Freie Universität Berlin

A Computer Vision Based System for the Automatic Analysis of Social Networks in Honey Bee Colonies

Dissertation zur Erlangung des Grades
eines Doktors der Naturwissenschaften (Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

von

Fernando Wario Vázquez

aus Mexiko

Berlin
Dezember 2017

A Computer Vision Based System for the Automatic Analysis of Social Networks in Honey Bee Colonies

Dissertation

Autor:	Fernando Wario Vázquez
Erstgutachter:	Prof. Dr. Raúl Rojas Freie Universität Berlin
Zweitgutachter:	Prof. Dr. Martin Nawrot Universität zu Köln
Tag der Disputation:	20 Dezember 2017

Diese Arbeit wurde vom Deutschen Akademischen Austauschdienst gefördert.

Summary

This thesis describes the development and implementation of the *BeesBook System*, a computer vision based solution for the automatic detection and analysis of behavioral patterns of honey bee colonies at the individual and collective level.

The behavioral analysis of honey bee colonies requires extensive data sets describing the behavior of individual colony members. These data sets must often be created manually - a time consuming and cumbersome activity. Consequently, behavioral data sets are usually restricted to small subsets of the colony's life, whether this regards to time, space or animal identity. By automating the data acquisition process, the *BeesBook* system allows the supervision of a higher number of individuals during more extended periods of time, opening the door to more sophisticated, inclusive and significant studies.

The *BeesBook System* uses unique binary markers attached to the bees to keep track of their position and identity via computer vision software. The markers' flexible design allows the implementation of a diversity of error-correcting codes, depending on the study's goals and the colony's population size. The markers adapt to the bee's thorax shape creating a surface that withstands heavy-duty activity in and outside of the hive.

Three recording seasons were conducted during the summers of 2014, 2015, and 2016 to evaluate and improve the performance of the system components. Each season extended over a period of nine weeks and generated approx. 65 million images. Prior to the beginning of each season, all members of a bee colony were individually tagged and transferred to an observation hive. The activity inside the hive was recorded using an array of four high-resolution cameras and stored for later analysis on one of the complexes of the North-German Supercomputing Alliance. Communication dances were identified in real-time using a second set of cameras comprised of two webcams running at high frequency. During the off-season, the experimental design was optimized to ensure that the generated data better serve the target of the experiment.

Stored images were processed using highly optimized computer vision software to obtain the position, orientation, and ID of every marked bee. These data are further processed to generate motion paths for the colony members, which, combined with data on the communication dances, constitute an unprecedented set of knowledge on the inner life of the honey bee colony. The information obtained through this system establishes the conditions for consolidating our understanding of already known behaviors. Furthermore, this research has identified previously unknown behavioral data which ultimately extend our knowledge of bee colonies and their collective intelligence.

A mis padres María Socorro y José Manuel.

Acknowledgments

First and foremost, I am sincerely grateful to Prof. Dr. Raúl Rojas for giving me the opportunity to be a part of his research group. His constant guidance made the successful completion of this thesis possible.

I am truly thankful to Prof. Dr. Tim Landgraf for his supervision throughout my doctoral research, his constant support and works ethics where primordial for the proper completion of this thesis.

I would also like to express my gratitude towards the whole BioroboticsLab Team, and all students who collaborated throughout the years in the BeesBook project. Especial thanks go to Benjamin and Leon for developing the Neural Network for the image analysis module. I thank Prof. Dr. Rojas, Prof. Dr. Landgraf, Benjamin, Franziska, and David as co-authors of my publications for their valuable insights. It was a pleasure work with them.

My life in Germany during the last five years has been a great human experience thanks to many great people I have crossed in my path. Thanks to Giovana and Arturo for their friendship and for listening patiently to my endless talks about bees. I thank Malte and Claas for their firendship and for being my main link to the German culture, introducing me to many nice people, especially to their parents Ursula and Joachim who made me feel like home in Uelzen. I thank my colleges in the research group Ricardo, Margarita and José for their camaraderie and Omar for the interesting discussions and for keeping up the common writing spirit. I thank my friends from the language courses Josué, Dao, and Sukanya whose friendship has lasted despite the distance.

I thank my friend Arturo for always being available to provide me with valuable advice during the writing of this thesis. Thanks to Sara for correcting my English in such a short notice. I also thank Andrés for his invaluable help during the very last stage of writing this thesis.

My greatest thanks to my parents María Socorro y José Manuel for accepting and supporting all my life choices and for their unconditional love. I thank my brothers Manuel, Jorge y Luis for their moral support during these last five years.

This project would have been impossible without the support of the Deutsche Akademische Austauschdienst (DAAD).

Contents

Summary i

1 Introduction 1
1.1 Honey Bee Colonies and Division of Labor 2
1.2 The Waggle Dance, Dancers, and Followers 3
1.3 Motivation for this Work 6
1.4 Overview of the Thesis 8

2 State of the Art of Animal Tracking Systems 10
2.1 The Path to Computational Ethology 10
2.2 Tracking Unmarked Animals 12
2.3 Tracking Marked Animals 20
2.4 Detecting and Decoding Communication Dances 22
2.5 Chapter Overview 25

3 Design and Implementation of the BeesBook System 26
3.1 Preliminary Work 27
3.1.1 Individual Binary Tags 27
3.1.2 Cameras and Camera Modification 31
3.1.3 Observation Hive, Scaffold and Illumination Setup 33
3.2 Image Acquisition Software 35
3.3 Image Analysis 38
3.3.1 CV-Pipeline 39
3.3.2 DCNN-Pipeline 42
3.4 Tracking and ID Filter 46
3.5 Automatic Detection and Decoding of Communication Dances 48
3.5.1 Target Features 49
3.5.2 Software Modules 50
3.6 Chapter Overview 58

4 Experimental Validation 60
4.1 Experimental Design of Recording Seasons 60
4.2 Highly Parallelized Image Processing on Cray Supercomputer 64

4.3 Image Analysis Performance 67
4.3.1 CV-Pipeline Performance 68
4.3.2 DCNN-Pipeline Performance 70
4.4 Tracking and Temporal ID Filter Performance 72
4.5 Automatic Waggle Dance Decoder 76
4.6 Conclusions 78

5 Case Studies **81**
5.1 Spatial Distribution of Foraging Groups on the Honey Comb 81
5.2 Mapping Dances Back to the Field 85
5.3 Discussion 87

6 Concluding Remarks **89**
6.1 Contributions 90
6.2 Directions for Future Work 91

Bibliography **93**

Chapter 1

Introduction

This thesis describes the development and implementation of the *BeesBook* system, a vision based solution for the automatic detection and analysis of behavioral patterns on honey bee colonies at the individual and collective level.

The collective behavior of the honey bee and its highly efficient foraging ecology have captivated humans for centuries. Already in ancient Greece, Aristotle documented his insights on the honey bee in his *Historia Animalium*, including an account of the *waggle dance*, a behavior observed in some bees after returning to the hive from collecting food or scouting for new nest locations. In 1946, more than 2,000 years later, Karl von Frisch proposed for the first time a correlation between some of the dance's characteristics and the location of the resources visited by the dancers [115], in what is, up to now, the only known case of abstract communication among insects.

After von Frisch's discovery, biologists have continued to study the waggle dance and the different ways in which it shapes the collective behavior of honey bee colonies [38, 95, 116]. As part of a general trend in ethology, the nature of these studies has gradually moved from an exclusively qualitative approach to a predominantly quantitative one. Quantitative studies, unlike their qualitative counterparts, require gathering and analyzing a substantial amount of data; this change of approach in ethology has led to a demand for new tools and methodologies that allow a more efficient acquisition and analysis of data.

It is under these conditions that the new field of *computational ethology* has emerged, bringing together advances in the areas of technology, engineering, and mathematics, and with the primary goal of automating measurement and analysis of animal behavior [3]. The *BeesBook* system, as a computer vision based solution for the automatic tracking of honey bees, is an excellent example of the scope intended by computational ethology.

To fully understand the motivation for this work, and appreciate its ambitions, it is imperative to review two of the features that have made the honey bee the interesting object of study that it is nowadays: division of labor and the waggle dance.

Figure 1.1: **The experimental methods developed by Karl von Frisch remained practically unaltered during decades.** Only until recent years, with the arrival of new technologies, these methods have been significantly modified. *(Image by: Nina Leen-Time Life Pictures, 1964).*

1.1 Honey Bee Colonies and Division of Labor

During foraging season, which usually extends over spring and summer, an average honey bee colony consists of some 20,000 members, from which the vast majority are workers- all these are females. The rest of the population consists of a few hundred males and the queen, mother of the colony. During winter the overall colony's population usually descends to just a few thousand bees, and the small male population is completely wiped out [95].

Population reduction is not the only seasonal change that takes place within the hive; bees also exhibit seasonal age polyethism. During foraging season, when food accumulation and growth rate are priorities, the colony optimizes these processes through age polyethism, this means that each bee performs a particular task depending on its age. During winter, when no new resources are available, and the colony must survive with the food gathered during the most recent foraging season, all workers become generalists [48].

Each of the activities performed by a worker bee during foraging season is closely related to her ongoing physiological stage, in particular to the maturity of its glandular system [95]. Biologists have hypothesized for a long time about the internal and external aspects that compel a worker to switch from one activity to another, resulting in the *temporal polyethism* observed in honey bee colonies. Johnson [48] conceptualizes the more representative results on honey bee's division of labor research into two models. The first model works at the behavioral level and explains the form and adaptive basis of age polyethism; it proposes that nurses are pushed from their caste by young bees ready to take their place, and mature workers are pulled from their caste as a consequence of interactions with members of the caste above them. The second model works at the proximate level and proposes that the colony level needs are translated into individual-level patterns of physiological development.

Regardless of which of the two models is closer to the truth behind the division of labor mechanisms; there is a clear correlation between the tasks performed by workers during foraging season and their age [48]. In fact, the type of tasks is allocated according to an specific age: **cell cleaner** (0 - 3 days old), **nurse** (4 - 12 days old), **food-storer** (13 - 20 days old) and **forager** (21 - dead around day 42).

Age polyethism allows honey bee colonies to constitute themselves into sophisticated biological organizations [95]. Thus, working as a unit at the group level enables the colony to transcend their individual capabilities and succeed in complex tasks such as; optimal exploitation of multiple food sources, control of the hive's micro-climate, and regulation of the hive's population. These complex and highly coordinated behaviors, which at first glance calls for the existence of a central intelligence, are in fact the result of myriads of interactions among colony members. Since each bee has only access to local stimuli [82], the flow of information across the colony must play a crucial role in the emergence of the behavioral patterns observed at the macroscopic level [46].

1.2 The Waggle Dance, Dancers, and Followers

When a bee discovers a highly valuable resource on the field, for instance, a patch of flowers or a potential new location where to nest, she shares this resource's location with her nest-mates via a series of symbolic body movements. This peculiar behavior, which usually takes place on the honeycomb surface, is now commonly known by the name of waggle dance [65]. Waggle dances consist of two distinct phases, the waggle run, and the return run. During the waggle run, the dancer vibrates her body from side to side while moving forward in a rather straight line. Each waggle run is followed by a return run during which the dancer circles back to the starting point of the previous waggle phase. Right and left return runs are alternated, this way the dancer describes a path resembling the figure eight (Fig. 1.2).

The importance of the waggle dance resides in the fact that some of its properties correlate to the resource's polar coordinates on the field and that it is used as a mean of communication by the honey bee. Firstly, the average orientation of consecutive waggle runs, measured from the hive's vertical, approximates the angle of the advertised resource and the solar azimuth as perceived from the hive (Fig. 1.2). Furthermore, the duration of the waggle run correlates to the distance between hive and resource

[19, 27, 116].

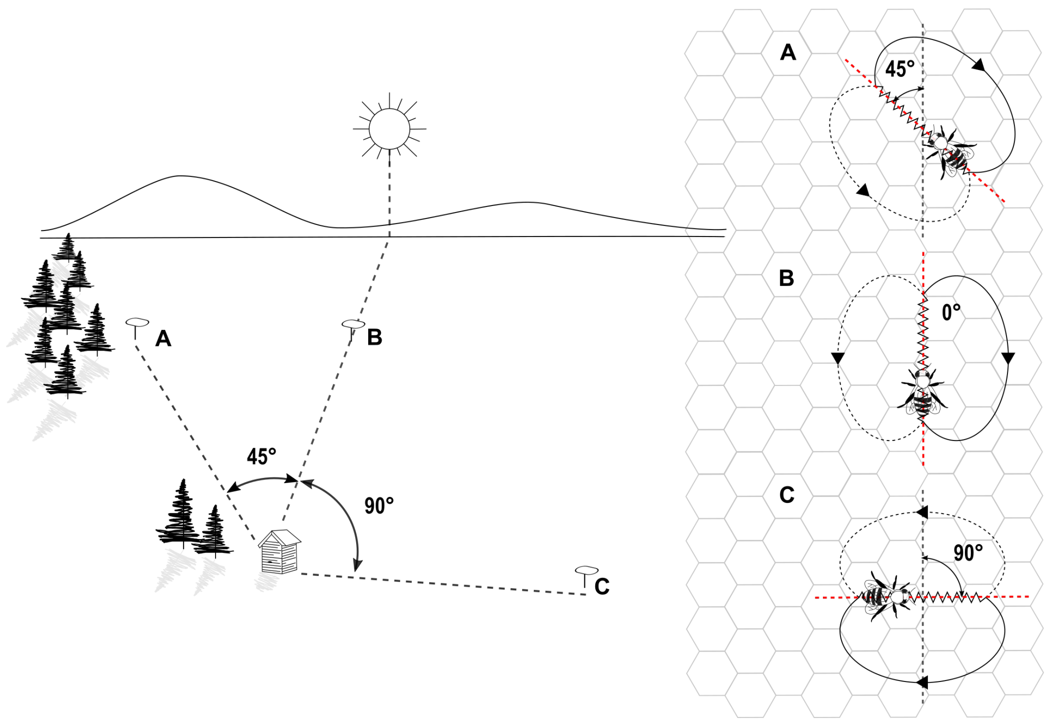


Figure 1.2: **Food source locations and their representations as waggle dance.** On the left, three food sources on the field located at (**A**) 45° counterclockwise (**B**) 0° and (**C**) 90° clockwise with respect to the azimuth and on the right the correspondent waggle dance paths on the surface of a vertical honeycomb.

A correlation between the profitability of the resource and the timing of the dance advertising its location has been observed. Similarly, return runs for highly profitable sites are considerably shorter than those for other not so profitable localities, yielding this way a higher waggle production rate [97]. Nevertheless, it remains unclear if dance attendants make use of this information or even if they are capable of reading it from the dance.

When a forager bee dances on the comb surface, one or more unemployed foragers might become interested, track her movements, translate the information encoded in the dance and search for the resource in the field [1, 8, 69, 91, 95]. This process is usually carried out multiple times since bees very rarely find the advertised resource on their first try [98]. Successful followers, once back in the hive, may also advertise the visited location, thereby incrementing the probability of new recruitment for their resource location.

Bees also exhibit a negative feedback mechanism known as stop signal. When a forager executes the stop signal, she knocks her head against the dancer in conjunction with a short burst of thorax vibration [55, 75, 76]. The stop signal is used as a mean to discourage recruitment to specific locations, for instance when a food source represents a threat due to aggressive food competition with other colonies or during nest selection when a different scout has found a better location to establish the colony. Waggle dance and stop signal complement each other. Furthermore, they constitute the mechanism that allows honey bees select the best nest site when swarming and to optimally distribute the population of foragers across the available food sources.

Pursuing a better understanding of the communication process carried out during the waggle dance, *Landgraf et al.* developed a robotic bee that mimics all potential stimuli known to be emitted by foragers during communication dances [62, 60]. Experiments conducted with the robotic dancer proved its ability to communicate new food locations to novice foragers. However, the response to the robotic dancer was not stable, some bees kept returning to the robot and attending to its dances, while other bees did not follow the robot at all.

A possible explanation for the irregular response to the robotic dances is that bees might forage in groups defined according to a social structure inherent to the colony [59]. If this hypothesis is correct, it will imply that the waggle dance is not self-contained and that the colony's social structure plays a major role in the decision of an unemployed forager whether to follow or not a particular dancer [1, 39]. These *foraging groups* would consist of nest-mates who have built a partnership in the base of frequent interactions, where each new encounter increases the probability for further collaborations. Since interactions between nest-mates are not limited to their life-stage as foragers, this record of interactions could extend over multiple stages, probably all the way back to their early days as nurses.

In support of the *foraging groups* hypothesis, some studies indicate that bees are capable of discriminating between members of their patriline -their super sisters- and members of other patrilines- their half-sisters- [35, 67, 72]. Furthermore, *Oldroyd et al.* [80] observed a preference among unemployed foragers for dances performed by their super-sisters over dances by their half-sisters. A later study [81] suggested a preference for certain foraging distances linked to genetic variances as a plausible explanation for this nepotistic behavior. This problem was revisited by *Kirchner and Arnold* in [54], who worked with two colonies of 2 and 17 subfamilies. After observing multiple dances, dancers and recruits were collected to conduct DNA studies and determined their subfamily. Results from this study indicate no preference from unemployed foragers for dances of their super-sisters over those of their half-sisters. However, it is worth noticing that the number of workers analyzed during this study is significantly small when compared to the colony's overall population; each of the two colonies consisted of about 6,000 bees, and only 100 workers from the 2 subfamilies colony and 134 from the 17 subfamilies colony were analyzed.

Most of the studies with a focus on honey bee colonies and their social structure face similar conditions to those encountered by *Kirchner and Arnold* in [54], where the number of individuals and the amount of data analyzed is strongly restricted by the manual and time-consuming nature of the data collection process.

1.3 Motivation for this Work

The main motivation for this work is the need in the field of ethology for new technologies that allow the generation of more extensive and reliable data sets. As pointed out during the introduction to this chapter, studies on animal behavior have become increasingly quantitative over time; unfortunately, technologies for behavior quantification have not developed at the same pace, and manual scoring remains the most common approach for this task [3].

Data collection from experiments with social insects like ants and termites, frequently involves tracking the activity of hundreds or even thousands of individuals. Unfortunately, most of the available methods for data collection remain to a large extent manual procedures that rely on human observation. As a consequence, there are two surmounting obstacles to any similar research: the man hours that the project demands, together with the subjectivity brought by the researchers. Even when assisted by special purpose software, manually scoring video recorded activity is a slow and tiresome task, which is why most of the quantitative studies are restricted to short periods of time and small groups of individuals. One technique that has been used to speed up the data collection stage and improve the depth and breadth of observation is parallelization of the scoring process. Unfortunately, parallelized scoring adds the extra difficulty of standardizing data from multiple sources [12].

Behavioral studies of the honey bee are not strange to quantitative analyses and the challenges that they pose. Studies as those conducted by von Frisch during the mid-20th century with a focus on the characterization of the waggle dance and its validation as a mean of communication [115], have been gradually replaced by studies with a focus on collective behavior and the analysis of interactions within groups of individually marked bees [95]. This kind of studies have been already conducted for a long time, and although there is no formal standard procedure, most of them follow the same general lineaments. For instance, one or more honeycombs are kept in a one layer array inside a glass cabinet known as *observation hive*. The limited space between honeycomb and glass panes forces the bees to move over the honeycomb surface without walking one over the other and maximizing the activity exposed to the observer. Colony members who are part of an ongoing experiment are marked with a paint stroke to ease their localization inside the hive, if the experiment requires it, they can be individually tagged by attaching numbered plastic markers to their thorax. The behavior annotation is then performed either on real-time, directly at the observation hive with the help of protractors and stopwatches [106, 116, 114, 117], or after the fact using digital video recordings (Fig. 1.3) and applying either manual [6], semi-automatic [23] or automatic techniques [62].

In recent years, some of the characteristic behaviors observed in honey bee colonies have gained attention in fields of knowledge other than ethology. In computer science, for instance, honey bee collective foraging has inspired the development of new optimization algorithms [50, 85, 126]. In complexity sciences, honey bee colonies have also

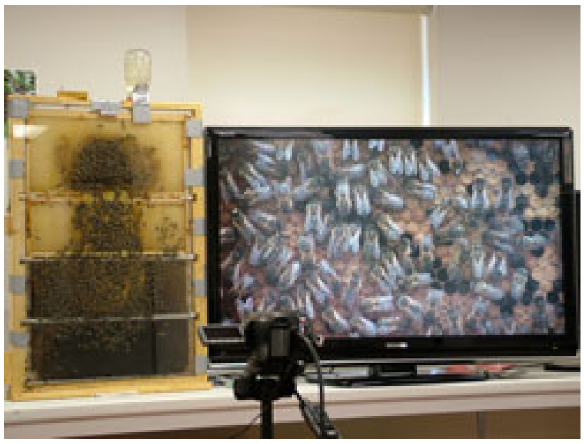


Figure 1.3: **Video recording of observation hive.** Video recording is now ubiquitous in ethology, it provides a permanent register of experimental raw data, and allows the conduction of more comprehensive studies after the fact *(Image by: Laboratory of Apiculture and Social Insects - University of Sussex)*.

become a common object of study, including successive studies that model their collective foraging [13, 24] and nest-site selection behavior [96, 83, 90, 89, 73]. In fact, with more detailed longitudinal studies and following the bottom-up approach described by Crick [18] and Seeley [95] (Fig. 1.4), it would be possible to actually define rules and mathematical models that are more in line with the actual mechanisms behind the colony's collective behavior.

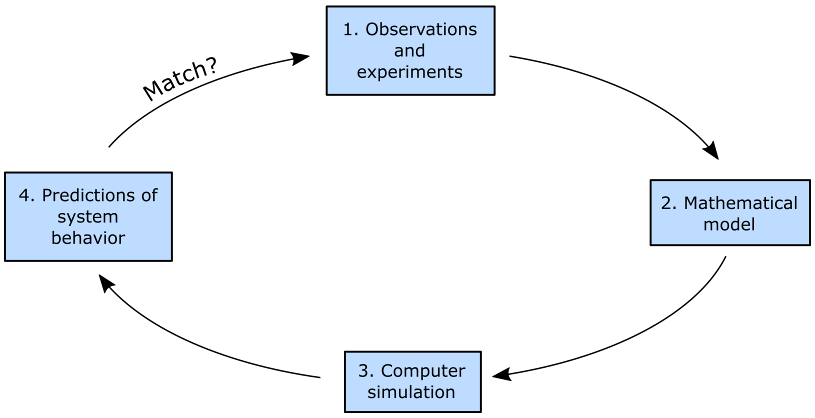

1. Observations and experiments
2. Mathematical model
3. Computer simulation
4. Predictions of system behavior
Match?

Figure 1.4: **Cycle of studies to understand a complex system.** Understanding a complex system like a colony of honey bees is an iterative process that begins with experimental observations, followed by the proposal of rules or a mathematical model that aims to reproduce the observed behaviors. The model can then be simulated and the output compared with the observed data, if necessary new observations are conducted, the model is adjusted, and the process starts all over again.

The need for new solutions to the data acquisition and analysis problem in studies of the honey bee is clear, moreover, its benefits are substantial. The ability to perform more comprehensive studies with deeper and broader data sets would allow us to gain a better understanding of the colony's social structure and the mechanism that shapes its collective behavior. An ideal solution to this problem would be one that allows tracking activity at the individual level during long uninterrupted periods of time.

Here is presented the *BeesBook* system, a computer vision based solution for the automatic tracking of behavioral activity in honey bee colonies. The system comprises recording and storage of high-resolution images, computer vision software for identifying uniquely marked bees (Fig. 1.5) and automatic detection and decoding of waggle dances. Unlike any other system of its kind, the *BeesBook* system is able to track all components of the waggle dance activity: Location, orientation and identity of every colony member inside the observation hive, 24 hours a day over several weeks. The system is conceived as a budget-priced framework for the incremental implementation of hardware and software modules.

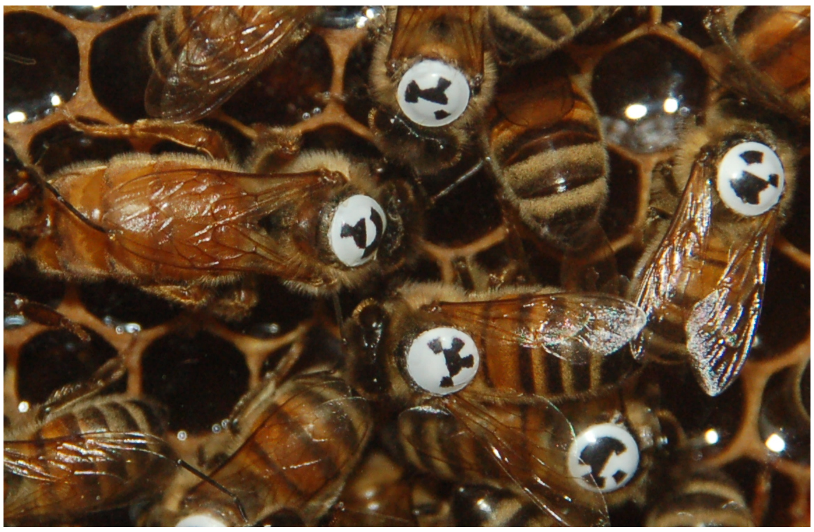


Figure 1.5: **Tagged bees inside the observation hive.** Prior to the beginning of each season, all members of a bee colony were individually tagged and transferred to an observation hive.

1.4 Overview of the Thesis

This work comprises of five additional chapters. The second chapter, devoted to the state of the art of animal tracking systems, provides a brief overview of the methods used so far in the study of behavioral patterns in social insects. The chapter pays particular attention to the work of Dankert et al. [20], Mersch et al. [70], and Crall et al. [17]; three notable examples of computer vision-based systems for the tracking of marked and unmarked social insects.

The third chapter provides a detailed description of the system components, beginning with the preliminaries such as the design of the individual binary tags and the recording setup. Next, it describes all elements involved in the image acquisition process. After that, it outlines the computer vision algorithms implemented to locate and decode the binary tags followed by the post-processing stage to trace the bee trajectories. The last section covers the design and implementation of the automatic waggle dance decoder sub-system.

Chapter four covers the experimental validation of the *BeesBook* system, providing full details of the experimental protocols implemented during the three recording seasons (2014, 2015,s and 2016), and presenting performance evaluation for the different modules that comprise the system.

In chapter five, two cases of study are presented to exemplify the usability of the system's generated data. The first one is an analysis of the spatial distribution of foragers on the honeycomb surface. The second case is a study of foraging ecology through the automatic mapping of detected dances using the waggle dance decoder sub-system.

Finally, general conclusions and discussion of the obtained results are presented in chapter six. Here are also described future experiments and tools for further investigation on honeybee colonies.

Chapter 2

State of the Art of Animal Tracking Systems

This chapter presents the state of the art of automated image-based tracking systems in the field of ethology, with emphasis on solutions applied to social insects. By way of introduction, it provides a brief outlook of ethology, describing how the study of animal behavior has changed through the years, moving from a purely qualitative approach to a predominantly quantitative one. It also discusses the impact that this change of approach had on the field, including the development of new methodologies for the quantitative description of animal activity. Then, it analyses the strengths and limitations of some of the developments in automated image-based tracking that have been applied to ethology. Finally, it analyses some of the methods proposed for the automatic detection and decoding of honey bee waggle dances.

2.1 The Path to Computational Ethology

Ethology is a discipline devoted to the objective study of animal behavior, typically of intact freely moving animals in their natural environment [3]. The origin of ethology as a formal branch of biology traces back to the 1930s and the notable work of Konrad Lorenz, Karl von Frisch and Niko Tinbergen, who in 1973 shared the Nobel Prize in physiology for their discoveries on the individual and social behavior of animals. During its early years, ethology was primarily a descriptive science and had as main purpose the observation and description of large collections of animal behavior. Eventually, ethology turned into a more analytical science, and the search for causal explanations to well-described behaviors rapidly became more and more important; consequently, the ratio between experimental analysis and description increased rapidly [111].

In ethology, the main purpose of experimental analyses is to provide a quantitative description of the animal activity. These descriptions can be very simple, and be limited to the account of incidences for specific behaviors, or they can be more complex and include a collection of detailed metrics for each behavioral episode, for instance, its duration, location or even the identity of the individuals taking part of the event.

Conducting experimental analyses and documenting their results have proven to be challenging tasks. Originally, these tasks were performed exclusively in real-time by trained biologists, usually with no more tools than paper and pencil [2]. This manual approach to documenting experimental analyses has many downsides, but they can be summarized in the following three: **First**, depth, and breadth of observation are constrained by human perception. **Second**, scoring of behaviors by human observers is subjective and difficult to standardize. **Third**, no raw record of events against which to compare collected information is preserved.

Looking to overcome these drawbacks, biologists resorted to several technologies, among which video recording became rapidly popular. Video recording was introduced as an aid for experimental analyses in ethology with the aim to allow more extensive studies and provide a permanent register of experimental raw data. Although the use of video recording represented an important improvement when compared to direct observations, the scoring technique remained in principle a manual process. Consequently, the overall process remained time-costly, and the extracted data had still the drawback of being tied up to the personal perception of a human observer.

When digital video recording became the norm, computer-assisted video analysis tools were developed, allowing observers to analyze the videos frame by frame, and to score well-defined behaviors with a simple click of the mouse [77]. From these measurements, it is now possible to obtain multiple metrics and statistics, extending the observation's dimensionality to scales beyond reach when the data was collected exclusively through manual scoring techniques. By speeding up the scoring phase, computer-assisted video analysis tools sped up the overall process. Nevertheless, since these new techniques still rely on human observers, breadth of observation and subjectivity of recorded values remain issues to address.

One common practice to speed up longitudinal studies is to divide the period to be analyzed into multiple sub-periods, this way, several chunks of information can be worked in parallel by more than one observer. Nevertheless, having many observers poses the challenge of conciliating their naturally biased scores. It becomes clear then that subjectivity is an issue in data obtained through manual techniques, whether one or multiple observers participate in the scoring process, and that the only reliable way to entirely avoid this problem is to remove the human factor from the scoring process. In principle, automating the data extraction from digital video recordings (Fig. 2.1) should allow broader and deeper observations with objective and unambiguous records.

The field of computational ethology [3] has taken significant steps towards the full automation of measurement and analysis of animal behavior. To this end, advances in technology, mathematics, and engineering have been brought together in this emerging field. If successfully automated both, measurement and analysis of behavior, solutions developed under the approach of computational ethology should overcome the

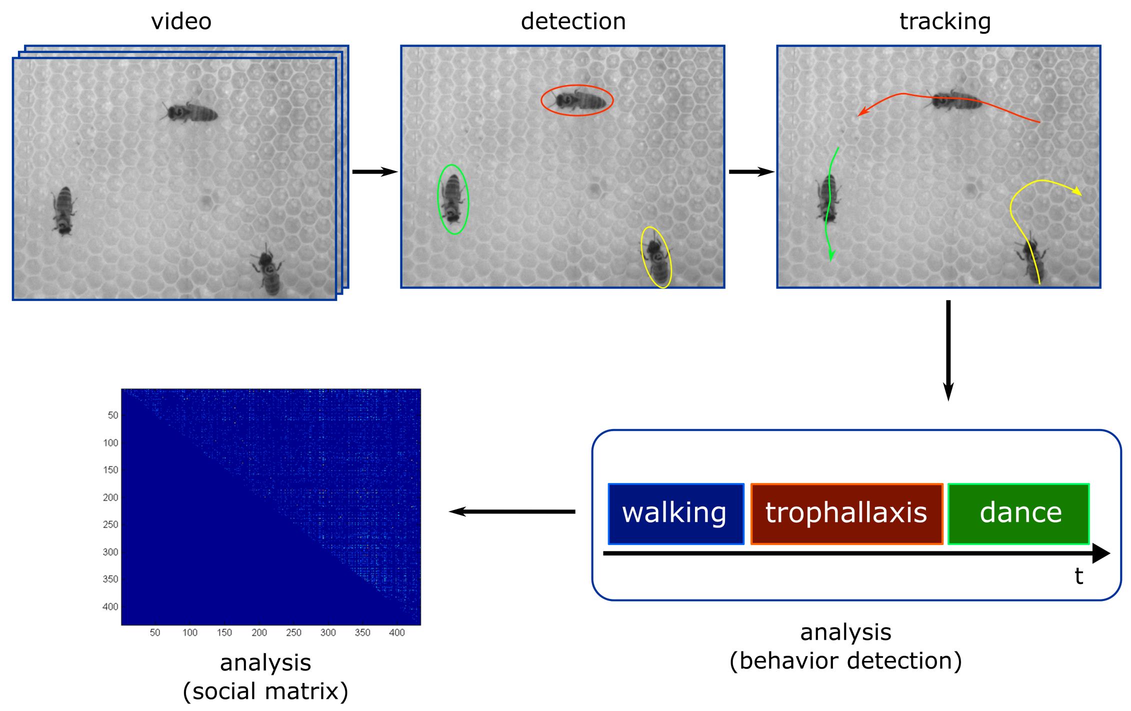


Figure 2.1: **General steps involved in the automated image-based analysis of behavior.** The first step in the process is to detect the individuals in the sequence of images that comprise the video. These detections are then linked to create individual trajectories for each animal in the video. The trajectories are analyzed to detect individual behaviors and interactions with other animals in the field of view.

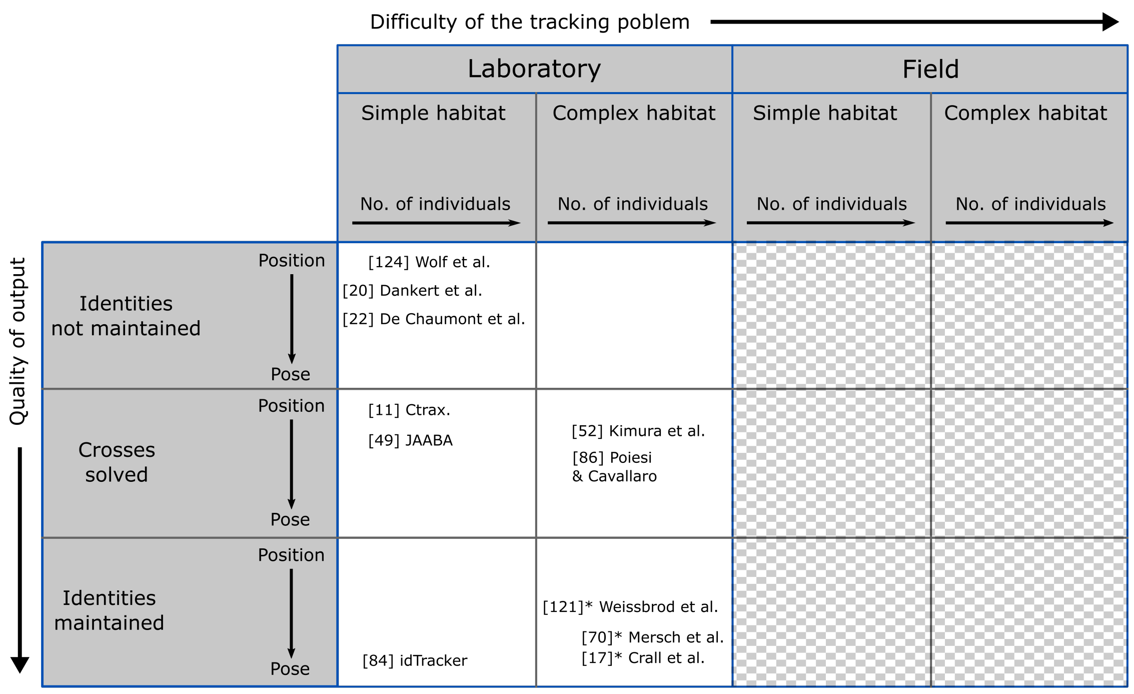
three main drawbacks of manually performed experimental analyses.

In recent years, multiple studies in the field of computational ethology have taken advantage of new advances in machine learning and computer vision to develop video tracking systems capable of simultaneously track many individuals, or to automatically measure multiple elements of behavioral patterns. Some of these studies are presented in the following sections; their strengths and limitations are discussed to conclude to which extent a solution developed under the approach of computational ethology could be used for the case of the honey bee colony that concerns this thesis.

2.2 Tracking Unmarked Animals

In [3], Anderson et al. describe ethology as a discipline devoted to the objective study of behavior, typically of intact freely moving animals in their natural environment. Based on this description of ethology, it is possible to come with a characterization for an ideal tracking solution in the frame of behavioral studies. An ideal solution should be able to extract the detailed location and pose of multiple freely moving animals in their natural environment; the solution should not require the animals to be handled in any way for a successful data acquisition.

In principle, although not a trivial task, it should be possible to develop a computer vision-based solution that complies with the characteristics expected from an ideal tracking solution. Nevertheless, not all tracking problems demand an ideal solution, and for those cases, developing such a system would represent a misuse of resources. According to Dell et al. [25], in automated image-based tracking, there is always a trade-off between the complexity of the tracking problem and the quality of tracking output. Hence, every tracking solution can be classified based on the complexity of the problems that it can deal with, and the quality of the data that it collects. In Fig. 2.2, this trade-off is represented over a plane. Following the analysis made in [25], the difficulty of the tracking problem is represented on the horizontal axis, ranging from small groups of individuals of the same species interacting in a simple habitat (left panel), to many individuals of multiple species interacting in complex habitats (right panel). The vertical axis classifies tracking systems according to the kind of output provided, specifically if the identity of the individuals is maintained through the analysis, and whether only the position or also the detailed pose are tracked.


Difficulty of the tracking poblem
Quality of output

	Laboratory		Field	
	Simple habitat No. of individuals →	Complex habitat No. of individuals →	Simple habitat No. of individuals →	Complex habitat No. of individuals →
Identities not maintained Position → Pose	[124] Wolf et al. [20] Dankert et al. [22] De Chaumont et al.		▨	▨
Crosses solved Position → Pose	[11] Ctrax. [49] JAABA	[52] Kimura et al. [86] Poiesi & Cavallaro	▨	▨
Identities maintained Position → Pose	[84] idTracker	[121]* Weissbrod et al. [70]* Mersch et al. [17]* Crall et al.	▨	▨

Figure 2.2: **Classification of automated image-based tracking solutions used for experimental analysis in ethology.** The horizontal axis separates the solutions based on the difficulty of the tracking problem (complexity of the habitat and number of individuals) and the vertical axis does it based on the quality of output provided (tracking of individual identities and detailed detection). *Require markers. +Tested for longitudinal studies.

To the parameters used in [25] to determine the difficulty of the tracking problems, two more parameters are here considered: The observation's span and whether the animals need to be marked to simplify their tracking and identification. Depending on the nature of the study being conducted, observations can last from only a few minutes to several days; long lasting observations represent a bigger challenge than short ones since the number of crosses between animals is significantly bigger in the former, also making more complicated to maintain identities through the whole observation. Since the problem presented in this work does not comprise tracking animals in open spaces, here are only discussed tracking systems that deal with animals kept in the laboratory.

Conducting observations in a laboratory offers the possibility to control many of the variables that influence, either directly or indirectly, its output. When using automated image-based tracking techniques, reducing the complexity of the habitat facilitates the job of the tracker significantly. In the simplest scenario, animals are filmed over homogenous surfaces that work as a high contrast background. This way, identifying the individuals can be achieved through simple operations, like binarization or background subtraction. This technique was used by Wolf et al. to develop a system capable of tracking the locomotor activity of up to 20 fruit flies at a time [124]. This system requires the flies to be placed into a translucent exposure chamber, they are then recorded from above, and identified by the software as dark objects on a light background; for each object, a path is traced between frames, as well as its speed and position. The system does not maintain identity nor solves crosses, placing its quality of output at a very basic level.

The information that can be obtained through segmentation based methods is not limited to the position of the individuals. In [20], Dankert et al. describe a system that monitors interacting pairs of flies, and at each frame computes a set of features describing body size, wing pose, and position and velocity of both flies. These features are later used to detect behaviors linked to aggression and courtship. In this system, just as in [124], the flies are placed inside a translucent exposure chamber and recorded from above. Recorded images are segmented in three classes (background, other parts, and body) using a thresholding method based on Gaussian mixture model (see Fig. 2.3). Once the image has been segmented, the method proceeds to compute a simplified model for each fly from where the set of features is computed.

A similar approach is used by de Chaumont et al. in [21], where geometrical primitives are used to model and track multiple objects in video sequences. This method requires defining a physics model by a set of primitives, and a set of joints linking the bodies together (see Fig. 2.4 **A, B**), for each of the objects to be tracked. For each sequence frame, two attraction maps are computed to fit each model to its correspondent object; a binary mask to attract the center of the model, and an edge map to adjust the peripheral bodies towards the object boundary. The physics engine solver combines these forces with the constraints of the physics model to adjust the model accordingly. Using this approach, de Chaumont et al. [22] developed a mouse-tracking method that works with pairs of interacting mice, providing position, orientation, dis-

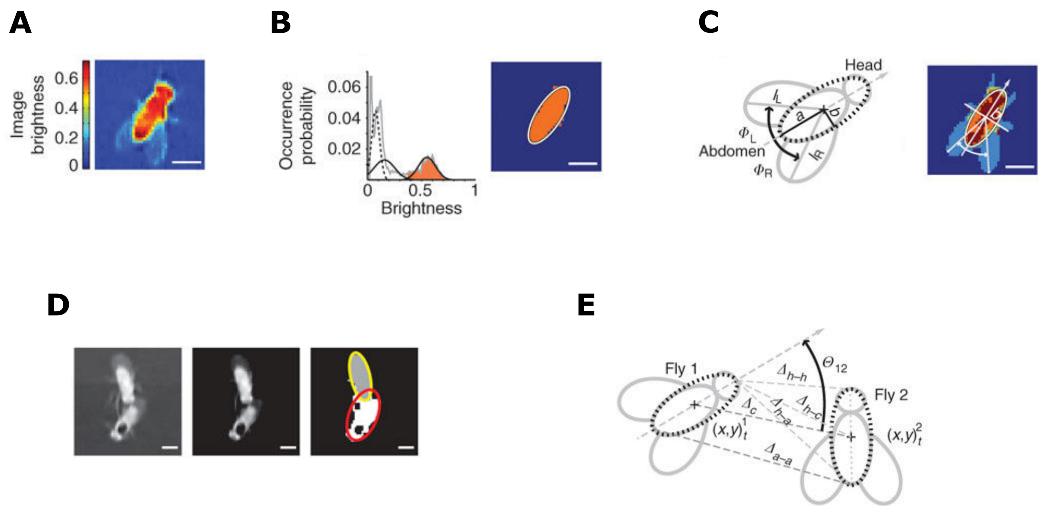


Figure 2.3: **Detection and modeling of fruit flies [20].** (**A**) The intensity map of the original image (**B**) is segmented in three classes (background, other parts, and body) using a thresholding method based on Gaussian mixture. (**C**) The pixels corresponding to the "body" class are fit with an ellipse. The position of the ellipse along with the pixels of the "other parts" class are used to compute the angle of the wings and orientation of the fly. (**D**) Touching flies can can be separated once the body ellipses have been defined. (**E**) Extra parameters are used to detect the interactions between individuals. *(Adapted by permission from Macmillan Publishers Ltd: Nature Methods [20], © 2009).*

tance and speed of each of the animals (Fig. 2.4).

Also using videos of animals in a high contrast planar arena to ease their detection and tracking through computer vision techniques, Branson et al. developed the software solution *Ctrax*. This software is capable of tracking many individuals without swapping identities [11]. Besides background subtraction, the software relies on the grouping of foreground pixels, if more than one group of pixels overlap, these are split by the tracker (see Fig. 2.5 **A, B**). The software computes a trajectory for each subject to maintain the identities throughout the video sequence. Trajectories are generated by connecting successive detections, each subject detected in frame *t* is associated with the trajectory from frame *t-1* that best predicts its position and orientation, this assuming a constant-velocity model (see Fig. 2.5 **C**). The results obtained using *Ctrax* are remarkable, in observations reported in [11], identity errors are estimated to occur every 40 min when working with a population of 50 flies.

Motivated by *Ctrax's* performance, multiple researchers found in it a reliable tool for tracking multiple animals on planar arenas. The machine learning-based system *JAABA*, developed by Kabra et al. [49] for the automatic annotation of animal behavior, relies on *Ctrax* tracking capabilities for the analysis of flies' behavioral patterns. *Ctrax* has also been used as a generic tracker in studies with other animals besides flies. In [88, 34], it was used to track the position of ants, while in [7] it was used to measure the walking speeds of cockroaches.

Despite their popularity, trajectory-led solutions like *Ctrax* face some significant

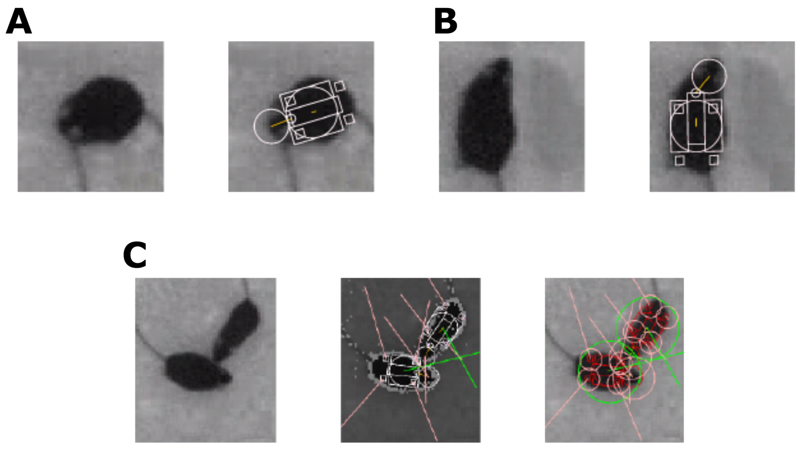


Figure 2.4: **Analysis of social interactions in mice using models based in geometrical primitives** [21, 22]. (**A-B**) Images of two different tracked mice and superposition of a physics model. (**C**) Image of two mice interacting and superposition of the physics model, including force vectors (green and pink lines) and boundaries of their attraction maps (green and pink circles). *(© 2009 IEEE).*

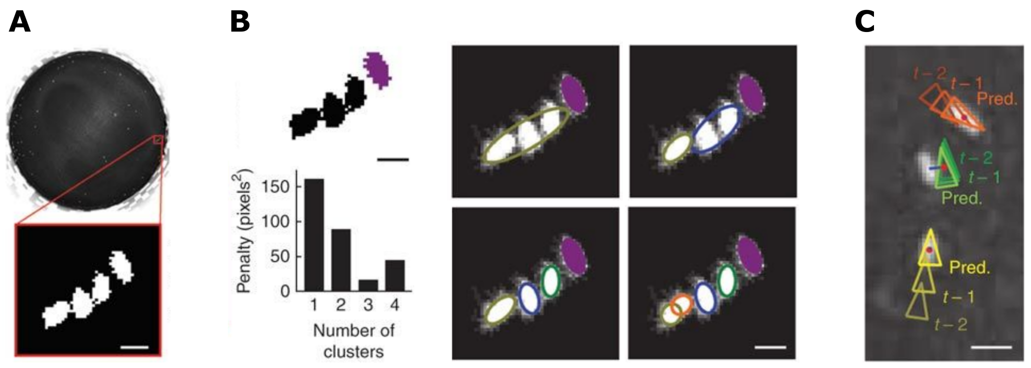


Figure 2.5: **Detection and tracking of fruit flies using ctrax** [11]. (**A**) Frame of the entire arena and detail with binarization of the foreground/brackground pixels. (**B**) The large group of pixels is divided into 1-4 subgroups and a penalty is computed for each option to find the optimal configuration. (**C**) Individual tracking is conducted through the matching of predicted and detected positions. *(Adapted by permission from Macmillan Publishers Ltd: Nature Methods [11], © 2009).*

limitations. Their performance is highly dependent on the number of crosses, which often result in the swapping of identities, an error that propagates through the rest of the sequence unless corrected manually. Identities assigned by the system are valid only within the same video sequence; plus, any animal that temporally disappears from view loses its original identifier.

To overcome these limitations, Pérez Escudero et al. developed *idTracker* [84], a tracking solution with an identity-led approach. *idTracker* extracts a characteristic fingerprint from each animal in the video sequence. The fingerprints are used to identify the individuals in each frame, regardless crossings, occlusions or temporary disappearance from view; and trajectories are traced by joining the centers of the labeled individuals. The identity-led approach of *idTracker* achieves higher levels of accuracy than algorithms based on the tracing of trajectories, and it maintains identities even across different videos.

On the downside, the performance of solutions like *idTracker* is highly dependent on the resolution and quality of the input video, and its computational cost is considerably higher than that of trajectory-led solutions. Its proved breadth of observation is also significantly narrow, it has been tested for groups of up to 20 animals, and minimal deterioration of identification is expected for up to 35 animals [84]. Even more important is that the method used to extract references can be compromised at high densities where crossings are ubiquitous. All these limitations are a clear indicator that *idTracker* is not a suitable solution to track animals in complex environments like a beehive.

Reducing the complexity of the environment eases the tracking process, usually at the cost of the output's significance. For the behavioral analysis of many animals this represents a fair trade-off, given that even observations in the most simple environments, provide reasonably significant data; but for the honey bee, this is hardly the case. Honey bee colonies are numerous, usually with populations in the many thousands [95], and their life in community is tightly linked to the structure of the honeycomb. The honeycomb not only is the surface where the interactions between colony members happen, but its cells are used to contain the larvae and store honey and pollen. For this reason, ethologists working with honey bees use observation hives for their studies. An observation hive essentially is a glass-walled cabinet inside which one or more honeycombs are kept in a one layer array, the limited space between honeycomb and glass panes forces the bees to move over the honeycomb surface without overlapping one another providing the observer with full visual access to activity within the hive. Due to its difficulty, few algorithms have addressed the task of automatically tracking bees in observation hives. Solutions designed to track animals in high contrast planar arenas [124, 20, 11, 84] perform poorly with large populations and nonuniform backgrounds; therefore new approaches have to be explored.

One of the solutions proposed for the observation hive problem is the method developed by Kimura et al. based on vector quantization and temporal contextual information [52]. In the proposed method, a *code book* is first generated by a 2×2 quantization process, and one of the found *code vectors* is used to obtain a honey bee-code image out of the original hive image (Fig. 2.6 **A, B**). From the honey bee-code image, two different kinds of regions are identified, single honey bee regions (SHR) and plural honey bee regions (PHR) (Fig. 2.6 **C**). Identification of successive SHRs is made by overlapping; it is assumed that consecutive SHRs with the largest overlapping correspond to the same bee. PHRs are divided into SHRs using preceding and posterior frames, it is always considered that a PHR is the result of the temporal merging of more than one SHR (Fig. 2.6 **D, E**).

For the solution proposed by Kimura et al. to perform satisfactorily, it is required

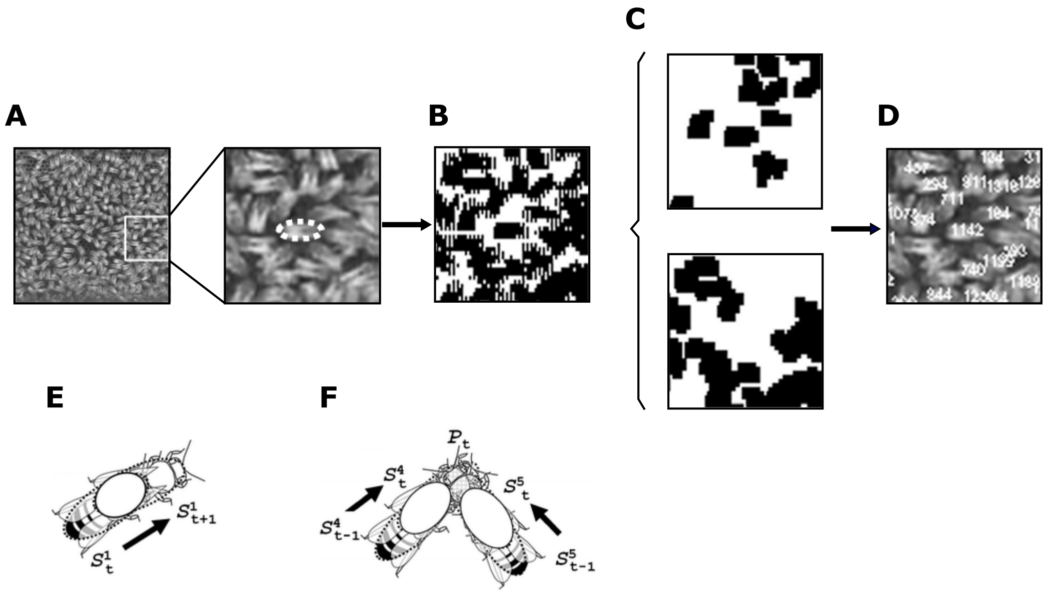


Figure 2.6: **Detection of bees in observation hive using vector quantization** [52]. (**A**) Observation hive image and detail with the position of one of the bees highlighted. (**B**) The original frame is processed using vector quantization to obtain a honey bee-code image. (**C**) The honey bee-code image is divided in SHR (top) and PHR (bottom). To track individual identities (**D**), it is assumed that (**E**) overlapping of consecutive SHR correspond to the same individual and (**F**) PHR are the result of the merging of more than one SHR. *(© INRA, DIB-AGIB and Springer Science+Business Media B.V., 2011. With permission of Springer).*

the frame rate of the input video to be high enough to allow overlapping of consecutive detections of the same bee. In [52], the analyzed videos had a frame rate of 29.97 fps, and the system reportedly was able to identify more than 72% of the 700 observable bees in the movie and trace trajectories for more than 50% of them. In general, the system allows locating the bees in the hive and compute their velocities, but it is not as effective tracking their movements and maintaining their identities. It also faces severe constraints regarding computing costs, in [52] the system was only able to process movies of up to 3 minutes duration due to memory limitations.

A different approach suitable for tracking bees in observation hive videos has been put forward by Poiesi and Cavallaro in [86]. They propose a composite solution integrated by a gradient-climbing technique, which works with a Markov chain Monte Carlo to fit a shape model iteratively onto the target locations; and an isocontour slicing approach for intensity maps to localize targets in videos with a high density of homogeneous targets (Fig. 2.7). The overall method profits from the strengths of both approaches by fusing their results. The trajectories are generated by recursively associating detections with a hierarchical graph-based tracker on temporal windows.

For experiments conducted with observation hive videos, the red channel (RGB color space), equalized and with a Gaussian filter applied to it was used as target-intensity map. As a shape model was used an ellipse with dimensions set to match the size with which bees appear in the video. Analyzing short video sequences of 500 frames of size $640 \times 350 px$ and recorded at 29.97 fps, the method performed with precision and recall values close to .90 for an average population of 30 bees per frame [86]. The tracking method also carried out with favorable values, switching IDs only an average of .22 per frame and 3.55 per track [86].

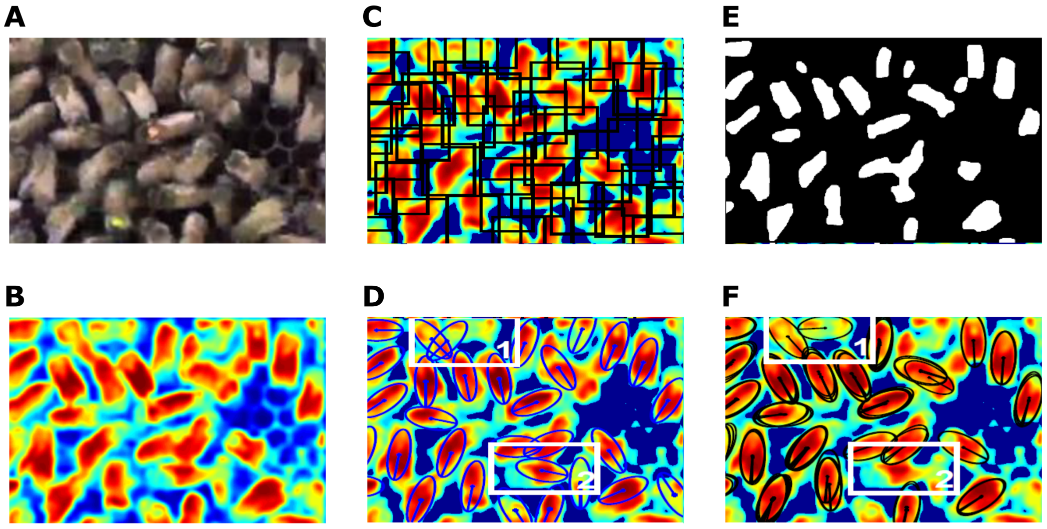


Figure 2.7: **Detection of bees in observation hive using the gradient-climbing based detector and hierarchical-isocontour based morphology** [86]. (**A**) Original frame and (**B**) target-intensity map. (**C**) Initialization of detections for the gradient-climbing technique and (**D**) resulting detections after the adjustment with a Markov chain Monte Carlo. (**E**) Binary map obtained after morphological operations for the isocontour slicing approach and (**F**) detected individuals. Regions 1 and 2 in (**D,F**) exemplify challenging cases that benefit from this composite approach. *(© 2015 IEEE).*

The overall performance of both methods [52, 86] is acceptable, detecting and tracking a significant amount of the observable bees. Nevertheless, there is still plenty of room for improvement, especially if these systems are intended to be used for behavioral studies that extend over long periods of time. Long-term analyses can significantly profit from methods that keep identities throughout the entire observation period, even if the subjects disappear temporally from view. From the solutions discussed in this section, only *idTracker* [84] is capable of keeping the identities of individuals that go out of view, a solution that on the other hand, is aimed for videos with small populations and high contrast background.

In general, conducting longitudinal studies of animal behavior, maintaining track of the identities of the subjects, is a complex task that for the moment seems to be out of reach if the animals are to be kept unmarked. It might be that marking the animals is not only a way to ease their detection but in some cases also the only way to keep track of their identities; e.g., honey bees in observation hives. Honey bees are not only hard to tell one from another, but many of them often get out of the supervised area to forage nectar and pollen or scout for new nest locations. This behavior is fundamental to the colony's life, and repressing it would rest significance to data collected during its study.

The following section discusses some of the proposed approaches for tracking marked animals, and which extra considerations can be taken into account to achieve a fair trade-off between their advantages and disadvantages.

2.3 Tracking Marked Animals

Most of existing methods for automatically track position and identity of animals assume that these are visible at all time [11] or that they have individual features which differentiate them from one another [84]. When these conditions are not met, biologists resort to different marking techniques to keep track of the animals identities. Some approaches take advantage of multiple techniques, like Weissbrod et al. [121] who combine video and radio frequency identified data to obtain detailed behavioral profiles at the individual and group level. However, most of the approaches use visible markers [17, 70, 77, 79], this way animal tracking can be reduced to a computer vision problem.

Handling and tagging animals can affect their stress levels and behavior [26, 103]. Therefore, much effort is put into preserving the environment as close to nature as possible. For instance, long-term observations are partly conducted under infrared lighting to simulate dark cycles. Furthermore, in some cases like honey bees in observation hives, it is convenient to carry out the whole observation period under infrared lighting to recreate the conditions in the natural environment. Infrared lighting renders colored markers useless, since they are not discriminable anymore, and narrows the options to black and white markers [17, 70].

Automatic image-based recognition of binary markers is widely known for its commercial applications like bar-codes and QR-codes. Nevertheless, binary markers have a broad variety of applications. Their high contrast structures make them optimal for use as fiducial markers; this along with their readability using computer-vision methods, have fostered their use in camera-calibration (*CALTag*, [4]) and augmented reality applications (*ARTag*, [31]). The same properties that make binary markers convenient for augmented reality make them a viable option to mark animals for automated high-throughput behavioral studies.

Proof of the suitability of binary markers for behavioral studies is the automated video tracking system used by Mersch et al. to track individuals in ant colonies [70]. The solution is based on the ARTag marker developed by Fiala [31], and it was used to track all members from 6 colonies (122 to 192 ants per colony). The marker itself consists of a 10×10 binary matrix array. A black or white outline of 2 units thickness encloses the 36 inner cells which encode the marker's ID (Fig. 2.8). Only 10 of the 36 bits within the delimiting outline are used to encode the marker's ID; the other 26 provide redundancy and uniqueness over the four possible rotations [31].

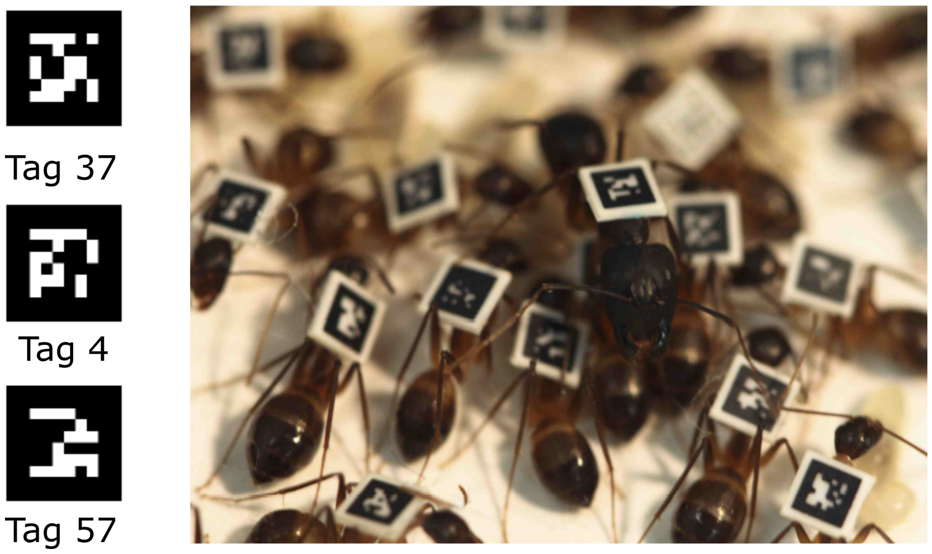


Figure 2.8: **Example of ARTag markers and detail of marked ants in the arena** [70]. The black outline is used by the decoder software to calculate a homography to define a grid over the inner elements. With 10 bits to encode the marker's ID it is possible to generate up to 1024 unique codes. The orientation of the tags is also used to obtain the orientation of the animal (equivalent to identify its head). *(From [70]. Reprinted with permission from AAAS.)*.

For their behavioral experiments with ants, Mersch et al. [70] used a $260 \times 160mm$ planar arena, and a video camera with a resolution of $4560 \times 3048px$ recording at a frequency of 2 frames per second. Under this setup, markers with a $1.6mm$ side length appear on recorded images with a resolution of $17-18px/mm$. With this configuration, Mersch et al. reported a tag detection probability of $88 \pm 17\%$, and a false positive and very low inter-marker confusion probability of 0.8×10^{-7} (supplementary material for [70]).

A similar solution to the one used in [70] is put forward by Crall et al. with a low-cost alternative denominated *BEEtag* (**BE**havioral **E**cology tag) [17]. *BEEtag* is distributed as a Matlab package, and the idea behind it is to provide all the benefits from other image-based tracking systems, but with the advantages of an open-source solution. The *BEEtag* markers are is similar to others used for visual tracking like *ARTag* [31] and *CALTag* [4]. The 25-bit matrix is divided into 3 columns (15 bits) used for the identity code, and 2 columns (10 bits) for error check. The valid 15-bit codes were limited to those that ensure matching with their error check only in one of the four possible orientations and at least a Hamming distance of 3, resulting in 7,515 valid tags (Fig. 2.9).

To test the system's performance, Crall et al. conducted an experiment with a bumblebee colony comprised of 100 workers [17]. All colony members were marked with

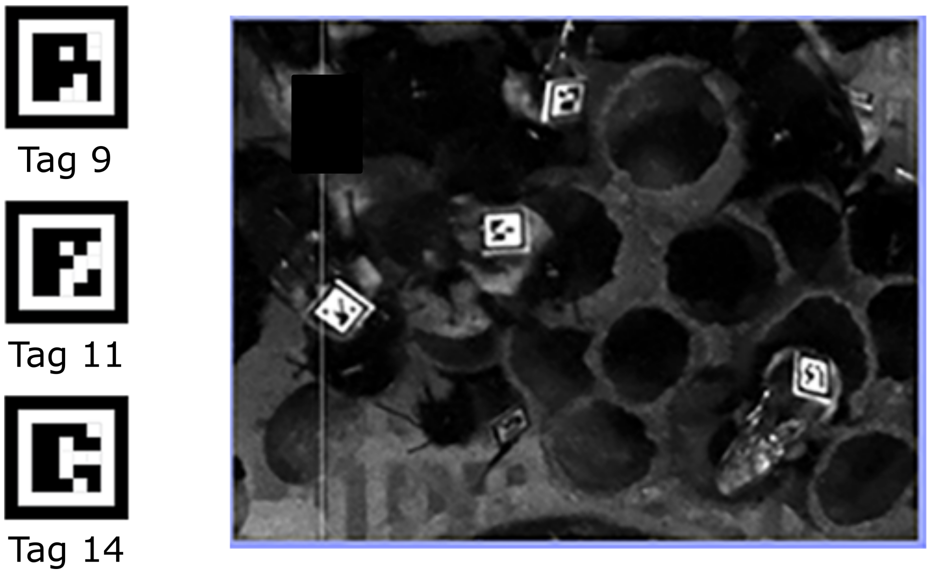


Figure 2.9: **Example of BEEtag markers and detail of marked bumblebees in their nest** [17]. The design consists of a 25 bits code matrix (5 × 5) of black and white cells surrounded by either a black or white one cell thick outline. The first 3 columns are used for the identity code, and the 2 remaining columns for error check. *(© 2015 Crall et al.)*.

unique tags of $2.1 \times 2.1mm$ size, and they could move freely on the nest. The nest was illuminated with red light, and a camera with a resolution of $6016 \times 4000px$ was used to record the entire nest area of around $21.5 \times 15.0cm$ (a resolution of $\sim 16px/mm$). The system detected an average of 90% of the recoverable tags in each frame. They also reported that 99.97% of the detected tags were decoded into valid IDs [17]. It is important to mention that this percentage does not necessarily account for accuracy since they did not consider inter-marker confusion.

In the end, the number of advantages provided by automated tracking systems based on marked individuals surpasses their limitations. While experiments with unmarked animals are usually limited to small groups interacting in simple habitats, experiments with marked animals can comprehend hundreds of individuals interacting in naturalistic environments. The use of tags also provides more certainty to the tracked identities regardless of the number of crossings and even allows individuals to go out of view temporarily. Finally, even though the marking procedure represents a potential factor of stress for the animals [26, 103], this can be significantly reduced if careful protocols are observed [93].

2.4 Detecting and Decoding Communication Dances

The waggle dance gained interest among ethologists after its significance as an abstract form of communication became clear thanks to the studies conducted by von

Frisch in 1946 [115]. The most common way in which ethologists gain access to the activity inside the beehive is by placing the colony to be studied in an observation hive. Keeping bee colonies in observation hives for their study is straightforward, and because its design forces the bees to move over the honeycomb surface without overlapping, it provides full access to the observable characteristics of the dance behavior.

Communication dances essentially encode in the orientation and duration of their waggle runs an ordered pair of polar coordinates for a field location. Hence, waggle dances can be mapped back to the field using the solar azimuth as a reference and a factor that correlates waggle run duration and distance in the field. To find this factor, first a calibration curve has to be built, usually by training bees to a feeding station at a known distance from the hive and timing their dances in the observation hive [114]. This way, without tracking the foragers' flight, one can deduce the distribution of foragers in the environment by establishing the distribution of dance-communicated locations. Couvillon et al. [16] used this method to investigate how the decline in flower-rich areas affects honey bee foraging, while Balfour and Ratnieks [5] used it to find new opportunities for maximizing pollination of managed honey bee colonies.

The information available through the waggle dance has not only been used to obtain the distribution of foragers in the field. In [42], Haldane and Spurway analyzed the spatial information contained in waggle runs following an information theory approach. Working with a data set produced by von Frisch [115] they concluded that the dance conveys 2.0 bits of information regarding the direction component. Later in [94], Schürch and Ratnieks reproduced this analysis, this time with a new and more complete data set [16], and computing also the information for the distance component. Their findings, 2.9 bits for direction and 4.5 bits for distance did not differ much from those of Haldane and Spurway [42]. Other researchers have studied the accuracy and precision with which bees represent spatial coordinates through waggle runs [23, 15] by manually decoding and analyzing thousands of waggle runs from digital video recordings. Landgraf and co-workers tracked honey bee dances in video recordings to build a motion model for a dancing honey bee robot [62, 60].

All the studies mentioned in the previous paragraph required the decoding of hundred of waggle dances. Different techniques have been used over time to decode waggle dances. During the first decades that followed von Frisch's discovery, the dances were analyzed in real time, directly from the observation hive with the help of protractors and stopwatches [116]. In subsequent years, when video recording was introduced as an aid for experiments, the decoding of dances started to be performed during a later process [95]. Nowadays, the use of digital video has become ubiquitous to extract the encoded information during a later process. Digital video allows researchers to analyze dances frame by frame and extract their characteristics either manually using the screen as a virtual observation hive [15], or assisted by computer software [23]. Although digital video recordings allow measurements with higher accuracy and precision, decoding communication dances continues to be a manual and time-consuming task.

Multiple automatic and semi-automatic solutions have been proposed to simplify and accelerate the dance decoding process. These solutions can be classified into two complementary groups. The first group of solutions focuses on tracing the bees' trajectories via a variety of tracking algorithms. For instance, in [52] vector quantization was used to track multiple individual bees behavioral patterns, including the waggle dance. Landgraf and Rojas [61] proposed a method capable of tracking unmarked dancing bees by clustering the motion of optical flow features using the Hough transform (Fig. 2.10). In [51] a Rao-Blackwellized particle filter-based tracker was used to track an unmarked bee. The second group of solutions analyzes the trajectories mapped by the tracking algorithms and extracts specific features such as orientation and duration of the waggle runs. The analysis is done using either a generic classifier trained on bee dances (see [29, 49, 78]) or methods based on manually defined features such as the specific spectral composition of the trajectory in a short window [62].

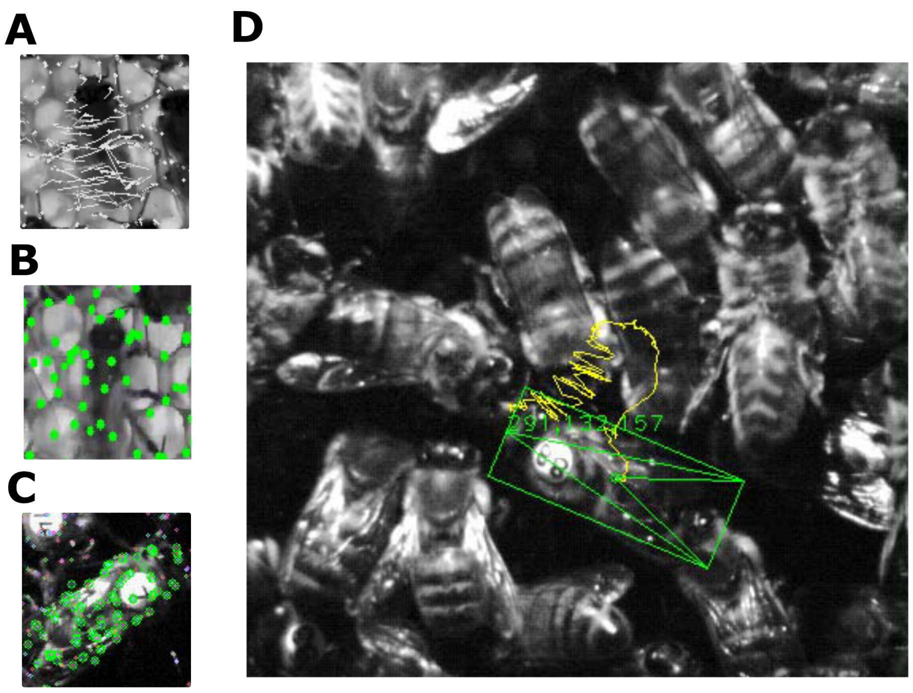


Figure 2.10: **Automatic tracking of dancing bee**. The method proposed by Landgraf and Rojas based on optical flow [61] only computes flow vectors for (**A**) promising points detected using a pyramidal implementation of the Lucas-Kanade tracker. (**B**) Sparce flow field for a dancing bee. (**C**) Points that exhibit the same direction are clustered together to obtain position and orientation of the bee. (**D**) Dancing bee and traceed trajectory. *(From [61]).*

Although methods from the two groups have been combined in a solution with the potential of an automatic detector and decoder of dances [30], its implementation has been limited to the automatic labeling of behaviors. What is expected from an automatic detector and decoder of communication dances, is the capability to measure the three basic properties of each dance, namely average waggle run duration, average orientation, and average return run duration. Once these measures are known, it is possible to read the information encoded in the dance. None of the solutions available in the literature has these characteristics.

2.5 Chapter Overview

In the introductory chapter of this thesis, it is presented as the main motivation for this work the lack of technologies that allow behavioral biologists to conduct longitudinal studies and obtain reliable data sets out of them. The work focuses on the particular case of the honey bee and its life inside the hive; it is expected that the data sets obtained from comprehensive studies would allow a better understanding of the colony's social structure and the mechanisms that shape its collective behavior.

The solution proposed to the lack of technologies for longitudinal studies is an image-based system that keeps automatic track of position and identity of the colony members and detects and decodes the communication dances. To maximize the relevance of the data sets obtained by the system, this must be able to work with freely moving individuals in naturalistic environments.

None of the technologies discussed in sections 2.2 - 2.4 can be used to fulfill the stated goals. The methods to automatically track marked animals are conditioned to the use of their designs, which does not adapt to the bee's curved thorax or the active environment within the hive. Regarding the technologies to detect and decode communication dances, none of them can perform both tasks, and those undertaking decoding do not compute the three fundamental properties of the dance: average waggle run duration, average orientation, and average return run duration. Nevertheless, these technologies do embody the advantages of the computational ethology approach, and work as inspiration for the development of a new solution capable of meeting the goals. In the next chapter is described this solution which uses its own binary markers design to keep track of the bees inside the observation hive and its capable of automatically detecting, decoding and mapping communication dances.

Chapter 3

Design and Implementation of the BeesBook System

This chapter provides a thorough description of the components that conform the *BeesBook* system. The overall system can be seen as integrated by two functionally independent units that operate simultaneously to harvest data from the colony's activity. The first sub-system works with high-resolution images captured by a four cameras array to extract position, orientation, and identity of each colony member. The second sub-system, from here on referred as "waggle dance decoder", detects and decodes communication dances via the analysis of video sequences generated by two web-cams running at high-frequency. As shown in Fig. 3.1 each operational unit is divided into three modules. At the end of the analysis phase, each unit writes its results to formatted files, from where it can be taken for further analysis.

The first section of this chapter covers the preliminaries to the data analysis components of *BeesBook*, these include the binary markers used to identify each colony member and the recording setup. In sections two to four, are presented the elements that process the high-resolution images to extract position and identity of marked colony members. The last section of the chapter is devoted to the waggle dance decoder subsystem.

The *BeesBook* system's status at the moment of writing this dissertation is the result of a continuous teamwork with members of the *Biorobotics Lab* of the Dahlem Center of Machine Learning and Robotics at the Institute of Computer Science. Some of the final system's features here reported were implemented by members of the group as part of their bachelor and master's theses; proper acknowledgment has been given to them throughout the text.

Some of the information provided in section 3.3.1 is also included in the research article *"Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees"* [118], published by *Frontiers in Ecology and Evolution.* Information in section 3.4 has been partly used for the research article *"Novel technological and methodological tools for the understanding of collective behaviors"*

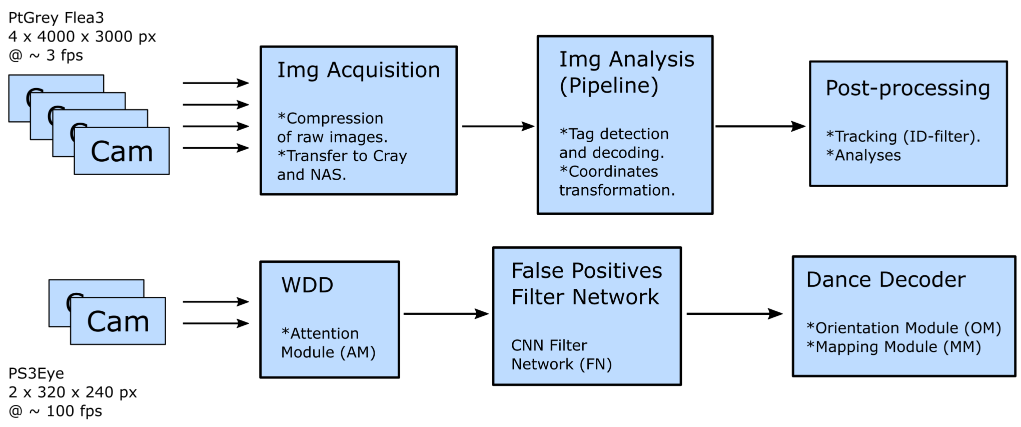


Figure 3.1: **BeesBook comprises of two sub-systems working in parallel.** The first sub-system works with high-resolution images obtained from a four cameras array. Images obtained from the high-resolution cameras are encoded and transferred for permanent storage to a remote server. These images are later processed for detection and decoding of the binary markers attached to the bees. Finally, detections are connected to build paths and correct misread IDs. The second sub-system works with two web-cams running at high-frequency. Images read from the web-cams are processed in real time to detect potential waggle dances. Sub-images of the region containing potential dances are stored locally and false positives filtered out using DNN. Finally detected dances are decoded through a process based on FFT.

[10], submitted to *Frontiers in Robotics and AI*. Some of the information in section 3.5 has been used in the preparation of the article *"Automatic detection and decoding of honey bee waggle dances"* [119], accepted for publication in *PLOS One*.

3.1 Preliminary Work

For a computer vision-based system, the image to analyze constitutes its raw material, and its quality has a substantial impact on the system's overall performance. For the specific goals pursued by this work, a number of challenges had to be addressed to generate high-quality images that allow the image processing stage to extract the targeted information with the highest accuracy possible. These problems and their solutions are described next.

3.1.1 Individual Binary Tags

Tracking individuals for behavioral studies in highly populated environments is a difficult task that can be significantly eased with the implementation of a reliable labeling method [41]. In the particular case of studies with honey bee colonies, the number of individuals rounds the thousands, supervised spaces are usually crowded, and most of the colony members are virtually identical, to tackle these challenges biologists have developed a series of inventive labeling methods to keep track of each individual bee.

Each of the labeling methods that biologists have at their disposal better serves a particular porpoise. When the number of bees to track is fairly reduced, the most common labeling technique is to color each bee with enamel paint, using either a toothpick or a fine-haired brush. Applying more than one color mark to each animal allows for multiple unique combinations. In [116], von Frisch describes a marking system based on five colors and four different body spots on the bee's thorax and abdomen; the system allowed von Frisch to mark up to 599 bees, each one with a unique colored code. Another highly popular method to individually label bees is to affix small numbered plastic disks to the animal's thorax. Numbered bee tags and the tools to apply them to the bees are commercially available. The tags are sequentially numbered from 00 to 99 and available in 5 different colors, making possible to mark up to 500 bees uniquely. In [95], Seeley mixed both methods to tag a whole colony with a total population of 4000 bees; each bee was marked using one of the 500 numbered tag plus a paint dot on the abdomen in one of eight different colors (Fig. 3.2).

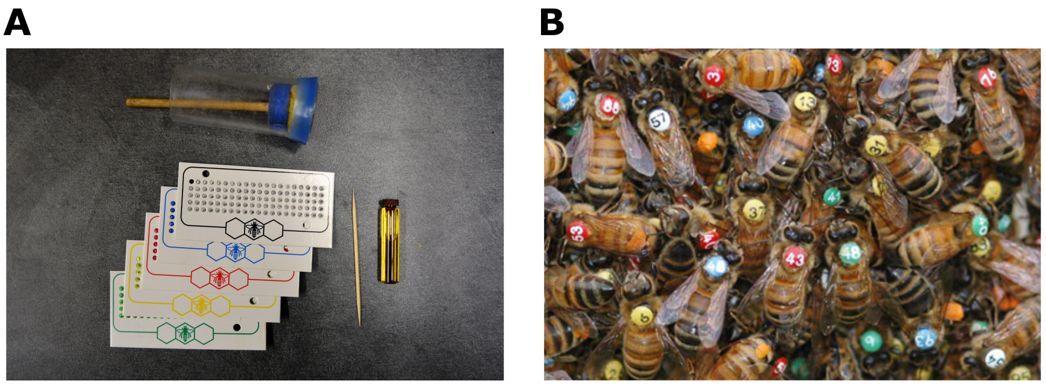


Figure 3.2: **Numbered bee markers.** (**A**) Tools to mark queen bees, including marking cage and numbered tags in five different colors are commercially available. (**B**) Using all 500 different numbered tags and adding a paint mark in one of eight different colors to the bees' abdomen, Seeley [95] has successfully labeled up to 4000 bees *(Image source: Seeley, 1995).*

The methods described have been for a long time the standard in the field and have repeatedly proved their suitability for studies with a scope on reduced groups of bees during short periods of time; unfortunately, they result inadequate for the set goal of automatically tracking all colony members, over uninterrupted extended periods of time. Leaving aside the problem of automatically reading the tags' numbers, since similar tasks have already been successfully addressed [40, 74], the main constraint of numbered tags resides in their color-dependency. To unequivocally mark over 100 bees using numbered tags, it is necessary to count with a source of light that allows for the distinction of different colors. That is, the supervised scene has to be illuminated using white light. In nature, bees tend to build their hives in places like tree cavities where they are protected from predators and weather conditions; as a result, hives are rarely exposed to natural light and introducing a source of white light, especially during night hours would certainly disturb the colony's environment.

A viable solution to the scene illumination problem is to use sources that only emit in wavelengths outside of the bees' visual range. The portion of the electromagnetic field visible to honey bees is slightly shifted towards the short wavelength end, regarding the human visual range; it spans from 300 to 650 nm [37]. Hence, using red or IR light is an acceptable option to white light. Regardless which of these options is implemented, the use of monochromatic sources demands an alternative, not color-dependent labeling method to uniquely mark each colony member.

For the labeling method, it is also necessary to take into consideration the number of individuals to be labeled. A fully functioning queen-right hive can be populated by as few as 1000 worker bees [45]. A solution to the problem is then the implementation of a binary design. For this sort of designs the marker is divided into a number of fields, and each field can adopt one of two different colors; this way each field can be interpreted as a bit in a visual representation of a binary code. A scheme containing 10 bits can produce up to 1024 different markers, in principle enough to tag a whole colony.

Binary markers have already been used in ethology, in [70] all members of six different ant colonies (between 122 and 192 per colony) were marked using barcode-like matrix generated using the ARTag system [31]. The ARTag markers are planar patterns that consist of a square layout with a 36-bit word encoded in it, the system allows for up to 2002 unique designs with small inter-marker confusion rates. Another example of binary markers in ethology is the *BEEtag*, developed by Crall et al. [17]. Similar to the ARTag design, the *BEEtag* markers' design is a square matrix divided into 25 cells, the design allows the codification of up to 7,515 different identifiers.

Although the experimental results reported in [70, 17] are encouraging for the use of ARTag and BEEtag markers in experiments with social insect colonies, this kind of design has some limitations that make it unsuitable for its application on bee colonies studies. The markers are planar an printed on paper [70, 17]. In the particular case of bees, a marker must fit the animal's curved thorax so that it doesn't interfere with the wings' movement; it also must be firmly attached to avoid it being removed by other colony members or the marked animal itself; finally they must be moist resistant since bees are frequently exposed to humid conditions, both in and outside of the hive.

Lacking an adequate solution to the labeling problem, I decided to devise my own marker. The resulting circular, curved tag (Fig. 3.3) was designed to create a marker that bears the heavy duty activity carried by worker bees during their full life-span while making the most of the available space on the bees' thorax. The tag adapts firmly to the thorax and displays two concentric circles. The smaller circle is divided into two semicircular segments, which are used to determine the tags orientation and to align the decoder properly. The ring formed by the overlapping of both circles is divided into 12 segments, each of which holds one bit of information. This design has multiple advantages; it adapts naturally to the circular form of the marker and the bee's thorax. Twelve bits of information allows the codification of up to 4096 different identifiers. In case more animals are to be marked, the ring can be divided into more cells. If the experiment comprises fewer animals, a coding scheme that allows for error detection or correction can be employed. If for instance, a single bit is spare, it could be used as a parity bit; if three or more bits are unused, Hamming coding [43] can be implemented.

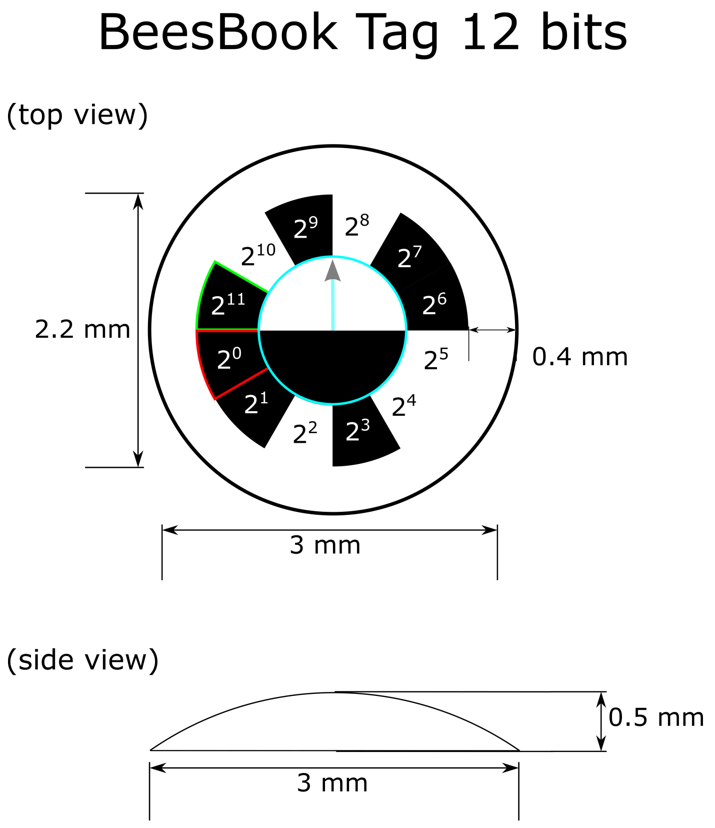


Figure 3.3: **The BeesBook tag design.** The circular matrix design is comprised of 14 regions. The two semicircular regions in the middle are used to encode the tag's orientation. The other 12 regions are used to represent a unique binary code. The design is printed on white material that offers a high contrast with the bee thorax.

Different materials were tested for resistance to humidity, extended durability and long lasting attachment to the bee's thorax. Finally, the design was printed on backlit polyester film by a commercial print shop. Each set of tags is generated using a python script and printed in a $210mm \times 405mm$ piece. The tags are later manually punched out from the polyester sheets with the help of two customized elements attached to the end and table of a stand-up drill (Fig. 3.4). The drill is switched off at all time, and it is only used for stability purposes, to assure that both elements remain in line during the process and the tag can be cut with maximum precision. The cutting procedure also bends the tags for optimal fit to the bees thorax.

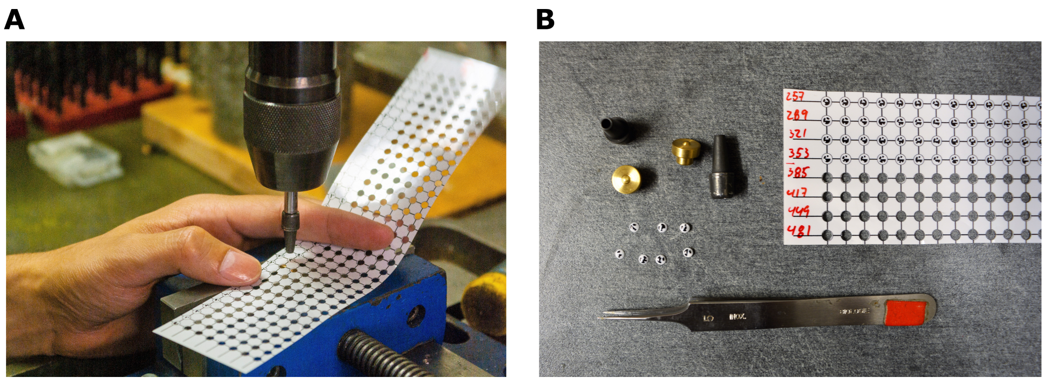


Figure 3.4: **Tags' punching out procedure.** (**A**) The markers are manually punched out from the polyester sheets using a stand up drill. (**B**) The hollow tip attached to the drill's end and the convex base provide the tag with the needed curved profile for an optimal fit to the bee's thorax.

3.1.2 Cameras and Camera Modification

Cameras are one of the neuralgic components of a vision-based tracking system. Their quality of image and transfer rates determine to a great extent the system's reach. For the *BeesBook* system, with the double goal of keeping track of the markers and detecting and decoding waggle dances, the requirements of image resolution and frame and transfer rates are by no means met by a single camera model, especially when trying to keep a low budget.

Detecting and decoding the markers is a task that relies heavily on the image quality, including image resolution, contrast, and sharpness; whereas the speed of the camera plays a secondary role, and any frame rate that allows keeping track of the bee inside the hive suffices. By contrast, detecting and decoding communication dances requires primarily of a camera with high frame rate capabilities, relegating image quality to a secondary role.

The system's setup comprises of a total of 6 cameras. Four PointGrey Flea3 (FL3-U3-88S2C-C) high-resolution cameras (Fig. 3.5), two per side, are employed to observe the surface of the comb. They feature a Sony IMX121 CMOS, 1/2.5", 1.55 μm color sensor with a resolution of 12 megapixels (4000 × 3000 pixels). Each camera is equipped with 12mm lenses (RICOH FL-CC1214A-2M) to capture half of the comb's surface. These cameras are connected to the central recording computer via USB3 and record 3 images per second. Two additional cameras (PS3Eye webcams) are connected to a second personal computer. Each of these cameras observes the full comb side at 320×240 pixels resolution. The PS3Eye cameras are low cost and deliver uncompressed images at 120 fps using a modified third party driver ("CL Eye").

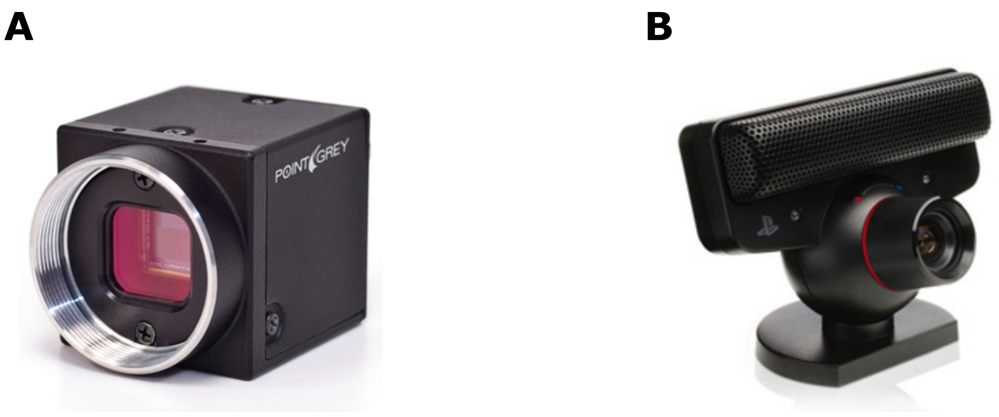


Figure 3.5: **The two model of cameras used in the *BeesBok* system.** (**A**) The Point Grey Flea3 (FL3-U3-88S2C-C) is used to record high-resolution images for the sub-system in charge of tracking position and identity of the colony members. (**B**) The PSEye capable of capturing video with frame rates of 120 hertz at a 320 × 240 pixel resolution is used to feed video to the waggle dance detector sub-system.

As mentioned before, the bees' visual range spans from 300 to 650 nm [37], what makes of red light (620-700 nm) a viable option to illuminate the recording setup without disturbing the colony. Nevertheless, lighting an scene exclusively with red light when working with color cameras has an important drawback. To create color images, digital cameras have a 2×2 mosaic array of color filters placed over the sensor; this way each photoreceptor is exclusively sensitive to one color (red, green or blue). The most popular pattern, known as Bayer filter, is 50% green, 25% blue and 25% red (Fig. 3.6). Consequently, images captured under red lighting only register real red values in 25% of the pixels, the other 75% corresponding to blue and green photoreceptors are interpolated from adjacent red photoreceptors (see Fig. 3.6**B**). Under these circumstances, the resulting images are in general of poor quality, and far from optimal for their use in computer vision tasks.

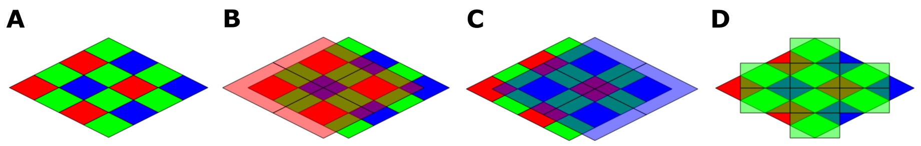


Figure 3.6: **Bayer filter.** (**A**) The Bayer filter mosaic array has twice as many green components as red or blue to mimic the physiology of the human eye. (**B, C, D**) To obtain a complete set of RGB values for each pixel, the two missing values are interpolated from neighboring pixels. It is worth to mention that while this means a total of 8 values for red and blue pixels, these are only 4 in the case of green pixels.

To avoid blurry images resulting from the demosaicing process and an scene exclusively illuminated with red light, the *BeesBook* setup uses infrared (IR) light sources (840 nm). The photoreceptors of digital cameras are sensitive to IR radiation, and color cameras are equipped with an IR block filter to minimize noise introduced by this portion of the electromagnetic spectrum. The IR block filter of the Flea3 was replaced with a cold light mirror with the same refraction index (bk Interferenzoptik bk-HT-45-B). This modification renders the Flea3 an inexpensive alternative to similar IR-sensitive cameras, that can go for as much as five times its price. To optimize the performance of the PS3Eye, not only the IR block filter was removed, but also the original lens was replaced with an 8mm IR corrected Megapixel lens (Lensagon BM8018), in order to fit the new lens to the camera a customized 3D printed lens mount was produced.

3.1.3 Observation Hive, Scaffold and Illumination Setup

Observation hive

A customized one frame observation hive was designed to be used during the experimental seasons (Fig. 3.7). Bees frequently smear small portions of honey, wax, and propolis on the observation hive glass, which impairs the image quality. Hence the glass panes have to be periodically replaced. Replacing the glass of a standard observation hive would require removing the cabinet from the recording setup, then the old glass has to be pulled off before the new one can be put in place, letting the honeycomb exposed. The customized design allows replacing the glass of the observation hive without need to remove it from the recording setup. It incorporates a set of guides that allows sliding the new glass in front of the old one before this is removed, keeping the cabinet closed at all time.

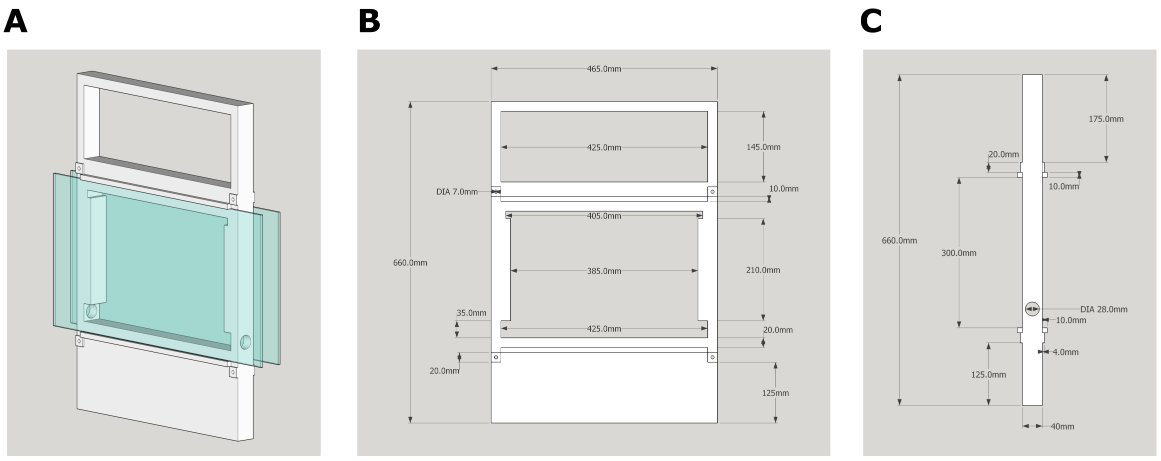


Figure 3.7: **Design of the customized observation hive.** (**A**) The customized design allows replacing the glass panes without having to open the cabinet. (**B**) Frontal and (**C**) Lateral detail of the design.

Scaffold and illumination setup

The observation hive stands in a scaffold that holds the IR lamps and the cameras (Fig. 3.8). The scaffold is built of a total of 26 aluminium profiles of the brand *item*. The scaffold comprises of two frame structures for the IR lamps, two for the Flea3 cameras and two arms for the PS3Eye cameras. The elements connecting the profiles enables the free movement of the components in three perpendicular axes to adjust the position of the lamps and cameras at convenience. The Flea3 cameras are mounted to the structure with a ball head (*CULLMANN CB2.7*), what effectively grants them a total of six degrees of freedom (three for position and three for orientation); the same is true for the PS3Eye cameras which are already shipped with a case that allows free rotation in three axes.

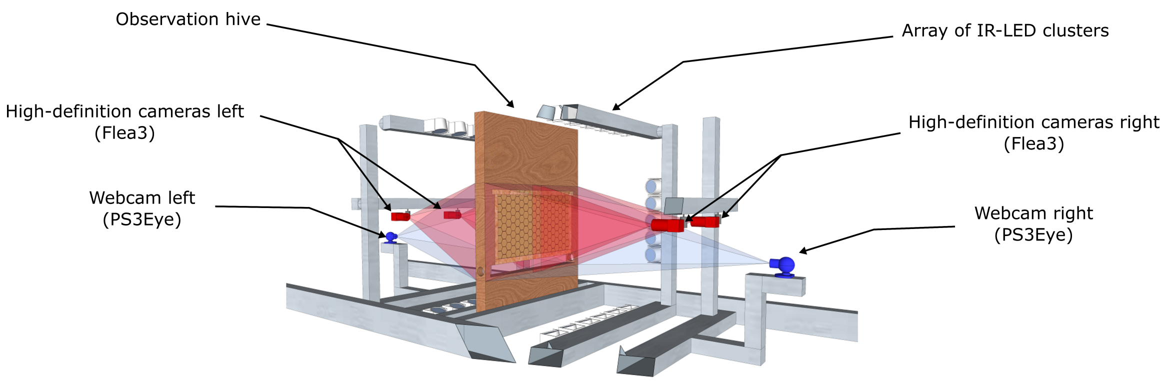


Figure 3.8: **Recording setup with IR LED clusters (840 nm wavelength), 4 high-resolution cameras (Flea3) and 2 webcams (PS3Eye).** The design of the scaffold allows multiple configurations for the lamps and grants the cameras 6 degrees of freedom to adopt practically any position and orientation.

The lighting setup comprises of 22 IR LED clusters (Abus TV6700) per side arranged in an adjustable frame. The entire skeleton is enveloped with IR reflector foil that has small embossments for light dispersion. The foil reflects 80% of infrared light and helps creating a homogeneous ambient lighting which reduces reflections on the glass pane or the tags. The IR LED clusters point towards the foil to create a homogeneous ambient lighting and at the same time reduce reflections on the glass panes. The observation hive is connected to one of the walls through a tunnel to allow colony members to leave and enter the hive.

Improvements to illumination setup for recording season 2016 [1]

Since the honeycomb is translucent to IR light, continuous lighting from both sides of the observation hive casts shadows from bees on one side of the comb to the op-

[1]The improvements here described were carried out by Hauke Mönck, member of the Biorobotics Lab. Further details can be found in his master's thesis [71].

posite side, plus it reduces the contrast in the images. To solve these problems and improve the image quality, the IR LED clusters used during the first two seasons were replaced for the season 2016 with asynchronous flashing IR LED boards (Fig. 3.9).

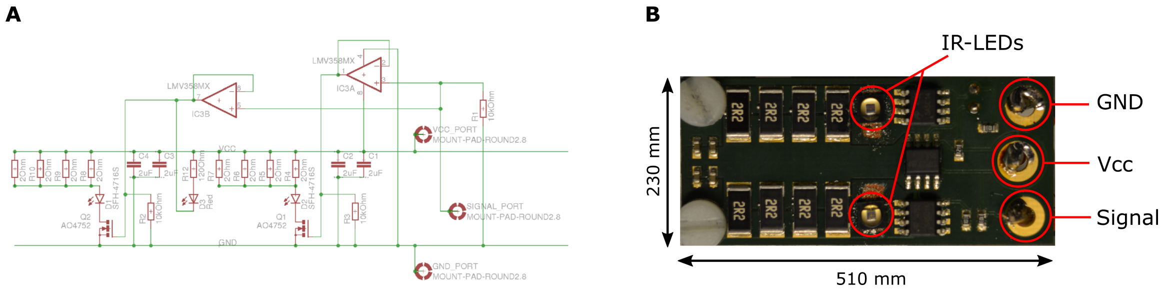


Figure 3.9: **The IR LED board design.** (**A**) Schematic diagram for the custom designed IR LED boards. (**B**) The design was implemented on $230 \times 510 mm$ PCB.

The new illumination setup consists of 5 custom designed IR LED boards at each side of the observation hive. Each board drives two high power IR emitters with a 150° beam angle (OSRAM SFH-4716S). Individuals sides of the setup flash asynchronously to avoid shadows and improve image contrast. Each set of LED boards flashes at a frequency of ∼ 3 Hz, and are synchronized to the Flea3 cameras on their setup side. An Arduino Duemilanove was used to trigger the asynchronous flashing of the LED boards and the synchronization with the corresponding cameras. The period of the flashing for the LED boards is defined according to the time of exposure of the Flea3 cameras, with an inherent limit of 1/6 of a second to fit the asynchronous flashing from both sides of the setup.

3.2 Image Acquisition Software

Each recording season produces about 65 M images ($4\,\text{cam} \cdot 3/\text{s} \cdot 60\,\text{s/min} \cdot 60\,\text{min/h} \cdot 24\,\text{h/d} \cdot 63\,\text{d} = 65{,}318{,}400$ images), depending on the compression ratio used to encode the images, this might represent ∼ 150 TB per season. Storing locally this amount of data is out of the scope of the *BeesBook* system, conceived as a budget-priced solution. From the early planning stage of the *BeesBook* system a solution to this problem was procured, enough computing and storage resources were granted to the project by the North German Supercomputing Alliance (HLRN). One of the HLRN's Cray XC30 supercomputer is housed in the Zuse Institute Berlin (ZIB), which is located just a few meters from the site of the recordings and offers gigabit ethernet connectivity. The Cray XC30 features 1872 compute nodes with 24 CPU cores each. The system has 117 TiB of RAM and 4.2 Petabyte of hard disk space, organized as RAID 6.

This section covers the details of the software module in charge of image acquisition. The module deals with a number of tasks, including retrieving of images from the Flea3 cameras, encoding them in JPEG format, and transferring them to the HLRN facilities for permanent storage.

Two applications were developed for the implementation of this module: bb _imageAcquisition and bb _imageStorage. Both applications were written in C++ and compiled in mingw64 gcc for portability. The applications also use the Qt framework to take advantage of the signals and slots mechanism for communication between objects. This section only covers the functionality of the applications, code and documentation of both applications is publicly available [2] [3].

During recording seasons bot applications run on the same computer. The computer was specially configured to administer the image acquisition module, an entry level workstation with an Intel® Xeon® processor E3-1280 (4 Cores), 32 GB of RAM, and two network cards with gigabit capabilities. One of the network cards is used to connect to the Cray. The second network card is connected to a NAS of 4 HDD of 3 TB each, organized as RAID 5. The RAID 5 configuration allows the recovering of any lost information upon the failure of a single drive. The NAS works as a cache if the Cray is not reachable and as permanent storage for *stripes* recorded over the day. The computer also runs a RAMDisk to enable part of the RAM as a virtual SSD. This allows buffering the images in RAM, instead of the HDD, before they are transferred to the HLRN. Using a regular HDD could delay the storing process and make it unstable.

From camera to computer - (bb_imageAcquisition)

In addition to the Qt framework, the `bb_imageAcquisition` application also uses the FlyCapture SDK to communicate with the Flea3 cameras. The application runs four threads in parallel, one for each of the Flea3 cameras. The software first initializes the cameras' parameters, including frequency. Each thread retrieves the images from the camera and saves them temporarily in the RAMdisk. The RAMdisk has a simple directory structure with one directory per camera. The bb_imageAcquisition thread saves the images in the corresponding directory and names the files following the schema:

```
Cam_ID_YYYYMMDDhhmmss_iii,
```

where YYYY = year, MM = month, DD = day, hh = hour, mm = minute, ss = second, and iii = fraction of second. For instance `Cam_1_20150812165048_333` corresponds to an image from camera 1 captured on August 12, 2015 at 16:50:48.333.

For season 2016 the original lighting setup was replaced with asynchronous flashing IR LED boards (see section 3.1.3). A program running on an Arduino Duemilanove synchronizes lights and cameras. A first pulse switches on the IR LED boards and it is

[2]bb_imageAcquisition: https://git.imp.fu-berlin.de/bioroboticslab/bb_imgacquisition

[3]bb_imageStorage: https://git.imp.fu-berlin.de/bioroboticslab/bb_imgstorage

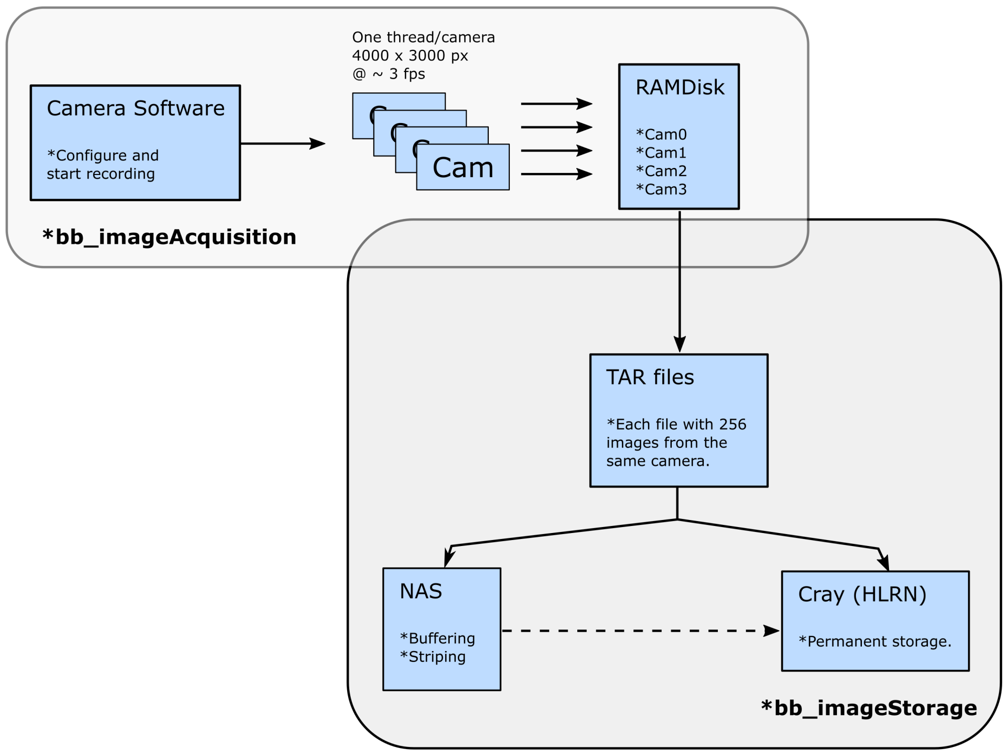


Figure 3.10: **Image acquisition module.** The module consists of two elements applications, namely `bb_imageAcquisition` and `bb_imageStorage`. `bb_imageAcquisition` configures the cameras parameters via software, initializes the recording process and encodes the images to `jpeg` format. `bb_imageStorage` accumulates the images in `TAR` files and transfers them to the Cray supercomputer (HLRN). The NAS is used as temporary storage if the connection with HLRN is broken.

hold up, a second pulse triggers the cameras' shutter, once the cameras' exposition has concluded the IR LED boards are turned off [4].

From computer to ZIB - (bb_imageStorage)

The `bb_imageStorage` application, originally developed by Christian Tietz [5], transfers the images from the local computer to the Cray supercomputer of the HLRN. It also saves a subset of the data, 6 stripes of 10min duration (distributed over the day), in the NAS. In addition to the Qt framework, this application also requires the `libcurl` library to connect with the Cray using SCP-protocol and keyfile authentification.

The application initializes four threads, one for each directory in the RAMDisk. Each thread manages one of the directories; when its directory reaches 256 files, the thread packs the images in a TAR file and transfers it to the Cray. If necessary, accord-

[4]Further details can be found in Hauke Mönck master's thesis [71].

[5]Further details on this application can be found in Christian Tietz master's thesis [110].

ing to the striping schedule, it also saves a local copy in the NAS. If the connection to Cray fails, a new child thread is initialized to temporarily copy the TAR file to the NAS and subsequently move it to the Cray, once the connection has been reestablished. All the manipulation of the TAR file is done in the RAMDisk, and the original data is only deleted after the transfer has been verified.

3.3 Image Analysis

The second software module also referred to as "pipeline", processes the high resolution images to detect and decode the binary markers (see Fig. 3.12). For each detected marker, the module obtains the following information:

- A sequential index of the detection, counted from 0 for every frame,
- *x*- and *y*- coordinates of the detection,
- a scale factor for the tag,
- rotation of the tag on all three axes (see Fig. 3.11), and
- the decoded marker ID.

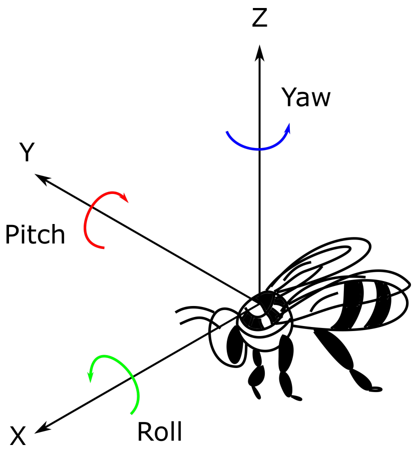


Figure 3.11: **The three rotation axes of a bee.** The position and orientation of the marker also tells the position of the bee. The x-rotation, or roll appears when the bee leans on her right or left side. The y-rotation, or pitch, appears when a bee leans into a comb cell. The z-rotation, or yaw indicates the direction in which the bee looks.

Two different approaches were implemented to solve this task. The first solution, based on a series of computer vision algorithms was first tested with images from season 2014. Although this approach is reliable for reading the position and orientation of the markers, it struggles with the id decoding phase. The second proposed solution is based on Deep Convolutional Neural Networks (DCNNs). DCNNs have shown remarkable results on many computer vision tasks. Unfortunately, their performance depends to a great extent on the amount of training data at hand. Generating labeled data for *BeesBook* is a time-consuming activity, which is why the implementation of this task was delayed till an alternative to annotating data manually became available.

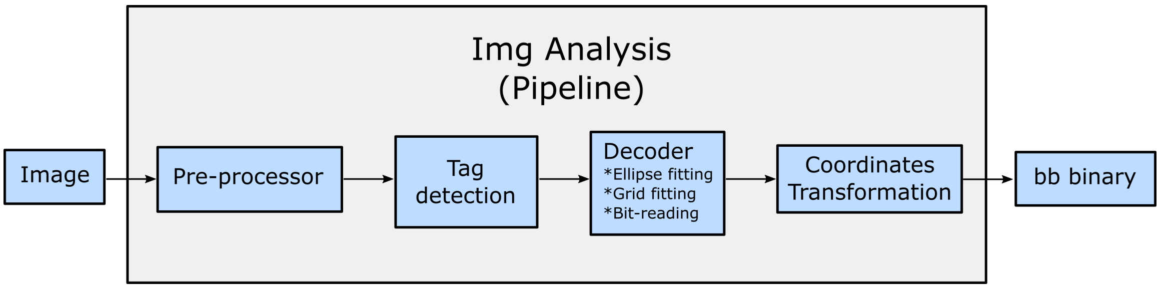


Figure 3.12: **Computer vision pipeline.** Schematic overview of the image analysis module.

3.3.1 CV-Pipeline

The computer vision pipeline (CV-Pipeline) used to analyze the data from season 2014 is organized in five layers. Each one of the listed layers processes the results of the previous one to extract more information from an initial detection (see Fig. 3.13). The software is written in C++ using the OpenCV framework and compiled in mingw64 gcc for portability. A description of the layers follows.

Image preprocessing. The first layer preprocesses the original images. To normalize brightness and contrast across different regions of the images, different cameras, and recording seasons, the images were preprocessed using the Contrast Limited Adaptive Histogram Equalization (CLAHE) [127].

Tag localizer. The second layer detects image regions containing strong edges in closed proximity and therefore more likely to display binary tags. First, the derivative of the image is computed using the Sobel operator. This image is then binarized using a threshold to create an edge map. The edge map is then eroded to remove noise, and dilated to join adjacent clusters of pixels and create bigger patches [99]. Large binary patches are reported as regions of interest (ROI) to the next layer.

Ellipse fitter. This layer detects elliptic contours in the ROI. The layer's input is a 50 × 50 px sub-image of the edge map centered at one of the ROI. A probabilistic Hough transform [125] is a used to find likely ellipse configurations defined by a high amount of edge pixels agreeing with an ellipse equation for a range of plausible parameter values (heavily rotated tags were excluded since those are likely to be decoded incorrectly).

Grid fitter. For each ellipse that has been detected, this layer fits a three-dimensional model (“Grid”) of the binary tag to the underlying image. When rotated in space, the contour of the circular tag becomes an ellipse in the camera image. There are two possible 3D rotations of a circular contour that project to a given ellipse in the image. The two rotation sets are computed from the ellipse parameters and used as starting points for the search of the best grid fit using gradient descent. The quality of the fit is expressed by a scoring function (Eq. 3.1) that evaluates the homogeneity of the pixel brightness in each cell and the match of the outer grid contour to the edge pixels:

$$S_G(G) = \alpha \cdot s_h + \beta \cdot s_e, \tag{3.1}$$

where α and β are coefficients to adjust the weight of the scoring function parts.

The latter part of the scoring function is easily computed as the number of pixel in the grid contour matching edge pixels in the image:

$$S_e(G) = \frac{\sum_{i=1}^{N_s} E_l(x_i, y_i)}{|N_s|}, \tag{3.2}$$

where E_l is a binary function that returns the value of (x_i, y_i) in the edge map, and N_s the contour pixels for the grid.

Respectively, the first part is computed using Fischer scoring [33]:

$$S_e(G) = \frac{|\mu_w - \mu_b|^2}{\sigma_w^2 + \sigma_w^2}, \tag{3.3}$$

where μ_i are the centroids for black and white classes and σ_i^2 their variance. The three best grid configurations are reported to the decoder layer.

Tag decoder. Each ring segment of the grid represents one bit of the marker ID and it can be either a “0” (black) or a “1” (white). Local contrast enhancement is applied to account for light intensity gradients on the binary tag. The decoder layer computes a statistic of the brightness of all underlying pixels for each ring segment to classify the cells to either of the two classes and reports the number as the marker ID.

Pipeline parameters optimization.

The pipeline has various parameters such as thresholds for the edge detection or the number of iterations of the morphological operations, a total of 48 parameters [6].

[6]For a full list of the parameters: https://github.com/BioroboticsLab/deeplocalizer_data

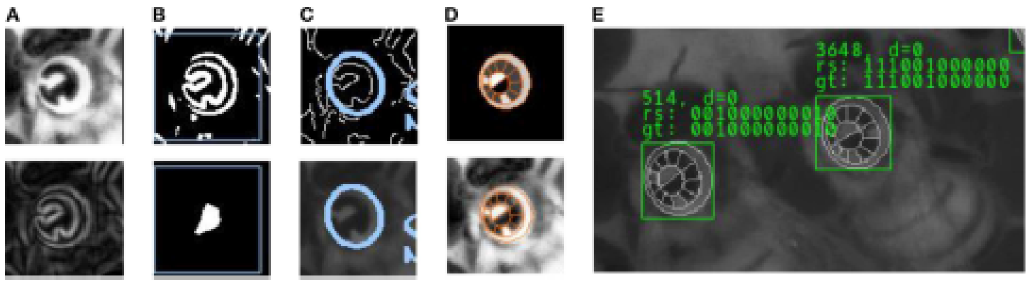


Figure 3.13: **The five layers of the computer vision approach to the pipeline.** (**A**) Histogram equalization and Sobel edge detection. (**B**) Edge binarization and morphological operations in the localizer layer. Only regions of interest (marked with a blue rectangle) are processed in the next layer. (**C**) Ellipse fit using probabilistic Hough transform. (**D**) 3D Grid model and fit to underlying image. (**E**) Result: The sequence of 0s and 1s is determined in the Decoder layer, based on the fit of the tag model.

Manually determining the optimal combination of parameters can be very time consuming and might result in a suboptimal performance. A global optimization library is used [66] to automatically select the best set of parameters.

Stitching and coordinates transformation[7]**.**

The coordinates of the tag reported by the grid fitter layer are given with respect to the image. Since each camera only records one fourth of the total observation hive surface, the detection coordinates have to be translated to a global reference that indicates the position of the bees on the hive surface. The field of view of two cameras recording at the same side of the hive share a small overlapping area. Prominent features are found in this area and matched on images from both cameras to obtain a homography. This way, images from both cameras can be stitched, duplicated detections filtered and the detection coordinates translated to a hive reference.

The bb-binary format.

The final output of the module is saved in a *BeesBook* specific data format (`bb-binary`) whose specification is publicly available [8]. The data format relies on a Cap'n Proto schema, a fast data interchange format and capability-based RPC system, it generates classes with accessors methods from the fields specified in the schema. Each tag detected by the *pipeline* is represented in the `bb-binary` format as a data point called Detection which contains the following information:

- A sequential index of the detection, counted from 0 for every frame,
- sequential index of the candidate per tag,

[7]This functionality was implemented by Peter Strümpel. Further details can be found in his master's thesis [107].

[8]Binary format for the BeesBook detections: https://github.com/BioroboticsLab/bb_binary

- sequential index of the grid/decoding per tag,
- x- and y- coordinates of the grid center wrt. the image,
- x- and y- coordinates of the grid center wrt. the hive,
- rotation of the tag on all three axes (see Fig. 3.11),
- scores for the localizer (ROI), ellipse fitter, and grid fitter, and
- the decoded id.

3.3.2 DCNN-Pipeline

The image analysis module based on DCNN operates in a similar way to its counterpart based on computer vision algorithms, with the difference that detecting and decoding stages (see Fig. 3.14) are replaced with two DCNNs. The first DCNN, named *localizer network*, processes the input image to generate a saliency map with the probability for each pixel to be located at the center of a marker. Morphological operations combined with a minimum threshold value are used to extract the local maxima. Regions of interest (ROI) centered at the local maxima are forwarded to the second DCNN, reducing this way the overall computational cost of the image analysis module. The *decoder network* finds the most likely configuration for the marker contained in the ROI. The output of the network includes the rotation of the marker, its radius, and a probability for each ID bit. The DCNN approach here described was developed by Leon Sixt and Benjamin Wild who compiled some of the following information in their theses [101, 123].

Localizer network.
Although DCNNs have shown remarkable performance on many computer vision tasks [57, 44], its performance is highly tied to the quantity and quality of labeled data available for their training. Hence, defining a quality data set whose production cost is not too high is the first step to designing a successful DCNN.

For the localizer network, whose task is finding ROIs with a high probility of containing a marker the data set was prepared as follows: Using a custom user interface, All position of markers were manually labeled in a small subset of the raw *BeesBook* image data. Small (100 × 100 px) subimages were randomly sampled from the labeled images. A measure for the probability that a marker is in the center of the subimage is defined as the probability density of the center point under a bivariate normal distribution centered at the next marker center with a fixed variance (Eq. 3.4).

$$f(x,y)=\frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}}exp\left[-\frac{z}{2\left(1-\rho^2\right)}\right], \tag{3.4}$$

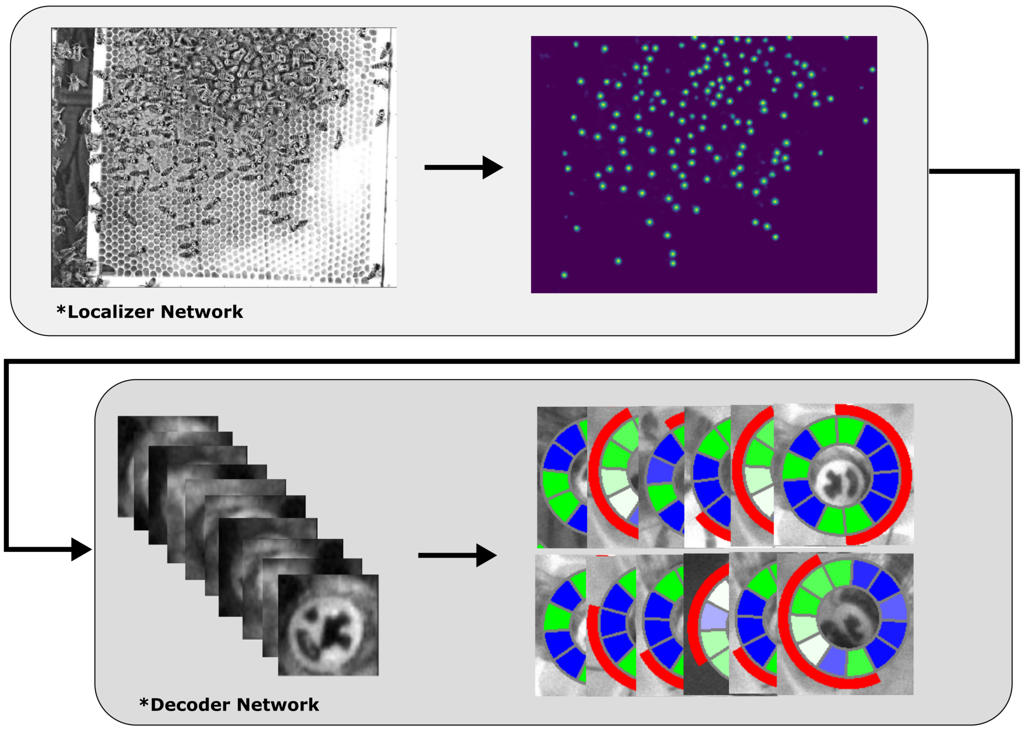


Figure 3.14: **Visualization of the DCNN-Pipeline stages.** The DCNN-Pipeline comprises of two DCNNs: Localizer network and decoder network. The localizer network generates a saliency map from where ROI with high probability to have a marker at the center are defined. The decoder network analyzes these ROIs and finds the most likely configuration for the marker in the image. In the visualization of the decoder's output, the color of the cells is a representation of the probability for each ID-bit to be set, fading from solid blue to solid green (0 to 1). The red semi-arc indicates the decoded orientation of the marker.

where

$$z \equiv \frac{(x-\mu_X)^2}{\sigma_X^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y} + \frac{(y-\mu_Y)^2}{\sigma_Y^2}, \tag{3.5}$$

x and y are the center coordinates of the subimage, μ_X and μ_Y are the center coordinates of the next marker in the image and

$$V_{XY} = \begin{bmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{bmatrix} \tag{3.6}$$

is a constant covariance matrix.

The localizer is a fully convolutional neural network with three convolutional layers with a kernel size of 5 px and strideof 2 px followed by a ReLU activation. No padding is applied before the convolutional layers so that the model can be easily applied to full images during inference and not only to the small regions of interest in the training

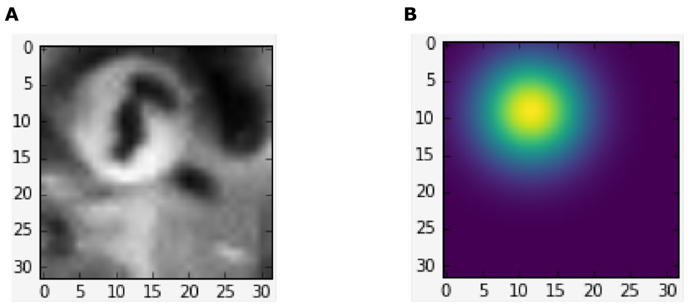


Figure 3.15: **A region of interest an its corresponding label.** The center of the marker in the original image (**A**) corresponds with the highest probability in the label (**B**).

data set. Dropout is applied following each convolutional layer to reduce overfitting. A final convolution layer with a kernel size of 1 px and a stride of 1 px followed by a sigmoid activation computes the probability of each pixel being in the center of a tag.

Because the described model is fully convolutional, it can be applied to images of various sizes; which is why the network is applied to the whole input image during inference (after preprocessing and downsampling). Processing the images in this manner is much faster than extracting subimages and applying the network individually to each subimage. The network generates a saliency map for the whole image, to which morphological operations and a maximum filter are applied to extract the pixels with the highest probability to be located at the center of a marker. From here the DCNN-Pipeline follows the process outlined in Fig. 3.14.

Decoder network.

The decoder network's goal is to find the most likely configuration for the marker contained in each ROI. The output of the network should include the rotation of the marker, its radius and the probability that each bit of the tag is set to one. The first attempt to decode *BeesBook* markers using a DCNN did not deliver significantly better results than those of the computer vision decoder. Being the main constraint the amount of labeled data available at that point in time. Furthermore, labeling the binary markers manually, even with the help of a custom user interface, proved to be a time consuming and tiresome activity, and the costs of generating a sufficiently extensive data set rendered the DCNN approach temporarily impractical. By the end of 2016, an extension to the Generative Adversarial Network (GAN) framework proposed by Sixt et al. [102] allowed generating realistic synthesized labeled images to train the DCNN (see Fig. 3.19) and significantly improve its performance.

The decoder network is a large CNN based on the ResNet architecture proposed by He et al. [44].

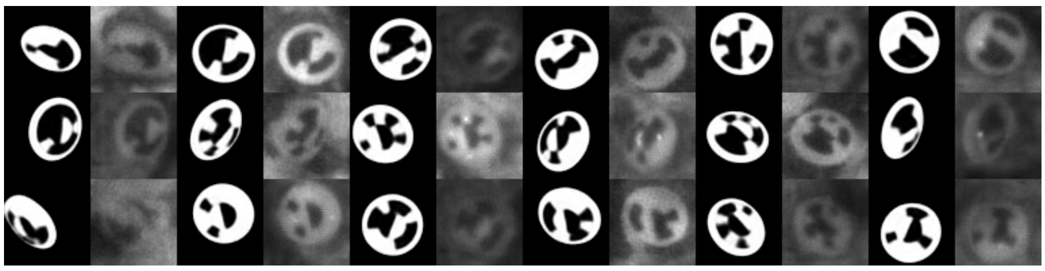


Figure 3.16: **Sample of synthesized labeled images generated with the RenderGAN framework.** The process begins with the automatic generation of an image of the marker's 3D model. This image is then modified by multiple stages that add realistic details without affecting the original label (rotation, radius and ID).

Layer name	Output size	Layer
input	64×64	-
conv1	32×32	3×3, 16, stride 2
conv2-x	32×32	$\begin{bmatrix} 3 \times 3, & 16 \\ 3 \times 3, & 16 \end{bmatrix} \times 3$
conv3-x	16×16	$\begin{bmatrix} 3 \times 3, & 32 \\ 3 \times 3, & 32 \end{bmatrix} \times 4$
conv4-x	8×8	$\begin{bmatrix} 3 \times 3, & 64 \\ 3 \times 3, & 64 \end{bmatrix} \times 6$
conv5-x	4×4	$\begin{bmatrix} 3 \times 3, & 128 \\ 3 \times 3, & 128 \end{bmatrix} \times 3$
id	12	fc 256, fc 12
params	6	fc 256, fc 6

Table 3.1: **Architecture of decoder network.** The brackets denote ResNet building blocks. The *id* and *params* layer receive both the output from layer conv5_3 as input. The output of the *params* layer represents the *orientation*, *position*, and *radius*. As in the original ResNet architecture, downsampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

As with the computer vision pipeline, the final output of the module is saved in a `bb-binary` data format containing the following information:

- A sequential index of the detection, counted from 0 for every frame,
- x- and y- coordinates of the grid center wrt. the image,
- x- and y- coordinates of the grid center wrt. the hive,
- rotation of the tag on all three axes (see Fig. 3.11),

- radius of the tag,
- saliency of the localizer network, and
- the decoded ID, the bit probabilitties are discretized to 0-255.

3.4 Tracking and ID Filter

Keeping track of the bees through time allows creating motion paths that can be later used for behavior analyses. Under optimal circumstances, tracking an individual and generating its path is as simple as connecting consecutive detections of the same ID. However, since *BeesBook's* pipeline does not have a 100% decoding accuracy, and the bees are not always visible, merely connecting successive appearances of the same ID would yield erroneous tracks (see Fig. 3.17). Therefore, the real problem to solve is finding correct correspondences between detections that might not exist in consecutive frames, and that might not exhibit matching IDs.

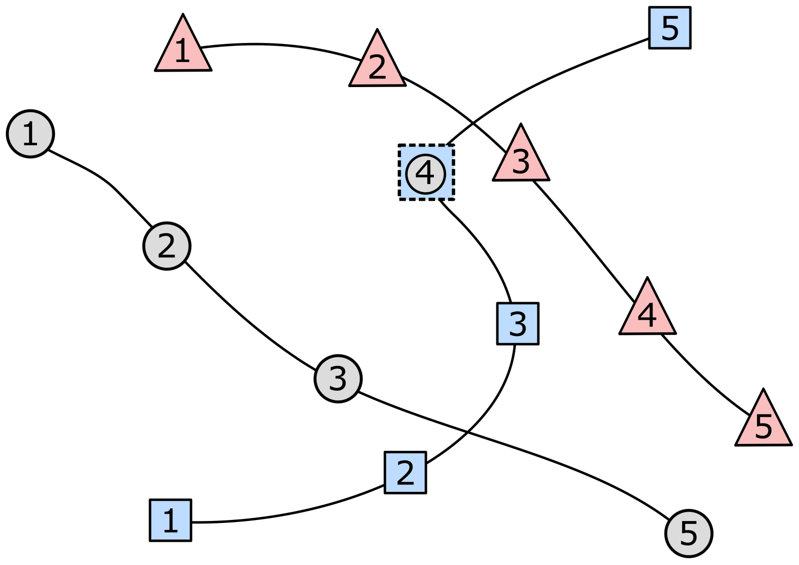


Figure 3.17: **Visualization of the tracking problem.** In an ideal scenario tracking identifiable moving objects is as easy as connecting consecutive detections with the same identity (triangles). In reality tracking is not a trivial task, especially in complex environments where the subjects are not always visible (circles), and their identities are sometimes misread (squares).

One way to deal with this problem is extending the search space for correspondence to more than one frame into the future and conducting and exhaustive search. Considering that *BeesBook* has an average rate of 200 detections per frame, using this approach would be computationally too expensive for the scope of the system. Furthermore, there are better approaches to the problem like the one here proposed[9]. The tracking module in *BeesBook's* addresses the tracking problem using a hierarchical approach based on probabilities [10]. During a first iteration, only detections from consecutive frames are considered to form trajectory fragments without gaps ("so-called tracklets"). A random forest classifier trained on a manually labeled data set is used to predict the probability of two detections belonging to the same track. Then, using the Hungarian algorithm (Kuhn-Munkres algorithm [58]) a global mapping of detections in time **t** to detections in **t+1** is generated. However, uninterrupted tracklets represent only part of all possible trajectories. Following the approach from Fasciano et al. [28], the tracking module links tracklets with temporal gaps during a second iteration.

Besides generating reliable motion paths of single individuals for behavioral analyses, the tracking module serves the second purpose of correcting misread IDs. An ID is computed for each trajectory as the bitwise averaged ID over all detections of the track. Eventually, after the last iteration of the module, the IDs of the individual detections are updated to that of their trajectory. A more in depth description of the first and second iteration follows.

Linking consecutive detections (first tracking iteration).

In an iterative scheme, all detections at time **t** fall in one of two classes: either the most recent detection of an open tracklet or a seed detection to start a new tracklet. A search radius of 200 px (approximately 12 mm, less than a bee's body length) is defined to create a list of potential successors for each of these detections. From each candidate pair (last detection of the tracklet and candidate detection), the following three features are extracted:

- **distance**: Euclidean distance in pixels.
- **id_distance**: Hamming distance between both detections' IDs.
- **z_rot_diff**: the difference of tag orientation between the last detection of the first fragment and the first detection of the second fragment. (see Fig. 3.11) in degrees.

A trained random forest classifier [64] is used to map these features to the probability that both detections correspond to the same individual. Finally the Hungarian algorithm is used to assign detections at time **t+1** to tracklets. Pairs with a score lower than a predefined threshold are discarded. Detections that could not be assigned to any tracklet are regarded as seed detections in the next step of the iteration. Tracklets that could not be assigned new detections are closed, i.e., they will not be regarded in the next step.

[9]The approach here described was implemented by Franziska Boenisch and Benjamin Rosemann of the Biorobotics Lab. Further details can be found in their theses [9, 92]

Linking tracklets. (second tracking iteration) After the first iteration the process is repeated, this time considering only tracklets and individual detections that were not assigned to any trajectory during the first iteration. With fewer candidates at each step of the iteration, it is possible to extend the search for candidates to no adjacent frames. The size of the affordable time gap for this step is not fixed and requires some analysis of the distribution of gaps among the data set produced by the first iteration. When selecting this parameter is important to note that allowing long gaps will be more computationally expensive than limiting the gaps to just a few frames, but the latter might require more iterations to finish the tracking analysis.

Tracklets with two or more detections allow more complex and discriminative features than those used during the first iteration. When linking tracklets over gaps, features that reflect a long-term trend, like the direction of motion are worthy to take into consideration, the following six were found to have a high predictive power:

- **distance**: Euclidean distance of last detection of fragment 1 to first detection of fragment 2.
- **id_distance**: Hamming distance of both fragments' bitwise averaged IDs.
- **forward_error**: Euclidean distance of linear extrapolation of last motion in first fragment to first detection in second fragment.
- **backward_error**: We extrapolate the starting point of the gap-spanning motion by projecting back the first motion segment in fragment two and compute the Euclidean distance to the last detection in fragment one.
- **confidence**: all IDs in both fragments are averaged (bitwise), the minimum difference of all averaged bit confidences to the value 0.5 (maximum uncertainty) is extracted from both fragments and compared (absolute difference).
- **z_rot_diff**: the difference of tag orientation between the last detection of the first fragment and the first detection of the second fragment.

Similar to the first iteration process, these features are computed at each step of the iteration, then a trained random forest classifier maps them to probabilities, and finally the Hungarian algorithm assigns candidate tracklets at time **t+n** to tracklets in **t**.

3.5 Automatic Detection and Decoding of Communication Dances

The detection and decoding of waggle dances is carried out by the waggle dance decoder (WDD) sub-module. The *WDD* relies on detecting the waggle run portion of communication dances. During waggle runs, bees pull their body from side to side at a frequency of $\sim 13Hz$, therefore the cameras must be capable to stream video at higher frequencies. The solution here proposed requires a frame rate of $\sim 100fps$ and a resolution of at least 1.5 pixels per millimeter. Thus QVGA resolution suffices to cover the whole surface of a "Deutschnormal" frame (370 mm x 210 mm). The four frame corners are used as a reference to rectify distortions caused by skewed viewing angles or camera rotations; If the frame corners are not captured by the camera, it is necessary to consider other reference points and their relative coordinates.

3.5.1 Target Features

As explained in section 1.2, forager bees share the location of valuable resources with their nestmates through the waggle dance (see Fig. 1.2). The relation of site properties (distance and direction to the feeder) and dance properties (duration and angle) has been documented thanks to systematic experiments [116]. Using these data, it is possible to approximate an inverse mapping of dance parameters to site properties.

$$r_R \sim f_d^{-1}(d_w). \tag{3.7}$$

$$\theta_R \sim atan2\left(\sum_{j=1}^{n} sin\alpha_{wj}, \sum_{j=1}^{n} cos\alpha_{wj}\right), n = 2k. \tag{3.8}$$

$$p_R \sim \frac{d_w}{d_r}. \tag{3.9}$$

Where r_R (Eq. 3.7) is the distance between hive and resource and is related to the average waggle run duration d_w through the function f_d that approximates the calibration curve [114]. θ_R (Eq. 3.8) is the angle between resource and solar azimuth (see Fig. 1.2), and it corresponds to the average orientation of the waggle runs with respect to the vertical, with an even number of consecutive runs to avoid errors due to the divergence angle [116, 120, 109, 62]. The resource's profitability p_R (Eq. 3.9) is proportional to the ratio between average waggle run duration d_w and average return run duration d_r. Dances for high-quality resources contain shorter return runs than those for less profitable resources located at the same distance, hence yielding a higher p_R value [97].

From Eq. 3.7 to Eq. 3.9 it follows that to decode the information contained in a communication dance three measurements are required: average waggle run duration d_w, average orientation α_w, and average return run duration d_r. Most of the current approaches available to decode communication dances require first to trace the dance path to then analyze it and extract its characteristics [51, 52]. In contrast to these methods, the waggle dance decoder of the *BeesBook* system directly analyzes video frames to obtain each waggle run starting timestamp, duration, and angle. The duration of return runs is calculated as the time difference between the end of a waggle run and the beginning of the next one (Fig. 3.18).

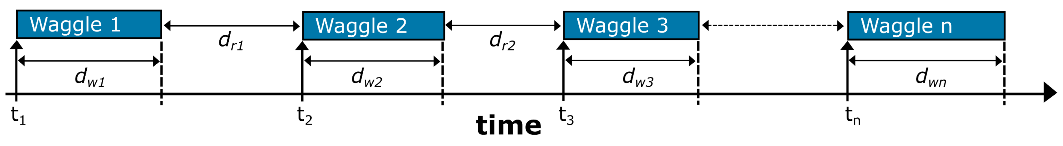


Figure 3.18: **Fundamental parameters of the waggle dance.** If the starting time (t_x) and duration (d_{wx}) for each waggle run is known, it is possible to calculate the return run durations as the duration of the gaps between consecutive waggle runs.

3.5.2 Software Modules

The dance decoder consists of four modules that are executed in sequence namely: attention module (*AM*) [10], filter network (*FN*), waggle orientation module (*OM*) and mapping module (*MM*) (Fig. 3.19). The AM analyzes either online video or video recordings in the search of waggle-like activity, if any is detected, a small sub-region of the video where the activity was detected is saved to disk. False positives detected by the AM are filtered out using a convolutional neural network *FN*. Later, the *OM* extracts the duration and angle of the filtered waggle runs. Finally, the mapping module (*MM*) clusters waggle runs belonging to the same dances and maps their parameters back to field coordinates. Although all modules can run offline on video recordings, long observations require large storage space; for this scenarios it is recommended to run the *AM* on real-time camera input to reduce the amount of stored data drastically. A detailed description of all four modules is given in the following sections.

Attention module (AM)

The waggle frequency of dancing bees lies within a particular range of values; here on referred to as *waggle band* [61, 36]. Each pixel in the video images covers a small area of the honeycomb surface and its intensity evolution through time reflects the region's activity. Since the bee body is well discriminable from the background under consistent lighting conditions, when a pixel intersects with a bee, its intensity time-series is a function of the bee's motion dynamics. Indeed, by using a camera with low spatial resolution, bees appear as homogeneous ellipsoid blobs without surface texture (Fig. 3.20). Thus the brightness of pixels that are crossed by waggling bees varies with

[10]The implementation of the AM was carried out by Alexander Rau as part of his master's thesis [87], conducted in the Biorobotics lab during the summer of 2014.

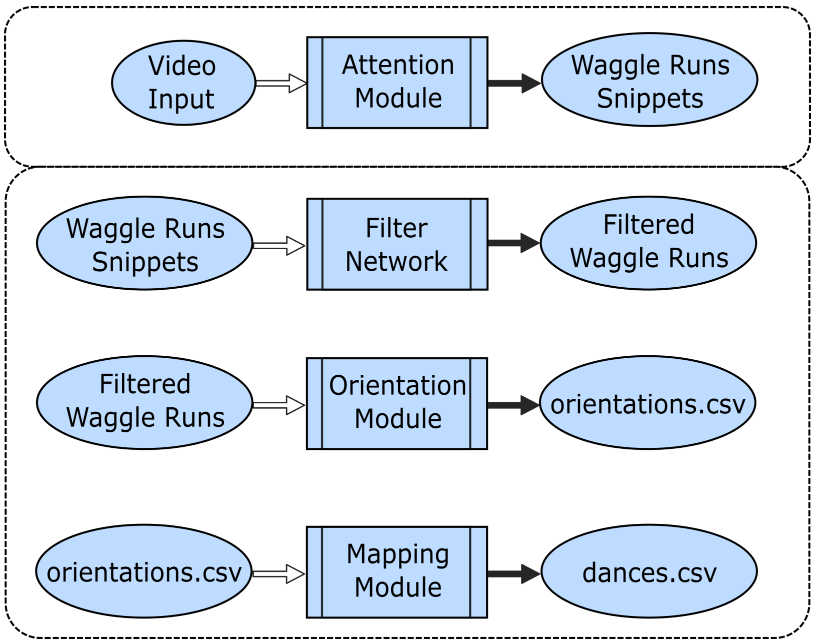


Figure 3.19: **Software modules of the waggle dance detector and decoder**. The automatic detector and decoder sub-system comprises of four modules. Even though all modules can run offline on video recordings, the AM is usually run online on live video streaming to minimize the disk space required for video storage.

the periodic waggle motion, and their frequency spectrum exhibits components in the *waggle band* or harmonics.

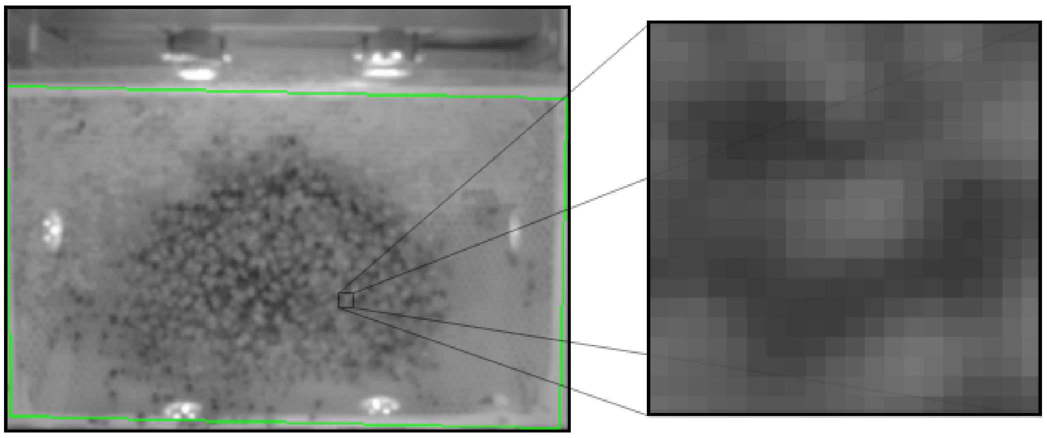


Figure 3.20: **A screenshot of the waggle detection camera stream.** (left) The green rectangle demarcates the comb borders. Despite the video low resolution, it is still possible to discern between the honey bees body and the background (detail shown on the right).

To detect these events, a binary classifier here on referred as *Dot Detector* (*DD*) is defined [87]; each pixel position $[i, j]$ is associated to a *DD* D_{ij}. The *DD* analyzes the intensity evolution of the pixel within a sliding window of width b. For this purpose, the last b intensity values of each pixel are stored in a vector B, which at the time n can be described as $B^n_{ij} = \left[v^{n-b+1}_{ij}, v^{n-b+2}_{ij}, \ldots, v^n_{ij}\right]$, where v^k_{ij} is the intensity value of the pixel $[i, j]$ at time k. We calculate a score for each of these time series using a number of sinusoidal basis functions, in principle similar to the Discrete Fourier Transform [14]:

$$s\left(\bar{B}^n_{ij}, r\right) = \sum_{m=1}^{b} \left(\left(\bar{B}^n_{ij}(m) \cdot cos\left(2\pi r \frac{m}{s_r}\right)\right)^2 + \left(\bar{B}^n_{ij}(m) \cdot sin\left(2\pi r \frac{m}{s_r}\right)\right)^2 \right), \tag{3.10}$$

where $\bar{B}^n_{ij}$ is the normalized version of B^n_{ij} with $\bar{B}^n_{ij} \in \{-1, 1\}$ and $min^n_{ij} = min\left(B^n_{ij}\right)$ and $max^n_{ij} = max\left(B^n_{ij}\right)$, s_r is the video's sample rate (100 Hz), and $r \in [10, 16]$ are the frequencies in the waggle band. If at least one of the frequencies in the *waggle band* scores over a defined threshold t_h, D_{ij} is set to 1.

To the analysis in the frequency domain follows a process divided into two layers. In the first layer those D_{ij} set to 1 are clustered together following a hierarchical agglomerative clustering (*HAC*) approach [100], using as a metric the Euclidean distance between pixels and with a threshold d_{max1} set to half the body length of a honey bee. To filter out false positives, only clusters formed by a minimum of c_{min1} *DDs* are regarded as positions of potential dancers.

In the second layer, positions found during the clustering step are used to form waggle runs (*WR*). Consecutive detections are considered as part of the same *WR* if they are located within a maximum distance d_{max2}, defined according to the average waggle forward velocity (see [62]). At each iteration new dancer detections are matched against open *WR* candidates, and either appended to a candidate or used as basis for a new one. A *WR* candidate can remain open up to g_{max2} frames without new detections being added, after what it is closed. Only closed *WR* candidates with a minimum of c_{min2} detections are retrieved as *WRs*. Finally, coordinates of the potential dancer along with 50 x 50 pixels image snippets of the *WR* sequence are stored to disk.

The operation of the *AM* can be seen as a three layers process (Fig. 3.21) summarized in the following points:

1. Layer 0, for each new frame I^n:
   (a) Update DDs' score vector.
   (b) Set to 1 DDs with spectrum components in the waggle band above t_h
2. Layer 1, detecting potential dancers:
   (a) Cluster together DDs potentially activated by the same dancer.

   (b) Filter out clusters with less than c_{min1} elements.
   (c) Retrieve clusters' centroids as coordinates for potential dancers.

3. Layer 2, detecting waggle runs:
   (a) Create waggle run assumptions by concatenating dancers positions with a maximum Euclidean distance of d_{max2}.
   (b) Assumptions with a minimum of c_{min2} elements are considered as real *WR*.

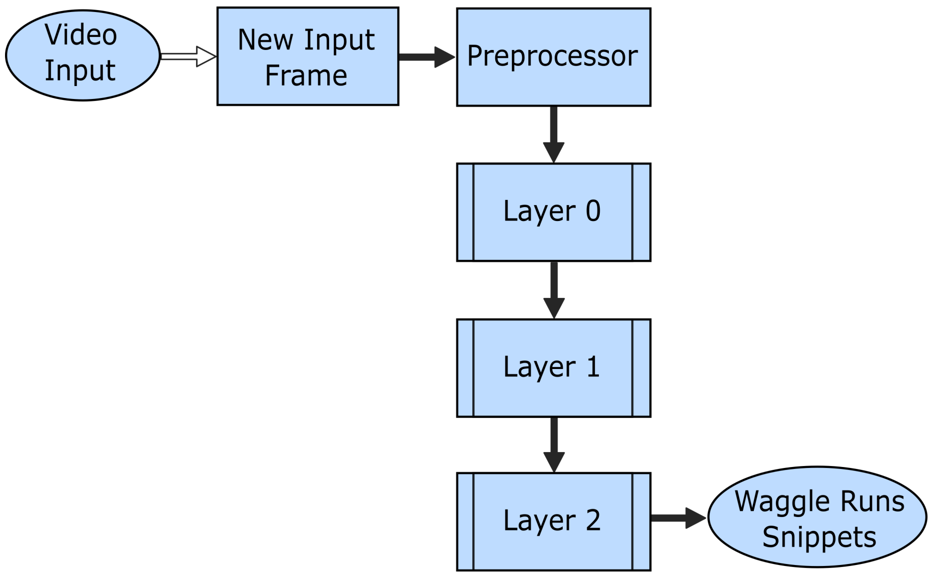


Figure 3.21: **Flowchart of the attention module (AM).** Each new input frame is preprocessed before its pixel's values are used to update the *DDs*' score. If after undergoing all three layers of the module a waggle run is detected, 50 × 50 px images snippets of the *WR* sequence are stored to disk.

Filtering network (FN)

The *AM* can be run offline on video recordings, but to reduce the storage requirements by longitudinal studies, it is advised to use it online with video from a camera streaming in real-time. Since only activity detected by the *AM* is stored to disk, this will reduce drastically the amount of storage required by the system. However, the threshold values of the *AM* must be set to fairly sensitive values to maintain the number of false negatives to a minimal, and reduce the risk of loosing significant data. This might lead in turn to a higher number of false positive detections and to the decrement of the detector precision.

To improve the system's precision, the detections can be filtered later using a binary classifier that classifies them either as correct detections or false positives. The solution here proposed to this end is a convolutional neural network designed and trained to process the sequences of 50 × 50 px images saved by the *AM*. This filtering network (*FN*) produces a scalar output that is then thresholded to predict whether the input sequence contains a waggle run. The network is a 3D convolutional network whose convolution and pooling layers are extended to the temporal dimension [47, 113, 105, 56]. The network architecture, three convolutional and two fully connected layers, is rather simple but suffices for the filtering tasks (Fig. 3.22).

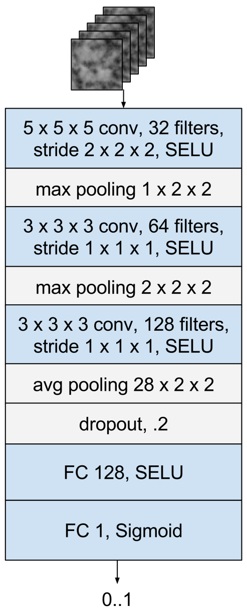

5 x 5 x 5 conv, 32 filters, stride 2 x 2 x 2, SELU
max pooling 1 x 2 x 2
3 x 3 x 3 conv, 64 filters, stride 1 x 1 x 1, SELU
max pooling 2 x 2 x 2
3 x 3 x 3 conv, 128 filters, stride 1 x 1 x 1, SELU
avg pooling 28 x 2 x 2
dropout, .2
FC 128, SELU
FC 1, Sigmoid
0..1

Figure 3.22: **Architecture of ConvNet.** The raw sequences of images are processed by two stacked 3D convolution layers with SELU nonlinearities. The outputs of the second convolutional layer are flattened using average pooling in all three dimensions. A final fully connected layer with a sigmoid nonlinearity computes the probability of the sequence being a dance or not. Dropout is applied after the average pooling operation to reduce overfitting.

The network was trained using a total of 8239 manually labeled AM detections from two separate days. During training, sub-sequences consisting of 128 frames were randomly sampled from the detections for each mini-batch. Detections with less than 128 frames were padded with constant zeros. Twenty percent of the manually labeled data was reserved for validation. To reduce overfitting, the sequences were randomly flipped on the horizontal and vertical axes during training. We used the Adam [53] optimizer to train the network and achieved an accuracy of 90.07% on the validation set. This corresponds to a recall of 89.8% at 95% precision.

Orientation module (OM)

While the duration of the *WRs* is estimated from the number of frames exported by the *AM*, its orientation is computed in a separate processing step here on referred as orientation module (*OM*). During waggle runs bees throw their body from side to side at a frequency of about 13 Hz [62]. Images resulting from subtracting consecutive video frames of waggling bees exhibit a characteristic pattern similar to a 2D Gabor filter, a positive peak next to a negative peak; since this pattern reflects the previous and current position of the dancing bee, its orientation is aligned with the its body Fig. 3.23.

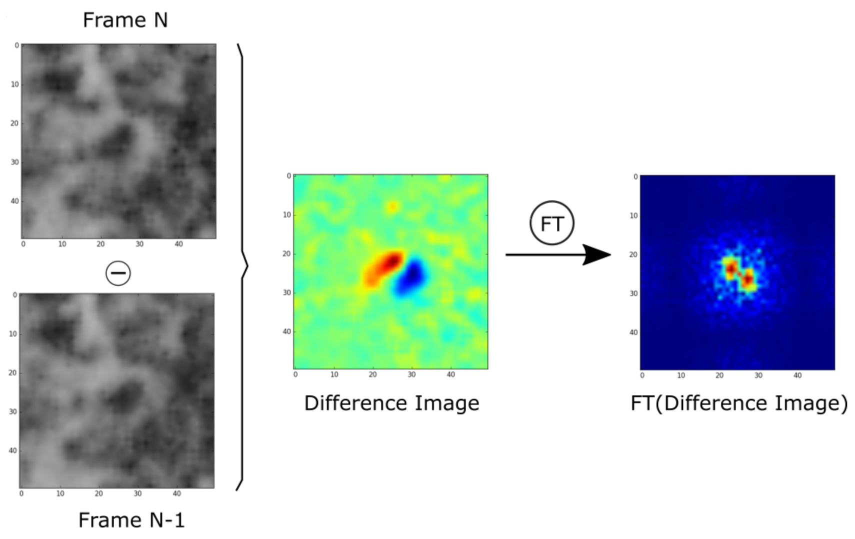


Figure 3.23: **Difference image and its Fourier transformation.** The image resulting from subtracting consecutive video frames of a waggling bee exhibits a characteristic Gabor filter-like pattern. While the peak location varies in image space along with the dancer's position, its representation in the Fourier space is location-independent, showing distinctive peaks at frequencies related to the size and distance of the Gabor-like pattern.

The location of the Gabor filter-like structure in the difference image is variable and determined by the dancing bee's location itself. Transforming the difference image to the Fourier space provides a location-independent representation of the bee's lateral movement and orientation (Fig. 3.23). The Fourier transform is the series expansion of an image into sinusoidal basis functions. Maxima in the Fourier space correspond to the frequency and orientation of the main sinusoidal components of the input image, in the case of the difference images, those matching the Gabor filter-like structure. Since the bee's longitudinal axis is orthogonal to her lateral movements, we can safely conclude that the orthogonal to the maxima in the Fourier space corresponds to the dancer's orientation.

Not all difference images in a given image sequence exhibit the Gabor pattern, it only appears when the bee is quickly moving laterally. To get a robust estimate of the average waggle orientation, it is convenient to sum all Fourier transformed difference images obtained from the *WR* sequence (Fig. 3.24**A**). To improve the quality of the orientation lecture, one can apply a bandpass filter to this cumulative sum. This is achieved by multiplying the resulting cumulative sum with a difference-of-Gaussians *DoG* (Fig. 3.24**B**). The radius of the ring needs to be tuned to the expected spatial frequency of the Gabor pattern. This frequency depends on the frame rate (100 Hz, for video from the PS3Eye webcam) and the image resolution (17 px/mm). With the lateral velocity of the bee (descriptive statistics can be found in [62]) one can compute the displacement in pixels (5 to 7 px/frame). Using Eq3.11 one can approximate the value of the expected frequency k:

$$k = \frac{I_{size}}{T} = \frac{I_{size}}{2x}, \tag{3.11}$$

where I_{size} is the input image size (50 px in this case) and T is the period for the Gabor filter-like pattern or twice the bee's displacement between frames.

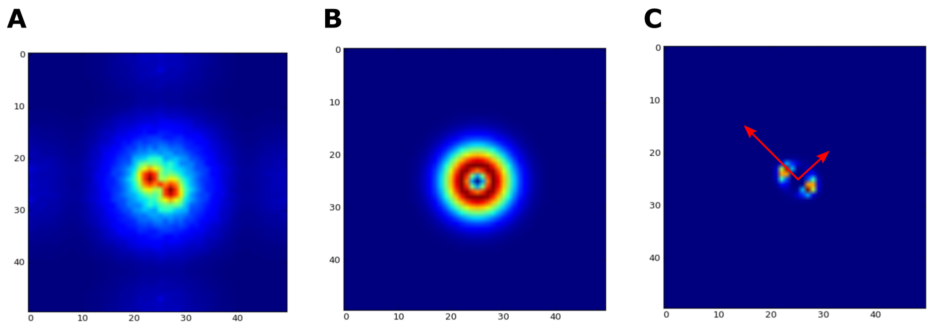


Figure 3.24: **Filtering cumulative sum of difference images in the Fourier space.** (**A**) The cumulative sum of the Fourier transformed difference images of a waggle run exhibit a strong pair of maxima in locations orthogonal to the dancer's orientation. (**B**) A *DoG* kernel of the Mexican hat type, properly adjusted to the waggle band, is used as a bandpass filter. (**C**) Bandpass filtering the cumulative sums emphasizes values within the frequencies of interest.

The orientation of the bees' lateral and forward movements is obtained from the resulting image through Principal Component Analysis (PCA). The principal component direction reflects the direction of the dancer's lateral movements, hence the bee's body axis must be orthogonal to the principal component. This axis represents two possible waggle directions. To disambiguate the alternatives, the *OM* processes the dot detector positions extracted by the *AM*. Each ordered pair in the waggle run metadata represents the average pixel position in which the *AM* detected brightness changes in the waggle band. In a typical waggle sequence, these points trace roughly the path of the dancer. The *OM* averages the first 10% ordered pairs of the waggle sequence and compute all *DD* positions relative to this average. Then it searches for the maximum values in the histogram of the orientation of all vectors and average their direction for a robust estimate of the main direction of the dot detector sequence. This direction is then used to disambiguate the two possible directions extracted during the *PCA*.

Mapping module (MM)

Each waggle dance encodes in its average waggle run duration and orientation a set of polar coordinates to a field location. The mapping module (*MM*) implements a series of steps to decode this coordinates and map the dances back to the field. The *MM* first reads the output of the *AM* and *OM*, essentially time, location, duration and orientation of each detected waggle run. Then, it clusters together waggle runs with a high probability of belonging to the same dance. To understand which waggle runs belong together, one can visualize a three-dimensional data space defined by the axes X and Y of the comb surface and a third axis T of time of occurrence. This way, each *WR* can be represented in the data space by (x, y, t) coordinates based on its comb location and time of occurrence (Fig. 3.25**A**). To maintain coherence between spatial and temporal values, the time of occurrence is represented in one-fourth of the seconds relative to the beginning of the day.

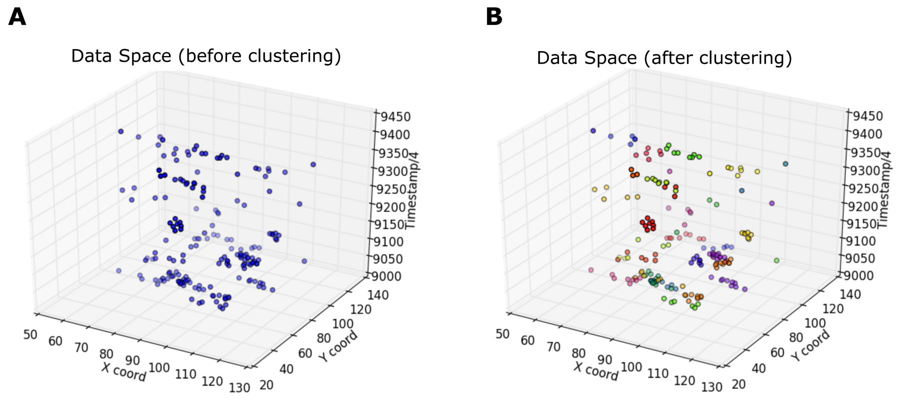


Figure 3.25: **Depiction of 200 waggle runs in the data space XYT**. (**A**) A set of 200 WRs plotted in the data space XYT, where values in the axes X and Y correspond to their comb location, and values in axis T to their time of occurrence. (**B**) Using an HAC approach, those WRs within a maximum Euclidean distance of dmax3 are clustered together and considered as belonging to the same dance.

The clustering process in the *MM*, just as with the *AM*, is carried out following a hierarchical agglomerative clustering (*HAC*) approach. An Euclidean distance d_{max3} is defined as a threshold for the clustering process. The value of d_{max3} is based on the average drift between *WRs* and the average time gap between consecutive *WRs* [62] (Fig. 3.26).

It is highly possible that during the clustering process either some waggle runs will end up in the wrong dance cluster or that some dances will be divided into multiple clusters. To deal with the latter issue, the *MM* only considers clusters with a minimum of 4 waggle runs as real dances, since the average values for a sample of 4 waggle runs from a dance is highly correlated with the mean of all its waggle runs [15]. The misclassification problem is solved using random sample consensus (*RANSAC*) [32], where the waggle runs mistakenly classified are perceived as outliers in the distribution of waggle run orientations. Waggle run duration and orientation are then averaged for all inliers and translated to field locations.

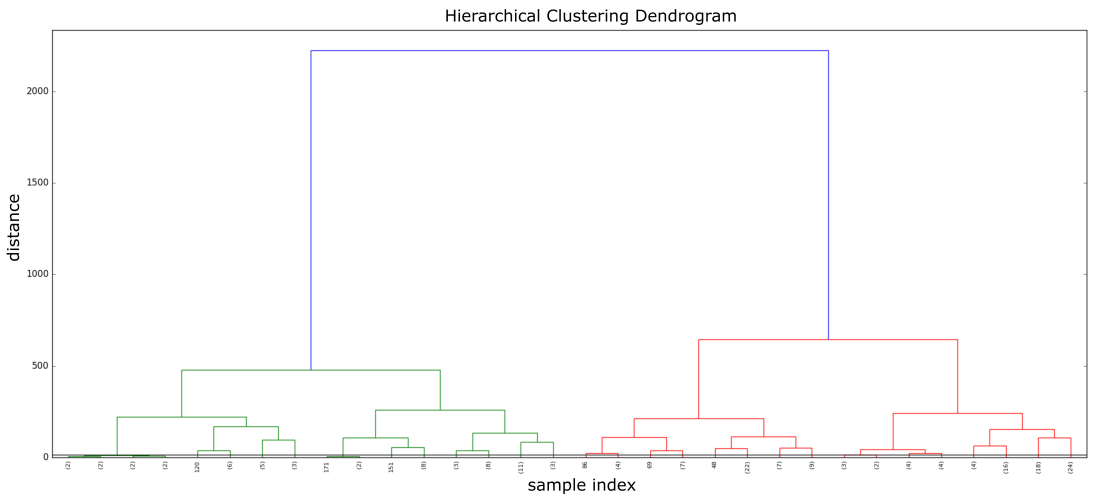


Figure 3.26: **Dendrogram for the hierarchical agglomerative clustering of 200 waggle runs**. The dendrogram provides a graphic representation of the HAC process. Each observation starts as its own cluster, at each iteration, the two closest clusters are merged, this process is performed iteratively till only one cluster remains. A maximum threshold distance d_{max3} is defined, and all clusters generated to this point are regarded as dances and their constituent elements as their waggle runs.

The mean waggle run duration is translated to meters using a conversion factor, and its orientation is translated to the field with reference to the azimuth at the time of the dance. The duration-to-distance factor is internally calibrated by each colony, and it depends on multiple factors. It is possible to approximate the value of this factor for a particular colony by training a group of their foragers to a known location and averaging the duration of all waggle runs signaling in its direction.

3.6 Chapter Overview

In this chapter were presented the components of the *BeesBook* system, a vision-based solution for the automatic data recollection in behavioral studies of bee colonies. The overall system can be seen as integrated by two functionally independent units that operate simultaneously. The first system comprises the automatic recording and storage of high-resolution images, computer vision software for identifying uniquely marked bees, and post-processing software to trace their trajectories. The second subsystem comprises image analysis software for detecting waggle dances on real time and post-processing software to decode the dances and map their coordinates back to the field. *BeesBook* has been conceived as a budget-priced framework for the incremental development of software and hardware components and its modular design facilitates the continuous improvement of the already existing components without disrupting the rest of the system.

Each component of *BeesBook* was carefully selected to maximize the quality and significance of the collected data. A marker that adapts to the physiology of the bee without limiting her everyday activity, and allows to uniquely identify all colony members (up to 4096 individuals at a time). A setup that allows video recording the activity of the hive 24 hours a day during extended periods of time without introducing any noticeable source of discomfort to the colony that could affect their natural behavior. To reduce costs, rather than using high end cameras as state of the art systems do, *BeesBook* relies on the implementation of powerful image analysis software.

All the elements of the system have been described thoroughly and in the case of the software elements links are provided to the repositories of the source code.

In the following chapter I present the experimental design for the three recording seasons (summers of 2014, 2015, and 2016) and discuss the performance of the system with the generated raw image data.

Chapter 4

Experimental Validation

This chapter discusses the experimental validation of the *BeesBook* system. The data here analyzed was collected during three consecutive summers (2014 - 2016). The first part of the chapter covers the experimental design of the recording seasons and the implementation of the image analysis solution on a Massive Parallel Processing (MPP) system. Due to the significant amount of media data gathered during each summer ($\sim$ 65 M images), its analysis had to be carried out in one of the complexes of the North-German Supercomputing Alliance (Norddeutscher Verbund zur Förderung des Hoch- und Hochstleistungsrechnens HLRN), requiring the development of a customized tool to batch analyze the recorded images. Int the second part of this chapter it is discussed the performance of the image analysis and post-processing modules on the images collected during the recording seasons. Finally, in the last section, it is discussed the performance of the waggle dance automatic decoder.

The results presented in section 4.3.1 have been previously published in *"Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees"* [118], published by *Frontiers in Ecology and Evolution.* Some of the data in section 4.4 has been partly used for the research article *"Novel technological and methodological tools for the understanding of collective behaviors"* [10] submitted to *Frontiers in Robotics and AI.* The results reported in section 4.5 have been used in the preparation of the article *"Automatic detection and decoding of honey bee waggle dances"* [119], accepted for publication in PLOS One.

4.1 Experimental Design of Recording Seasons

To evaluate and improve the performance of the different components of the *BeesBook* system, three recording seasons were conducted from 2014 to 2016. The recordings were conducted exclusively during summer to capture the colony's activity at the peak of its yearly cycle. Life expectancy of bees during summer ranges from 30 to 60 days [68]. Hence, each season was programed to last $\sim$ 60 days to document at least one full cycle of renovation of the colony's population. Each season represents a significant improvement over the previous one, both for the experimental design and the components of the system.

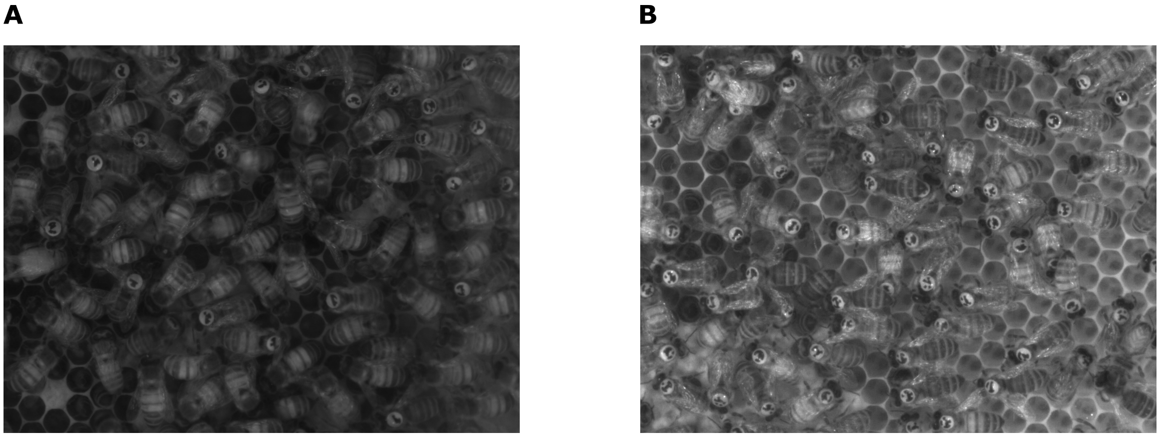


Figure 4.1: **Comparative of images obtained from first and last season.** (**A**)Season 2014. (**B**) Season 2016.

In Fig. 4.1 are shown side by side details of images captured during (**A**) season 2014 and (**B**) season 2016. Although both images were recorded by the middle of the season, there are some clear differences between them. For a start, Fig. 4.1**B** seems brighter and sharper than Fig. 4.1**A**. Also, while all bees in Fig. 4.1(B) seem to be marked, this is not the case for Fig. 4.1**A**. Many lessons had to be learned during seasons 2014 and 2015 to achieve the quality of image observed in Fig. 4.1**B**, the details are given in the following paragraphs.

Season 2014

The first recording season was conducted during the summer of 2014 (from July 24 to September 25, 2014 - 63 full days). The process started weeks before the recordings with the installation of a standard observation hive containing an unmarked colony. The observation hive was connected to the exterior through one of its entrances to allow bees to fly in and out and get accustomed to the environment. The bees were marked previous to the start of the video recording with help of members of the Biorobotics Lab over a period of three days. The panels of the observation hive were prepared in advance to allow extracting the bees without opening the cabinet (Fig. 4.2**A**). Batches of approximately 50 bees were extracted from the original hive into containers with a vacuum cleaner (Fig. 4.2**A**,**B**) and were distributed to the collaborators to be marked.

Each bee was marked individually, first the bee is put in a marking cage, then hair is removed from the thorax and shellac applied onto it to fixate the tag; finally, the tag is attached with the white semi-circle rotated toward the bee's head (Fig. 4.3**A**). Once all

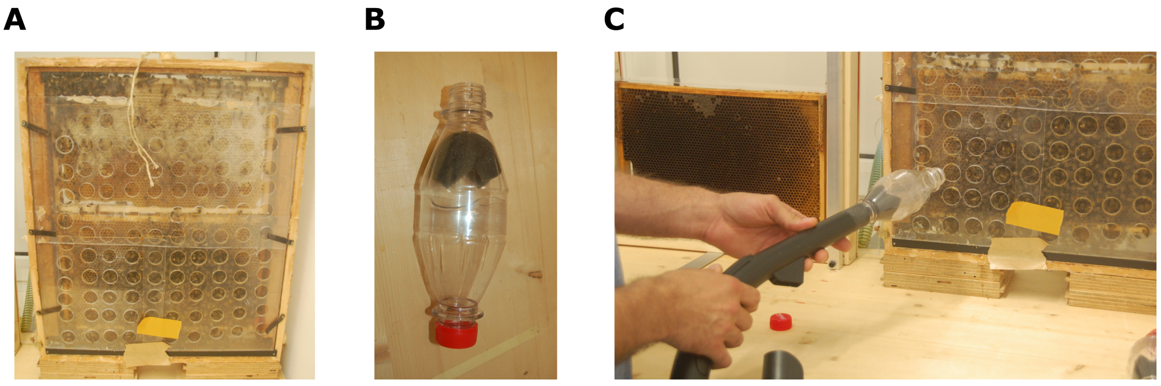


Figure 4.2: **Extracting bees from original observation hive.** (**A**) The panels of the observation hive containing the unmarked colony were perforated and covered with a second layer of panels. The second layer could be re-arranged to open just one of the holes at a time and allow extracting bees from that region of the hive. (**B**) Bees were extracted from the original hive into containers (**C**) using a vacuum cleaner.

bees in the batch were marked, they were poured at the hive entrance (now connected to the customized observation hive) (Fig. 4.3**B**). This process was repeated till all bees in the original hive had been marked (A total of $\sim 2,000$ bees were marked). The queen was marked in the same way as the workers, but she was introduced to the observation hive through the other entrance (on the inside of the room) (Fig. 4.3**C**).

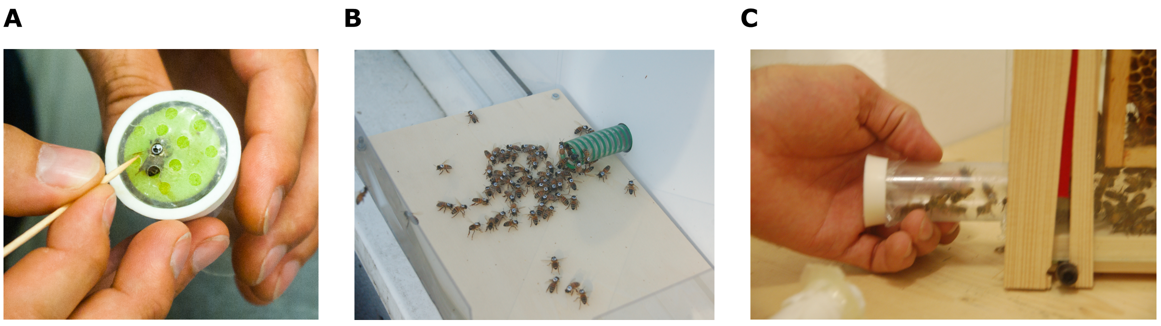


Figure 4.3: **Marking procedure.** (**A**) Each bee was marked individually using a marking cage. (**B**) Marked bees were placed on a platform at the entrance of the observation hive (on the outside of the building). (**C**) The queen was also marked but introduced to the hive through its second entrance (on the inside of the building).

The season 2014 was intended as a proof of concept, to test and adjust protocols of the experimental design and components of the setup; it would also provide the first batch of images under real circumstances to put to the test the image analysis modules. The structure used during season 2014 (Fig. 4.4) did not have many of the features projected for the final design; the lighting setup consisted of only six IR lamps per side, and the cameras could just adopt a very limited number of positions.

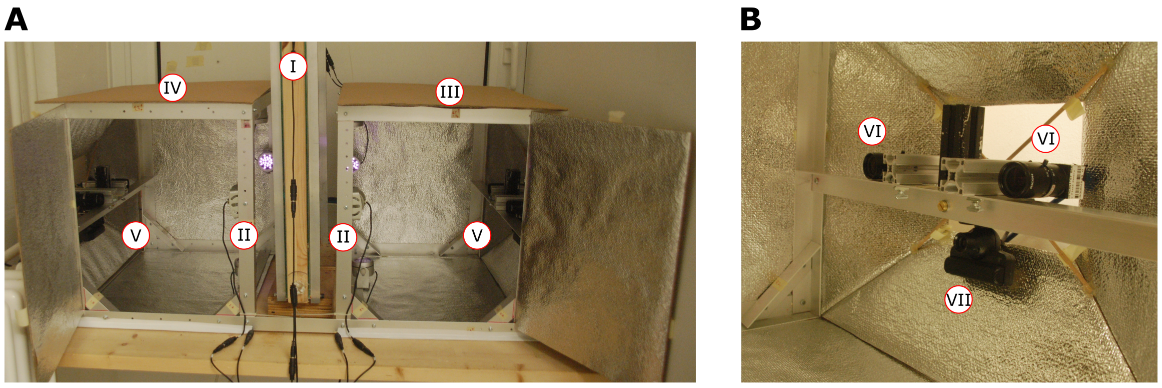


Figure 4.4: **Experimental setup season 2014.** (**A**) (I) Observation hive, (II) infra-red lamps, (III) right side of observation cage, (IV) left side of observation cage, and (V) camera array. (**B**) Detailed view of the camera array: (**VI**) Flea3 cameras and (**VII**) PS3Eye webcam.

Another particularity of season 2014 was that the same brood comb was used throughout the whole season. Worker honey bees take 21 days to emerge from their cells. Hence, three weeks after the start of the recording season new unmarked bees started to hatch. To maintain a significant population of the colony marked, young bees were raised in an incubator, marked and introduced to the observation hive. Following this procedure, a total of 441 bees were added to the colony over a period of 30 days. By the end of the season all of the first generation of marked bees, except for the queen, had already died and most of the colony members were unmarked.

Season 2015

Some minor but important changes were made to the experimental design for season 2015. The recording season lasted seven weeks (70 days), from August 18 to October 26, 2015. The colony from 2014 survived the winter and was used also for this season. A total of 1953 colony members were marked previous to the beginning of the recordings following the same protocol from 2014. Also this season young bees raised in incubator were marked and introduced to the observation hive, a total of 822 bees over a period of 49 days. This year, a register with the marker ID and hatching day was kept for the incubated bees. This way, the age of incubated bees was known throughout all season (not so for original members of the colony). To avoid young, unmarked bees hatching within the observation hive, the brood comb was replaced for a honeycomb every three weeks. Disregarding those bees who loosed their tag, season 2015 had an entirely marked colony from beginning to end. The most noticeable changes for season 2015 were done to the recording setup, which was completely renewed to adopt the originally projected design described in section 3.1.3 (see Fig. 4.5). The new lighting setup with 22 IR lamps per side enabled the acquisition of brighter and sharper images.

Season 2016

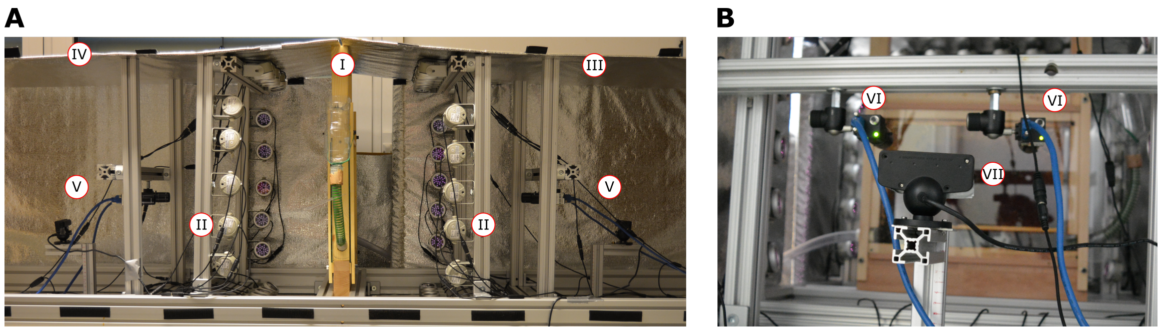


Figure 4.5: **Experimental setup seasons 2015-2016.** (**A**) (I) Observation hive, (II) infra-red lamps, (III) right side of observation cage, (IV) left side of observation cage, and (V) camera array. (**B**) Detailed view of the camera array: (**VI**) Flea3 cameras and (**VII**) PS3Eye webcam.

During season 2016 the marking of the bees was conducted differently. The tagging period began three weeks previous to the recording season (June 28), and only young bees raised in incubator were marked. Each day a batch of bees was taken from the incubator, bees were tagged individually, and introduced to the hive. A record of the tags used during the marking sessions was maintained to keep track of the bees' age. The recording season began on July 19, already with 1424 marked bees in the hive, and lasted 63 days. Young bees were marked and introduced to the hive on a daily basis till two weeks before the end of the recording season. By the end of the tagging period, a total of 3,191 bees had been marked and added to the observation hive. As in 2015, this season the brood comb was also exchanged to prevent unmarked bees to hatch within the observation hive. Regarding the recording setup, it practically remained the same as in 2015; only the illumination setup was modified (see section 3.1.3).

The marking protocol from season 2016 has two advantages over those implemented in 2014 and 2015. First, it does not require handling adult bees; young bees are easier to handle since they still do not fly and rarely sting. But most important is the fact that this protocol allows the system to compute the age of each colony member at any point in time, opening the possibility to conduct new and more significant studies with the collected data.

4.2 Highly Parallelized Image Processing on Cray Supercomputer

Each recording season produces about 150 TB of data in images (see section 3.2). The image processing reduces the raw image data to just a few hundred Gigabytes in binary files (see 3.3), but the computing time required for this task results prohibitive, even if performed on a high performance desktop. Considering an average computing time of one second per image, the analysis of a whole season would take more than two years. To speed up this process, the analysis had to be parallelized.

From the early planning stage of *BeesBook*, computing and storage resources were granted to the project by the North German Supercomputing Alliance (HLRN). The HLRN's provide the project with access to their Cray XC30 supercomputer housed in the Zuse Institute Berlin (ZIB), which is located just a few meters from the site of the recordings and offers gigabit ethernet connectivity. The Cray XC30 features 1872 compute nodes with 24 CPU cores. The system has 117 TiB of RAM and 4.2 Petabyte of hard disk space, organized as RAID 6. Since the recording seasons' data is already stored on the supercomputer's file system, using it to process the images circumvents additional data transfers.

The Cray XC30 supercomputer works under a Single Program Multiple Data (SPMD) approach, which means that the same program can be executed in multiple instances, working on separate data. In the particular case of *BeesBook*, this means that multiple instances of the image analysis module can be run in parallel, with each instance accessing to an exclusive batch of images.

To access the compute nodes of the Cray XC30, a batch script has to be submitted to the job queue. The batch script contains information about the resources requested (number of nodes, wall time, modules, etc...) and a call to the binary. Once the script has been submitted, is up to the system scheduler to decide when to execute the job. Many parameters are taken into account to make this decision, but the number of nodes and time wall are the most important, the fewer resources are requested, the sooner the script is run[1]. Therefore, the correct strategy to accelerate the image analysis on the Cray XC30 is to submit numerous jobs with small requirements. Manually submitting the jobs and keeping track of their status is time-consuming, to automate this task, an *observer* program was implemented[2] [3]. The script is divided into three modules, namely a context container, a job queue manager, and an image provider.

The ***BeesBook* context module** is a container for all the configuration constants, among them directory paths, job properties, average computing time per image. These constants are used by the other two modules and during the initialization of the analysis when the batch scripts are automatically generated.

The **job queue manager module** manipulates the internal job queue and implements all related functions, including submitting jobs, checking the status of submitted jobs, moving the results from a job directory to the global results directory and recover the job queue if the process is unexpectedly interrupted.

The **image provider module** extracts the images from their containers to the work-

[1]Full documentation of the process in the HLRN website: https://www.hlrn.de/home/view/System3/WebHome.

[2]Code and documentation available online: https://git.imp.fu-berlin.de/biorob oticslab/bb_observer.

[3]The final version of this tool is based on the one proposed by Simon Wichmann as part of his master's thesis in the biorobotics lab [122]

ing file system from where they are distributed to the jobs' directories to be submitted. As a way to keep a record of the images that have already been processed by a running job, these are deleted from its internal directory. Once the job has been terminated by the system scheduler, all images remaining in its internal directory are collected by the image provider module and reassigned to new jobs.

When the *observer* is initialized, it first creates a directory structure (see Fig. 4.6) with all necessary files to keep track of the production. The *Job Script Slot* files are the batch scripts required to submit jobs to the compute nodes. They are generated by the *observer* when initializing the work directory. The *Image Heap* contains images ready to be assigned to a slot. The *Job Outputs* directory contains the command line output of the batch jobs (e.g., for error reporting). The *Results* directory contains the result files of the image analysis. The *Job Slots* directory contains the image data for the batch jobs. Each *Slot* directory contains one *Proc* directory for each process in the batch job. The image blocks are provided here for their respective job. Each HLRN account has a limited number of batch jobs (**N**) that can be either in the queue or running. The observer creates one extra script and slots directory that will be ready to be submitted as soon as a slot opens.

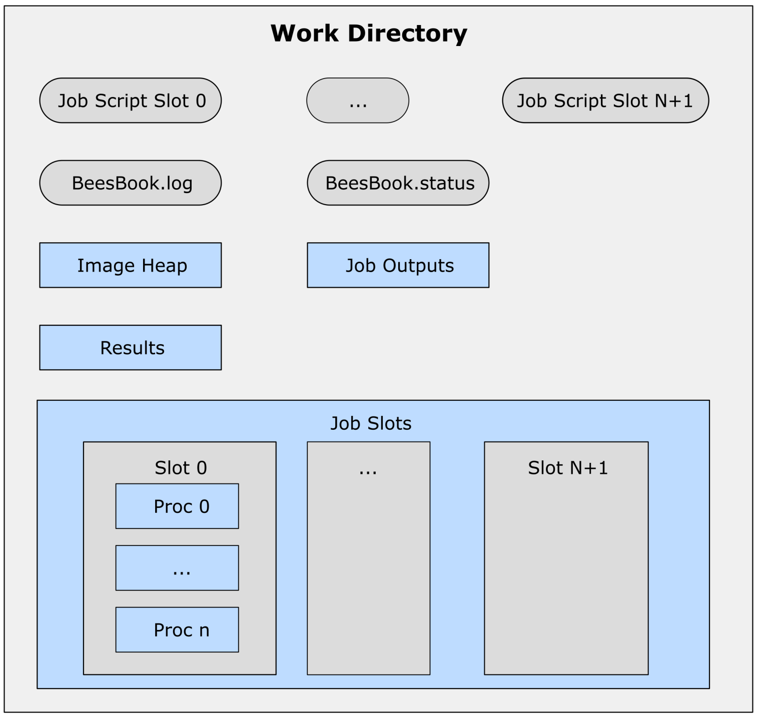


Figure 4.6: **Work directory structure.** The Observer's work directory is used to maintain a job queue and the image blocks provided for the separate jobs. Files are depicted as round boxes and directories as rectangular boxes.

The major functionality of the observer program is depicted in Fig. 4.7. First the *observer* initializes the directory structure or recoveries from an interrupted process, in which case cleans the Slots directories and submits new jobs. Then, while there are images left to process, the image provider module prepares a slot directory. Then, when a slot opens, it submits the new job, collecting before results files if necessary. This process is repeated iteratively till all images have been processed and all results collected.

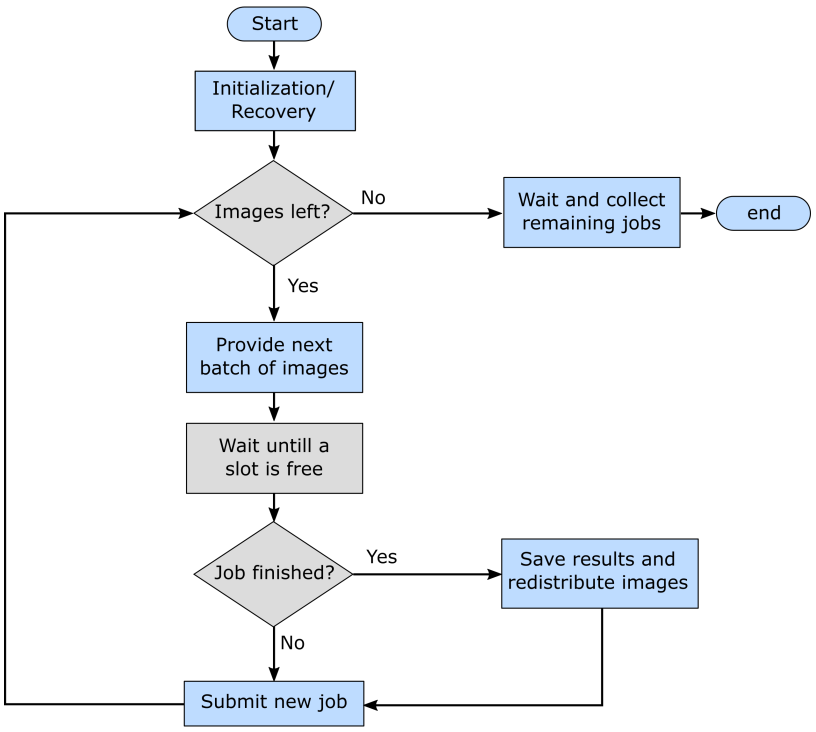

Start
Initialization/ Recovery
Images left?
No
Wait and collect remaining jobs
end
Yes
Provide next batch of images
Wait untill a slot is free
Job finished?
Yes
Save results and redistribute images
No
Submit new job

Figure 4.7: **Main routine of the external observer.** The observer keeps the queue and slots directories full till all images have been processed, after what it waits till the last job is completed to collect the latest batch of results.

4.3 Image Analysis Performance

This section presents the performance evaluation of the two implemented pipelines, namely CV-Pipeline and DCNN-Pipeline. The *BeesBook* system was initially conceived with a computer vision pipeline carrying out the tasks of the image analysis module, and many of the design decisions (particularly the marker design) were made keeping this approach in mind. The analysis of real data generated during the first experimental season (2014) showed that, although the CV-Pipeline is a robust approach to the image analysis problem [118], there is room for improvement, especially in the bit-reading stage. For this reason, a new approach, based on DCNN was proposed. Presenting the results of both approaches side by side highlights the value of *BeesBook's* modular design, that did not require changes to the rest of its structure to exchange the CV-Pipeline for the better performing DCNN-Pipeline.

Ground truth

To validate the performance of the image analysis module, a ground truth data set was generated using a custom user interface[4] (see Fig. 4.8). The user interface allows to manually fit a three-dimensional model of a tag, with control of the size, position, and rotation in three axes. The interface also allows setting the value of each of the 12 code cells to black, white or undetermined. To minimize the user bias, the data set was generated with the help of 13 coworkers. Using this process, it took approximately 39 hours to label 26 images, what amounts to a total of 2,808 tags.

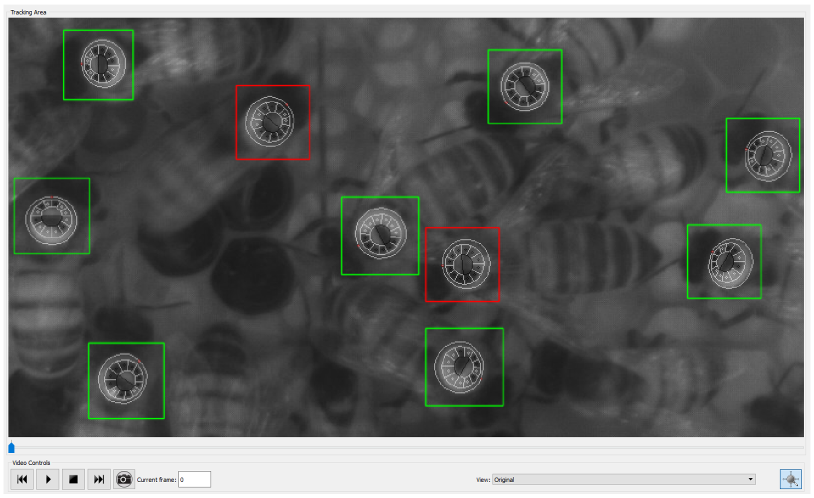
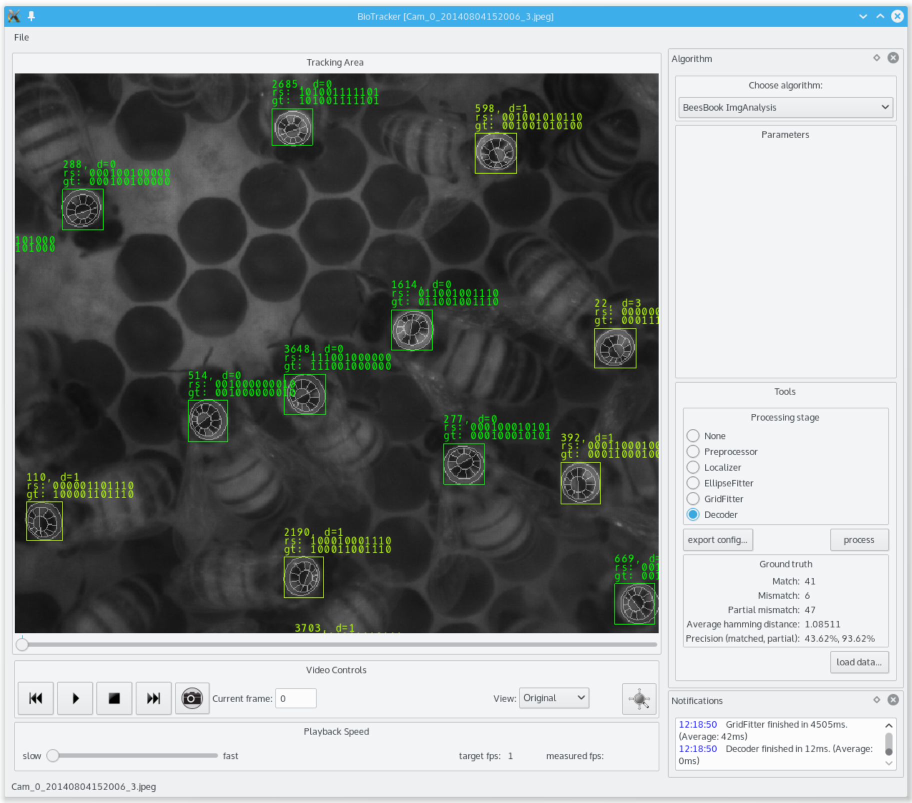


Figure 4.8: **A GUI was used to generate ground truth data manually.** A custom user interface allows setting new tags and modify all its parameters via mouse interaction.

4.3.1 CV-Pipeline Performance

The GUI used to generate the ground truth data was also enabled to read and display the output from the pipeline. This way one could see what makes a marker hard to be detected or decoded and modify the pipeline in function of this information.

[4]This utility was implemented as a module of the biotracker framework, developed in the biorobotics lab. A repository is available online: https://github.com/BioroboticsLab/biotracker_beesbook.



Current frame: 0

View: Original

target fps: 1 measured fps:

Cam_0_20140804152006_3.jpeg

BeesBook ImgAnalysis

None
Preprocessor
Localizer
EllipseFitter
GridFitter
[✓ Decoder]

Ground truth
Match: 41
Mismatch: 6
Partial mismatch: 47
Average hamming distance: 1.08511
Precision (matched, partial): 43.62%, 93.62%

12:18:50 GridFitter finished in 4505ms. (Average: 42ms)
12:18:50 Decoder finished in 12ms. (Average: 0ms)

Figure 4.9: **Validation of pipeline results with GUI.** The user interface developed to generate ground truth can also be used to explore the pipeline output.

The performance of each layer of the CV-Pipeline was evaluated individually. As measures of performance were chosen recall and precision. In the context of a binary classification problem, recall is defined as the proportion of real positives that are correctly predicted positive, and can be expressed as:

$$Recall = \frac{t_p}{t_p + f_n}, \tag{4.1}$$

where t_p is the number of correctly predicted positives (or true positives) and f_n is the number of unpredicted positives (or false negatives). In a similar way, precision is defined as the proportion of predicted positives that are real positives, and can be expressed as:

$$Precision = \frac{t_p}{t_p + f_p}, \tag{4.2}$$

where f_p is the number of erroneously predicted positives (or false positives). A result generated by a pipeline layer is considered as a true positive (t_p) if it exists in the ground truth data set, otherwise is regarded as a false positive (f_p); the elements in the ground truth data set that are not predicted by the layer are considered as false negatives (f_n).

All four evaluated layers (localizer, ellipse fitter, grid fitter, and decoder) scored high values of recall and precision (see Tab. 4.1). Almost 90% of the tags are detected by the localizer. The ellipse fitter performance is outstanding, finding the correct parameters for 97% of the identified markers that represents 88% of the ground truth set. The grid fitter estimates the parameters correctly for 88% of the tags. Finally, although only 66% of the detections can be decoded successfully, up to 94% of the decoded IDs have less than three misidentified bits. These detections are still valuable as their correct ID can be retrieved using the tracking module. As an additional way to measure the decoder's accuracy, the Hamming distance [43] was also computed with an average value of 1.08.

	Recall	**Precision**
Localizer	0.90	0.84
Ellipse fitter	0.88	0.97
Grid fitter	0.88	0.89
Decoder		0.66/0.94
		(0/<3 hamming distance)

Table 4.1: **CV-Pipeline performance.** Recall and precision values for the CV-Pipeline layers.

4.3.2 DCNN-Pipeline Performance

During training, a neural network (NN) learns to minimize its *loss*, a summation of the error made for each example in training set. Ideally, a model will decrease its *loss* with each iteration; the lower the *loss*, the better a model. Therefore, *loss* can be seen as a metric for performance. As a matter of fact, customary metrics to report the performance of a NN are: the loss and accuracy (the percentage of correctly classified examples) of the model with the test set.

Training and validation losses are here reported as a metric for performance of the two models that comprise the DCNN-Pipeline. In addition to the loss, and to be able to compare the performance of the DCNN-Pipeline with that of the CV-Pipeline, four more metrics were computed: Recall and precision for the localizer network, and mean Hamming distance and accuracy for the decoder network.

Training and performance of the localizer network

For the localizer network, several millions of samples were generated with the process described in section 3.3.2. The 100×100 px image regions were downsampled to a a size of 32×32 px using bilinear interpolation; this increases the speed of the final model and helps to avoid overfitting. The training set consists of 17,956 samples from which 10% is used for testing and 10% for validation. The network was trained to minimize the cross entropy between its output and the training labels using stochastic gradient descent (SGD) with back propagation [63]. The network is trained for a fixed model number of 100 epochs after which the loss does not decrease significantly anymore (see Fig. 4.10).

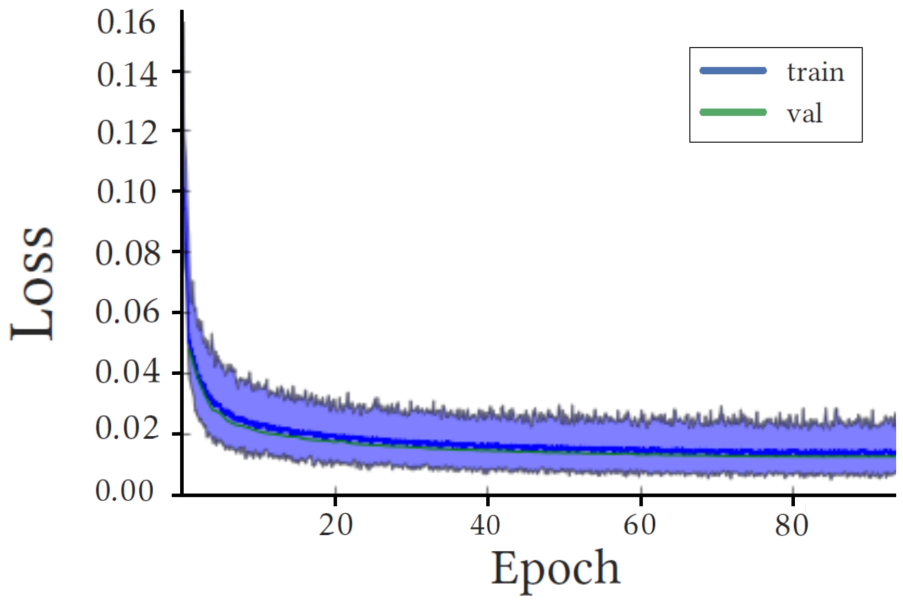


Figure 4.10: **Training and validation losses of localizer.** Binary cross entropy between outputs and training labels is used as loss.

Training and performance of the decoder network

For the decoder network, the training set was generated using the RenderGAN framework [102], each epoch consisting of 1000 batches with 128 samples. Early stopping was used to select the best parameters of the network. As a validation set was used the manually generated ground truth data, with a total of 66,000 samples. The loss for the decoder network is the binary cross entropy between the true values of the individuals bits and the mean squared error between the true orientations and its predictions. As with the localizer network, the training was done using stochastic gradient descent (SGD) with back propagation [63]. The training and validation losses of the decoder network are plotted in Fig. 4.11.

Both models are validated with the manually generated ground truth data from season 2014. The overall performance of the Pipelines is compared using five different metrics: Recall, precision, mean Hamming distance (MHD), accuracy of the decoded IDs, and computing time per tag. As Table 4.2, shows, the DCNN-Pipeline clearly out-

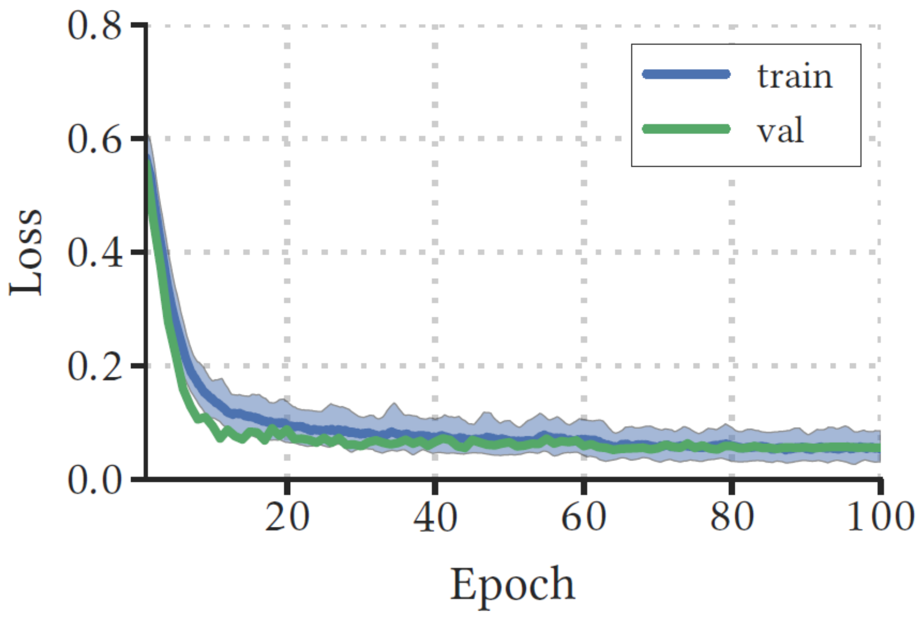


Figure 4.11: **Training and validation losses of decoder.** Binary cross entropy for the bits and the mean square error for the orientations are used as loss.

performs its counterpart in accuracy. Furthermore, the computing time required by the DCNN-Pipeline is less than one-hundredth the time of its counterpart.

	CV-Pipeline	**DCNN-Pipeline**
Recall (Localizer)	0.90	0.98
Precision (Localizer)	0.84	0.99
MHD (Decoder)	1.08	0.17
Id Accuracy	0.66	0.87
Time per tag (ms)	177.54	1.43

Table 4.2: **Pipelines performance comparison.** The DCNN-Pipeline clearly outperforms the CV-Pipeline in all five metrics, yielding higher accuracy values and consuming less computing resources.

4.4 Tracking and Temporal ID Filter Performance

This section presents the performance evaluation of the tracking module described in section 3.4. The module implemented in python and is available for download from its github repository (https://github.com/BioroboticsLab/bb_tracking). The module uses the random forest classifier available in python's scikit learn library[5]. To train the classifier two ground truth data sets (**2015.1** and **2015.2**) were manually defined using a custom user interface that allows marking the position of an animal and defining its ID [6].

Ground truth data sets

The data set **2015.1** comprises two sequences of images. The first one obtained from camera 0, starting at 11:36, and lasting 100 frames; the second was obtained from camera 1, starts at 04:51, and extends over 101 frames (more derails in Tab. 4.3). The data set **2015.2** consists of a single set of images from camera 0, starting at 13:36, and extending over 200 fps. To avoid overfitting, the data sets were chosen to contain both high activity (around noon) and almost no activity (in the early morning hours).

The random forest classifier was trained and validated with the data set **2015.1** using stratified split. The data set **2015.2** was used for testing. Hyperparameters of the random forest classifier were tuned using hyperopt-sklearn (https://github.com/hyperopt/hyperopt-sklearn)

Data set	Date	Time	Frames	Detections	F-Positives	Ind.
2015.1	18.09.2015	11:36 & 04:51	201 (3 fps)	18,085	222 (1.23 %)	144
2015.2	22.09.2015	13:36	200 (3 fps)	10,945	82 (0.75 %)	98

Table 4.3: **Ground truth data sets.** The number of detections corresponds to number of tags localized and decoded by the DCNN-Pipeline over all frames in the data set.The number of false positives (F-Positives) shows how many times the deep learning pipeline detects a detection when there is none. The number of individuals (Ind.) indicates how many different bees are present in the data set.

Performance

After training and tuning of the classifier, the data set **2015.2** is processed using the tracking module. Two iterations are performed, the first with a gap of 0 frames and the second with a maximal gap of 14 frames. To compare the quality of the data before and after being processed with the tracking module, four *pre-tracking* properties are computed for the data set:

- **track_ids_correct_truth**: Proportion of ground truth tracks with correctly predicted ID.

[5]Scikit learn library: http://scikit-learn.org

[6]The tracking software is available to download from github: https://github.com/BioroboticsLab/bb_tracking

- **detection_ids_correct_truth**: ID averaging on ground truth tracks, proportion of detections with correctly predicted ID.
- **detection_ids_correct_default**: Proportion of correct detections after thresholding each bit.
- **tracks_in_scope**: Relative number of tracks that in the best case could be found given a gap size 0 or 14, for tracking step 1 or 2, respectively.

For the data set **2015.2**, only 83% of all detections entering the tracking module have a correct ID (*detection_ids_correct_default*). Given a perfect tracking algorithm, averaging individual bits would assign correct IDs to 92% of all tracks (*track_ids_correct_truth*) and more than 99% of all detections (*detection_ids_correct_truth*). During the first iteration only 35% of all possible tracks could be generated (tracks with no gaps). Around 78% of all tracks have gaps of up to 14 frames (tracks_in_scope).

To measure the effect that each new iteration has in the data set, eleven *post-tracking* properties are computed after each one of them:

- **fragment_ids_correct_tracking**: Proportion of fragments with correctly predicted ID, refers to all fragments obtained from respective tracking step.
- **detection_ids_correct_tracking**: Proportion of detections with correctly predicted ID after ID averaging on fragments found in respective tracking step.
- **tracks_complete**: Number of correctly assembled tracks, relative to *tracks_in_scope.*
- **track_fragments_complete**: Number of correctly assembled fragments (tracklets with 0 or 14 gaps, respectively).
- **track_detections_complete**: Post-tracking, number of correctly associated detections.
- **fragments_inserts**: Number of incorrectly inserted fragments (instead of gap).
- **detections_inserts**: Number of incorrectly inserted detections (instead of gap).
- **fragments_id_mismatches**: Number of incorrectly inserted fragments (instead of other fragments).
- **detections_id_mismatches**: Number of incorrectly inserted detections (instead of other detections).
- **fragments_deletes**: Number of incorrectly missing fragments.
- **detections_deletes**: Number of incorrectly missing detections.

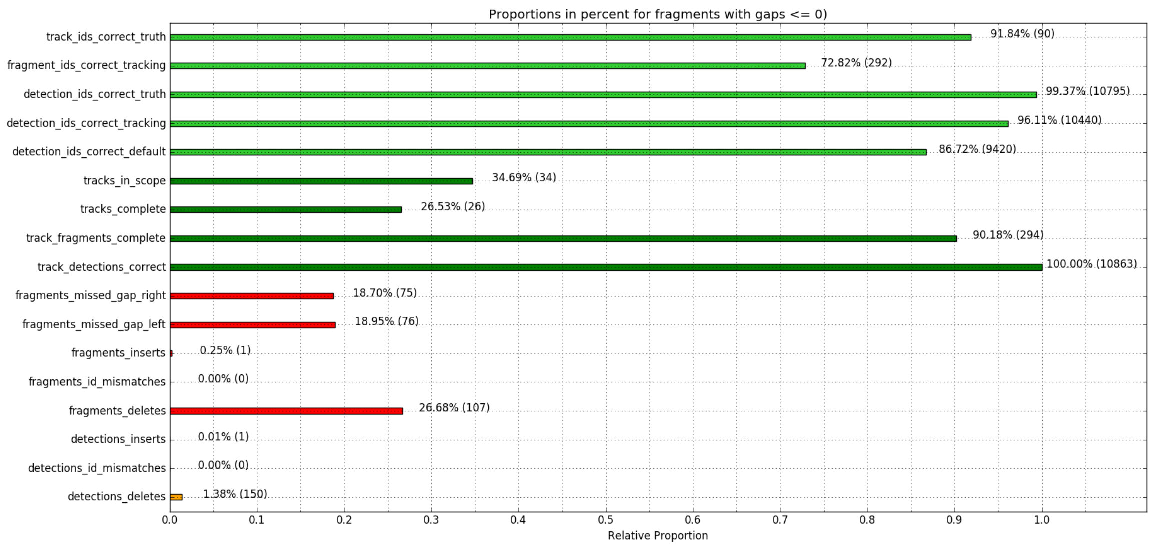


Figure 4.12: **Results from first iteration.** While many detections were correctly identified during the first iteration, the fragments were generally shorter and only a small fraction of detections were be assigned to complete fragments

The ID correctness for the tracks as well as for individual detections is measured after each iteration. The first tracking iteration produces 292 correctly identified fragments (73%); The second iteration yields 112 fragments with correct ID, since these are longer fragments that include some fragments from the previous iteration, this accounts for 82% of the fragments (*fragment_ids_correct_tracking*). The remaining incorrect fragments are only a small part of the detections this due to their short length.

While the first iteration yields 96% correctly identified detections, the second iteration assigns correct IDs to 98% of the detections (*detection_ids_correct_tracking*), a small increase in accuracy that has a significant impact on the completeness of the tracks. Only 26% of the theoretically possible fragments (*relative to tracks_in_scope*) were correctly found after the first iteration, a second iteration improves finds 69% of the tracks with no error (*tracks_complete*).

Incomplete or incorrect fragments might exhibit deletions (missing detections), mismatches (incorrect detection replaces the correct one) or inserts (incorrect detection instead of a gap). After both iterations only one of the fragments contains an insert (*fragment_inserts*) of only one detection (*detection_inserts*). No mismatches were introduced in the test set (*fragments_id_mismatches* and *detections_id_mismatches*). After the first iteration were found 107 fragments (27%) with deletions and 25 fragments (18%) were found after the second iteration (*fragments_deletes*). These deletions, however, correspond to only $\sim 2\%$ of the whole set (*detections_deletes*).

After the tracking and the median-ID assignment, more than 98% of all detections obtain the correct ID. This represents an improvement of 15% over the original 83% returned by the DCNN-pipeline.

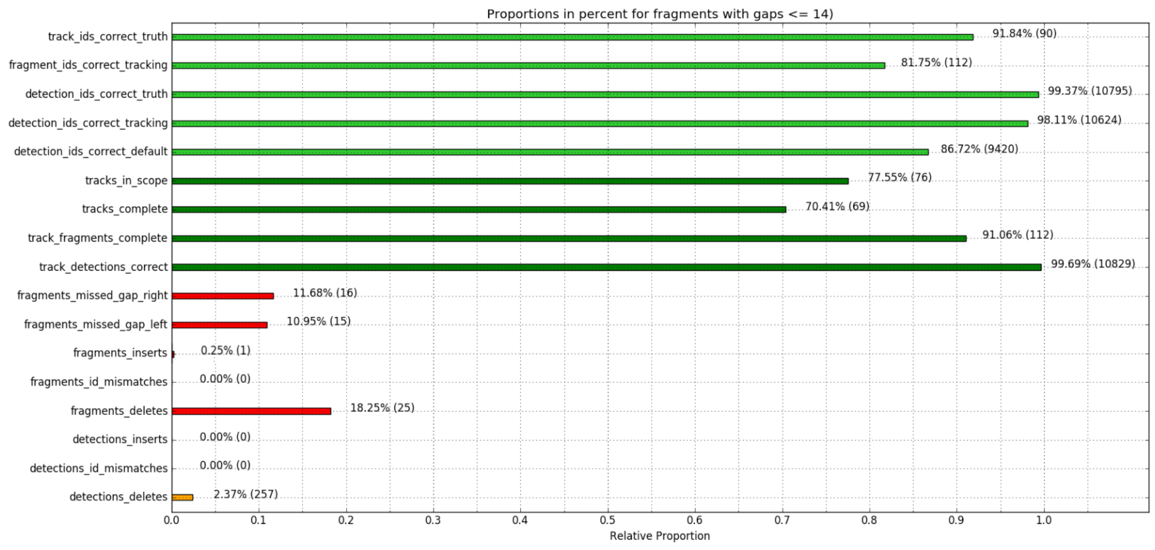


Figure 4.13: **Results from second iteration.** The second iteration yields longer and more complete fragments. More than 98% of the detections were correctly identified

4.5 Automatic Waggle Dance Decoder

The working principle behind the automatic waggle dance decoder is the detection of pixel clusters pulsating at a characteristic frequency. This oscillation of intensity is observed in pixels that capture dancing bees during the waggle run. For this phenomenon to occur, it suffices that the bee body is easily discriminable from the background. The performance of the waggle dance decoder is not affected by the modifications made to the experimental design or the illumination setup throughout the recording seasons. Therefore, the results discussed in this section can be considered valid for any of the recording seasons.

Accuracy of attention module

The attention module (*AM*) is the only module of the waggle dance decoder subsystem running in real time. It was implemented in C++ and uses the OpenCV library. The hardware requirements are reasonably modest; during recording seasons both PS3Eye cameras were connected to the same computer over USB 2.0 interface at two separate USB controllers. The computer had a CPU Intel i7-26000 with four cores @ 3.4GHz and 4 GB of RAM. Both processes combined consume less than 300 MB of RAM, and depending on the sensitivity settings and the liveliness of the colony, it produces about 1GB of data per day.

To evaluate the WR duration accuracy of the waggle dance decoder, the AM was executed on a set of video sequences containing a total of 199 waggle runs [87]. The results obtained by the AM were compared then to manually defined values. The duration of the detected sequences and human-generated values was translated from number of frames into time units using the frame rate. Two different kind of errors were computed, the signed gap for the first and last frame, and the absolute difference for the WR duration. It was observed an overall delay in the detections made by the AM. On average, the AM shows an average delay of 110.75 ms (SD = 110.92) for the start time of the WR, and M = 208.54 (SD = 79.03) for the end. This translates into an overestimation of the WR duration of M = 98 ms (SD = 139 ms).

Accuracy of the orientation module

The accuracy of the orientation module *OM* is evaluated with a test set of 200 waggle run sequences. First, the orientation for each waggle run in the test set was manually defined by a group of eight observers using a custom user interface. Using the user interface, each observer traces the line that best fits the dancer's body during the waggle run. The average of the eight manually extracted angles is considered as the reference value (ground-truth), and the distribution of values among the user's observations is noted as a reference for acceptable performance of the OM. The human-generated data showed a normal distribution with a SD = 6.66° (Fig. 4.14A). Running the mapping module (MM) on the test set a total of 53 dances were detected, only 23 of which consisted of 4 or more waggle runs. An average angle was computed for each waggle dance and each user; as expected, the standard deviation for manually extracted angles is smaller (SD = 3.67°) when computed at the dance level (Fig. 4.14B).

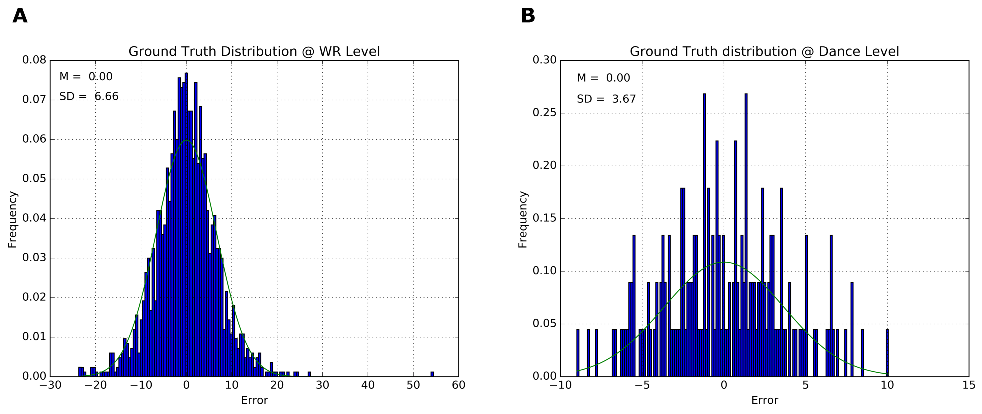


Figure 4.14: **Distribution of the manually extracted angles for a 200 waggle runs test set.** A group of eight observers extracted manually the angle of 200 WRs. (**A**) The data shows a normal distribution with a SD = 6.66° at the WR level and (**B**) a SD = 3.67° at the dance level.

The 200 WRs in the test set were analyzed using the OM and the results compared to the ground truth values defined by the human observers. Although the mean error of the values obtained by the OM is relatively low (M = –5.18°), a closer examination of the data reveals that ∼ 10% of the predicted values have an error close to ±180° (Fig. 4.15A). These outliers are a direct consequence of the method used by the OM to disambiguate the direction of the WR. Recalling section 3.5.2, the method used by the OM to disambiguate between WR orientations is based on the locations of the dancer during the WR. Although this method works for most of the cases, has some difficulties with short WRs. Ignoring the outliers, the error distribution of the OM (M = –2.92°, SD = 7.37°) gets closer to the one observed in the manually defined angles (Fig. 4.15B).

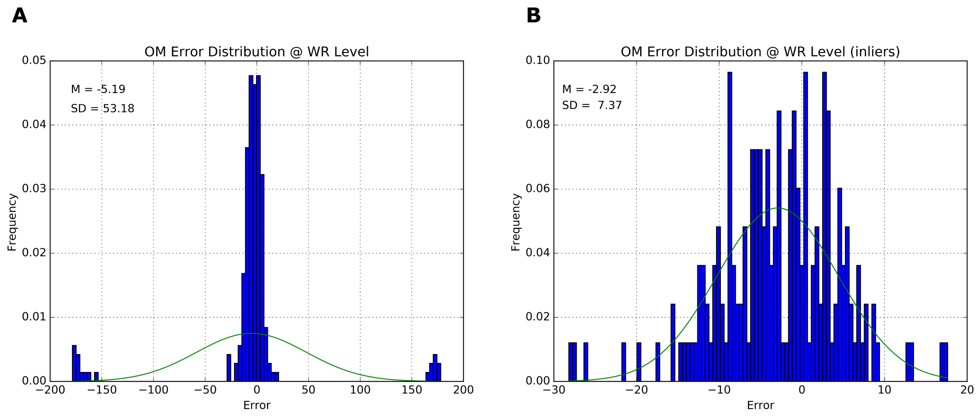


Figure 4.15: **OM error distribution at the waggle run level.** (**A**) The values returned by the OM for the test set contain ~ 10% of outliers. (**B**) Removing the outliers, the remaining values have a distribution with a SD = 7.37°, close to the one for manually defined angles.

Accuracy of the mapping module

The mapping module (MM) clusters together waggle runs that potentially belong to the same dance based on their location in the data space XYT (see Fig. 3.25). To evaluate the performance of the MM, an average value was computed for each of the 23 dances detected in the test set using the values returned by the OM. The resulting error distribution at the dance level has a mean value of –3.27° and an SD of 5.48° (Fig. 4.16A), which as expected is bellow the standard deviation at the WR level.

The MM includes a RANSAC step to compute the average value for each dance disregard the outliers. To demonstrate the effect that the RANSAC step has at the WR level, a new error distribution was computed using exclusively the WRs used by the RANSAC step to calculate the angles for the detected waggle dances. The new error distribution has a mean value of –2.02° and an SD of 6.13°, well within the SD observed in the human defined angles (Fig. 4.16B).

4.6 Conclusions

BeesBook has been in constant improvement since its conception. Major modifications were done to the setup after the first recording season and software modules

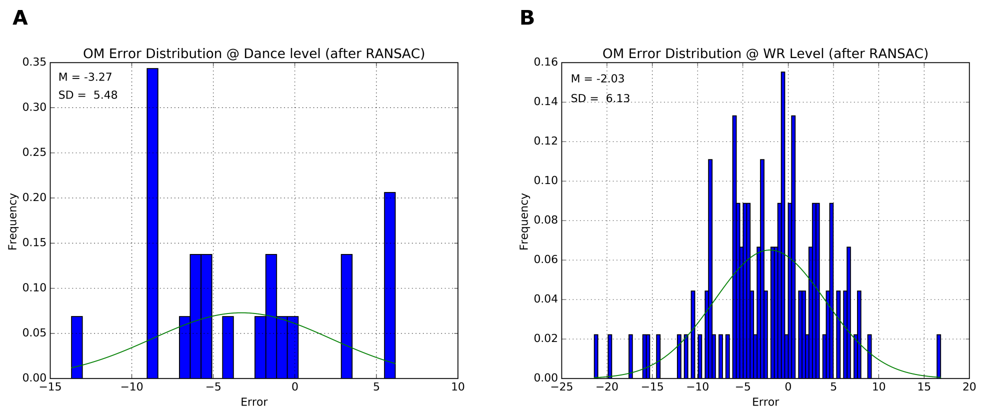


Figure 4.16: **MM error distribution.** (**A**) Error distribution at the dance level has a mean value of −3.27° and an SD of 5.48°. (**B**) The outliers removing effect of RANSAC is observed when mapping dances back to WRs (M = −2.02°, SD = 6.13°).

are constantly tuned to obtain a better performance. The modular architecture of the system allows substituting specific modules without changing the rest of the structure.

In the first part of this chapter, I describe the experimental design of the three recording seasons. Each of the seasons represents an improvement over the previous one. It is noteworthy that before *BeesBook*, conducting experimental observations with a fully marked colony over uninterrupted extended periods of time was unfeasible with any available technology. Consequently, there was no experimental protocol for a longitudinal study of these characteristics. The refined protocol for the 2016 season, not only allows having a fully marked colony during the nine weeks of recording, but it also allows knowing the age of all members at any point in time, adding even more value to the data set generated by *BeesBook*.

The image analysis module based on DCNN outperformed the original CV-Pipeline. Using a DCNN to address the most computationally intensive module in the system reduces the overall demand for computing power; what hopefully will allow processing the *BeesBook* images on general-purpose computers instead of recurring to supercomputers. Thanks to the improvement in decoding accuracy brought by the DCNN-Pipeline, the tracking module can determine the correct ID for more than 98% of the individuals in the hive. This number is even higher than state-of-the-art technologies [70] that unlike *BeesBook* use high-end cameras and redundancy in the markers of up to 26 bits. With 99.69% of all detections being assigned to their correct trajectory, the tracking module provides an accurate image of the movement paths of bees. These paths can then be used for analyses of the bees' experiences during their lifetime.

The WDD performs with a detection accuracy of 90.07%. The decoded waggle orientation has an average error of M = −2.92°, SD = 7.37°, well within the range of human error with a SD = 6.66°.

Having validated the capabilities of the system, the following chapter presents two case studies that show the potential of *BeesBook*.

Chapter 5

Case Studies

This chapter presents two different case studies that exemplify the type of behavioral and ecological questions *BeesBook* can address. The first case analyzes the spatial distribution inside the hive of individuals observed foraging from the same artificial food source. This study was designed to test the hypothesis that bees forage in groups defined according to a social structure inherent to the colony [59], a question that to a great extent first motivated the development of *BeesBook* due to the lack of technologies that allow conducting this type of studies. In the second case, waggle dance activity detected in the observation hive was mapped back to the field. Dance decoding provides integrated information about the food sources available to the colony. This data can be used for monitoring the floral resources in the landscape around the hive. A total of 571 dances detected during a period of 6 hours were automatically decoded and mapped back to the field. Using the Waggle Dance Decoder from *BeesBook*, a task that using traditional methods would require numerous hours of manual work is reduced to just a few seconds[1] [2].

5.1 Spatial Distribution of Foraging Groups on the Honey Comb

During recording season 2016 groups of foragers from the bee colony were trained to 5 different artificial feeders (see Fig. 5.1). At the feeding station, one collaborator kept track of the animals foraging from the place. Those individuals visiting the same location more than once, during the same day, were regarded as belonging to the same

[1]The training of foragers to the artificial feeders was done in both experiments by Franziska Lojewski during recording season 2016.

[2]The data analysis for the first case study was conducted by Maria Sparanberg, member of the Biorobotics Lab. Further details can be found in his bachelor's thesis [104].

forager group. Unmarked bees were tagged to keep control o their visits but were not considered as part of the forager groups since their age was unknown.

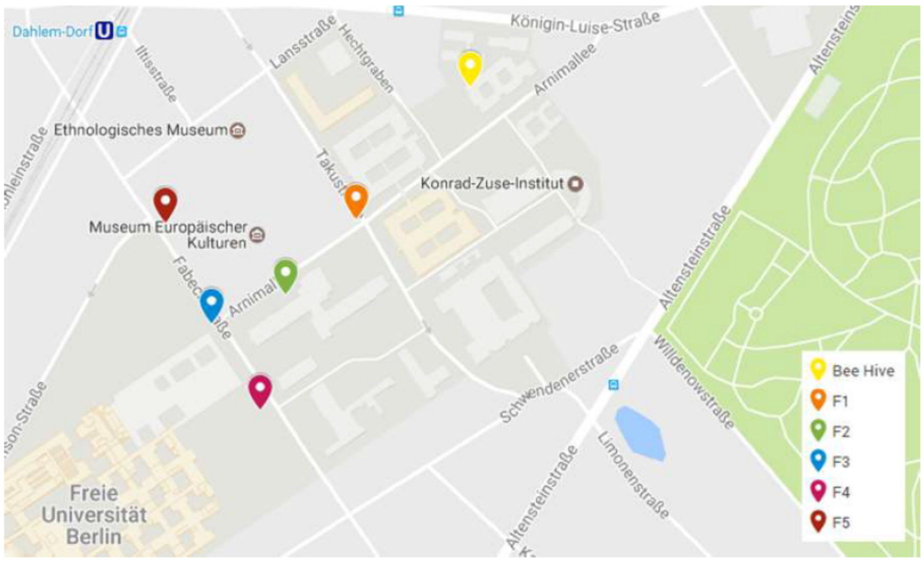


Figure 5.1: **Artificial feeders locations.** The feeding station was located at different distances from the hive (here in yellow). The first days it was located at a distance of 210 m from the hive (F1-orange marker), after that at 340m (F2-green marker), 425m (F3-blue marker), 400m (F4-magenta marker), and 540m (F5-brown merker).

Using these guidelines a total of 25 groups with an average of 10 members were identified. During recording season 2016, we kept a record of the hatching day for each bee as she was marked, using this information it is possible to obtain the age of the foragers. The average age of the 25 groups goes from 24 to 42 days.

Using the positional data extracted by *BeesBook* this study analyzed the spatial proximity of members from the same group when in the hive. This study aims to identify indicators of potential social structures between bees foraging from the same feeding location. As an example here are presented in detail the results obtained for one of the foraging groups. The group in question was detected foraging from the feeding station 1 (F1), it consists of 8 bees and has an average age of 31 ± 3 days.

Spatial distribution.

In Fig. 5.2 and Fig. 5.3 is showed the spatial distribution of members of the forager group over a period of one hour. Each member of the group is represented in a different color. Each of the graphics corresponds to one of the hives' side. For the graphic at the top (left side of the hive), the entrance to the hive is located at the lower left corner, whereas for the second graphic (right side of the hive) the entrance is at the lower right corner.

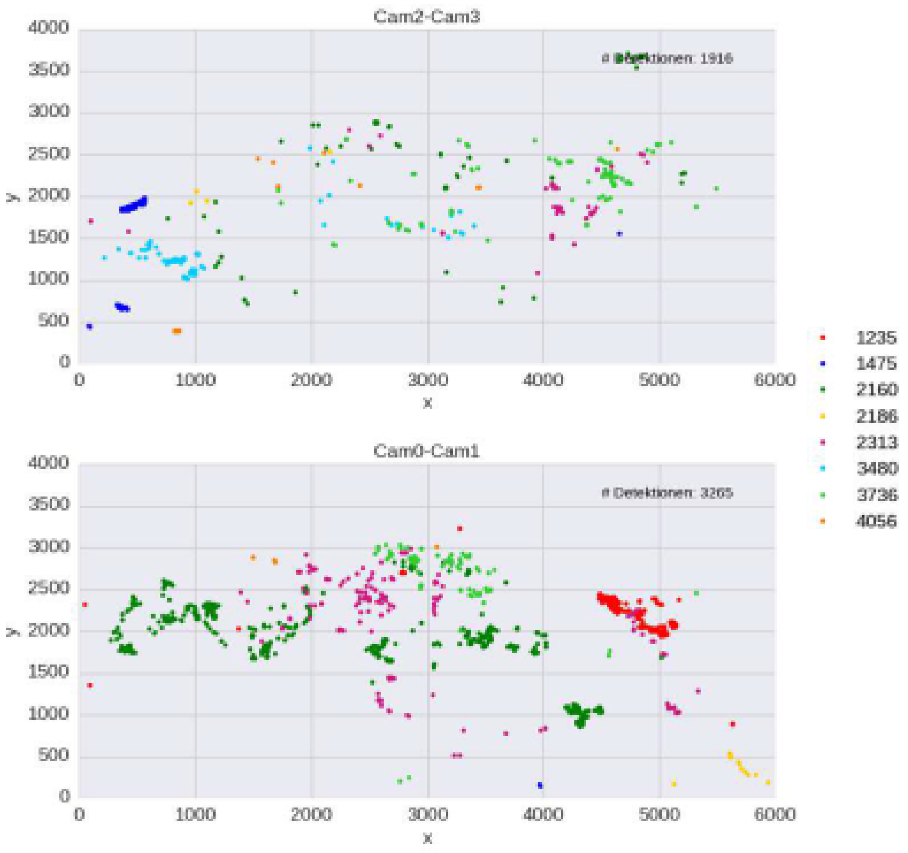


Figure 5.2: **Spatial distribution of the foraging group (08/02/2016 from 4:00 to 5:00 AM).** The image on the top corresponds to the left side of the hive (Cam2-Cam3) and the image on the bottom to the right side (Cam0-Cam1). Each member of the group is represented by a different color. Since these data correspond to a period early in the morning before dawn, only a few of the bees show a high level of activity, in particular, the bee 2160 *(Image from [104])*.

Euclidean distance.

The Euclidean distance between members of the group was computed to analyze if the proximity within members of the group could be considered an indicator of the existence of a social structure. Every time that at least two of the members of the forager group are detected inside the hive their Euclidian distance is calculated and averaged over one second. The measures are accumulated for periods of one hour to find their median and average values over the 24 hours of the day. For the left side of the hive the median distance variates between 371 and 2,378 px with an average of 1,429 px, for the right side the median distance oscillates between 871 and 3,003 px with an average of 1,712 px.

These data is compared with values computed for 1,000 randomly generated groups with the same average age and number of members. Again, for each second in which at least two of the members from the forager group were detected in the hive, their av-

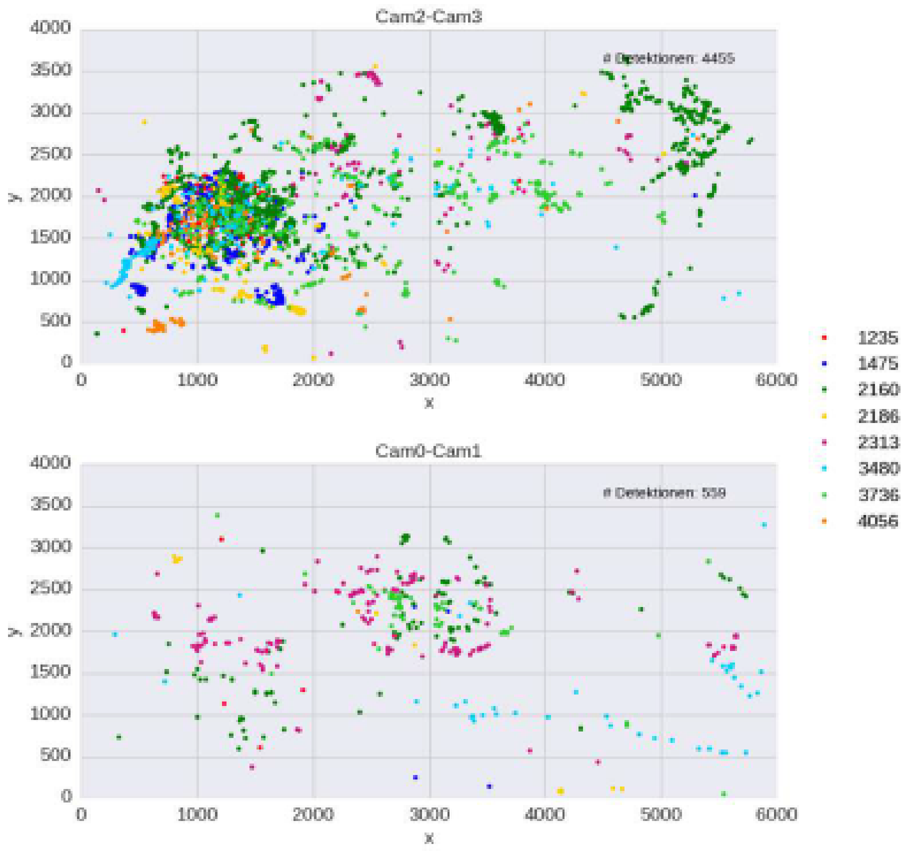


Figure 5.3: **Spatial distribution of the foraging group (08/02/2016 from 2:00 to 3:00 PM).** The image on the top corresponds to the left side of the hive (Cam2-Cam3) and the image on the bottom to the right side (Cam0-Cam1). The data plotted in the graphics correspond to a busy period in the hive when most of the foraging takes place. It is easy to notice that most of the activity is located close to the entrance to the hive located at coord. (0,0) for the image on the top and at coord. (6000,0) for the image on the bottom *(Image from [104])*.

erage Euclidean distance was computed. At each time one of these events happened the Euclidean distance for all random groups with at least two members in the hive was also computed. This information was used to calculate the percentile for the forager group. For the left side of the hive the percentage fluctuated between 2% and 73% during the day with an average of 42%. For the right side the percentile oscillated between 25% and 81% with an average of 51%.

Although these data cannot be used to validate the foraging groups' hypothesis, the intention of presenting this kind of analysis in this chapter to show the technical capabilities of the system rather than to draw conclusions. I consider this type of work out of the scope of this thesis, but I hope that the case studies here presented will help the reader to recognize the potential of the tool system.

5.2 Mapping Dances Back to the Field

To exemplify the use cases of the automatic detection of waggle dances, dance activity detected in the observation hive was mapped back to the field. During the recording season of 2016, a group of foragers from the colony was trained to an artificial feeder placed 342 m southwest of the hive. On the 16th of August, dance activity detected by the attention module (AM) during a period of 6 hours (11:00 - 17:00) was collected and further processed by the rest of the modules of the automatic waggle dance decoder. The 539 detected dances are depicted in Fig. 5.4 as purple circles. The saturation of each circle is proportional to the number of waggle runs associated to that particular dance, where white denotes the minimum (4 WRs) and deep purple the maximum (17 WRs) (5.8 in average).

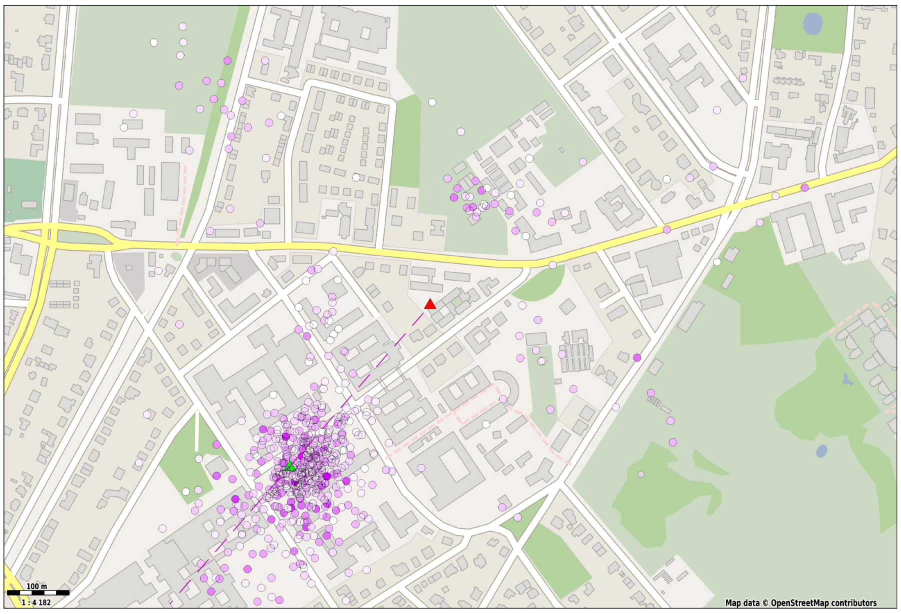


Figure 5.4: **Detected dances mapped back to the field.** Dance detections were averaged over at least four waggle runs and translated to a field location with respect to the sun's Azimuth. A linear mapping was used to convert waggle duration to metric distance. The saturation of each circle is proportional to the number of waggle runs associated to the dance. The hive and feeder positions are depicted with a white and green triangle, respectively. The dashed line represents the average dance direction.

A total of 3,287 WRs were detected and processed by the orientation module (OM); while the mapping module (MM) identifies 539 dances with 4 or more associated WRs. The angles obtained by the MM and OM were translated to angles on the field with reference to the sun's azimuth at the time the WRs were detected. As the dots in Fig. 5.4 and the error distributions in Fig. 5.5 show, most of the detected dance activity clusters around the artificial feeder. The mean angular error with respect to the artificial feeder is of only 0.69° at the WR level and 2.35° at the dance level.

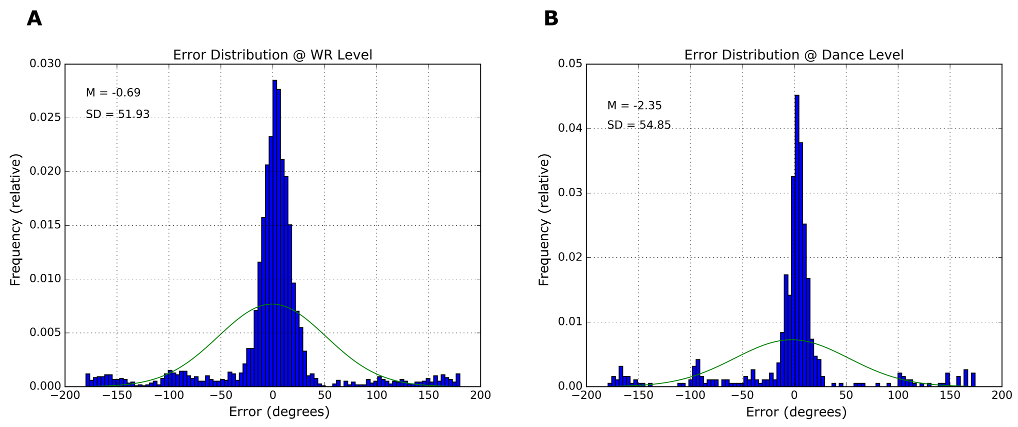


Figure 5.5: **Error distribution with respect to the orientation of the artificial feeder.** (**A**) From the error distribution at the waggle run level (M = −0.69°, SD = 51.93°) (**B**) and at the dance level (M = −2.35°, SD = 54.85°) it is clear that most of the dance activity pointed in the direction of the artificial feeder.

In Fig. 5.4, it can be noted that the dances around the artificial feeder are rather scattered. Although this distribution of the mapped dances can be misinterpreted as an error from the OM, it is actually caused by the dancers themselves [120, 112, 108]. This divergence in the orientation of waggle runs correlates negatively with the distance to the advertised location. To analyze the precision of the dancers advertising the artificial feeder, dances and WRs with an angular error of ±45° were selected. The error distribution for the filtered data shows a SD = 11.12° at the dance level and SD = 14.37° at the WR level (see Fig. 5.6), what is consistent with previous observations documented by Landgraf et al. [62].

To map dances back to the field, in addition to their orientation with respect to the solar azimuth, it is necessary to transform their average WR duration to distance using a linear factor. Bees encode accumulated optical flow rather than metric distances in their waggle runs. Hence, the distance/duration factor is particular to each colony and geographical situation. To calibrate the distance/duration factor of the colony in this study, the duration of all WRs signaling towards the artificial feeder (±10°) was averaged (M = 582.79 ms, SD = 196.09 ms). The mean duration of 582.79 ms was considered as reference for the 342 m distance to the artificial feeder and the rate of both values (342 m/582.79 ms) as the linear factor to map the rest of the dances.

It is unlikely that the set of waggle runs used to compute the distance/duration factor would have contained waggle runs signaling a different food source since natural food sources were scarce in that time of the year. Furthermore, the coefficient of variation of ≈ 0.34 (196.09/582.79) for the duration distribution is consistent with the value

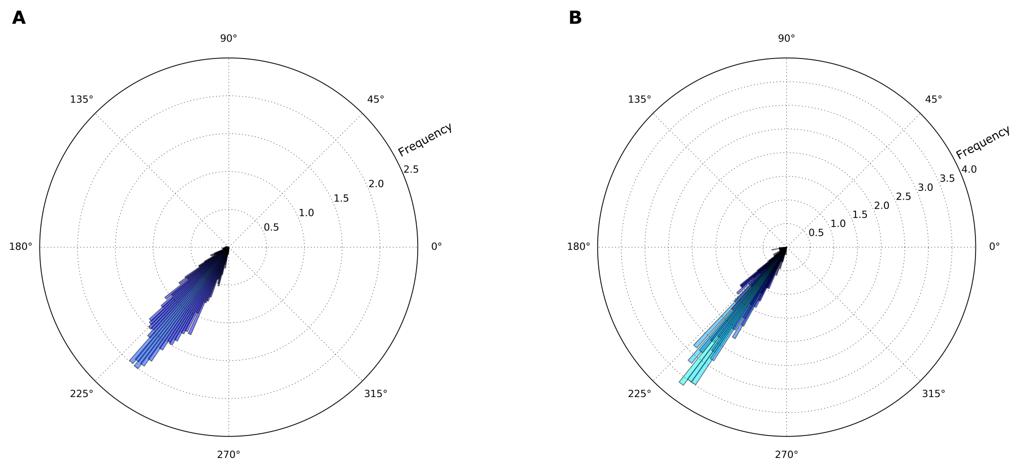


Figure 5.6: **Precision of the waggle dance.** Dance activity pointing towards the artificial feeder (±45°) is very accurate but not as precise. (**A**) It has mean error of 2.58° with an SD = 14.37° at the waggle run level, (**B**) and a M = 2.33° with an SD = 11.12° at the dance level.

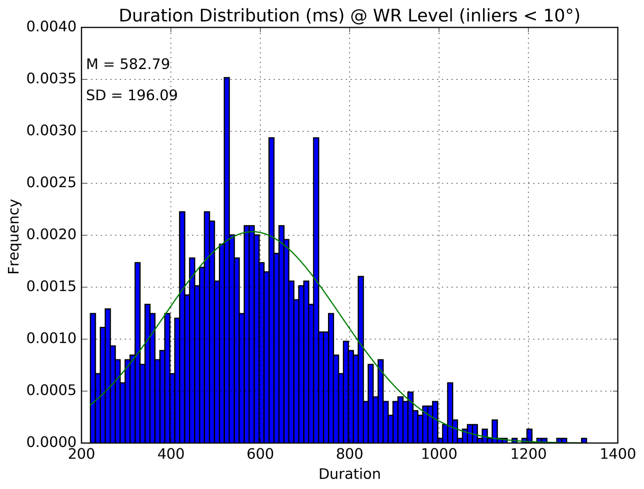


Figure 5.7: **Duration distribution.** The duration of 2024 WRs signaling towards the artificial feeder (±10°) was averaged (M = 582.79 ms, SD = 196.09 ms) to calibrate the distance/duration factor of the colony.

documented by Landgraf et al. [62].

5.3 Discussion

The two case studies presented in this section show how, using *BeesBook*, researchers gain access to a variety of data that up to now was either unavailable or only partially accessible. The data extraction effort, which using alternative methods would require a significant manual work, is reduced to a matter of minutes with the system.

It is important to remember that *BeesBook* was conceived as a tool for researchers looking to reduce the manual effort required to collect and decode data. Likewise, the experiments described in this chapter aim to showcase precisely the technical capabilities of the system rather than to give an extensive list of them. It is important to note, therefore, that the focus of this work is first to lay the technical foundation for future case studies. I consider this type of work out of the scope for this thesis, but I hope that researchers from other fields will recognize the potential of the tool and develop experiments of their own which make use of the technologies presented here.

Chapter 6

Concluding Remarks

This dissertation introduces the *BeesBook* system, a successfully implemented vision-based solution for the automatic tracking of behavioral activity in honey bee colonies. The system consists of a custom-made recording setup and software modules for identifying uniquely marked bees and decoding of waggle dances. *BeesBook* is unique in its approach and capabilities: it can identify and decode up to 90% of all the waggle dance activity, the location, orientation, and identity of almost all colony members ($\sim$ 98%) inside the observation hive, 24 hours a day over several weeks. No other available technology has even comparable capabilities.

BeesBook was developed in response to the manifest need for automation of behavior quantification in studies of bee colonies. While studies on animal behavior have become increasingly quantitative over the last decades, technologies for behavior quantification have not developed at the same pace, and manual scoring is still to this day the most common approach to this task. Relying on human observers to track a high number of individuals, as is the case in bee colonies, poses several disadvantages, chief among which is the introduction of subjectivity as well as constraints on the depth and breadth of the observations.

By automating the data acquisition process, *BeesBook* is able to monitor a high number of animals during extended periods of time. Likewise, the data extraction effort, which with alternative methods require days of manual work, is reduced to a matter of minutes with the use of the system. Furthermore, removing the human factor from the data collection process ensures the objectivity of the gathered data.

BeesBook has been conceived with a modular architecture of reasonably self-contained elements in mind. This structure favors the incremental development of software and hardware components as well as the improvement of the already existing elements without requiring significant adaptations to the rest of the system. Thanks to the architectural style of the system, it was possible to iteratively and independently improve the recording setup, most of the software elements of the waggle dance decoder, and the image analysis module. The accuracy of *BeesBook* on determining the correct IDs of marked animals in the hive is higher than that reported by Mersch et al. [70] (98% vs.

95% respectively). Whereas the current state-of-the-art system developed by Mersch et al. [70] relies on high-end cameras and markers with 26 bits of redundancy, *BeesBook* circumvents this need for expensive hardware and bit redundancy through the use of a highly reliable decoder.

Using less expensive hardware means lower image quality, which in turn introduces uncertainty when it comes to identifying elements in images. In *BeesBook*, this manifests itself in the first layer of abstraction as potentially misread IDs, which then propagate down the data pipeline. *BeesBook* tackles this challenge by quantifying the uncertainty early in the process and adding algorithms that are able to make more intelligent decisions based on probabilities. The image analysis module delivers the marker IDs as a set of probabilities rather than discrete IDs as would be the case in systems with high-end cameras where the identification usually has a high level of certainty. Given that behavioral studies in bees rely on being able to track a given individual through time, it is clear how this can be accomplished by linking consecutive frames containing the same ID in the ideal scenario. Once uncertainty is introduced, tracing reliable trajectories requires to consider additional parameters. This is accomplished by the tracking module within the system, which takes into consideration position and orientation of individuals in addition to their identity, thus enabling the tracing of trajectories despite possible single faulty detections within a given path. Unlike the case with discrete IDs, adding additional parameters makes our trajectory tracing method more robust once the cost of the hardware is taken into consideration.

6.1 Contributions

This thesis makes four major contributions:

First, the system itself. The modular design of the system favors the incremental development of software and hardware components and allows for the constant improvement of multiple elements at a time. The hardware components and all the software elements that precede the image analysis, ultimately have a significant impact on the quality of the data delivered by the last stage of the system. Therefore, is of great importance to continually search room for improvement in these early stages of the system. Starting from the marker, that was specially designed to be used in longitudinal studies of bee colonies within the hive, i.e., that it could unambiguously identify thousands of individuals, functional under infrared light, and that could withstand the particular intense activity of the bee during the summer without restricting its mobility. The current iteration of *BeesBook*, while using budget components for the recording setup, manages to generate high-quality raw data for the rest of the processing pipeline through either the adaptation of existing hardware (as it is the case with the cameras) or custom-built components. The hardware elements of the recording setup are complemented by software modules for image acquisition and storage designed to operate over extended periods of time without pause with high levels of reliability. *BeesBook's* image analysis module has a significant weight on the overall performance of the system; it is mainly thanks to the level of accuracy of this module that *BeesBook* outperforms state-of-the-art technologies like the one proposed by Mersch et al. [70]. Furthermore, *BeesBook* not only outperforms similar techniques on the level of accuracy but it also enables conducting experimental observations with unprecedented levels of breadth and depth.

A second significant contribution is the experimental design from season 2016. Previous to the development of *BeesBook*, conducting experimental observations with the depth and breadth from the three recording seasons documented in this thesis was not pursued, since processing the generated raw data with available technologies would require a enormous amount of resources. Consequently, there was no experimental design for similar cases to *BeesBook's* recording seasons and one had to be designed from zero. The original procedure followed during recording season 2014 was improved for 2015 and by season 2016 *BeesBook's* experimental design has achieved a level of refinement that reflects in well-defined procedures and an overall process that takes full advantage of *BeesBook's* potential.

A third contribution is the waggle dance detector. Although this system operates under *BeesBook*, and it is expected that data from both processing pipelines can be fully integrated to augment the variety of data provided by the system, it can also operate as an entirely independent system. With a detection accuracy of 90.07% and a decoding error well within the range of average human error, the waggle dance decoder can provide valuable information and save hundreds of hours of manual work to behavioral biologists and ecologists by decoding and mapping the dance activity of the hive. Furthermore, its implementation is very simple and does not require high-end equipment for video recording or high performance computing, what makes it attractive to a broader public than the full *BeesBook* system.

I consider as a fourth major contribution of this thesis the three recording seasons. The data sets stemming from the three recording seasons are unique in their extension and number of animals under observation. For each season around 60 million images were recorded, what translates to more than 6 billions of detections. With the high level of accuracy of he most recent implementation of the pipeline, that can determine the correct ID for more than 98% of the individuals in the hive, no other study conducted until now comes even remotely close to the characteristics of *BeesBook's* data sets. In particular, the season 2016 also adds the age of each one of the individuals to the data set, what makes it even more interesting to researchers.

6.2 Directions for Future Work

BeesBook was conceived as a framework for the incremental implementation of hardware and software modules. As such, there will always be room for improvements such as the automatic detection of focal behaviors or the implementation of the tag decoder in a portable device to ease the data recollection at the feeding station and closing the loop of automated data extraction in behavioral experiments with honey bees. Nevertheless, I consider the most important and urgent work the **analysis and interpretation of the existing data sets** stemming from the recording seasons. With feedback from biologist and researchers from other fields, new lines of development can be discovered what in turn will generate new data sets, setting in motion a loop of feedback that at each iteration will provide a clearer picture of the life at the colony and the mechanism laying at the core of its collective behavior. The ultimate goal of *BeesBook* is using this information to build a virtual hive in which hypothesis related either to biology or collective intelligence can be reliably tested.

A waggle dance decoder based on deep convolutional neural networks (DCNN). The main constraint for the successful implementation of a DCNN is the availability of labeled data. Using the automatic waggle dance decoder presented in this work as a generator of labeled data, a DCNN based version of the software could be developed. Tests have already been conducted in the *Biorobotics Lab* and they have delivered promising results. The transition of technology in the waggle dance decoder would potentially translate into a better performance and the speed up of the decoding process as it happened with the image analysis module within *BeesBook*. This sort of improvements could lead to the mapping of foraging activity in real time, a valuable resource for researchers in the field of ecology.

Bibliography

[1] Hasan Al Toufailia, Christoph Grüter, and Francis L. W. Ratnieks. Persistence to unrewarding feeding locations by honeybee foragers (apis mellifera): The effects of experience, resource profitability and season. *Ethology*, 119:1096–1106, 2013.

[2] Jeanne Altmann. Observational Study of Behavior: Sampling Methods. *Behaviour*, 49(3):227–266, jan 1974.

[3] David J Anderson and Pietro Perona. Toward a science of computational ethology. *Neuron*, 84(1):18–31, 2014.

[4] B Atcheson, F Heide, and W Heidrich. CALTag: High Precision Fiducial Markers for Camera Calibration. In Reinhard Koch, Andreas Kolb, and Christof Rezk-Salama, editors, *Proceedings of the Vision, Modeling, and Visualization Workshop 2010*, pages 41–48, Siegen, Germany, 2010.

[5] Nicholas J. Balfour and Francis L. W. Ratnieks. Using the waggle dance to determine the spatial ecology of honey bees during commercial crop pollination. *Agricultural and Forest Entomology*, dec 2016.

[6] M. Beekman and Francis L. W. Ratnieks. Long-range foraging by the honey-bee , Apis melliera L . *British ecological society*, 14(4):490–496, 2000.

[7] John A. Bender, Elaine M. Simpson, Brian R. Tietz, Kathryn A. Daltorio, Roger D. Quinn, and Roy E. Ritzmann. Kinematic and behavioral evidence for a distinction between trotting and ambling gaits in the cockroach Blaberus discoidalis. *Journal of Experimental Biology*, 214(12), 2011.

[8] Jacobus C. Biesmeijer and Thomas D. Seeley. The use of waggle dance information by honey bees throughout their foraging careers. *Behavioral Ecology and Sociobiology*, 59:133–142, 2005.

[9] Franziska Boenisch. Feature engineering and probabilistic tracking on honey bee trajectories. Master's thesis, Freie Unviersität Berlin, 2017.

[10] Franziska Boenisch, Benjamin Malte Rosemann, Benjamin Wild, Fernando Wario, David Dormagen, and Tim Landgraf. Novel technological and methodological tools for the understanding of collective behaviors. (in press), 2017.

[11] Kristin Branson, Alice A Robie, John Bender, Pietro Perona, and Michael H Dickinson. High-throughput ethomics in large groups of Drosophila. *Nature Methods*, 6(6):451–457, may 2009.

[12] Katherine S. Button, John P. A. Ioannidis, Claire Mokrysz, Brian A. Nosek, Jonathan Flint, Emma S. J. Robinson, and Marcus R. Munafò. Power failure: why small sample size undermines the reliability of neuroscience. *Nature Reviews Neuroscience*, 14(5):365–376, apr 2013.

[13] Scott Camazine and James Sneyd. A model of collective nectar source selection by honey bees: Self-organization through simple rules. *Journal of Theoretical Biology*, 149(4):547–571, 1991.

[14] James W. Cooley and John W. Tukey. An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, 19(90):297, apr 1965.

[15] Margaret J Couvillon, Fiona C Riddell Pearce, Elisabeth L Harris-Jones, Amanda M Kuepfer, Samantha J Mackenzie-Smith, Laura A Rozario, Roger Schürch, and Francis L W Ratnieks. Intra-dance variation among waggle runs and the design of efficient protocols for honey bee dance decoding. *Biology open*, 1(5):467–72, may 2012.

[16] MargaretăJ. Couvillon, Roger Schürch, and Francis L. W. Ratnieks. Waggle Dance Distances as Integrative Indicators of Seasonal Foraging Challenges. *PLoS ONE*, 9(4):e93495, 2014.

[17] James D. Crall, Nick Gravish, Andrew M. Mountcastle, and Stacey A. Combes. BEEtag: A low-cost, image-based tracking system for the study of animal behavior and locomotion. *PLoS ONE*, 10(9):e0136487, 2015.

[18] F. C. Crick. *What Mad Pursuit: A Personal View of Scientific Discovery.* Basic Books, New York, 1988.

[19] M. Dacke and M. V. Srinivasan. Two odometers in honeybees? *Journal of Experimental Biology*, 211(20):3281–3286, 2008.

[20] H Dankert, L Wang, E D Hoopfer, David J Anderson, and P Perona. Automated monitoring and analysis of social behavior in Drosophila. *Nat Methods*, 6(4):297–303, 2009.

[21] F. de Chaumont, A. Dufour, and J.-C. Olivo-Marin. Tracking articulated objects with physics engines. In *2009 16th IEEE International Conference on Image Processing (ICIP)*, pages 885–888. IEEE, nov 2009.

[22] Fabrice de Chaumont, Renata Dos-Santos Coura, Pierre Serreau, Arnaud Cressant, Jonathan Chabout, Sylvie Granon, and Jean-Christophe Olivo-Marin. Computerized video analysis of social interactions in mice. *Nature Methods*, 9(4):410–417, apr 2012.

[23] Rodrigo J De Marco, Juan M Gurevitz, and Randolf Menzel. Variability in the encoding of spatial information by dancing bees. *The Journal of experimental biology*, 211(Pt 10):1635–1644, may 2008.

[24] Han de Vries and Jacobus C. Biesmeijer. Modelling collective foraging by means of individual behaviour rules in honey-bees. *Behavioral Ecology and Sociobiology*, 44(2):109–124, nov 1998.

[25] Anthony I. Dell, John a. Bender, Kristin Branson, Iain D. Couzin, Gonzalo G. de Polavieja, Lucas P J J Noldus, Alfonso Pérez-Escudero, Pietro Perona, Andrew D. Straw, Martin Wikelski, and Ulrich Brose. Automated image-based tracking and its application in ecology. *Trends in Ecology and Evolution*, 29(7):417–428, 2014.

[26] R. L. Dennis, R. C. Newberry, H.-W. Cheng, and I. Estevez. Appearance Matters: Artificial Marking Alters Aggression and Stress. *Poultry Science*, 87(10):1939–1946, oct 2008.

[27] Harald E. Esch, Shaowu Zhang, Mandyan V. Srinivasan, and Juergen Tautz. Honeybee dances communicate distances measured by optic flow. *Nature*, 411(6837):581–583, may 2001.

[28] Thomas Fasciano, Hoan Nguyen, Anna Dornhaus, and Min C. Shin. Tracking multiple ants in a colony. In *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 534–540. IEEE, jan 2013.

[29] Adam Feldman and Tucker Balch. Automatic Identification of Bee Movement. In *International Workshop on the Mathematics and algorithms of social insects*, pages 53–59, Georgia, 2003. Georgia Institute of Technology.

[30] Adam Feldman and Tucker Balch. Representing Honey Bee Behavior for Recognition Using Human Trainable Models. *Adaptive Behavior*, 12(3-4):241–250, dec 2004.

[31] Mark Fiala. ARTag, a Fiducial Marker System Using Digital Techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 590–596. IEEE, 2005.

[32] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, jun 1981.

[33] R A Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[34] Elizabeth L. Franklin and Nigel R. Franks. Individual and social learning in tandem-running recruitment by ants. *Animal Behaviour*, 84(2):361–368, aug 2012.

[35] Wayne M Getz, D Brückner, and Katherine B Smith. Conditioning honeybees to discriminate between heritable odors from full and half sisters. *Journal of Comparative Physiology A*, 159:251–256, 1986.

[36] Mariana Gil and Rodrigo J De Marco. Decoding information in the honeybee dance: revisiting the tactile hypothesis. *Animal Behaviour*, 80:887–894, 2010.

[37] Timothy H. Goldsmith. The nature of the retinal action potential, and the spectral sensitivities of ultraviolet and green receptor systems of the compound eye of the worker honey-bee. *The Journal of general physiology*, 43(4):775–99, mar 1960.

[38] Christoph Grüter, M Sol Balbuena, and Walter M Farina. Informational conflicts created by the waggle dance. *Proceedings. Biological sciences / The Royal Society*, 275(March):1321–1327, 2008.

[39] Christoph Grüter and Francis L. W. Ratnieks. Honeybee foragers increase the use of waggle dance information when private information becomes unrewarding. *Animal Behaviour*, 81(5):949–954, 2011.

[40] Qiang Guo, Jun Lei, Dan Tu, and Guohui Li. Reading numbers in natural scene images with convolutional neural networks. In *Proceedings 2014 IEEE International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pages 48–53. IEEE, oct 2014.

[41] J R Hagler and C G Jackson. Methods for marking insects: current techniques and future prospects. *Annual review of entomology*, 46:511–543, 2001.

[42] J. B. S. Haldane and H. Spurway. A statistical analysis of communication in Apis mellifera and a comparison with communication in other animals. *Insectes Sociaux*, 1(3):247–283, sep 1954.

[43] R.W. Hamming. Error detecting and error correcting codes, 1950.

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv preprint*, dec 2015.

[45] Irmgard Hoffmann. Über die Arbeitsteilung in weiselrichtigen und weisellosen Kleinvölkern der Honigbiene. *Z. Bienenforsch*, 5:267–279, 1961.

[46] Bert Hölldobler and Edward Wilson. *The Superorganism: The Beauty, Elegance, and Strangeness of Insect Societies.* W.W. Norton & Company, 1 edition, 2008.

[47] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, jan 2013.

[48] Brian R. Johnson. Division of labor in honeybees: Form, function, and proximate mechanisms. *Behavioral Ecology and Sociobiology*, 64(3):305–316, 2010.

[49] Mayank Kabra, Alice A Robie, Marta Rivera-Alba, Steven Branson, and Kristin Branson. JAABA: interactive machine learning for automatic annotation of animal behavior. *Nature Methods*, 10(1):64–67, 2012.

[50] Dervis Karaboga. AN IDEA BASED ON HONEY BEE SWARM FOR NUMERICAL OPTIMIZATION (TECHNICAL REPORT-TR06, OCTOBER, 2005). Technical report, Erciyes University, Engineering Faculty Computer Engineering Department, 2005.

[51] Zia. Khan, Tucker. Balch, and Frank. Dellaert. A Rao-Blackwellized Particle Filter for Eigentracking. In *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 980–986. IEEE, 2004.

[52] Toshifumi Kimura, Mizue Ohashi, Ryuichi Okada, and Hidetoshi Ikeno. A new approach for the simultaneous tracking of multiple Honeybees for analysis of hive behavior. *Apidologie*, 42(5):607–617, 2011.

[53] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, dec 2014.

[54] Wolfgang H. Kirchner and Gerard Arnold. Intracolonial kin discrimination in honeybees: do bees dance with their supersisters? *Animal Behaviour*, 61(3):597–600, 2001.

[55] Wolfgang H. Kirchner and Am Hubland. Hearing in honeybees: the mechanical response of the bee's antenna to near field sound. *Journal Of Comparative Physiology A Neuroethology Sensory Neural And Behavioral Physiology*, pages 261–265, 1994.

[56] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-Normalizing Neural Networks. *arXiv preprint*, jun 2017.

[57] Alex Krizhevsky, Ilya Sutskever, and Geoffrey 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, 2012.

[58] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, mar 1955.

[59] Tim Landgraf. *ROBOBEE: A BIOMIMETIC HONEYBEE ROBOT FOR THE ANALYSIS OF THE DANCE COMMUNICATION SYSTEM.* PhD thesis, Free University Berlin, 2013.

[60] Tim Landgraf, Michael Oertel, Andreas Kirbach, Randolf Menzel, and Raúl Rojas. Imitation of the honeybee dance communication system by means of a biomimetic robot. In *Lecture Notes in Computer Science*, volume 7375 LNAI, pages 132–143. Springer-Verlag, 2012.

[61] Tim Landgraf and Raúl Rojas. Tracking honey bee dances from sparse optical flow fields. Technical Report June, Freie Universität Berlin, Berlin, 2007.

[62] Tim Landgraf, Raúl Rojas, Hai Nguyen, Fabian Kriegel, and Katja Stettin. Analysis of the waggle dance motion of honeybees for the design of a biomimetic honeybee robot. *PLoS ONE*, 6(8):e21354, 2011.

[63] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, December 1989.

[64] Andy Liaw and Matthew Wiener. Classification and Regression by randomForest. *R News*, 2(3):18–22, 2002.

[65] Martin Lindauer. Schwarmbienen auf Wohnungssuche. *Zeitschrift für Vergleichende Physiologier*, 37(4):263–324, 1955.

[66] Ruben Martinez-Cantin. BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. *Journal of Machine Learning Research*, 15:3915–3919, 2014.

[67] M D Meixner and R F A Moritz. Clique formation of super-sister honeybee workers (Apis mellifera) in experimental groups. *Insectes Sociaux*, 51:43–47, 2004.

[68] R Menzel and M Eckoldt. *Die Intelligenz der Bienen: Wie sie denken, planen, fühlen und was wir daraus lernen können.* Knaus, 2016.

[69] Randolf Menzel, Andreas Kirbach, Wolf Dieter Haass, Bernd Fischer, Jacqueline Fuchs, Miriam Koblofsky, Konstantin Lehmann, Lutz Reiter, Hanno Meyer, Hai Nguyen, Sarah Jones, Philipp Norton, and Uwe Greggers. A common frame of reference for learned and communicated vectors in honeybee navigation. *Current Biology*, 21(8):645–650, 2011.

[70] Danielle P. Mersch, Alessandro Crespi, and Laurent Keller. Tracking individuals shows spatial fidelity is a key regulator of ant social organization. *Science*, 340(6136):1090–3, 2013.

[71] Hauke Mönck. Large Scale Supercomputer Assisted and Live Video Encoding with Image Statistics. Master's thesis, Freie Unviersität Berlin, 2016.

[72] R F A Moritz and T Heisler. Super and Half-Sister Discrimination by Honey-Bee Workers (Apis-Mellifera L) in a Trophallactic Bioassay. *Insectes Sociaux*, 39(4):365–372, 1992.

[73] Thiago Mosqueiro, Chelsea Cook, Ramon Huerta, Jürgen Gadau, Brian Smith, and Noa Pinter-Wollman. Task allocation and site fidelity jointly influence foraging regulation in honeybee colonies. *Royal Society Open Science*, 4(8):170344, aug 2017.

[74] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. *NIPS workshop on deep learning and unsupervised feature learning*, 2011(2):5, 2011.

[75] James C. Nieh. The Stop Signal of Honey-bees - Reconsidering Its Message. *Behav Ecol Sociobiol*, 33(1):51–56, 1993.

[76] James C. Nieh. A Negative Feedback Signal That Is Triggered by Peril Curbs Honey Bee Recruitment. *Current Biology*, 20(4):310–315, 2010.

[77] Lucas P. J. J. Noldus, Andrew J. Spink, and Ruud A. J. Tegelenbosch. EthoVision: A versatile video tracking system for automation of behavioral experiments. *Behavior Research Methods, Instruments, & Computers*, 33(3):398–414, aug 2001.

[78] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Learning and Inferring Motion Patterns using Parametric Segmental Switching Linear Dynamic Systems. *International Journal of Computer Vision*, 77(1-3):103–124, may 2008.

[79] Shay Ohayon, Ofer Avni, Adam L. Taylor, Pietro Perona, and S.E. Roian Egnor. Automated multi-day tracking of marked mice for the analysis of social behaviour. *Journal of Neuroscience Methods*, 219(1):10–19, sep 2013.

[80] Benjamin P. Oldroyd, Thomas E. Rinderer, and Steven M. Buco. Honey bees dance with their super-sisters. *Animal Behaviour*, 42:121–129, 1991.

[81] Benjamin P. Oldroyd, Thomas E. Rinderer, Steven M. Buco, and Lorraine D. Beaman. Genetic variance in honey bees for preferred foraging distance. *Animal Behaviour*, 45(2):323–332, 1993.

[82] Robert E Page. *The Spirit of the Hive*. Harvard University Press, 2013.

[83] Kevin M. Passino and Thomas D. Seeley. Modeling and analysis of nest-site selection by honeybee swarms: the speed and accuracy trade-off. *Behav Ecol Sociobiol*, 59:427–442, 2006.

[84] Alfonso Pérez-Escudero, Julián Vicente-Page, Robert C Hinz, Sara Arganda, and Gonzalo G de Polavieja. idTracker: tracking individuals in a group by automatic identification of unmarked animals. *Nature Methods*, 11(7):743–748, 2014.

[85] D T Pham, A Ghanbarzadeh, E Koç, S Otri, S Rahim, and M Zaidi. The Bees Algorithm A Novel Tool for Complex Optimisation Problems. Technical report, Manufacturing Engineering Centre, Cardiff University, 2005.

[86] Fabio Poiesi and Andrea Cavallaro. Tracking multiple high-density homogeneous targets. *IEEE Trans. on Circuits and Systems for Video Technology*, 25(4):623–637, 2015.

[87] Alexander Rau. Realtime Honey Bee Waggle Dance Decoding System. Master's thesis, Freie Universität Berlin, 2014.

[88] Chris R. Reid, David J. T. Sumpter, and Madeleine Beekman. Optimisation in a natural system: Argentine ants solve the Towers of Hanoi. *Journal of Experimental Biology*, 214(1), 2010.

[89] Andreagiovanni Reina, James A. R. Marshall, Vito Trianni, and Thomas Bose. A model of the best-of-N nest-site selection process in honeybees. *arXiv*, nov 2016.

[90] Andreagiovanni Reina, Gabriele Valentini, Cristian Fernández-Oto, Marco Dorigo, and Vito Trianni. A Design Pattern for Decentralised Decision Making. *PLoS ONE*, 10(10):e0140950, oct 2015.

[91] J R Riley, Uwe Greggers, a D Smith, D R Reynolds, and Randolf Menzel. The flight paths of honeybees recruited by the waggle dance. *Nature*, 435(7039):205–207, 2005.

[92] Benjamin Malte Rosemann. Ein erweiterbares System für Experimente mit Multi-Target Tracking von markierten Bienen. Master's thesis, Freie Unversität Berlin, 2017.

[93] Ricarda Scheiner, Charles I Abramson, Robert Brodschneider, Karl Crailsheim, Walter M Farina, Stefan Fuchs, Bernd Grünewald, Sybille Hahshold, Marlene Karrer, Gudrun Koeniger, Niko Koeniger, Randolf Menzel, Samir Mujagic, Gerald Radspieler, Thomas Schmickl, Christof Schneider, Adam J Siegel, Martina Szopek, and Ronald Thenius. Standard methods for behavioural studies of Apis mellifera. *Journal of Apicultural Research*, 52(4):1–58, jan 2013.

[94] Roger Schürch and Francis L. W. Ratnieks. The spatial information content of the honey bee waggle dance. *Frontiers in Ecology and Evolution*, 3:22, mar 2015.

[95] Thomas D. Seeley. *The Wisdom of the Hive*. Harvard University Press, 1995.

[96] Thomas D. Seeley and S. C. Buhrman. Nest-site selection in honey bees: How well do swarms implement the "best-of-N" decision rule? *Behavioral Ecology and Sociobiology*, 49:416–427, 2001.

[97] Thomas D. Seeley, Alexander S. Mikheyev, and Gary J. Pagano. Dancing bees tune both duration and rate of waggle-run production in relation to nectar-source profitability. *Journal of Comparative Physiology - A Sensory, Neural, and Behavioral Physiology*, 186(9):813–819, oct 2000.

[98] Thomas D. Seeley and WilliamF. Towne. Tactics of dance choice in honey bees: do foragers compare dances? *Behavioral Ecology and Sociobiology*, 30(1):59–69, mar 1992.

[99] Jean Serra. Introduction to mathematical morphology. *Computer Vision, Graphics, and Image Processing*, 35:283–305, 1986.

[100] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, jan 1973.

[101] Leon Sixt. RenderGAN: Generating Realistic Labeled Data. Master's thesis, Freie Universität Berlin, 2016.

[102] Leon Sixt, Benjamin Wild, and Tim Landgraf. RenderGAN: Generating Realistic Labeled Data. *arXiv preprint*, nov 2016.

[103] Keith W Sockman and Hubert Schwabl. Plasma Corticosterone in Nestling American Kestrels: Effects of Age, Handling Stress, Yolk Androgens, and Body Condition. *General and Comparative Endocrinology*, 122(2):205–212, may 2001.

[104] Maria Sparanberg. Analyse der raumlichen Verteilung von Sammlerinnen innerhalb der Honigbienenkolonie. Master's thesis, Freie Unviersität Berlin, 2017.

[105] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[106] Ingolf Steffan-Dewenter and Arno Kuhn. Honeybee foraging in differentially structured landscapes. *Proceedings. Biological sciences / The Royal Society*, 270(1515):569–575, 2003.

[107] Peter Strümpel. Kamerakalibrierung und (teil-) automatisiertes Stitching. Master's thesis, Freie Universität Berlin, 2016.

[108] David A. Tanner and Kirk Visscher. Do honey bees tune error in their dances in nectar-foraging and house-hunting? *Behavioral Ecology and Sociobiology*, 59(4):571–576, feb 2006.

[109] David A. Tanner and P. Kirk Visscher. Adaptation or constraint? Reference-dependent scatter in honey bee dances. *Behavioral Ecology and Sociobiology*, 64(7):1081–1086, 2010.

[110] Christian Tietz. Entwurf und Implementierung einer Speicherarchitektur für Bildmassendaten zur Verhaltensanalyse von Honigbienenkolonien. Master's thesis, Freie Universität Berlin, 2015.

[111] N. Tinbergen. On aims and methods of Ethology. *Zeitschrift für Tierpsychologie*, 20(4):410–433, apr 1963.

[112] William F. Towne and James L. Gould. The spatial precision of the honey bees' dance communication. *Journal of Insect Behavior*, 1(2):129–155, apr 1988.

[113] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. *arXiv preprint*, dec 2014.

[114] P. Kirk Visscher and Thomas D. Seeley. Foraging Strategy of Honeybee Colonies in a Temperate Deciduous Forest. *Ecology*, 63(6)(6):1790–1801, 1982.

[115] Karl von Frisch. Die Tänze der Bienen. *Österreich Zool . Z.*, 1:1–48, 1946.

[116] Karl von Frisch. *Tanzsprache und Orientierung der Bienen*. Springer, Berlin, 1965.

[117] Keith D Waddington, P. Kirk Visscher, Thomas J Herbert, and Monica Raveret. Comparisons of forager distributions from matched in suburban honey bee colonies environments. *Behavioral Ecology and Sociobiology*, 35:423–429, 1994.

[118] Fernando Wario, Benjamin Wild, MargaretăJ. Couvillon, Raúl Rojas, and Tim Landgraf. Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees. *Frontiers in Ecology and Evolution*, 3(September):1–14, 2015.

[119] Fernando Wario, Benjamin Wild, Raúl Rojas, and Tim Landgraf. Automatic detection and decoding of honey bee waggle dances. (in press), 2017.

[120] Anja Weidenmüller and Thomas D. Seeley. Imprecision in waggle dances of the honeybee (Apis mellifera) for nearby food sources: Error or adaptation? *Behavioral Ecology and Sociobiology*, 46(3):190–199, 1999.

[121] Aharon Weissbrod, Alexander Shapiro, Genadiy Vasserman, Liat Edry, Molly Dayan, Assif Yitzhaky, Libi Hertzberg, Ofer Feinerman, and Tali Kimchi. Automated long-term tracking and social behavioural phenotyping of animal colonies within a semi-natural environment. *Nature Communications*, 4, 2013.

[122] Simon Wichmann. Storage and Parallel Processing of Image Data Gathered for the Analysis of Social Structures in a Bee Colony. Master's thesis, Freie Universität Berlin, 2014.

[123] Benjamin Wild. Large scale image analysis in the study of honeybee social behaviour. Master's thesis, 2017.

[124] Fred W. Wolf, Aylin R. Rodan, Linus T.-Y. Tsai, and Ulrike Heberlein. High-Resolution Analysis of Ethanol-Induced Locomotor Stimulation in Drosophila. *Journal of Neuroscience*, 22(24), 2002.

[125] Yonghong Xie Yonghong Xie and Qiang Ji Qiang Ji. A new efficient ellipse detection method. *Object recognition supported by user interaction for service robots*, 2(c):957–960, 2002.

[126] Xin-She Yang. Engineering Optimizations via Nature-Inspired Virtual Bee Algorithms. In *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, pages 317–323. Springer Berlin Heidelberg, 2005.

[127] Karel Zuiderveld. Contrast limited adaptive histogram equalization. In Paul S. Heckbert, editor, *Graphics Gems IV*, pages 474–485. Academic Press Professional, Inc., San Diego, CA, USA, 1994.

Zusammenfassung

Diese Dissertation beschreibt die Entwicklung und Implementierung eines *BeesBook* Systems, welches ein Bildverarbeitungsverfahren zur automatisierten Erkennung und Analyse des Verhaltens von Bienenstöcken auf der Ebene einzelner Individuen sowie des kollektiven Verhaltens ermöglicht.

Verhaltensanalysen von Bienenpopulationen setzen umfangreiche Daten voraus, die das Verhalten einzelner Mitglieder der Population beschreiben. Diese Daten müssen in der Regel manuelle erzeugt werden, welches eine zeitintensive und aufwändige Aufgabe darstellt. Folglich waren Verhaltensdaten bisher nur auf kleine Teilbereiche (bezogen auf Zeit, Raum und Identifizierung der Bienen) des Populationslebens beschränkt. Die automatisierte Datengewinnung des *BeesBook* Systems erlaubt es, eine hohe Anzahl von Individuen über längere Zeiträume zu beobachten, woraus sich zahlreiche Möglichkeiten für umfassende und inklusive Untersuchungen ergeben.

Das *BeesBook* System verwendet eindeutige, binäre Markierungen, um die Position und Identität einzelner Individuen mit Hilfe von Bildverarbeitungssoftware zu bestimmen. Abhängig von der Populationsgröße und den Zielen der Untersuchung erlaubt dieses flexible Design der Markierungen die Implementierung vielfältiger fehlerkorrigierender Codes. Die an die Thoraxform der Biene angepassten Markierungen bilden eine Oberfläche, die den durch die verschiedenen Aktivitäten in- und außerhalb des Bienenstocks hervorgerufenen Belastungen standhält.

Um die Leistung der einzelnen Systemkomponenten bewerten und verbessern zu können, wurden insgesamt drei Experimente durchgeführt. Die Untersuchungen wurden im Sommer der Jahre 2014, 2015 und 2016 durchgeführt und dauerten jeweils neun Wochen. Insgesamt wurden ca. 65 Millionen Bilder aufgenommen. Vor Beginn der jeweiligen Untersuchung wurde jedes Mitglied der Bienenpopulation markiert und in einen Beobachtungsstock überführt. Die Aktivitäten innerhalb des Bienenstocks wurden mit vier hochauflösenden Kameras aufgenommen. Die so erzeugten Daten wurden auf einem Komplex des norddeutschen Verbundes für Hoch- und Höchstleistungsrechnen gespeichert. Die Schwänzeltänze wurden in Echtzeit mit einem zweiten Set von Kameras identifiziert, welches aus zwei Hochgeschwindigkeits-Webcams bestand. Während der drei Untersuchungszeiträume wurde das experimentelle Design hinsichtlich der Eignung der erzeugten Daten zur Analyse des kollektiven Verhaltens optimiert.

Um die Position, Orientierung und ID jeder markierten Biene zu erfassen, wurden die gespeicherten Bilder unter Zuhilfenahme optimierter Bildverarbeitungssoftware verarbeitet. Anschließend wurden diese Daten weiterverarbeitet, um Bewegungspfade zu erzeugen, welche in Kombination mit den Informationen der Schwänzeltänze ein neuartige Einblicke in das Innenleben eines Bienenstocks erlauben. Die durch dieses System gewonnen Informationen ermöglicht es bereits bestehende Erkenntnisse Bienenverhalten zu validieren. Darüber hinaus hat diese Forschungsarbeit bisher unbekannte Verhaltensdaten erzeugt, die letztendlich unser Verständnis von Bienenstöcken und seinen Schwarmintelligenz erweitern kann.

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich alle Hilfsmittel und Hilfe angegeben habe und auf dieser Grundlage die Arbeit selbstständig verfasst habe.
Ich erkläre weiterhin, dass die Arbeit nicht schon einmal in einem früheren Promotionsverfahren eingereicht wurde.

Berlin, den 22 Dezember 2017

Fernando Wario Vázquez