competing used CNN features.

3.3.1 MobileNet based Long-term Tracking by Detection (MBMD)

The MBMD tracker [ZWW+18] is the winner of the VOT2018 long-term tracking challenge [KLM+18]. MBMD is a long-term tracker that consists of three main parts: a regression network, a verification network and a dynamic switch scheme to recognize if the tracked object is present or absent and switch between local search and re-detection, accordingly (see Figure 3.7).

The **regression network** gets as input a template (the patch cropped from the initial ground truth bounding box) and the search region: a region in the current frame centered on the location of the tracked object in the previous frame. Both are fed into convolutional neural networks (MobileNet architecture [HZC+17]), which serve as feature extractors. The resulting feature maps are fused and put into a region proposal network [RHGS15] that outputs candidate bounding boxes with corresponding similarity scores that reflect the similarity of the candidate to the template. The regression network is trained only off-line to avoid accumulating errors introduced by on-line training – the template it compares the current frame to remains the object's appearance in the first frame.

The output of the regression network (the candidate bounding boxes with similarity score) is fed into the **verification network**. This network starts with the candidate that received the highest similarity score and classifies it with a CNN (VGGM architecture [CSVZ14]), going down the list of candidates ranked by similarity score until it classifies a candidate as foreground. The tracker then proceeds by making a decision whether the object is still present based on the similarity score of the selected candidate and the classification score the CNN assigned to that candidate. If the object is deemed absent a search of the full image is conducted otherwise the object is searched in the proximity of the tracked object. The verification network is updated on-line to adapt to appearance changes of the object.

Advantages of MBMD:

- cannot erroneously learn background during tracking in the first part of the network by using only the template from the first frame as a reference image to compare against

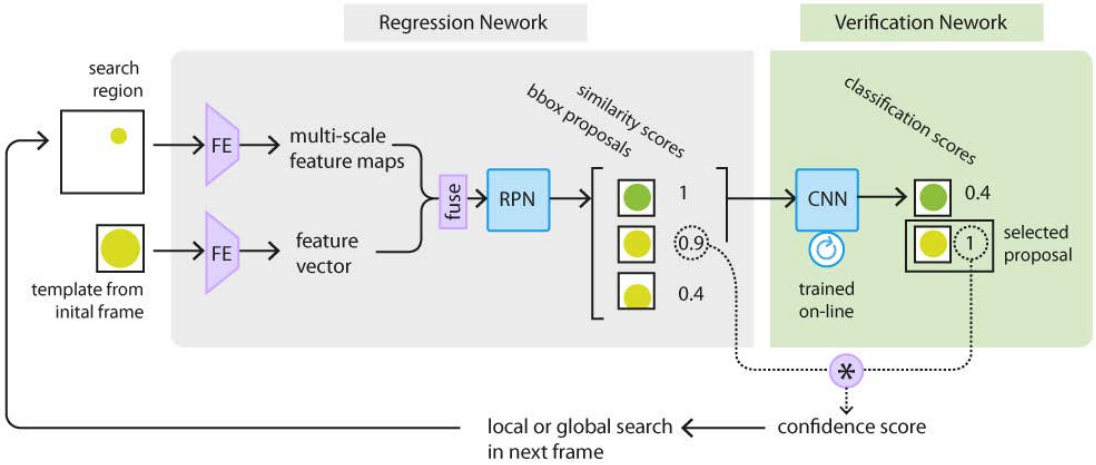

Figure 3.7: Diagram of MBMD. MBMD's network takes as input the current search region and the template from the initial frame. From both, features are extracted (FE) which are fused and feed into the region proposal network (RPN), which proposes potential bounding boxes and corresponding similarity scores. The bounding box proposals receive a classification score from the verification network which is combined with the similarity score assigned by the regression network into a confidence score. Depending on that confidence score the model switches between local and global search. [ZWW+18].

Disadvantages of MBMD:

- By not updating the first part of the model (the regression network) this tracker might not be able to handle strong appearance changes.

3.3.2 Distractor-Aware Long-term Tracking (DA Siam LT)

DA Siam LT [ZWB+18] placed second in the VOT2018 long-term tracking challenge [KLM+18]. This tracker is based on the Siamese region proposal network (SiameseRPN) [LYW+18], which is a Siamese network followed by a region proposal network. The network is trained off-line and updated on-line during tracking. The input to the SiameseRPN is a pair of images, the Siamese network measures their similarity and the final output of SiameseRPN are region proposals inside a given search region. In order to increase the number of object classes in the training dataset during off-line training, DA Siam LT uses not only frames from video datasets, but also image pairs generated from large detection datasets by applying data augmentation techniques, such as translation, re-scaling and artificial motion blur.

During tracking, there is only one positive example of the tracked object available per frame, while the entire background can be sampled for negative examples. However, if the background is randomly sampled, most samples will typically not contain objects, which makes them easy to recognize as background and thus not useful in improving

the discriminative ability of the network. Instead, the region proposal network looks for negative examples that are harder for the network to classify, called distractors. Distractors can also be objects from the same class as the tracked object. While the region proposal of the RPN which received the highest score is chosen as the target, the rejected region proposals with similarity scores above a threshold are used as distractors. By including the distractors in the similarity metric, it becomes domain-specific.

DA Siam LT enables long-term tracking through a local-to-global search region strategy: if low detection scores indicate tracking failure, the search region is increased by a constant step size, which allows the tracker to re-detect the target once it re-appears. DA Siam LT runs faster than real-time.

Advantages of DA Siam LT:

- offline learning on large-scale datasets, including still images augmented with artificial motion blur
- strategy to focus on learning with difficult and thus more useful negative samples
- long-term tracker: can re-detect target object
- real-time performance
- more flexible than fixed search region size: grows search region if object is not detected, which might help to find a fast moving object

3.3.3 Efficient Convolution Operators for Tracking (ECO)

ECO [DBKF17] is a short-term tracker based on the Discriminative Correlation Filter (DCF) method and is the improved version of C-COT (Continuous Convolution Operator Tracker) [DRSKF16], which was one of the top performing trackers in the VOT2016 challenge [KLM+16]. Like C-COT, ECO is based on a CNN pre-trained on ImageNet. The CNN is used to extract feature maps from the search region in the current frame. These feature maps consist of the input image patch and the convolutional layers from the CNN. Based on the feature maps, both trackers learn continuous convolution filters. When convolved with the feature maps, these filters produce a continuous confidence score inside an image region centered on the previous location of the target. The target is localized where the confidence score reaches its peak (Fig. 3.9).

C-COT learns a convolution filter for every feature map – many of which barely have an influence on target localization. In contrast, ECO fuses the convolution filters into multi-channel filters instead of learning one filter per feature channel. This results in a convolution operator with a much smaller number of parameters (80% less than C-COT) that need to be trained. This less complex model avoids overfitting and is faster to train.

Other trackers store one training sample for each frame for online training. Due to space limitations, however, some samples have to be removed from the training set – typically

these are the oldest samples. ECO uses a more refined approach: It models the training data as a mixture of Gaussian components, where each component models a different form of the target's appearance (see Figure 3.8). In each frame, for the new training sample, a new component is added. If the number of components exceeds a given limit, the new component is either discarded (if its weight is too small) or the two most similar components are merged. This representation of the training set is more compact than storing all previously seen samples and thus enables a more diverse representation of the target's varying appearances with a reduced risk of overfitting to more recent target appearances.

For further speed-up, the convolution operator is not updated at every frame but only when a significant change to target appearance has occurred. ECO realizes this by performing an update after a fixed number of frames. They even report better tracking results when updating at every fifth frame, compared to updating at every frame.

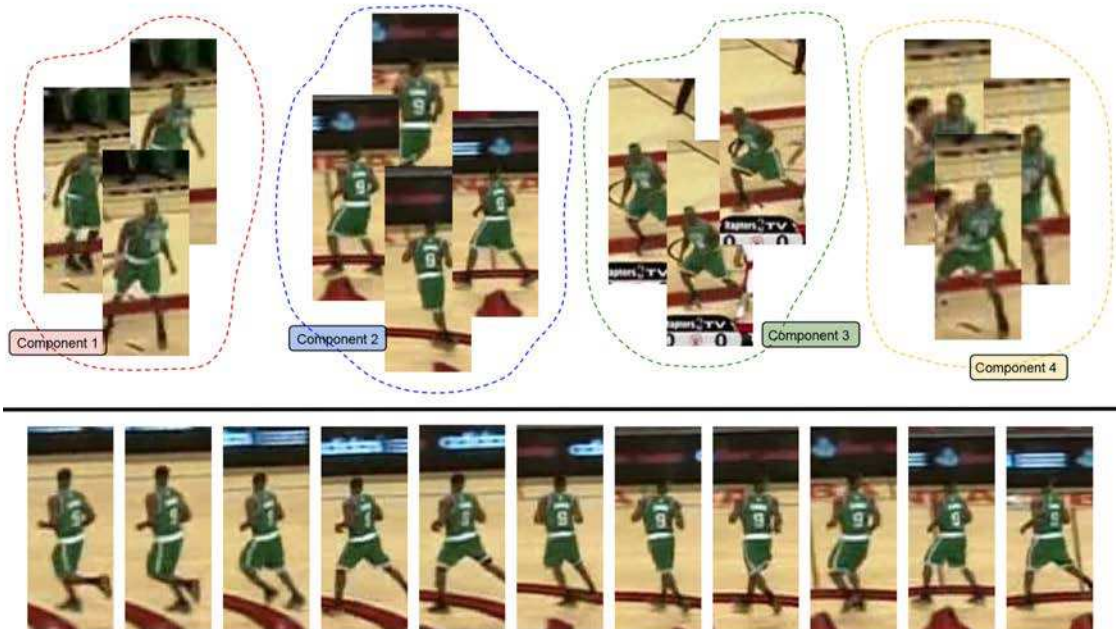

Figure 3.8: Instead of simply storing a fixed number of previously seen frames (bottom row), the ECO tracker represents the previously seen frames as a mixture of Gaussian components (top row). Each component captures a different version of the target's appearance [DBKF17].

Advantages of ECO:

- computing a continuous score enables sub-pixel accuracy
- uses information from multiple convolutional layers of the CNN

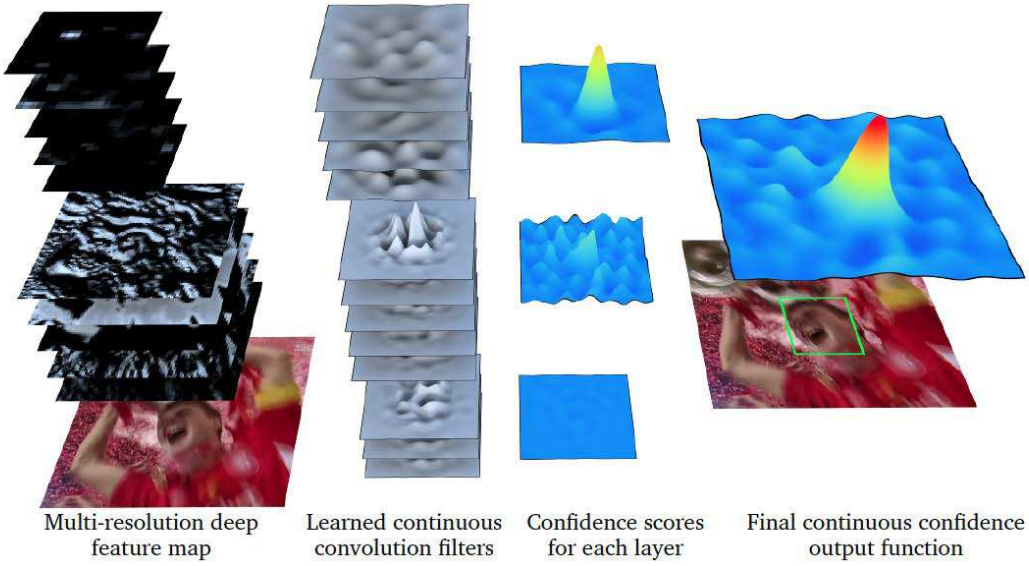

Figure 3.9: Continuous convolution filters are applied to a multi-resolution feature map to produce a continuous confidence score of the target. The target's center in the current frame is assumed to be at the peak of the confidence score [DRSKF16].

- keeps multiple representations of previously seen target appearances. For tracking the manakin, this can be useful because it has quickly varying but reoccurring kinds of appearances (on perch, jumping, wings open, etc.).

3.3.4 Generic Object Tracking Using Regression Networks (GOTURN)

GOTURN [HTS16] is a short-term tracker based on a convolutional neural network (CNN). The CNN takes two images as input: the target in the previous frame padded to include some context around the target and the search region in the current frame, which is the current frame cropped with the target bounding box of the previous frame enlarged by a constant factor. The output of the CNN are the coordinates of the new target bounding box within the search region obtained though regression (see Figure 3.10).

The CNN is trained only offline on large-scale video and image datasets but not updated online during tracking. To train with a video, two consecutive frames are randomly taken from a video sequence; to train with a still image the image is cropped and resized in order to simulate movement of the depicted object. Although the training data created from still images is not as realistic as the frames taken from videos as they only contain motion in the form of translation and size change and no other types of deformation, these samples help to increase the diversity of objects that the network sees during training.

GOTURN runs in real-time as it skips the time-consuming CNN update as well as performing CNN inference only once per frame.

Advantages of GOTURN:

- very fast

Disadvantages of GOTURN:

- cannot handle occlusion
- no online learning: does not adapt to target appearance
- input to network is crop from previous and current frame: potential overfitting to most recent target appearance

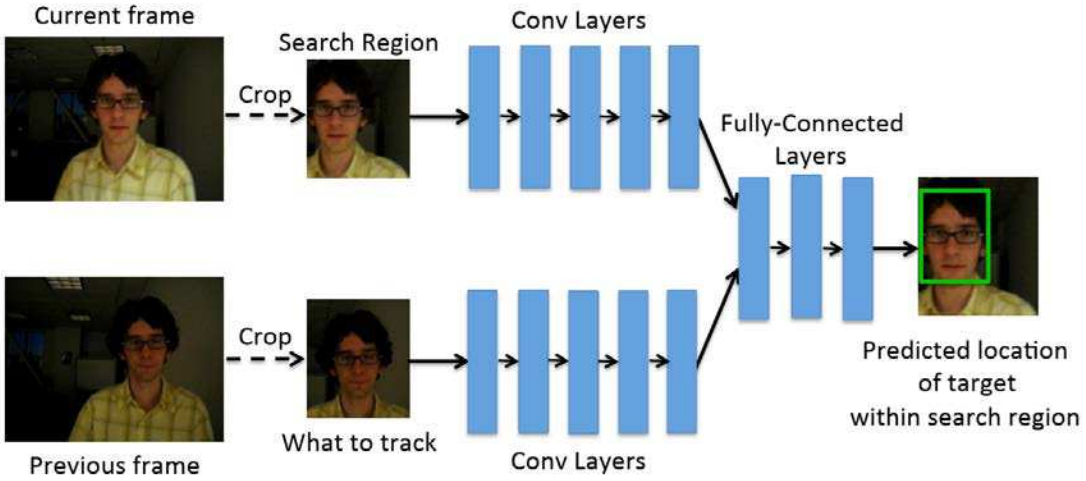

Figure 3.10: GOTURN's CNN takes the current and previous frame as input and outputs the coordinates of the target bounding box (green box) in the search region inside the current frame. [HTS16].

3.3.5 Comparison of Video Trackers

In this section, we compare the trackers presented above and analyze their usefulness for our datasets described in section 2.

Online learning. The trackers employ different strategies to update their model during tracking. GOTURN and MedianFlow do not perform any updates: while this makes tracking faster, the tracker cannot adapt to the changing appearance of the target object or the background.

CSRT, KCF, MOSSE, CSRT, MIL, and ECO only update on positive samples, which are extracted from the current frame based on the predicted target location.

The MBMD tracker updates only the second part of the network (verification) to avoid incorrect learning in the first part. In the first part of the network, this tracker uses the target object from the first frame (for which the groundtruth was given) as a reference and proposes potential new target location to the verification part of the network. As a consequence, candidates that are rejected by the first stage never reach the verification stage, which adapts to target appearance. Thus, this tracker relies on the target appearance staying close to its appearance in the initial frame.

MOSSE extracts a training sample in every frame and weighs the set of training samples using a moving average so that the most recent appearances contribute more strongly to its appearance model. Sampling at every frame, however, can lead to redundancies in the training set and lead to overfitting to more recent target appearances. ECO addresses this issue by updating its convolution operator only after a fixed number of frames to increase the likelihood that sufficient change to the target's appearance has taken place.

Some trackers employ strategies to improve the sampling of training examples. ECO tries to avoid storing redundant training samples and instead keeps diverse representations of the target appearances observed in the training samples. DA Siam LT samples negative training samples that are difficult for the CNN to classify and are thus more useful in making the network more discriminative.

All of the positive samples that are acquired during tracking – except for the initial one – are based on the tracker's own prediction and are thus unreliable. Therefore, some trackers use strategies to increase the likelihood of updating only on training samples that have a higher chance of showing only the target object.

Offline learning. Offline learning refers to training a tracker's classifier on available data before the trackers starts tracking the video. The traditional trackers (boosting, MIL, Median Flow, KCF, TLD, MOSSE, CSRT) do not train offline.

All of the deep trackers presented above (MBMD, DA Siam LT, ECO, GOTURN) use CNNs pre-trained on frames extracted from video datasets. Additionally, GOTURN and DA Siam LT generate training samples from large-scale image datasets by simulating pairs of consecutive video frames from still images.

Features. Features are extracted from the video frames using feature extractors. Good features represent some aspect of the target object that makes it possible to discriminate it from the background. Traditional trackers use hand-crafted features. The most simple feature is just the raw image itself. Boosting uses Haar-like features, histogram of oriented gradients, and local binary patterns; MIL also uses Haar-like features; Median Flow – and TLD, which uses Median Flow as its tracking component – uses the brightness of the points it tracks; KCF and MOSSE apply a correlation filter directly to the image; CSRT uses histogram of oriented gradients, Colornames, and the gray-scale image.

The histogram of oriented gradients relies on edges (usually these are the borders of the object [MC11]). The bird's silhouette changes significantly between frames, however, because of the strong motion blur and because the bird is a non-rigid object.

The trackers use different strategies to combine multiple features: The boosting tracker weighs features based on the current background and feature appearance. Similarly, MIL uses an algorithm inspired by AdaBoost. CSRT uses channel weights to control the influence of the individual features on the final output.

Deep trackers use "deep" features, i.e. features extracted by the convolutional layers of a deep neural network. MBMD and DA Siam LT input deep features into a region proposal network. DA Siam LT additionally uses a CNN to get a classification score.

The advantage of deep features is that they are more powerful than shallow machine learning methods. Deep neural networks are able to extract high-level features [ZZXW19]. With increasing layer depth the features extracted by the convolutional layers become more high-level.

One disadvantage of deep features is that the deep neural network require more space. MBMD, for example, requires two networks that take up about 100 MB in total. Deep features are also less interpretable than hand-crafted features.

Motion model. All the trackers presented above except for the Median Flow tracker predict the movement of the target only by using search regions centered at the previous target location, which can be interpreted as a motion model that assumes that the target location in the current frame is close to the target location in the previous frame and a movement in every location is equally likely. The Median Flow tracker is based on the Lucas-Kanade Tracker which estimates optical flow.

There is a trade-off when it comes to the size of the search region: while a smaller search region is faster to process, a larger search region is more likely to include the target in its new location. When the target moves fast a search region of fixed size might be too small to include the target in the current frame. DA Siam LT grows the search region iteratively when the tracker fails to locate the target.

Localization. Based on the features extracted in the current and previous frame as well as the bounding box predictions for previous frames, the tracker estimates the location of the bounding box in the current frame. The tracker needs some kind of scoring function to select the location which reaches the maximum score among the potential locations [MC11].

Boosting and MIL use an AdaBoost classifier fusing multiple weak classifiers. MOSSE, KCF, CSRT, and ECO use correlation filters to locate the target in an image patch. A correlation filter takes an image patch or feature map as input and outputs a function that should peak at the location of the target. The correlation filter is learned with training images and a corresponding 2D output that has its maximum at the target center. MOSSE and KCF use the raw images as input, CSRT learns correlation filters for multiple feature channels (all hand-crafted features), and ECO trains correlation filters

on multiple feature maps extracted from a CNN. MBMD combines scores obtained from a region proposal network with the classification score from a CNN and selects among the region proposals the one that reached the maximum combined score. DA Siam LT also uses a region proposal network and selects the proposals which received the highest score, the other proposals are used as negative training samples.

Runtime. Real-time runtimes are required for certain applications, such as surveillance, where the frames need to be processed at the same (or faster) speed as they are produced by the camera, which is usually at 60 frames per second. Fast run times can be achieved by using some of the following options: choosing features that are fast to compute, avoiding online updates, smaller size of the search region, limiting the number of image patches that are classified, or using powerful hardware.

For our task, real-time runtimes are not required as the manakin videos are already recorded when the tracker is applied.

Long-term / Short-term tracking. Long-term tracking places additional challenges on a tracker: long occlusions and the target leaving the frame and re-appearing must be handled. Some short-term trackers can handle limited amounts of occlusion. MOSSE, for example requires the target not to move outside of the search region while it is occluded. TLD, MBMD, and DA Siam LT are designed to be long-term trackers, while the other presented trackers are short-term trackers.

In our task, the bird leaves the frame which requires re-detection, therefore a long-term tracker is necessary.

3.4 Behavior Recognition and Representation

This work investigates behavior recognition and representation of birds. In section 2.2.4 we describe the behaviors that the golden-collared manakin has been observed to display during its courtship dance. While there has been little research into recognizing and representing the behavior of birds, researchers have tried to automatically detect the behaviors of other animal species as well as humans.

3.4.1 Behavior Recognition

The behavior of animals is studied to improve animal welfare on farms with the aim of increasing productivity [LXC+18], inside laboratories [CHP+14], to understand sexual selection [Per17], to study the effects of drugs in pharmaceutical research [SGK15], or to study, conserve and manage wild animals [NNK+17].

3.4.2 Behavior Recognition for Birds

Traditionally, when studying the behavior of birds, observers watch the birds in the field and manually record the duration and type of behaviors they see [DuV07]. In some studies, researchers additionally record videos so that details of the behaviors can later

be analyzed by watching the videos frame-by-frame [DuV09, SGL17]. For example, Ota et al. [OGS15] analyzed the courtship dance of the blue-capped cordon-bleu, which resembles a step dance, by counting the number of steps the bird executes using standard video editing software.

Manual Software Tools

Software tools are available that assist biologists in quantifying the bird's behaviors, such as JWatcher [Sta08] which lets the user watch a video and records key presses to indicate the start and end of a behavior. The software then calculates statistics based on the duration of the individual behaviors. JWatcher has been used by Lukianchuk et al. [LMD14] to analyze the courtship dance of the long-tailed manakin. Noldus Observer XT [ZBW+09] is another software tool that lets users analyze videos frame by frame. Ullrich et al. [UNS16] used it to code the behavior of the zebra finch's choreography.

ImageJ [RSH+17, ASSRE12] is an image processing software for scientific images that can be used to measure distances in individual frames extracted from a video in order to describe differences between courtship displays in more detail. Ribeiro et al. [RdCGMM19] used ImageJ to measure flight height and the distance between male and female birds; Manica et al. [MGPM16] measured the jump heights of a songbird (blue-black grassquit) with ImageJ. Similarly, Miles et al. [CMF18] used a software tool (MB-Ruler Pro) to measure angles in individual video frames to identify the deepest bow and widest wing position in the choreography of a tropical bird (Montezuma oropendola).

These tools do not enable automatic behavior recognition but merely assist the researchers in recording the behaviors manually. The researchers are still required to watch every frame of the entire video dataset, which is typically many hours long, and decide which behavior they want to assign to what they saw in each video frame. Thus, such approaches are very time-consuming. They rely on the subjective perception of the researchers, so these decisions may be biased. Additionally, since watching hour-long videos is a very monotonous and tiring process, errors can occur when the observer becomes inattentive.

3.4.3 Automatic Behavior Recognition for Animals

Automatic behavior recognition software is available for animals that are more commonly used as model species in pharmaceutical research, such as rats. EthoVision XT [vDvdHtB+13] provides a software module that can automatically recognize rat behavior. It recognizes some behaviors based on the trajectory of three body points on the rat (nose, center and tail base), as well as its body shape. For this method to work, the body points need to be visible and consistently recognizable. In our task, however, the complex movement of the bird and clutter in the scene cover body points that might be considered for this approach [Per17]. Additionally, severe blur distort potential body points on the bird.

EthoVision XT also deducts behaviors from the rat's location in the cage: for example the rat can be classified as feeding when the rat is in close proximity to the feeder inside

its cage. The system was developed for videos recorded in a cage by an overhead camera. However, our datasets were recorded from side-views and in an environment that contains strong clutter. This tool was developed specifically for rats, which exhibit a specific set of behaviors that differs from the behaviors of other animals. There exists no such tool for automatically recognizing the behavior of golden-collared manakins or any other comparable species.

Behavior Recognition based on Trajectory

Some approaches to behavior recognition aim to recognize certain behaviors based on the trajectory of one or more body points. Matsumoto et al. [MUU+14] studied how rats explore novel objects based on the 3D trajectory of the rat's nose. Steward et al. [SGK15] propose tracking zebrafish to examine how their motion patterns change under the influence of different drugs. The behavior of the fish is classified with decision trees that are constructed based on features of the 3d trajectory, such as velocity, angular velocity and position in the tank. Winkler et al. [Win18] monitor the behavior of honey bees by tracking bees in videos and based on the trajectory, measure the time a bee spends outside of the hive and how often it leaves the hive. Liang et al. [LXC+18] recognize the behavior of cows based on their trajectories.

Behavior Recognition based on Location

Some approaches determine the behavior based on the animal's spatial position in the frame. Ings et. al [IWC12] use the 3D trajectory of bees to detect which flowers they choose to land on. Nakarmi et al. [NTX14] track hens using background subtraction and determine their behavior based on their location: for example, if a hen visits the feeding area a feeding behavior is registered. Such approaches require "meaningful" sections in the recorded area. For the manakin such an approach is only applicable in a limited way: the perching behavior is linked to the manakin resting on a sapling; for the flip the manakin moves towards the ground. Other behaviors, such as the wing-snap can take place anywhere in the arena. However, we do not have an annotation available that segments the frame into semantic regions.

Behavior Recognition based on Image Patches

In addition to the trajectory, Nie et al. [NTIM11] use the silhouette of segmented animals (mice in their case) to detect behaviors based on the area of the body. They also use frame-to-frame differences to get the frequency of very fast, repetitive limb movements to identify scratching in different parts of the mouse's body. Similarly, Crispim-Junior et al. [CdAN17] automatically recognize the behavior of laboratory rats by classifying central tendency and variance of the mice's body area and length as well as distance travelled and the number of changed pixels with a Multilayer Perceptron network.

Deep-Learning Approaches

DeepLabCut [MMC+18, NMC+19] and LEAP [DPAW+19] extract posture data by tracking body points in videos using deep-learning. These systems are trained on images in which the body points are marked. From the body points poses can be inferred which can then be categorized into behavior states.

DeepLabCut is an open-source software that uses a deep convolutional network to track user-selected body points without requiring physical markers on the animal. DeepLabCut uses ResNets pre-trained on ImageNet, where the classification layer have been replaced by deconvolutional layers. The output of the network is a map that gives the probabilities that any of the selected body points is present in a particular location in the input the image. It requires training data of about 50 to 200 frames labeled with body points to achieve human labeling accuracy. From the trajectories of the body points, the researchers can deduce the behavior of the recorded animal.

LEAP (LEAP Estimates Animal Pose) is a framework for estimating the position of body points on animals with deep learning. It also contains a graphical interface to let the user label body points of interest. The network requires a minimum of 100 labeled frames for training. It uses a 15-layer, fully convolutional neural network that outputs a probability distribution for each body point's location in the image. The network is trained iteratively: the user first labels ten images that are used to train the network and create preliminary labels for the other images in the dataset that the user can adjust to produce the ground truth. Both LEAP and DeepLabCut need about 500 labeled images to reach human-level accuracy [NMC+19].

These approaches can be extended to analyze the behaviors of other species by training the neural networks on data annotated with body points of the desired species. To use such an approach hundreds of images with body point labels would be needed which would require considerable effort and thus exceeds the scope of this thesis. Our datasets only contain bounding box annotations that mark the entire bird. However, since details such as the legs or the beak are not discernible due to blur when the bird is moving fast, it is questionable if it would be possible to annotate body points throughout the videos.

Instead of basing the detection of behavior states on the trajectory of an animal, Stern et al. [SHY15] use a Convolutional Neural Network (CNN) to classify video frames. They trained a CNN to recognize if a fruit fly (Drosophila) is on egg-laying substrate or not, achieving a success rate of 99.9%.

Norouzzadeh et al. [NNK+17] automatically identify the species and behavior of wild animals in camera-trap images using CNNs. The CNNs were trained on a large-scale dataset (3.2 million images) of camera-trap images from the Snapshot Serengeti project [SKL+15]. Every image has labels concerning the species depicted, the number of animals present and the behaviors, such as eating or resting, shown. As the images come from camera traps, they are imperfect: animals can be far away or too close, they might be occluded and lighting conditions differ. Their networks reach 94% accuracy at identifying

animals. This approach, however, requires behavior labels – our dataset only contains bounding box labels indicating the location of the birds.

Human Action Recognition

Several deep-learning architectures have been proposed for human action recognition [CZ17]. However, these networks have to be trained on hundreds of images representing each action class. There are large-scale datasets for human action recognition [KCS+17] that encompass hundreds of human action classes – such as hugging, punching, swimming – that have been successfully used to pre-train action recognition models for videos [CZ17]. Such datasets, however, offer little value for animal action recognition as animals and their behaviors (e.g. sniffing, flying) have little visual resemblance to humans and the behaviors (e.g. playing the violin) that they display.

3.4.4 Behavior Representation

There are different ways to represent the behavior of humans and animals. Andrienko et al. [AA18] describe how state transition graphs can be used to summarize movement data, so that the meaning behind the transitions between locations can be analyzed. They use locations in space as states and the movements between those locations as the (oriented) edges. Figure 3.11 shows a state-transition-graph that visualizes the behavior of inhabitants of a city by depicting specific, meaningful locations in the city (such as the person's home or company) as circles and movement from one location to another as lines that connect those circles, where thicker lines indicate more transitions. This approach requires locations and the movement between them to have meaning.

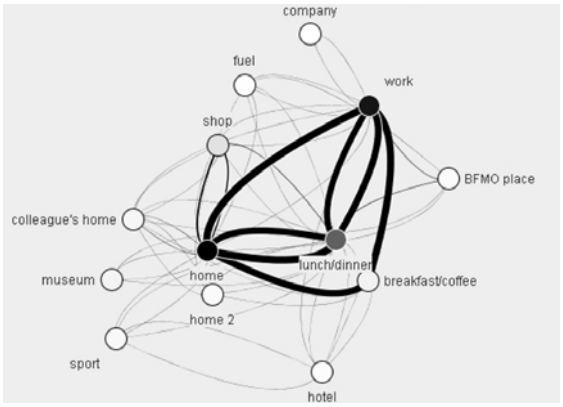

Figure 3.11: State-transition-graph for analyzing the behavior of people by showing their movements between meaningful locations in a city (image from [AA18]).

Ribeiro et al. [RdCGMM19] and Ullrich et al. [UNS16] use a similar diagram (see Figure 3.12) to visualize the behavior of birds but they use the observed behaviors, such as head movement or hopping, as states (depicted as rectangles) and arrows to indicate the transition from one behavior to another. Numbers on the arrows indicate the likelihood

of a transition. This type of diagram does not show the order in which the behaviors take place or where they occurred in space.

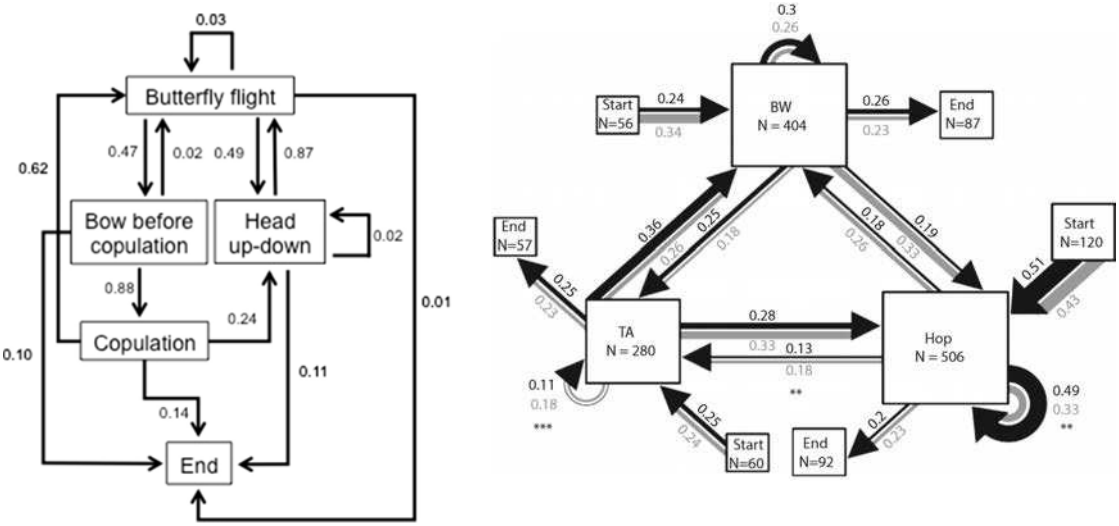

Figure 3.12: Right: Diagram of Swallow-tailed Manakin's courtship dance. The numbers stand for the probabilities of a transition. (image from [RdCGMM19]); Left: Sequence diagram of zebrafinch's courtship dance. (image from [UNS16])

In sports, representing a sequence of actions is of interest as it enables coaches and other stakeholders to analyze games and find ways to improve the performance of the players. Ono et al. [ODS18] show the course of baseball games in a timeline, which depicts how players move between locations marking interesting states. Trajectory data typically suffer from occlusion, which are avoided in this type of visualization.

Berman et al. [BCBS14] visualize a fly's leg movements over time by segmenting the fly in every frame of a video and aligning the segmented images. The images are transformed into a set of time series, which are then transformed to spectrograms. Every point in time is then mapped to 2D space. In the 2D representation, peaks represent more common behaviors (see Figure 3.13).

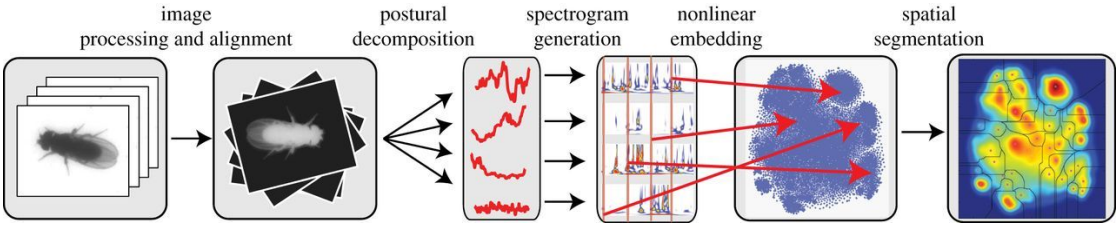

Figure 3.13: Visualization of fly behavior. Right image: Peaks (red) represent stereotypical behaviors (image from [BCBS14]).

CHAPTER 4

Tracking and Recognizing the Behavior of the Manakin

4.1 Visual Tracker

The tracker provides the location of the bird throughout the videos under the challenging conditions outlined in Section 2.2.1 while making use of the fact that the camera is stationary. The tracker consists of two main parts: an appearance model to discriminate between the tracked bird and the background and a motion model, which estimates the movement of the bird. During tracking, the motion model provides potential locations in a given frame and the appearance model classifies these locations to determine the final location of the bird in that frame.

4.1.1 The ManakinTracker

We propose a tracker, that

- detects moving blobs in the scene with a Mixture Of Gaussians model (MOG) [SG99],
- decides if a candidate location visually resembles a male golden-collared manakin with a fine-tuned Convolutional Neural Network (CNN),
- estimates the location of the target using a Kalman filter [WB95] to track the bird through occlusion and to verify blob-based predictions.

4.1.2 Blob Detection: Mixture Of Gaussians

Making use of the fact that the videos were recorded with a stationary camera, we use a method based on background subtraction to segment the foreground. We chose Mixture

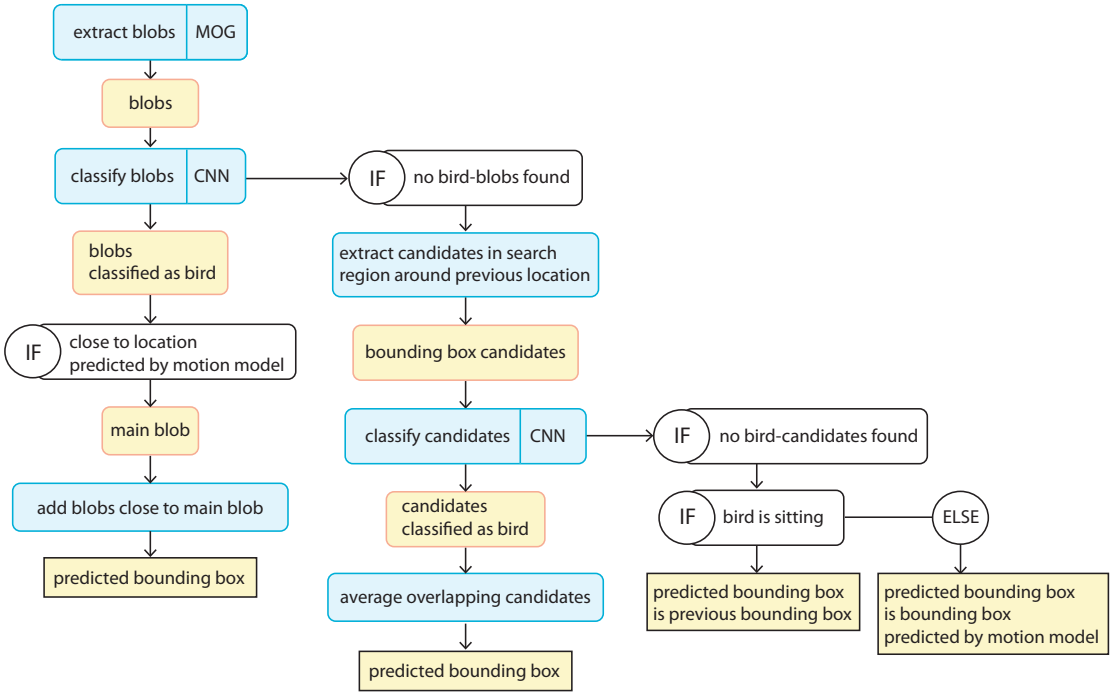

Figure 4.1: Flowchart of ManakinTracker.

Of Gaussians (MOG) because it can handle small movements in the background. For every frame, MOG generates a foreground mask from which we extract moving objects, called blobs (Figure 4.2). Out of these candidate blobs, we aim to select the ones that contain the target – and discard those that contain other objects such as moving leaves, branches or the female bird.

Figure 4.2: Foreground mask (right) generated by Mixture Of Gaussians model of frame (left). The blue box marks the extracted blob.

4.1.3 Appearance Model: CNN architecture

To decide which candidates contain the target we use a CNN, since CNNs have shown top performance in object classification in images [ZZXW19]. Our CNN is based on AlexNet [KSH12], which is a CNN pre-trained on ImageNet, that takes 227x227 pixels RGB images as input. We use the pre-trained layers of AlexNet for our CNN, except for the last 3 layers, which we replace with a new fully connected layer, a new softmax-layer and a new output layer to match our two classes: target (i.e. male golden-collared manakin) and background (i.e. forest). The output of our CNN is a background score and a target score in [0, 1]. We fine-tune this new CNN with image patches of male golden-collared manakins and background cropped from a set of sequences in our dataset, that does not contain the sequence in which we currently track the bird. During tracking the CNN is not updated further.

4.1.4 Motion Model: Kalman Filter

We use a Kalman Filter to predict the target location in the absence of reliable visual information. Reliable visual information is given by a blob, or candidate location(s) extracted around the target's previous position, that receives a high target score by our CNN. We also use the Kalman Filter's location estimation if we find more than one blob. In this case, we select the blob that is closest to the Kalman Filter's location estimation. The linear Kalman Filter is initialized with the ground truth bounding box in the first frame and updated with the estimated target location at each frame after tracking. The linear Kalman Filter estimates an ongoing movement of the target at constant velocity, based on the target's previous locations.

4.1.5 Tracking

The male bird's location is initialized with the ground truth bounding box in the first annotated frame. For each frame, we detect blobs and classify them with our CNN. Out of the blobs that receive a target score greater than t_1, we select the one that is closest to the location predicted by the Kalman filter as our main blob. The bird can be partly occluded (e.g. by the sapling it sits on), so we add blobs to the main blob, which were classified as target (target score greater than t_2) and which are close to the main blob. Since we do not know the width of the saplings, we consider two blobs as close if the distance between the boundaries of the two bounding box is less than the largest width or height of one of the two blobs. If we find a blob or combination of blobs, that fits these conditions, it becomes the target bounding box for the current frame (Figure 4.3).

If we find no blobs in a frame or none that receive a high enough target score we search the bird in the region around its previous location. This usually happens, when the bird is not moving and thus not recognized as a blob. We shift the previous bounding box to the left, right, top, bottom and diagonally and crop image patches at these candidate locations. We resize these image patches to fit the CNN's input size and classify them with the CNN. A candidate location which receives a target score of t_3 or above, is

Figure 4.3: The two small blobs (small blue bounding boxes) are combined into a bigger blob (big blue bounding box). The white text indicates the blobs' target scores.

Figure 4.4: Bird is sitting and no blob was found (red box: candidate locations with target score $\geq t_3$, white box: final bounding box)

assumed to contain the target. To avoid false positives, we exclude candidate positions, which do not overlap with the majority of overlapping candidate positions that were classified as target. The average of these candidate locations is the bounding box for the current frame (Figure 4.4). Since the bird tends to sit still at the start of the video, we keep the initial bounding box if the CNN assigns a target score of t_4 or above to the image patch cropped at the bird's initial location and if we find no blob in that frame.

In our experiments, we achieved good results when we set the thresholds t_1 and t_3 to 0.8 and 0.9, respectively. t_3 is used for image patches that are not based on blobs and thus typically contain sitting birds. t_3 is set to 0.9 instead of 1, because we noticed that when the bird is sitting, the ground truth location usually gets a score of 0.9 and above, but that candidate locations which receive a score of 1 are often not a better estimation than those with a slightly lower score. t_1 is set to a lower value than t_3 because t_1 is used for image patches that are based on blobs. In those cases the bird is typically moving which can make it harder to recognize for the CNN due to motion blur and shape distortion.

In case we find no candidate locations, which we can assume to contain the target – because the bird is largely or fully occluded or unrecognizable to the CNN – we will rely on the location predicted by the Kalman filter (Figure 4.5). This prediction can be unreliable because the bird might either move or sit while it is not visible. The Kalman filter estimates the bird's movement as a continuation of its previous movement, which is only useful if the bird keeps moving while it is occluded. Because of this, we only use the Kalman filter's estimation if we assume that the bird is moving (i.e. not sitting). Otherwise, we use the bird's previous location as the current target bounding box.

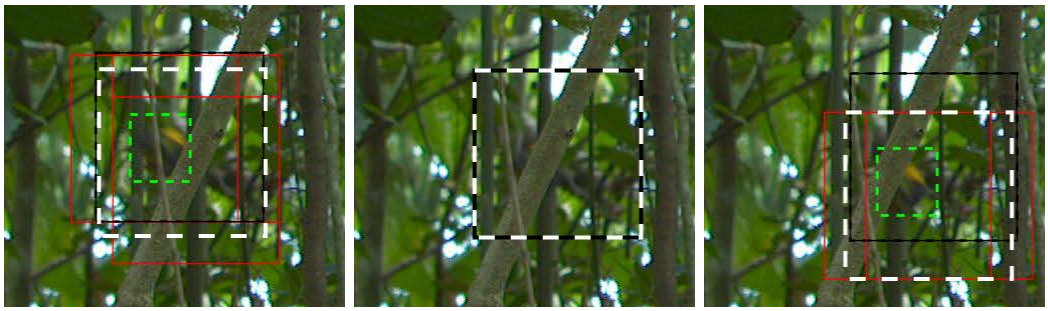

Figure 4.5: Middle: The Kalman filter's location estimation (black box) is used as the bounding box output when the bird becomes invisible to the camera during a jump. Left, right: The bird is visible and can thus be recognized by the CNN. (green boxes: ground truth; red boxes: candidate locations classified as target; white boxes: bounding box output for current frame)

We assume that the bird is perching if the difference between image patch $crop_t$ cropped with the previous bounding box from the current frame t and the image patch $crop_{t-1}$ cropped from the previous frame at the same location is below a threshold and the CNN score of the image patch from the current frame is above 0.5. In that case the previous bounding box is the predicted bounding box for the current frame.

$$\frac{1}{N}\sum_{i=1}^{N} crop_t - crop_{t-1}$$

N ... number of pixels in the image patch.

If the bird is recognized as sitting, the Kalman filter is re-initialized to the current location, because the bird tends to jump in a different direction than before it landed, and the Kalman filter would predict an ongoing movement in the same direction as before the landing. If the bird leaves the frame – i.e. we estimate its location to be partially outside of the frame – candidate locations are placed along the edges of the frame to detect the bird when it re-enters the frame. To avoid a false positive detection while the bird is outside the frame, the target bounding box estimate must be based on a blob in this case.

Advantages of the ManakinTracker:

- uses a Kalman Filter and MOG, and thus does not rely solely on the visual information in a single frame but detects motion based on multiple frames. This allows the ManakinTracker to handle target movement during occlusion.
- uses a CNN fine-tuned to recognize male golden-collared manakins (including highly blurry and partially occluded) with high accuracy
- In most cases (particularly during jumps) blobs give accurate bounding boxes and no further correction of the bounding boxes' dimensions is necessary.
- can track the bird efficiently in most frames by classifying only a limited number of image patches extracted from blobs (usually 1-4 per frame)

Disadvantages of the ManakinTracker:

- is fine-tuned to track golden-collared manakins. To track a different target, the CNN needs to be trained with groundtruth data of that target.
- requires video sequences as input that were recorded with a stationary camera.

4.1.6 Open Issues

Using blobs as bounding boxes works well as long as the entire bird moves – if the bird sits still and moves only partly, for example only its head, the bounding box enclosing the blob will only contain the moving part of the bird.

When the bird lands on a thin sapling, the sapling moves along with the bird, which leads to a blob that includes parts of the sapling along with the bird. This could be corrected in post-processing by adjusting bounding boxes, that strongly increased in size shortly before the bird started sitting.

In some cases the tracker recognizes the female bird as the target, even though the CNN was only trained on male birds. This suggests that the CNN is not dependent on the male bird's yellow neck for a correct classification. The downside of this is, that if the male and female bird are both present in a frame the two have to be distinguished by the tracker. One solution would be to train a CNN on images of female birds also, which would require ground truth bounding boxes for the female birds. Currently, we handle this issue by choosing the blob that is closest to the location predicted by the Kalman filter if there are multiple blobs that get a high target score.

4.2 Automatic Behavior Recognition

Based on the trajectory and the image patches cropped with the bounding boxes predicted by the tracker, we aim to recognize the bird's behavior in each frame by identifying the behaviors described in Section 2.2.4.

B1: On perch. When the bird is on perch, it moves only slightly, so this behavior can be recognized as the distance between the centers c_{t-1} and c_{t+1} of the bounding boxes before and after the current frame t being less than a small threshold $thresh_{jump}$.

$$perch = \begin{cases} true, & \text{if } \sqrt{(c_{t-1} - c_{t+1})^2} < thresh_{jump} \\ false, & \text{otherwise} \end{cases}$$

B2: Jump. The start and end of a jump can be identified through the distance between bounding box centers c_{t-1} and c_{t+1}: when the bird starts jumping it moves away from a sapling until is stops moving when it has reached another sapling.

$$jump = \begin{cases} true, & \text{if } \sqrt{(c_{t-1} - c_{t+1})^2} \geq thresh_{jump} \\ false, & \text{otherwise} \end{cases}$$

B3: Wingsnap. When the bird does a wingsnap, it lifts the wings above the body. This can be recognized as the bounding box area increasing between consecutive frames t-1 and t during a jump.

$$jump \;\&\; \frac{area_t}{area_{t-1}} > thresh_{wingsnap}$$

B4: Beard-up. The beard-up pose occurs while the bird is perching. As the biologists are interested in the moment when the bird is fully stationary in the beard up, we can measure the level of blur or measure correlation between consecutive patches, as the correlation should strongly increase as soon as the bird does not move anymore. Additionally, we will detect the male's yellow neck based on color values and identify the direction it points at by determining in which of the three views $view_1, view_2, and\ view_3$ the yellow region appears largest in. Regions of yellow color will be identified using the Colornames feature [VDWSVL09].

$$perch \;\&\; \max_{view_1, view_2, view_3} (yellow\ area)$$

4.2.1 Detecting the Yellow Neck

While it is easy for humans to assign colornames to image pixels, this is not as straightforward for a computer. In the computer the colors are represented by three values per

pixel (R,G,B) that have to be mapped to a color name, in our case yellow. However, color values are strongly impacted by lighting conditions.

The biologists hypothesize that the male bird presents its yellow neck to the female during the beard-up posture. To investigate this hypothesis, we aim to detect the yellow neck and determine towards which of the three camera views the male bird points its neck.

We have considered three approaches to detect the male's yellow neck:
1. using thresholds
2. finding the color closest to perfect yellow (R,G,B) = (1,1,0)
3. using the Colornames feature [VDWSVL09]. Colornames is a feature that assigns a color name, such as "yellow") to every pixel based on the pixel's (R,G,B) values. Colornames was trained on images found through Google Image Search, where images that contain e.g. the color yellow where found by searching the term "yellow". Since these images where labeled by humans, the colors found in those images reflect how humans perceive colors in real-world images.

Approach 1, setting specific thresholds to color values is sensitive to changes in lighting conditions. As the videos in our dataset vary strongly in lighting it is difficult to define a range of values between fixed thresholds that work for all videos.

Approach 2 will detect a region of a different color, which is the closest to perfect yellow, if there is no yellow region present in the image patch.

Approach 3 also labels regions as yellow that are more beige and do not look like the yellow of the bird's neck.

As all of these approaches are based on color values, they will all fail if the yellow neck is not visible in the image. This can be the case if the neck is occluded or the bird faces in a different direction. An issue for all approaches is yellow clutter in the image patch that is larger than the bird's yellow neck.

We will use the Colornames feature since it is the most robust to varying shades of yellow as well as shadows and is able to detect the absence of a yellow region.

4.3 Visualization of Trajectory and Behavior

We created two types of interactive visualizations for the bird's behavior during its courtship dance: a trajectory Visualization and a sequence Visualization. The visualizations are based on the trajectory obtained by the visual tracker and the result of the behavior recognition. The purpose of the visualizations is to represent the data in such a way that the biologists can compare multiple videos of courtship performances to find similarities and differences among them in order to allow them to find characteristics that make a courtship performance successful.

4.3.1 Trajectory Visualization

In the trajectory visualization (see Figure 4.6), a red line represents a jump, a green circle represents perching – the radius of the circle endcodes the duration of perching. The wing snap is shown by a cross along the jumping path. The trajectory can be optionally superimposed on the selected video frame to compare the video data with the detected behavior.

4.3.2 Sequence Visualization

In the sequence visualization, every frame is represented by a rectangle. Each rectangle encodes a behavior. The rectangles are aligned in a sequence. A red rectangle represents jumping, while a green rectangle represents perching. A smaller white rectangle inside of a red rectangle indicates a wingsnap. On top of the sequence of rectangles that show jumping, perching and wing snap, another sequence of rectangles indicates the camera view in which the bird mainly points its beard during the beard-up pose. The three camera views are represented by three different colors (magenta, cyan, and yellow).

4.3.3 Behavior Representation: Interaction

The two visualizations are connected: when the user moves the slider along the sequence visualization, the trajectory is shown up until the frame selected with the slider. The slider enables the user to scroll through the sequence and select a frame of interest. The same behavior is encoded with the same color in both visualization types. The user can select which videos to compare. The selected video is shown from all three view points.

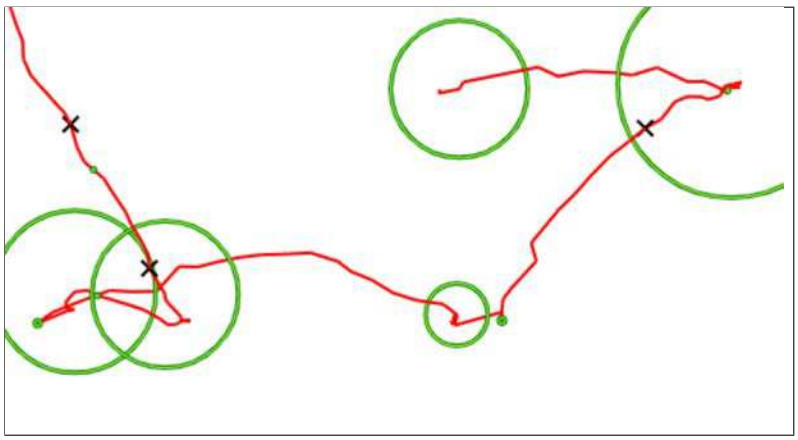

Figure 4.6: Trajectory visualization. Red lines: jumps, radius of circles: duration of perching, crosses: wing snaps.

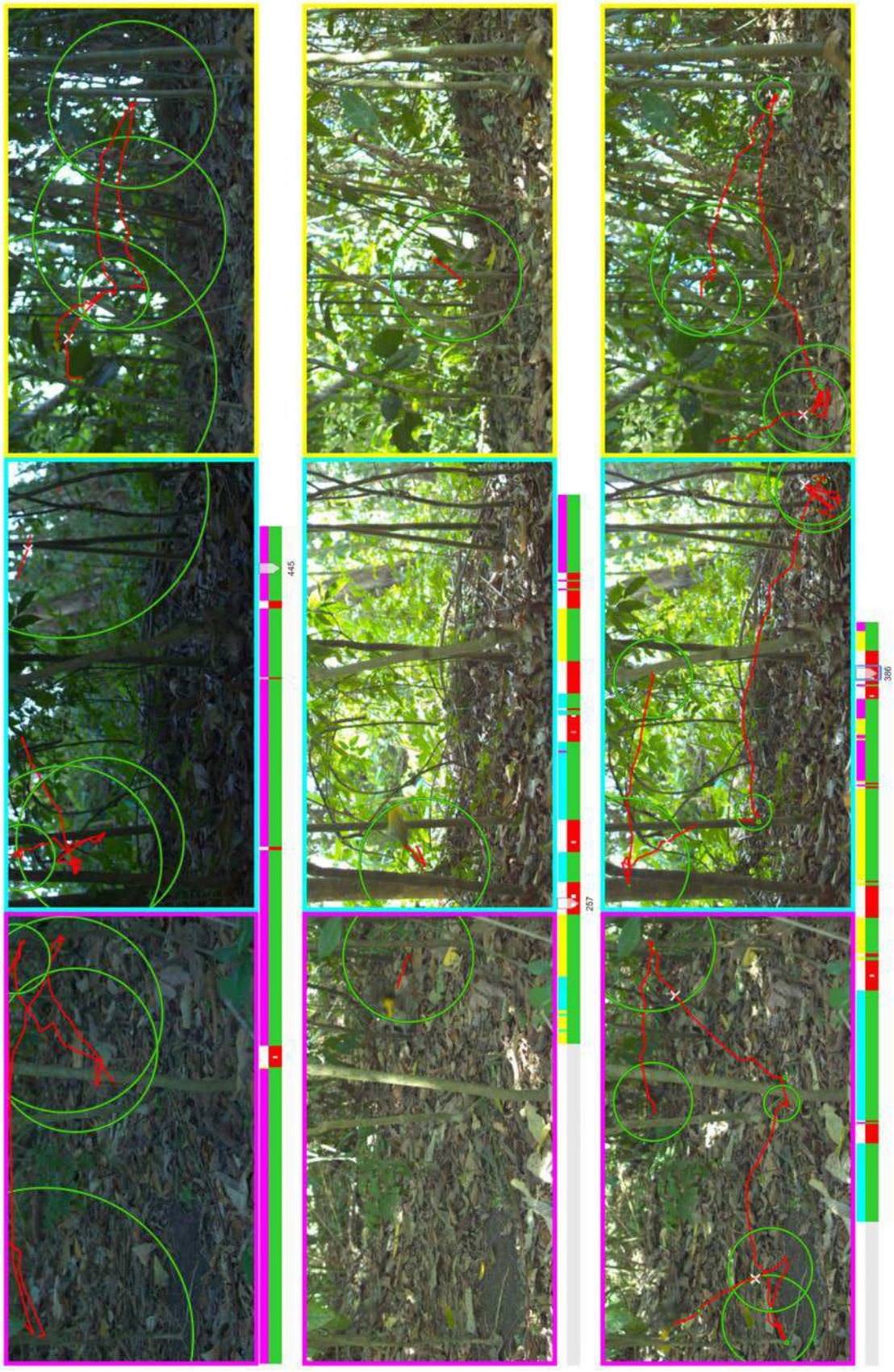

Figure 4.7: Screenshot of Visualization.

CHAPTER 5

Evaluation

5.1 Visual Tracking

We tested all trackers presented in Sections 3.2 and 3.3 on the videos in the two Manakin datasets (see Section 2). To test the ManakinTracker we split dataset-2017 into two parts and trained two CNNs on each halve. We then tested each sequence using the CNN that was trained with the training set that did not contain that sequence. The ManakinTracker was tested on dataset-2018 with a CNN trained using only dataset-2017.

5.1.1 Performance Metrics

To measure the trackers' performance, we measure accuracy and robustness.

Accuracy is measured as the Intersection over Union (IoU) [KLM+18] between the groundtruth bounding box (gt) and the bounding box predicted by the tracker (pred) over all frames for which a groundtruth annotation is available:

$$\frac{1}{N}\sum_{t=1}^{N} IOU(gt_t, pred_t)$$

$$IOU = \frac{area(gt \cap pred)}{area(gt \cup pred)} \in [0,1] \begin{cases} 0, & \text{if gt and pred have no overlap} \\ 1, & \text{if gt and pred are identical} \end{cases}$$

N ... total number of frames in all videos with groundtruth annotation

Robustness is measured as the number of frames in each video until the first tracking failure occurs, starting from the frame in which the bird first moves.

In the beginning of the videos the bird typically sits before it starts jumping (on average 65.18 and 24.58 frames in dataset-2017 and dataset-2018, respectively). Since we wanted

to avoid counting tracking of the sitting bird as successful tracking (which could be achieved by simply keeping the initial ground truth bounding box by default), we start counting the number of tracked frames as soon as the bird has started moving. Starting from the first frame in which the bird has moved so far that its bounding box does not overlap with the bounding box in the initial frame anymore, we count the number of frames until tracking fails, i.e. zero overlap with ground truth.

5.1.2 Performance on Manakin Datasets

We ran the trackers described in Sections 3.2 and 3.3 on the 83 videos in the Manakin datasets (described in Section 2) that contain at least 50 frames with ground truth annotation after the point where the bird first starts moving. Our goal was to evaluate the trackers' robustness and to find out if the trackers are suitable for tracking the videos in the Manakin dataset.

The ManakinTracker is the most robust among all the tested trackers for both datasets. It tracks 3 times more frames in the dataset-2017 and 2.2 times more frames in dataset-2018 than the second most robust tracker ECO.

The ManakinTracker also achieves the highest accuracy for both datasets. The second highest accuracy on dataset-2017 is achieved by the ECO tracker, which is a deep tracker aimed at high accuracy. CSRT reaches a higher accuracy than ECO on the dataset-2018, however, as it only tracks about 17 frames per sequence in that dataset, the accuracy is not as meaningful. The same applies for the accuracy values given for TLD, KCF, MIL and BOOSTING trackers for dataset-2017 have limited significance as the number of tracked frames is below 1 frame per sequence on average.

Among the traditional trackers, all but the CSRT tracker fail at the bird's first move in at least 85% of the sequences. MOSSE and Median Flow performed the worst: they tracked no frames successfully in any of the videos. This is probably caused by the strong appearance change the bird exhibits when starting to move.

Among the traditional trackers, only the CSRT tracker was able to track the bird in more than 50 frames (in 13% of the videos in dataset-2017). However, the CSRT tracker still failed to track the bird when it first moves in 58% of the videos in dataset-2017.

The two deep trackers MBMD and GOTURN only track about 4 frames per sequence on average. Both of those trackers do not adapt to a strong change in target appearance: MBMD does not update the part of the network that proposes potential bounding boxes and GOTURN does not perform any on-line learning.

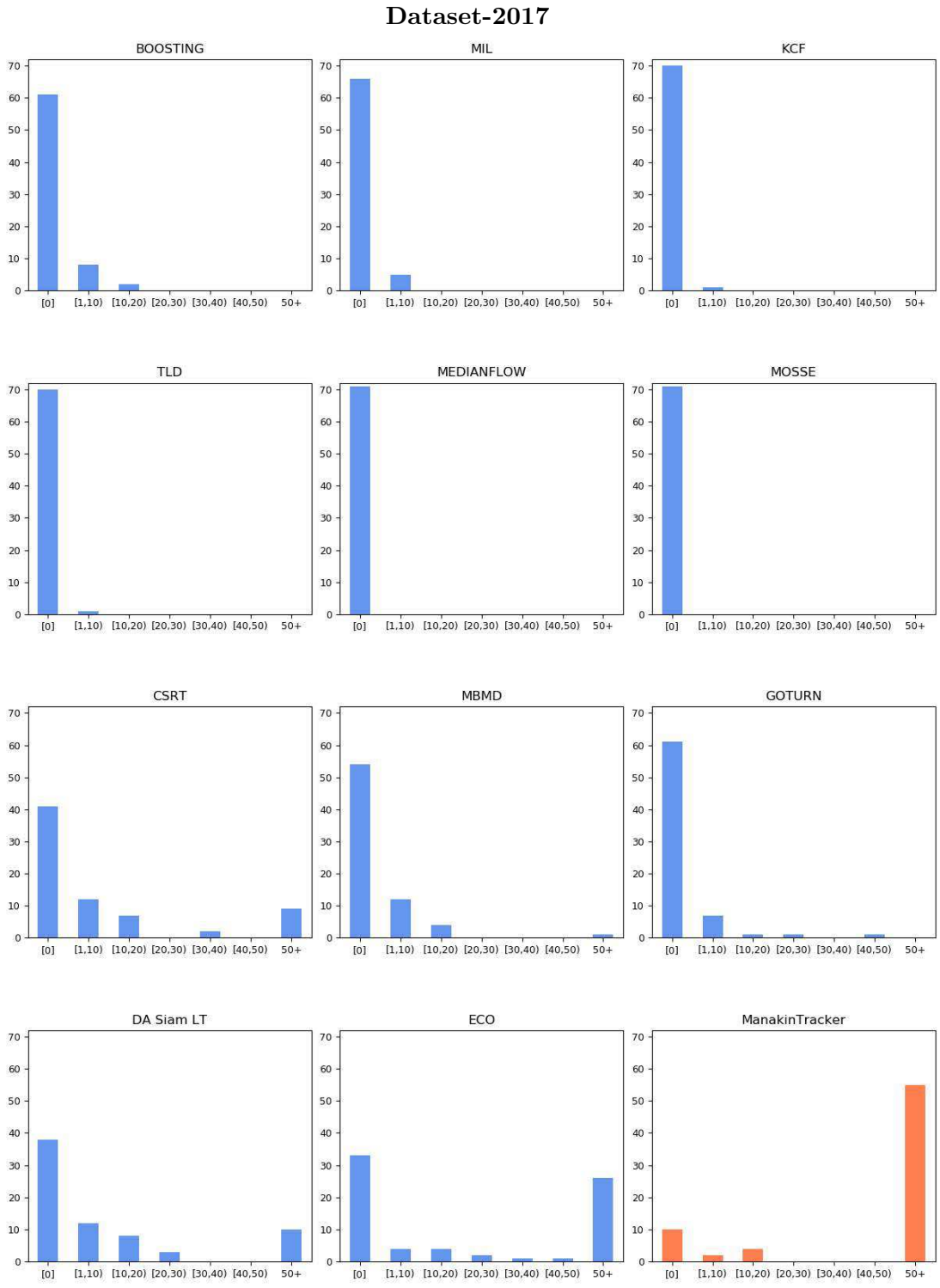

Figure 5.1: The histograms show the number of frames per sequence in **dataset-2017** each of the trackers was able to track the bird after it first starts moving. (Our tracker highlighted in orange)

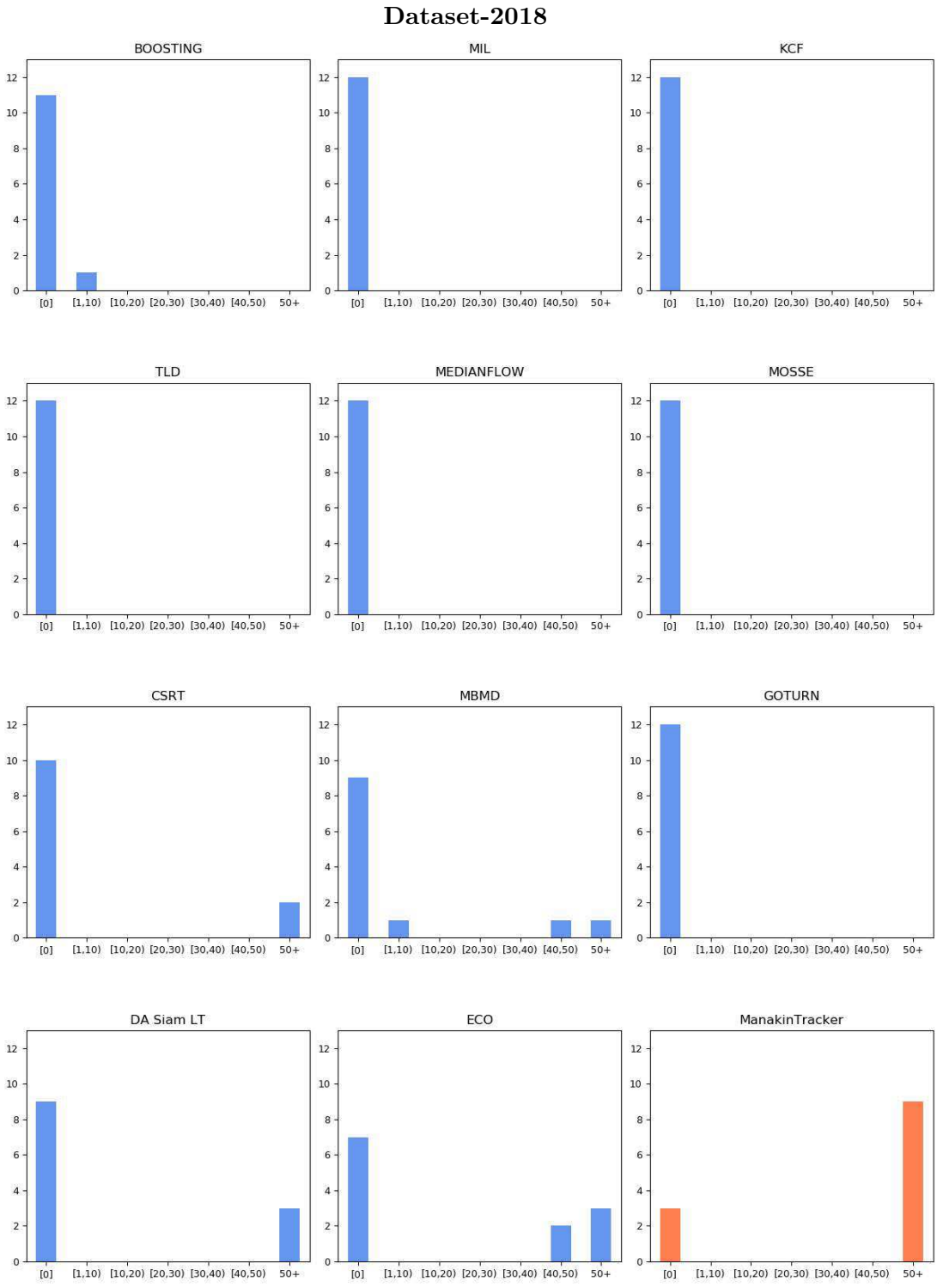

Figure 5.2: The histograms show the number of frames per sequence in **dataset-2018** each of the trackers was able to track the bird after it first starts moving. (Our tracker highlighted in orange)

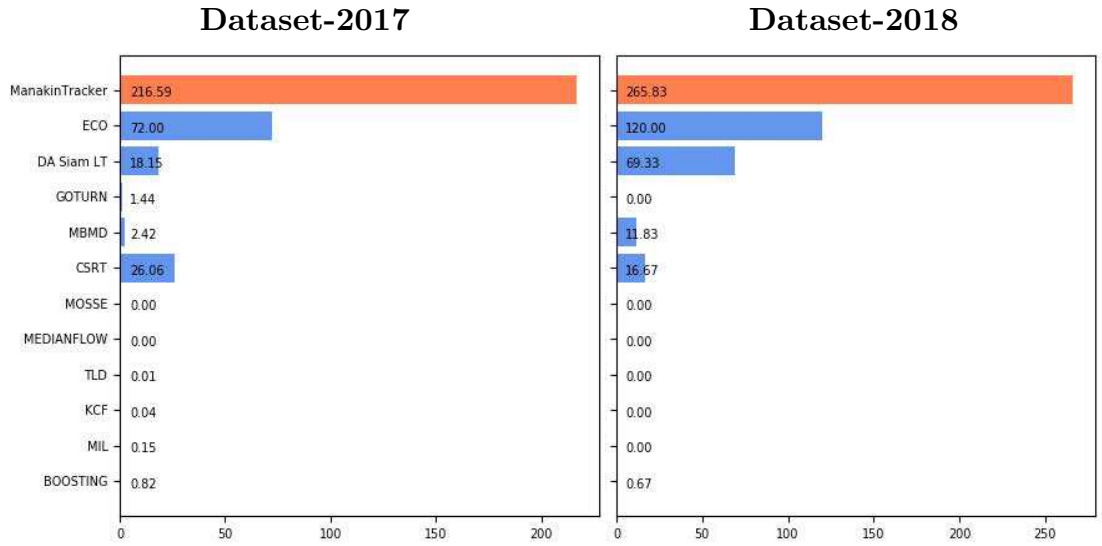

Figure 5.3: Bar chart showing the **average number of successfully tracked frames per sequence** per tracker in dataset-2017 (left) and dataset-2018 (right).

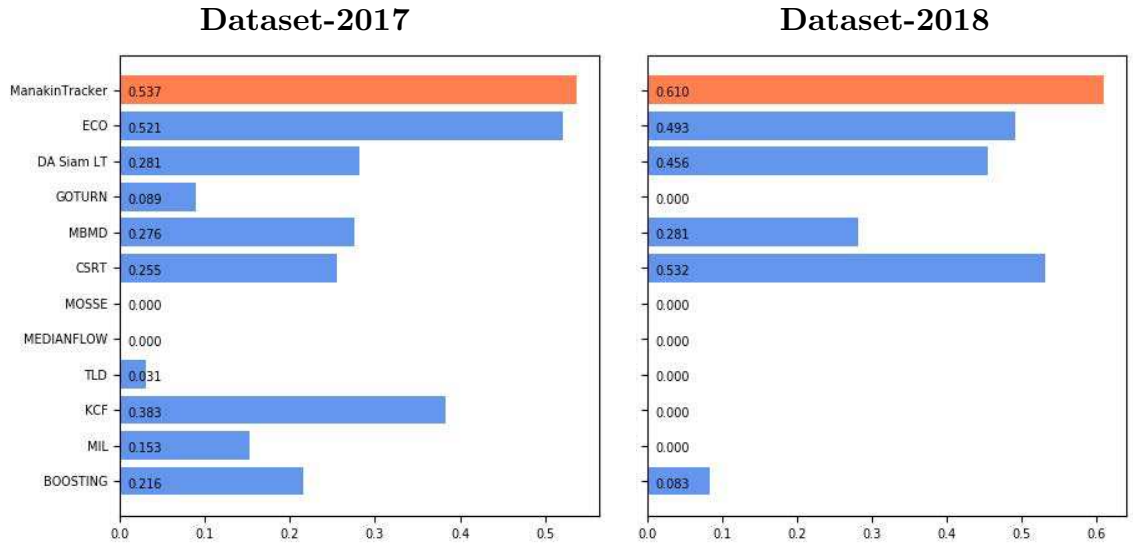

Figure 5.4: Bar chart showing the **average accuracy** achieved by each tracker in dataset-2017 (left) and dataset-2018 (right).

5.1.3 Failure Analysis

To gain a better understanding of the causes of tracking failures, we inspected the frames on which the tracking failure occurred.

Tracking failures can occur when the male bird ...

- is occluded or strongly distorted by motion blur and a piece of clutter that visually resembles the bird is in close proximity (see Figure 5.5).
- jumps directly behind the female bird and the female resembles the male bird (see Figure 5.6).
- sits on a sapling, only the head is detected and it then proceeds to "swing" its head first behind and then to the other side of the sapling (see Figure 5.7).
- is affected by strong motion blur when it starts a jump and is thus not recognized (see Figure 5.8).
- has moved outside of the frame and a piece of clutter or the female bird is tracked instead.

In a technical report [Gos19], we evaluated how motion blur influences the effectiveness of the ManakinTracker. If the CNN used in the ManakinTracker relies on motion blur to recognize the bird, this could pose a problem when the biologists use a different camera with higher frame rate in future recordings because under the same lighting conditions a higher frame decreases motion blur. We simulated the bird's appearance by superimposing three colored circles and found that stronger motion blur lowered the confidence of the CNN and lead to more tracking failure. Thus, we conclude that the ManakinTracker will perform better on sharper videos.

5.2 Behavior Recognition and Representation

Using the distance that the bird has moved between the previous and next frame to distinguish between perching and jumping works in most cases. An incorrect prediction can occur when the bird jumps slowly or either moves while perching or the predicted bounding box is inaccurate. If perching is incorrectly recognized as jumping this will show up in the trajectory view as multiple smaller circles in the location where the bird is sitting. If a frame during a jump is recognized as perching a small circle will appear on the red line that encodes the jump.

Wing-snaps are recognized based on the change of bounding box area between consecutive frames. This wing motion is very similar to the flapping of the wings that the bird does when starting and landing. In the trajectory view, wingsnaps and flapping can be distinguished by their distance to the sapling: flapping occurs close to a sapling. The correct detection of the extension of the wings depends on the level of motion blur.

Figure 5.5: Example of background clutter being falsely recognized as the bird. Right column: Close-ups of the clutter. In the top row the clutter resembles the bird viewed from behind, in the bottom row the clutter resembles the bird's yellow and black head. Green: groundtruth bounding box, red: predicted bounding box.

Figure 5.6: Example of the female being falsely tracked when the two birds' paths cross. Green: groundtruth bounding box, red: predicted bounding box.

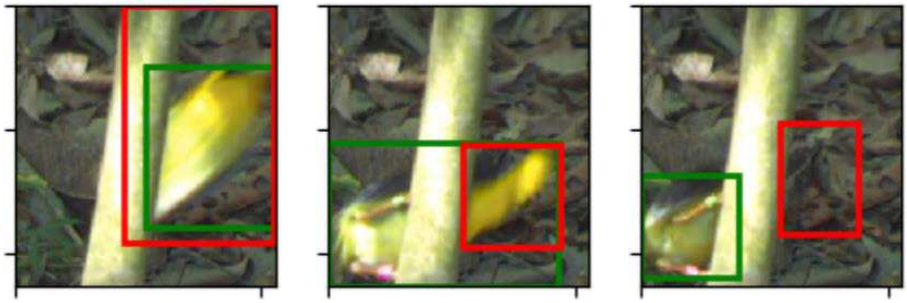

Figure 5.7: Example of the tracker losing the bird when the bird moves its head behind the sapling. Green: groundtruth bounding box, red: predicted bounding box.

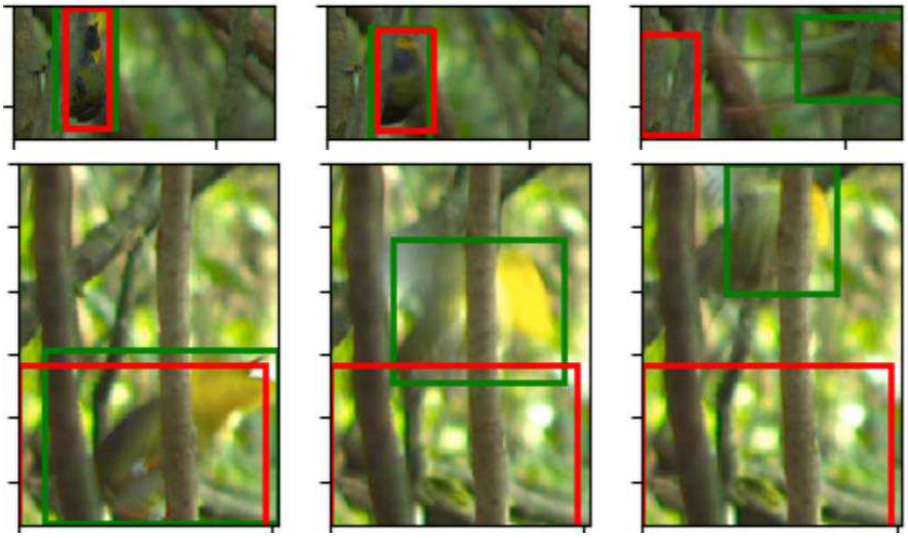

Figure 5.8: Example of the tracker losing the bird when it starts a jump. Green: groundtruth bounding box, red: predicted bounding box.

If motion blur is too strong, the wings will not be included in the blob and thus the predicted bounding box.

To find the direction in which the bird points its yellow neck, we find the largest connected region of yellow pixels among the three camera views. This can fail if the bird is closer to one of the cameras and the neck is projected larger onto the frame recorded by that camera.

An advantage of the trajectory visualization is that it preserves spatial information: for example the user can see the shape of the bird's trajectory and see which saplings the bird lands on most often. The user can see if the bird lands on the same spot – the circles will be concentric if the bird sits again in the exact same location. A disadvantage of the trajectory visualization is that – as the bird jumps back and forth inside its arena – lines

cross and it can be hard to tell different jumps apart. This is mitigated by the three views shown side-by-side where different parts of the choreography can be seen more clearly from a different angle. Additionally, a slider allows the user to show the sequence only until a selected frame and follow the bird's path by scrolling through the video.

The sequence visualization is very compact: it enables the user to compare a large number of sequences at the same time. There are no line crossings – every behavior can be clearly seen. However, as the sequence visualization contains no spatial information (i.e. no information in what place of the frame a behavior occurred), it is beneficial to combine this type of visualization with the trajectory visualization.

While the visualization's main purpose is to gain insight into the bird's courtship dance and to compare different displays, it can also be useful to spot errors in the automatic behavior recognition. For example the user can detect a wingsnap in the video that was not recognized as an area change during automatic behavior recognition or check if a detected wingsnap is instead the flapping of the wings that the bird does when starting and landing.

CHAPTER 6

Conclusion

We collaborated with a team of biologists led by Prof. Dr. Leonida Fusani from the University of Vienna, who provided us with two datasets that show courtship dances performed by a tropical bird called golden-collared manakin. Both datasets have per-frame bounding box annotations of the male bird. The birds were recorded in the jungle, in a strongly cluttered environment, and move at a rapid speed which leads to significant motion blur. Automatic behavior recognition is available for human behaviors and some species that are commonly recorded in laboratories but no such software has been developed for the golden-collared manakin or a comparable species in a highly challenging environment.

We developed the ManakinTracker, a novel visual tracker that is able to handle the specific challenges of the videos in manakin datasets. The ManakinTracker exploits the fact that the videos were recorded with a stationary camera by using a background subtraction method to extract moving objects from the frame that serve as potential locations. It uses a convolutional neural network trained offline on manakin videos to handle the strongly varying appearance of the bird. By estimating the movement of the bird with a Kalman filter, the ManakinTracker can track the bird when it moves through occlusions.

Based on the trajectory obtained through tracking, we extract the bird's moved distance, change in area and the largest yellow region to recognize typical behaviors and visualize them in two linked views. The behavior visualization makes the behaviors visible at one glance and enables the biologists to compare different courtship displays.

Our tracker achieved better robustness and accuracy on the manakin videos than 11 state-of-the-art visual trackers. Automatic tracking and behavior recognition enables the biologists to process the videos much faster than watching the videos frame-by-frame and manually label the position and behavior of the bird. By visualizing each video with two different visualizations the behaviors can be compared in a compact way while preserving spatial information.

List of Figures

1.1 Close-ups of male (from left: first, second) and female (from left: third, fourth) golden-collared manakins 1
1.2 Images of birds affected by strong motion blur. 2

2.1 Example images from dataset-2017. Left: perching bird; right: jumping bird 8
2.2 Example images from dataset-2018. Left: perching bird; right: jumping bird 8
2.3 This drawing shows the golden-collared manakin's courtship dance (image from [FGDS07]) 11

3.1 Birds recorded in a custom-built arena [KvBL15]. 15
3.2 Birds recorded against a distant background [SX08]. 15
3.3 Pigeon with body markers [RBB+11]. 15
3.4 Block diagram of the Median Flow tracker. (image from [KMM10]) . . . 17
3.5 Cyclical shifts produce erroneous wrapped-around edges. Note, for example, the ground that is below the cyclist in the middle frame was shifted above him in the left-most image. (image from [HCMB14]) 18
3.6 To localize the target an image region is convolved with correlation filter channels. The resulting correlation responses are weighted by channel reliability weights and summed up. The peak in the summed correlation responses indicates the new target location. (image from [LVČZ+17]) 21
3.7 Diagram of MBMD. MBMD's network takes as input the current search region and the template from the initial frame. From both, features are extracted (FE) which are fused and feed into the region proposal network (RPN), which proposes potential bounding boxes and corresponding similarity scores. The bounding box proposals receive a classification score from the verification network which is combined with the similarity score assigned by the regression network into a confidence score. Depending on that confidence score the model switches between local and global search. [ZWW+18]. 23
3.8 Instead of simply storing a fixed number of previously seen frames (bottom row), the ECO tracker represents the previously seen frames as a mixture of Gaussian components (top row). Each component captures a different version of the target's appearance [DBKF17]. 25

3.9 Continuous convolution filters are applied to a multi-resolution feature map to produce a continuous confidence score of the target. The target's center in the current frame is assumed to be at the peak of the confidence score [DRSKF16]. 26
3.10 GOTURN's CNN takes the current and previous frame as input and outputs the coordinates of the target bounding box (green box) in the search region inside the current frame. [HTS16]. 27
3.11 State-transition-graph for analyzing the behavior of people by showing their movements between meaningful locations in a city (image from [AA18]). . 34
3.12 Right: Diagram of Swallow-tailed Manakin's courtship dance. The numbers stand for the probabilities of a transition. (image from [RdCGMM19]); Left: Sequence diagram of zebrafinch's courtship dance. (image from [UNS16]) 35
3.13 Visualization of fly behavior. Right image: Peaks (red) represent stereotypical behaviors (image from [BCBS14]). 35

4.1 Flowchart of ManakinTracker. 38
4.2 Foreground mask (right) generated by Mixture Of Gaussians model of frame (left). The blue box marks the extracted blob. 38
4.3 The two small blobs (small blue bounding boxes) are combined into a bigger blob (big blue bounding box). The white text indicates the blobs' target scores. 40
4.4 Bird is sitting and no blob was found (red box: candidate locations with target score $\geq t_3$, white box: final bounding box) 40
4.5 Middle: The Kalman filter's location estimation (black box) is used as the bounding box output when the bird becomes invisible to the camera during a jump. Left, right: The bird is visible and can thus be recognized by the CNN. (green boxes: ground truth; red boxes: candidate locations classified as target; white boxes: bounding box output for current frame) 41
4.6 Trajectory visualization. Red lines: jumps, radius of circles: duration of perching, crosses: wing snaps. 45
4.7 Screenshot of Visualization. 46

5.1 The histograms show the number of frames per sequence in **dataset-2017** each of the trackers was able to track the bird after it first starts moving. (Our tracker highlighted in orange) 49
5.2 The histograms show the number of frames per sequence in **dataset-2018** each of the trackers was able to track the bird after it first starts moving. (Our tracker highlighted in orange) 50
5.3 Bar chart showing the **average number of successfully tracked frames per sequence** per tracker in dataset-2017 (left) and dataset-2018 (right). . 51
5.4 Bar chart showing the **average accuracy** achieved by each tracker in dataset-2017 (left) and dataset-2018 (right). 51

5.5 Example of background clutter being falsely recognized as the bird. Right column: Close-ups of the clutter. In the top row the clutter resembles the bird viewed from behind, in the bottom row the clutter resembles the bird's yellow and black head. Green: groundtruth bounding box, red: predicted bounding box. 53
5.6 Example of the female being falsely tracked when the two birds' paths cross. Green: groundtruth bounding box, red: predicted bounding box. 53
5.7 Example of the tracker losing the bird when the bird moves its head behind the sapling. Green: groundtruth bounding box, red: predicted bounding box. 54
5.8 Example of the tracker losing the bird when it starts a jump. Green: groundtruth bounding box, red: predicted bounding box. 54

List of Tables

2.1 Manakin Datasets 5
2.2 Properties of groundtruth bounding boxes (in dataset-2017). 6
2.3 Properties of groundtruth bounding boxes (in dataset-2018). 6

Bibliography

[AA18] Natalia Andrienko and Gennady Andrienko. State transition graphs for semantic analysis of movement behaviours. *Information Visualization*, 17(1):41–65, 2018.

[AQRSM12] Douglas L Altshuler, Elsa M Quicazán-Rubio, Paolo S Segre, and Kevin M Middleton. Wingbeat kinematics and motor control of yaw turns in anna's hummingbirds (calypte anna). *Journal of experimental biology*, 215(23):4070–4084, 2012.

[ASSRE12] Caroline A Schneider, Wayne S Rasband, and Kevin Eliceiri. Nih image to imagej: 25 years of image analysis. *Nature Methods*, 9, 07 2012.

[BBDL10a] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2544–2550. IEEE, 2010.

[BBDL10b] David S. Bolme, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2544–2550, 2010.

[BCBS14] Gordon J Berman, Daniel M Choi, William Bialek, and Joshua W Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672, 2014.

[BSF15] Julia Barske, Barney A. Schlinger, and Leonida Fusani. The presence of a female influences courtship performance of male manakins. *The Auk*, 132(3):594 – 603, 2015.

[BSWF11] Julia Barske, Barney A Schlinger, Martin Wikelski, and Leonida Fusani. Female choice for male motor skills. *Proceedings of the Royal Society B: Biological Sciences*, 278(1724):3523–3528, 2011.

[BYB09] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 983–990, 2009.

[C+35] Frank Michler Chapman et al. The courtship of gould's manakin (manacus vitellinus vitellinus) on barro colorado island, canal zone. *Bulletin of the American Museum of Natural History 68*, page 472–521, 1935.

[CdAN17] Carlos Fernando Crispim, Fernando Mendes de Azevedo, and José Marino Neto. What is my rat doing? behavior understanding of laboratory animals. *Pattern Recognition Letters*, 94:134–143, 2017.

[CHP+14] Marion Coulon, Laurence Henry, Audrey Perret, Hugo Cousillas, Martine Hausberger, and Isabelle George. Assessing video presentations as environmental enrichment for laboratory birds. *PloS one*, 9(5):e96949, 2014.

[CMF18] Meredith C. Miles and Matthew Fuxjager. Animal choreography of song and dance: a case study in the montezuma oropendola, psarocolius montezuma. *Animal Behaviour*, 140, 05 2018.

[CSVZ14] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *British Machine Vision Conference, BMVC*, 2014.

[CZ17] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the Kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[DBKF17] Martin Danelljan, Goutam Bhat, Fahad Khan, and Michael Felsberg. ECO: Efficient convolution operators for tracking. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6931–6939, 07 2017.

[DM13] Kanjar De and V Masilamani. Image sharpness measure for blurred images in frequency domain. *Procedia Engineering*, 64:149–158, 2013.

[DPAW+19] Talmo D. Pereira, Diego Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S.-H. Wang, Mala Murthy, and Joshua W. Shaevitz. Fast animal pose estimation using deep neural networks. *Nature Methods*, 16, 01 2019.

[DRSKF16] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016.

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, June 2005.

[DuV07] Emily DuVal. Social organization and variation in cooperative alliances among male lance-tailed manakins. *Animal Behaviour*, 73:391–401, 2007.

[DuV09] Emily DuVal. Cooperative display and lekking behavior of the lance-tailed manakin (chiroxiphia lanceolata). *The Auk*, 124:1168–1185, 01 2009.

[EGK+08] Dennis Eckmeier, Bart RH Geurten, Daniel Kress, Marcel Mertes, Roland Kern, Martin Egelhaaf, and Hans-Joachim Bischof. Gaze strategy in the free flying zebra finch (taeniopygia guttata). *PLOS ONE*, 3(12):e3956, 2008.

[FFG+17] Matthew J. Fuxjager, Leonida Fusani, Franz Goller, Lisa Trost, Andries Ter Maat, Manfred Gahr, Ioana Chiver, R. Miller Ligon, Jennifer Chew, and Barney A. Schlinger. Neuromuscular mechanisms of an elaborate wing display in the golden-collared manakin (manacus vitellinus). *Journal of Experimental Biology*, 220:4681–4688, 2017.

[FGDS07] Leonida Fusani, Marta Giordano, Lainy Day, and Barney Schlinger. High-speed video analysis reveals individual variability in the courtship displays of male golden-collared manakins. *Ethology*, 113:964–972, 09 2007.

[GGB06] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. *Proceedings of British Machine Vision Conference (BMVC)*, 1:47–56, 01 2006.

[Gos19] Anna Gostler. The effect of motion blur on the ManakinTracker. Technical Report PRIP-TR-148, PRIP, TU Wien, 2019.

[HCMB14] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):583–596, 2014.

[HTS16] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer, 2016.

[HZC+17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[IWC12] T. C. Ings, M.-Y. Wang, and L. Chittka. Colour-independent shape recognition of cryptic predators by bumblebees. *Behavioral Ecology and Sociobiology*, 66(3):487–496, Mar 2012.

[KCS+17] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[KLM+16] Matej Kristan, Aleš Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Čehovin Zajc, Tomas Vojir, Gustav Häger, Alan Lukežič, and Gustavo Fernandez. The visual object tracking VOT2016 challenge results. *ECCV Workshops*, Oct 2016.

[KLM+18] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, et al. The sixth visual object tracking vot2018 challenge results. *ECCV Workshops*, 2018.

[KMM10] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *2010 20th International Conference on Pattern Recognition*, pages 2756–2759. IEEE, 2010.

[KMM11] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, 2011.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[KvBL15] Daniel Kress, Evelien van Bokhorst, and David Lentink. How lovebirds maneuver rapidly using super-fast head saccades and image feature stabilization. *PLOS ONE*, 10(6):1–24, 06 2015.

[LK81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[LMD14] Katrina Lukianchuk and S M Doucet. Cooperative courtship display in long-tailed manakins chiroxiphia linearis: Predictors of courtship success revealed through full characterization of display. *Journal of Ornithology*, 155:729–743, 03 2014.

[LS12] Wen Li and Dezhen Song. Automatic video-based bird species filtering using periodicity of salient extremities. *Department of Computer Science and Engineering, Texas A&M University, Tech. Rep. TR2012-08-2*, 2012.

[LVČZ+17] Alan Lukezic, Tomas Vojir, Luka Čehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6309–6318, 2017.

[LXC+18] Yun Liang, Fuyou Xue, Xiaoming Chen, Zexin Wu, and Xiangji Chen. A benchmark for action recognition of large animals. *2018 7th International Conference on Digital Home (ICDH)*, pages 64–71, 2018.

[LYW+18] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, June 2018.

[MC11] Emilio Maggio and Andrea Cavallaro. *Video tracking: theory and practice*. John Wiley & Sons, 2011.

[MGPM16] Lilian T. Manica, Jeff A. Graves, Jeffrey Podos, and Regina H. Macedo. Multimodal flight display of a neotropical songbird predicts social pairing but not extrapair mating success. *Behavioral Ecology and Sociobiology*, 70(12):2039–2052, Dec 2016.

[MMC+18] Alexander Mathis, Pranav Mamidanna, Kevin M. Cury, Taiga Abe, Venkatesh N. Murthy, Mackenzie W. Mathis, and Matthias Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 2018.

[MUU+14] Jumpei Matsumoto, Takashi Uehara, Susumu Urakawa, Yusaku Takamura, and Hisao Nishijo. 3D video analysis of the novel object recognition test in rats. *Behavioural Brain Research*, 272:16–24, 2014.

[NMC+19] Tanmay Nath, Alexander Mathis, An Chi Chen, Amir Patel, Matthias Bethge, and Mackenzie W Mathis. Using deeplabcut for 3D markerless pose estimation across species and behaviors. *Nature Protocols*, 2019.

[NNK+17] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Ali Swanson, Craig Packer, and Jeff Clune. Automatically identifying wild animals in camera trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115, 03 2017.

[NTIM11] Yuman Nie, Takeshi Takaki, Idaku Ishii, and Hiroshi Matsuda. Behavior recognition in laboratory mice using hfr video analysis. *2011 IEEE International Conference on Robotics and Automation*, pages 1595–1600, 2011.

[NTX14] Akash D Nakarmi, Lie Tang, and Hongwei Xin. Automated tracking and behavior quantification of laying hens using 3D computer vision and

radio frequency identification technologies. *Transactions of the ASABE*, 57(5):1455–1472, 2014.

[ODS18] Jorge H. Piazentin Ono, Carlos Dietrich, and Claudio T. Silva. Baseball Timeline: Summarizing Baseball Plays Into a Static Visualization. *Computer Graphics Forum*, 2018.

[OFE13] Jan I. Ohlson, Jon Fjeldså, and Per G.P. Ericson. Molecular phylogeny of the manakins (aves: Passeriformes: Pipridae), with a new classification and the description of a new genus. *Molecular Phylogenetics and Evolution*, 69(3):796 – 804, 2013.

[OGS15] N. Ota, M. Gahr, and M. Soma. Tap dancing birds: The multimodal mutual courtship display of males and females in a socially monogamous songbird. *Scientific Reports*, 5, 2015.

[OR01] Nikunj Chandrakant Oza and Stuart Russell. *Online ensemble learning*. University of California, Berkeley, 2001.

[Per17] Elisa Perinot. Novel approaches to study the elaborate courtship behaviour of the golden-collared manakin. Master's thesis, Università degli Studi di Padova, Italy, 2017.

[RBB+11] Ivo G. Ros, Lori C Bassman, Marc A Badger, Alyssa N Pierson, and Andrew A. Biewener. Pigeons steer like helicopters and generate down- and upstroke lift during low speed turns. *Proceedings of the National Academy of Sciences of the United States of America*, 108(50):19990–19995, 2011.

[RdCGMM19] Pedro Leite Ribeiro, André de Camargo Guaraldo, R. H. Macedo, and Lilian Tonelli Manica. Variation within and between courts in visual components of swallow-tailed manakin (chiroxiphia caudata) display. *Journal of Ornithology*, 160:485–496, 2019.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[RSH+17] Curtis T. Rueden, Johannes Schindelin, Mark C. Hiner, Barry E. DeZonia, Alison E. Walter, Ellen T. Arena, and Kevin W. Eliceiri. Imagej2: Imagej for the next generation of scientific image data. In *BMC Bioinformatics*, 2017.

[SBD+13] Barney A. Schlinger, Julia Barske, Lainy Day, Leonida Fusani, and Matthew J. Fuxjager. Hormones and the neuromuscular control of courtship in the golden-collared manakin (manacus vitellinus). *Frontiers in Neuroendocrinology*, 34(3):143–56, July 2013.

[SDF08] Barney A Schlinger, Lainy B Day, and Leonida Fusani. Behavior, natural history and neuroendocrinology of a tropical bird. *General and comparative endocrinology*, 157(3):254—258, July 2008.

[SG99] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking, 02 1999.

[SGK15] Adam Michael Stewart, Robert Gerlai, and Allan V. Kalueff. Developing higher-throughput zebrafish screens for in-vivo cns drug discovery. *Frontiers in Behavioral Neuroscience*, 9:14, 2015.

[SGL17] Edwin Scholes, Julia M Gillis, and Timothy G Laman. Visual and acoustic components of courtship in the bird-of-paradise genus astrapia (aves: Paradisaeidae). *PeerJ*, 5:e3987, 2017.

[SHY15] Ulrich Stern, Ruo He, and Chung-Hui Yang. Analyzing animal behavior via classifying each video frame using convolutional neural networks. *Scientific reports*, 5:14351, 2015.

[SKL+15] Alexandra Swanson, Margaret Kosmala, Chris Lintott, Robert Simpson, Arfon Smith, and Craig Packer. Snapshot serengeti, high-frequency annotated camera trap images of 40 mammalian species in an african savanna. *Scientific Data*, 2:150026, 06 2015.

[Sta08] Theodore Stankowich. Quantifying Behavior the JWatcher Way. Daniel T. Blumstein and Janice C. Daniel. *Integrative and Comparative Biology*, 48(3):437–439, 02 2008.

[SX08] Dezhen Song and Yiliang Xu. Monocular vision-based detection of a flying bird. Technical report, Texas A & M University, 2008.

[UNS16] Robert Ullrich, Philipp Norton, and Constance Scharff. Waltzing taeniopygia: Integration of courtship song and dance in the domesticated australian zebra finch. *Animal Behaviour*, 112:285–300, 02 2016.

[vDvdHtB+13] Elsbeth A van Dam, Johanneke E van der Harst, Cajo JF ter Braak, Ruud AJ Tegelenbosch, Berry M Spruijt, and Lucas PJJ Noldus. An automated system for the recognition of various specific rat behaviours. *Journal of neuroscience methods*, 218(2):214–224, 2013.

[VDWSVL09] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1523, 2009.

[VJ+01] Paul Viola, Michael Jones, et al. Rapid object detection using a boosted cascade of simple features. *CVPR*, 1(511-518):3, 2001.

[WB95] Greg Welch and Gary Bishop. An introduction to the Kalman filter, 1995.

[Win18] Gernot Winkler. Behaviour monitoring from honey bees based on video analysis. Master's thesis, Technische Universität Wien, Austria, 2018.

[ZBW+09] Patrick H Zimmerman, J Elizabeth Bolhuis, Albert Willemsen, Erik S Meyer, and Lucas PJJ Noldus. The Observer XT: A tool for the integration and synchronization of multimodal signals. *Behavior research methods*, 41(3):731–735, 2009.

[ZWB+18] Zheng Zhu, Qiang Wang, Li Bo, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *European Conference on Computer Vision*, 2018.

[ZWW+18] Yunhua Zhang, Dong Wang, Lijun Wang, Jinqing Qi, and Huchuan Lu. Learning regression and verification networks for long-term visual tracking. *CoRR*, abs/1809.04320, 2018.

[ZZXW19] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–21, 01 2019.