Freie Universität Berlin
Fachbereich Mathematik und Informatik

Realtime Honey Bee Waggle Dance Decoding System

Alexander Rau
alexander.rau@fu-berlin.de

September 30, 2014

Erstgutachter:	Dr. Tim Landgraf
Zweitgutachter:	Prof. Dr. Raúl Rojas
Typ:	Masterarbeit

Statutory Declaration

I declare that I have authored this thesis independently and that I only used the explicitly specified sources and tools. I further declare that the present work has not been submitted to any other academic award.

Berlin, den September 30, 2014 (Alexander Rau)

Abstract

The goal of this thesis is to present the first realtime honey bee waggle dance decoding system, and to establish a benchmark of detection rates and decoding quality. Honey bee research concerns a wide scientific community, but there are no automatic waggle dance detection and decoding systems in use. Hence the goal of this thesis is to create one.

This master's thesis first introduces the honey bee waggle dance communication and motivates the need for an automatic decoding system. Then it examines related work and concludes that no such systems exist. First I define the software requirements, then I present a detailed implementation. Finally, the presented algorithm is applied to video recordings and used in two experiments. Decoding quality is evaluated with precise video material. Detection results of six and four weeks live observations are evaluated. Decoder system's capability to reconstruct the location of a trained artificial food source is tested.

Evaluation results show that the realtime honey bee waggle dance decoding system can achieve $96.4\,\%$ precision rate and $89.5\,\%$ recall rate with recorded videos. The system shows $78.6\,\%$ and $67.4\,\%$ precision rates for live observations. On average decoded dance positions show $0\,$mm error and a low standard deviation error. Waggle run duration decoding averagely exhibits an error of additional $97.79\,$ms and a standard deviation error of $138.52\,$ms. On average the decoding of honey bee dance directions shows $-1.06°$ error and a standard deviation error of $35.83°$. In the end I discuss different evaluation results and presents further improvements for future works.

Contents

1 Introduction and Motivation 3
1.1 Honey Bees and the Waggle Dance Communication System . . . 3
1.2 Specific Need for Realtime Honey Bee Waggle Dance Decoding . 4
1.3 Summary 6

2 Related Work 7
2.1 Honey Bee Hive Surveillance 7
2.2 Honey Bee Waggle Dances Analysis 8
2.3 Summary 8

3 Implementation 9
3.1 Software and Hardware System Requirements 9
3.1.1 Software Requirements 9
3.1.2 Hardware Requirements 11
3.2 Software Implementation 11
3.2.1 Process and Layers Overview 11
3.2.2 Initial Setup 13
3.2.3 VideoFrameBuffer 13
3.2.4 Input Frame Preprocessing 14
3.2.5 Waggle Dance Decoder Layer 1: DotDetectors 15
3.2.6 Waggle Dance Decoder Layer 2: Signals 18
3.2.7 Waggle Dance Decoder Layer 3: Waggle Runs 19
3.2.8 Decoding and Saving Waggle Runs 19
3.3 Summary 19

4 Evaluation 21
4.1 RoboBee Evaluation 21
4.1.1 RoboBee Source 21
4.1.2 Evaluation Method 22

4.1.3 RoboBee Evaluation Results 22
4.2 Beehouse and BeesBook Evaluation 27
4.2.1 Beehouse and BeesBook Source 27
4.2.2 Evaluation Method 28
4.2.3 Beehouse and BeesBook Evaluation Results 29
4.3 Conditioned Bees Evaluation 30
4.3.1 Conditioned Bees Source 30
4.3.2 Evaluation Method 31
4.3.3 Conditioned Bees Evaluation Results 31
4.4 Summary 32

5 Discussion and Outlook **34**
5.1 Evaluation Results Interpretation 34
5.1.1 Waggle Run Detection 34
5.1.2 Waggle Run Position Decoding 36
5.1.3 Waggle Run Duration Decoding 36
5.1.4 Waggle Run Orientation Decoding 37
5.1.5 Beehouse and BeesBook Evaluation Difference 38
5.1.6 Decoder System Design Drawbacks 38
5.2 Outlook 40

A Appendix **44**

Chapter 1

Introduction and Motivation

In biology the use of Computer Vision (CV) for measuring animal behavior has increased in recent years. Advantages of such approaches include a faster analysis, absence of human bias and the possibility to process big amounts of data. In fact, detection of honey bee waggle dances is an adequate example of such a scenario. In this master's thesis I present the first automatic realtime honey bee waggle dance decoding system.

Background information about the honey bee waggle dance is briefly introduced in the following section. In the subsequent section, I motivate the need of a realtime waggle dance decoder system. For this purpose one specific project is presented, which already uses the waggle dance decoder. Additionally, another recent study is shown, which could have benefited from an automatic waggle dance decoder system. A short summary and the outline of this master's thesis structure is given in the last section.

1.1 Honey Bees and the Waggle Dance Communication System

It was von Frisch's research and discovery which initially started growing interest in the honey bee waggle dance within the scientific community[5]. Due to his publications the honey bee waggle dance is recognized as one of the most sophisticated forms of communication in the animal kingdom. This communication is crucial.

For the survival of the honey bee colony nutrition in form of nectar and pollen is needed. Those food resources are collected by foraging honey bees from blooming plants, which change with time of the season. Multiple research proof that successful forager, upon their return to the hive, tend to communicate locations of discovered profitable food sources[10, 11, 13]. Other honey bees, which at-tend dances closely, can decode locations and therefore help to quickly exploit rich nutrition sources.

The process of information transfer can be observed and is known as the *honey bee waggle dance*. In order to cause a waggle dance the distance between hive and indicated food source has to be approximately further then $150\,\mathrm{m}$[13]. I distinguish the *waggle run phase* and the *return phase*. Waggle dances consist of concatenated waggle run and return phases.

Usually honey bees start with a waggle run phase. Hereby honey bees slowly move on an ideal straight line, while vigorously flipping their tail from one to the other side. Under natural conditions and with red light illumination honey bees can not see in the dark hive. But for humans this single waggle run visually encodes all necessary geographical information to find an advertised food source. Bearing and distance is communicated as waggle run direction and duration.

Honey bees translate the direction of the food source from one reference system to another. On the outside, honey bees use the sun's azimuth[1] as a reference for navigation. Inside the hive, this reference is translated into the opposite direction of gravitation force. Abrevations from the opposite gravitation direction are translated to accroding deviations to suns's azimuth.

With the hive's GPS coordinates and a dance timestamp one can recalculate the sun's azimuth. Finally, any recorded waggle run direction can be translated into a real world bearing.

The distance towards a food source is encoded in the waggle run duration. Studies show that communicated distances depend on the perceived optical flow[2] and therefore suggest that a linear mapping should be calibrated for each individual hive location. Other research also assume a linear function which maps waggle run duration to distance[1, 2, 13]. After a waggle run, in the return phase honey bees walk a half circle shaped path back to the same starting location. Because dancers alternately go back turning left or right, one can depict the movement paths as a figure eight (see figure 1.1).

1.2 Specific Need for Realtime Honey Bee Waggle Dance Decoding

Research on honey bees has been conducted for decades, but still questions remain unanswered. Details about function and mechanisms of honey bee waggle dance communication are one of the open fields in current research.

[1]angle between north pole and sun's perpendicular projection towards horizon on the earth's surface

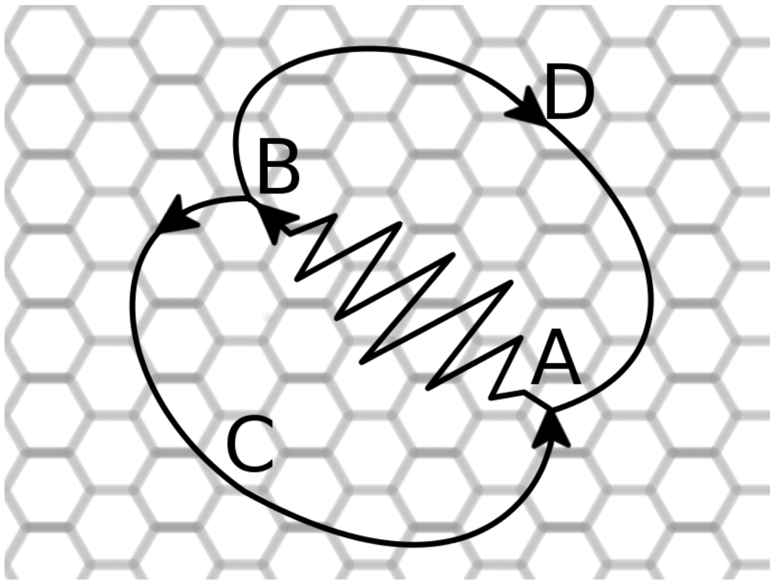

Figure 1.1: Ideal honey bee waggle dance path. Waggle run phase starts at ‘A’ and continues until ‘B’. Then path ‘C’ or ‘D’ are alternately walked in the return phase, back to ‘A’.

In order to learn how precisely honey bee dance followers decode transmitted information Landgraf et. al.[10] analyzed detailed video recordings of waggle dances as part of his RoboBee project. Using movement statistics, an artificial honey bee dance model was created, which could perform waggle dances as instructed. With this RoboBee project a successful recruitment behavior could be observed.

However, one result of this experiment formed the hypothesis that honey bees could prefer certain hive areas for dancing. Furthermore, honey bees could form social subgroups, because follower bees seemed to prefer specific dancing bees.

In order to answer this questions, the social behavior of honey bee colonies has to be tracked and analyzed. The resulting social graph can be used to unveil cliques of honey bees, which tend to interact among themselves. Project BeesBook was initiated in order to perform these surveillance experiments and gain detailed social data.

Currently, all studies, which concern honey bee waggle dance research, share a common approach. This technique involves huge hard drive capacities for storage and the burden of manual review[1, 2, 3, 10, 11, 12]. As a result the amount of possibly analyzed data is limited in some way. One reason is that high video frame rate and optical resolution are necessary for manual review, which makes video files grow fast. Another reason is the time consumption while manually searching recordings for waggle dances. In order to overcome such limitations, in the context of the BeesBook project, I created a new automatic realtime honey bee waggle dance decoding system.

Another study at the University of Sussex[1] experienced the drawbacks of manual waggle dance decoding. Over the period of two years one hour per day of hive observation was recorded. In total approximately 5000 waggle dances were decoded by hand.

In their work Couvillon et al.[1] use decoded waggle dance distances as indicators of seasonal foraging challenges. It is claimed that decoded, far flight distances may be used as a signal to inform about shortages in hive food supply. With prevailing honey bee extinction in mind, the presented work could become a useful tool. In case a nutrition scarcity is detected artificial food could be deployed.

However, this tool is another example for automatic realtime honey bee waggle dance decoder application. The presented approach has no practical use if decoding has to be done manually.

1.3 Summary

The honey bee waggle dance is a unique body language and communication form among the animal kingdom. Research fields of understanding biological dance signal processing still remain open. Real demand for automatic waggle dance decoding is exemplified with the BeesBook project and Sussex study.

In the following chapter 2 related work in the field of honey bee hive and dance observation is presented. It is showed that there are no comparable automatic waggle dance decoding solutions to find. Waggle dance decoding system requirements and a detailed implemented solution is presented in chapter 3. In subsequent chapter 4 evaluation results for recorded high precision data, two live observation sites, and live honey bee conditioning is presented. Decoder system capacity is shown to achieve $89.5\,\%$ precision rate and $96.4\,\%$ recall rate with recorded videos. Live observation experiments show precision rates of $78.6\,\%$ and $67.4\,\%$. Discussion of evaluation results and an outlook on future developments are presented in the final chapter 5.

Chapter 2

Related Work

In this section related work in the field of honey bee observation is presented. It is indicated if involved methods are capable to **automatically detect and decode** honey bee waggle dances in **realtime**.

First studies related to honey bee colony tracking and observation are examined. Then research concerning honey bee waggle dance analysis is presented. Finally, a section summary is given.

2.1 Honey Bee Hive Surveillance

New insights on the social behavior can be gained by observing and tracking honey bee colonies. Some experiment setups allow to collect data down to the scale of individual honey bees.

Computer aided observation and classification of brood cells helps to identify "hygienic bees"[9]. In another study Klein et. al.[8] tracks about 100 honey bees over two years and analyzes their sleeping positions. For a continuous identification the observed honey bees were marked with color codes on their thorax.

According to Kimura et. al.[6, 7] tagging of honey bees could lead to abnormal behavior and therefore he introduces new sophisticated tracking methods. Using a "vector quantization method" Kimura et. al.[6] successfully tracks approximately 3 out of 4 bees in a colony. However, this results are achieved by evaluating only 3 randomly chosen frames out of a 10 seconds record, which contains 300 frames. For his results Kimura et. al. claims to successfully extract 2 waggle dance trajectories. There is no clear description if Kimura et. al.'s system is able to automatically detect a waggle dance. However, the presented solution is unsuitable for realtime honey bee waggle dance decoding.

2.2 Honey Bee Waggle Dances Analysis

A number of research involves analysis of honey bee waggle dances[1, 2, 3, 10, 11, 12]. Over a time span of 15 years, methods to analyze waggle dances stay almost the same. Honey bees are recorded with high resolution and fast frame rates, then manual decoding follows. One can observe improvements in the field of computer aided dance data extraction.

For creation of high-precision waggle dance trajectory data Landgraf et. al.[10] uses an automatic tracking system. The initial position has to be setup manually and subsequent tracking is overseen manually. Landgraf et. al.'s system does not automatically detect waggle dances, but decode them. With this method Landgraf et. al.[10] statistically evaluates 109 waggle dances, which are composed of 1009 waggle runs. In the end, this approach is not applicable in realtime and it lacks automatic waggle run detection.

Another study investigates the importance of dance follower's tactile stimuli with respect to successful information decoding. In her work Gil et. al.[3] evaluates behavior of follower bees within a set of 40 dances. Dance data is extracted by manual review. Neither automated waggle dance detection nor decoding is used.

Investigating honey bee swarm decision-making Seeley et. al.[12] observes waggle dances which advertise new hive locations. Dance detection and information extraction is manually executed. In his report one can identify the burden of manual waggle dance decoding. Seeley et. al. says that the analysis of 16 h movies takes more then 250 h work time.

Even 15 years later in the year 2014, a current study conducted by Couvillon et. al.[1] uses the same dance decoding method. Researchers daily record three separate hives for one hour, over 189 days per year in two consecutive years. Movies are evaluated by slow motion replay and manual extraction.

There are no signs of an automatic honey bee waggle dance detection or decoding system which can execute in realtime.

2.3 Summary

In this part it is shown that there exist a number of research topics, which are related to honey bee waggle dance communication and it's visual decoding.

But only two works[6, 10] exhibit capabilities to automatically decode waggle dances. However, both approaches lack the ability to automatically detect waggle dances and to perform in realtime.

Chapter 3

Implementation

In this section implementation details of the realtime honey bee waggle dance decoding system are presented. In the first part software and hardware system requirements are described. Technical implementation details are shown in the second part. In the end a section's summary is presented.

3.1 Software and Hardware System Requirements

Because system requirements are determined by their goals, I introduce a verbal definition of the target. Essentially, one can comprehend the target as follows:

"Suppose an observation honey bee hive and adequate hardware, then monitor comb surface and use Computer Vision to automatically detect and decode waggle runs in realtime."

Terms "observation honey bee hive" and "adequate hardware" are intentionally described vaguely. Details about laboratory setups play a subordinate role, because this master's thesis main focus is to contribute a new software approach. The knowledge how to prepare observation hives is widely available.

In the following, key features of software requirements are investigated. Subsequently, an example of utilized hardware specifications is presented.

3.1.1 Software Requirements

Recalling the previously formulated goal one can extract the following key software features:

a) automatic

b) waggle run detection

c) waggle run decoding

d) realtime

Subsequently, each feature and reasons for it's importance are explained.

Automatic

The ability to automatically detect and decode waggle runs is a fundamental aspect of the realtime honey bee waggle dance decoder system. Currently, users have the burden to manually review movie material and execute numerous monotonous decisions and decoding actions. An automatic aspect therefore multiplies volume of data that can be analyzed during the same amount of time. It removes limits on observation time, because the system can perform automatically 24 h a day. However, implementation of the automatic algorithm has to guarantee to meet the realtime processing requirement.

Waggle Run Detection

The first step of the waggle run decoding is it's detection. Because multiple waggle dances can appear simultaneously, the system must be able to detect them all. Information within a single detection has to contain dancing bee's position and timestamp of detection.

Waggle Run Decoding

When a waggle run is detected, visually observable information has to be decoded. Decoding addresses three main dance properties. A decoder system needs to extract positions over the period of the dance, amount of dance time, and dance direction. Dancing bees are uniquely identified with collected positions. Detection timestamp, dance duration, and dance direction can be combined and translated into GPS coordinates of the advertised food location. Basically, this three informations are similar to manually decoded waggle dance data.

Realtime

It is significantly important that the waggle dance decoding system can operate in realtime. Conventionally, studies save high resolution video material and manually process it later. If the waggle dance decoding system can process video input streams in realtime storage of full movies becomes obsolete. Because information of interest is automatically extracted “on the fly” only necessary data is saved. One can conduct long time observations using common hard disk drives. Maintenance due to an overloaded harddrive can be virtually avoided. This use case is referred to as “online” or “live”, because system processes visual input which is gained just in that moment and no video record is available.

Consequently, there exists an opposite “offline” or “recorded” use case, i.e. if movies of waggle dances already exist. In this scenario, a realtime property is also beneficial: waggle dance decoder processing time has an upper bound equal to the amount of time recorded. In the end, it is the user who can save many working and processing hours.

3.1.2 Hardware Requirements

The hardware requirements I note in this section are reduced to a minimum and are based on values I use for my research. Additional equipment that is needed for an observation setup is left out of scope. Because details of experiments are subject to individual setups, it is impossible to cover all possibly involved hardware details.

Generally, hardware specifications for a live operating realtime honey bee waggle dance decoding system are very common. The two main classes of items are a camera and a computer.

I suggest a camera that is capable of 100 Hz frame rate. For my live observation slightly modified PlayStation Eye Cameras are used. At their setup position all four cameras observe the comb surface with approximately 0.43 dpmm resolution. The video frames are downsampled to 160 px width and 120 px height.

A standard computer is able to concurrently process two cameras, which are connected over USB 2.0 interface at two separate USB controllers. As CPU an Intel i7-26000 with four cores, running at 3.4 GHz is utilized. One process consumes less then 150 MB RAM. Size of detection files usually is less then 1 GB per day.

3.2 Software Implementation

In this part implementation details of the realtime honey bee waggle dance decoding system is explained. First, a top down view is presented and the initial setup process is outlined. Then all consecutive steps in the process of dance decoding are described in detail.

3.2.1 Process and Layers Overview

When a new observation experiment starts the decoder system for the first time, a simple manual configuration process has to be done. Afterwards, the schema de-

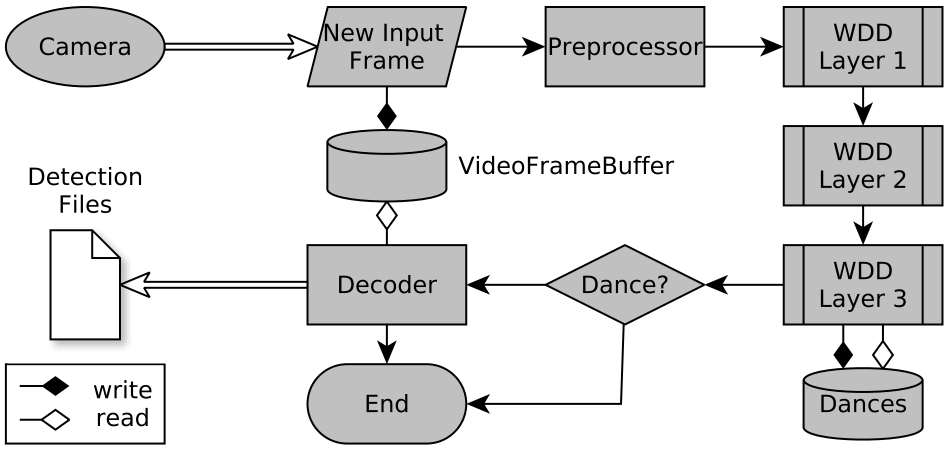

Figure 3.1: Overview of decoder system when processing a new video frame. New input frame is saved to VideoFrameBuffer before being preprocessed. Then all waggle dance decoder layer subsequently handle further signals. If a new dance detection is created the decoding step finally saves results to harddrive.

picted in figure 3.1 illustrates video frame processing steps. In detail the following actions are executed:

1. Acquire next input frame and temporarily store it
2. Transform input frame into target frame
3. Run signal detection (layer 1): DotDetectors with new target frame
4. Run signal detection (layer 2): layer 1 signal clusters
5. Run signal concatenation (layer 3): layer 2 signals over time
6. Extract terminated waggle runs: write decoded dances to files

In the first step, the camera driver is responsible to acquire each new video frame from the camera device. Any camera device and API may be used, as long as "raw frames" can be stored within *cv::Mat*[1] objects.

An entity that manages temporary image storage is called *VideoFrameBuffer*, a basic ring buffer implementation with additional features.

[1]as part of OpenCV 2.4.9 library it is used to store image data

The second preprocessing step assures that input frame formats are transformed into the expected target frame format. At this point the interface between observation input and further internal processing is embodied.

All remaining steps are part of the *WaggleDanceDecoder* entity, that for itself consists of different process layers and entities.

3.2.2 Initial Setup

The goal of an initial setup is to save absolute pixel position values of a rectangle that overlays the observed honey comb frame (see figure 5.1 for an illustration example). This setup is started if the waggle dance decoder system can not find a fitting configuration on hard drive. Then a live view, with camera's maximum video frame resolution, is drawn on the screen. The user can first adjust camera position to fit the scene and then finally select four corners of an *arena rectangle*. After user confirmation, an internal camera id is mapped to the camera hardware id and it's corresponding arena configuration. Finally, this setup is persistently saved into a file. In case of a system restart, camera hardware ids are matched against known configurations, which are read from file. Therefore the initial setup has to be executed once, if no change in observation conditions occurs.

The arena configuration has multipurpose use. First, if one knows real world size of the mapped comb frame it is possible to calculate real world distances, approximate video resolution in dpmm, and rectify camera views. Second, one is able to verify that no significant alteration in camera position occurred, because camera video stream is printed live on screen and the arena is drawn on top. Third, the processed detection area can be reduced to the size of the arena.

3.2.3 VideoFrameBuffer

The purpose of VideoFrameBuffer is to provide access to a short duration of recent video records. Stored frames are copies of camera input frames. As an upper bound for waggle run durations I use 600 frames of history. Like a ring buffer, the oldest frame is continuously overwritten with new frames; static memory allocation is implemented. Additional methods permit access and copy of a cropped image sequences. Hereby extract size can be configured according to input image resolution and images center view position is passed as an argument.

The VideoFrameBuffer is used to provide a visual sequence of detected waggle runs, when decoded data is written to harddisk. However, other utilization of this image sequences are imaginable, as I discuss in section 5.2.

3.2.4 Input Frame Preprocessing

The main goal of preprocessing is to achieve a transformation of any *input frame format* into *target frame format*. In the following the two formats are described and subsequently a transform function is presented.

Target Frame Format

In order to describe implementation details of the realtime honey bee waggle dance decoding system I define some symbolic representations.

I write f^i when the *i*-th video frame is in target format and call it target frame. When I refer to properties of f^i, it is expressed as f^i_{width}, f^i_{height} and $f^i_{colorspace}$. Attributes *width* and *height* are positive integer values. The property *colorspace* represents the name of the color space model. A colorspace representative is *GRAYSCALE*, which encodes image intensity as 8 bit integers, with values from 0 to 255. For any frame object Ξ the function $\mathrm{format}(\Xi)$ evaluates attributes as:

$$\mathrm{format}(\Xi) = [\Xi_{width}, \Xi_{height}, \Xi_{colorspace}]$$

Because target frame format is final, any frame has the same properties. Target frame format therefore can be written as:

$$\mathrm{format}(f^i) = \mathrm{format}(f) = [f_{width}, f_{height}, \text{GRAYSCALE}]$$

Values for f_{width} and f_{height} are application depended.

Input Frame Format

Like a target frame, I name the *i*-th video frame with input format $\widetilde{f^i}$ and call it input frame. Property types are identical with a target frame. Input frame format therefor can be defined as:

$$\mathrm{format}(\widetilde{f^i}) = \mathrm{format}(\widetilde{f}) = [\widetilde{f}_{width}, \widetilde{f}_{height}, \widetilde{f}_{colorspace}]$$

For an input frame format no assumptions about the colorspace are made.

Transform Function

In essence the preprocessing step has to apply a transform function $t()$, that ensures following equation:

$$\mathrm{format}(t(\widetilde{f_i})) = \mathrm{format}(f)$$

Because only subsampling makes sense, in any case it is true that:

$$f_{height} \leq \widetilde{f}_{height} \quad \wedge \quad f_{width} \leq \widetilde{f}_{width}$$

It is possible to avoid input frame preprocessing if the acquired input frames already have the target frame format. However, in my experiments input frames have to be transformed. OpenCV offers different subsample methods. I chose to use the simplest cv::INTER_AREA method, without any blur filter. This method simply calculates mean value of pixels and does not obliterate neighbor pixels. Regarding algorithm execution time it is desirable to downsample as far as possible. However, one has to pay attention to decoder system's lower bound on dpmm resolution. For my evaluation I used subsampled images with values between 0.43 dpmm and 0.48 dpmm resolution.

3.2.5 Waggle Dance Decoder Layer 1: DotDetectors

The foundation and core element of the waggle dance decoding system is a concept named *DotDetector*. The term "dot" is equivalent to "pixel" and it illustrates the way detection works. Every DotDetector D_j is assigned to a position $p_j = [x_j, y_j]$, with reference to target frame format properties:

$$0 \leq x_j < f_{width} \quad \wedge \quad 0 \leq y_j < f_{height}$$

The principal goal of a single DotDetector is to produce a signal, when a pattern of certain frequency is observed over time. In the following details that form the basis of a DotDetector and it's calculations of detection signal are presented.

Buffer and Buffer Size

For each D_j a constant buffer B_j with size b is defined. This property controls the amount of consecutive pixel values each DotDetector stores. Given one D_j and it's position $p_j = [x_j, y_j]$ a corresponding pixel value of the *i*-th frame is equal to:

$$v_j^i = f^i(x_j, y_j)$$

At the moment of processing the *n*-th frame f^n, where $n \geq b$, the system processed consecutive frames $f^{n-b+1}, f^{n-b+2}, .., f^n$. At this point, each D_j has filled it's buffer with it's b least pixel values:

$$B_j^n = [v_j^{n-b+1}, v_j^{n-b+2}, .., v_j^n]$$

Each DotDetector saves the pixel values it "sees" as a history of b consecutive frames in it's own buffer. Later this values are used to extract waggle signal features. When the n-th frame arrives, a reference to the m-th element of B_j^n is denoted as:

$$B_j^n(m) = v_j^{n-b+m} \quad \text{with } m \in [1; b]$$

Frequency Configuration

Recent research results show that honey bees waggle with 13 Hz[3, 10] during a dance. DotDetectors are build to detect this movement pattern. During my research, I find the following frequency range and steps to work successfully:

$$\text{FREQ}_{min} = 11 \quad \text{FREQ}_{max} = 17 \quad \text{FREQ}_{step} = 1$$

$$\text{FREQS} = [11, 12, 13, 14, 15, 16, 17]$$

This configuration means that DotDetectors scan their buffers for frequencies from 11 Hz until 17 Hz, with intermediate steps of 1 Hz.

Sine and Cosine Sample Creation

For later frequency score calculation one needs appropriate $\sin$ and $\cos$ samples. In order to save processing time, this values are calculated once at system startup. Pre-calculation requires knowledge of frequency configuration FREQS and video input sample (frame) rate s_R. Number of samples calculated per frequency is equal to s_R. Functions $\sin$ and $\cos$ are sampled at values:

$$2\pi r \frac{1}{s_R} k, \quad \text{with } r \in \text{FREQS},\ k \in [0; s_R - 1]$$

In the following I reference the m-th sample value of frequency $r \in$ FREQS as:

$$\sin_r^m \qquad \text{and} \qquad \cos_r^m$$

Sample values are guaranteed to form a cycle for each frequency, therefore it is true that:

$$\forall r \in \text{FREQS} : \sin_r^{s_R+1} = \sin_r^1 \ \wedge \ \cos_r^{s_R+1} = \cos_r^1$$

One of the advanced implemented techniques is a *continuous calculation*; details on this are shown later.

Frequency Score Calculation

Assume that the *n*-th frame f^n, where $n \geq b$, is to be processed. In preparation of the frequency score calculation each B_j^n is normalized to $\bar{B}_j^n \in \{-1;1\}$ using values $\min_j^n = \min(B_j^n)$ and $\max_j^n = \max(B_j^n)$. Then function $\mathrm{score}()$ calculates for each D_j and frequency $r \in$ FREQS:

$$\mathrm{score}(\bar{B}_j^n, r) = \sum_{1 \leq m \leq b} (\bar{B}_j^n(m) * \sin_r^m)^2 + (\bar{B}_j^n(m) * \cos_r^m)^2$$

At this point, each DotDetector describes *how similar* it's recent b normalized values are to configured oscillation frequencies; amplitude information is not considered, yet. However, it is not always true that frequency score value calculation involves all b buffer positions. Before I finish the implementation details I explain continuous calculation.

Time Boost with Continuous Calculation

Previously, the $\mathrm{score}()$ function calculates values based on *all* values of buffer $\bar{B}_j^n$. However, this is the worst case scenario in terms of computation time consumption. If a new frame f^{n+1} arrives, and at D_j the new pixel value v_j^{n+1} is within range of former pixel values, then only a fraction of full calculation time is needed. Therefore condition for time boost is $\min_j^n \leq v_j^{n+1} \leq \max_j^n$. Using previously saved values one can normalize the single v_j^{n+1} and calculate it's frequency scores using the *next* $\sin$ and $\cos$ samples. Each DotDetector therefore has it's own pointer of it's next sample to use. Because of guaranteed sample value cycle it is possible to retrieve only new additional score values *continuously*. Of course "oldest" score values need to be removed.

Signal Generation

At this stage every DotDetector has it's appropriate frequency scores saved and a detection decision has to be made. With the following steps for this purpose a *potential* is calculated for each D_j. First, frequency scores are sorted ascending. Then the sum of all frequency scores is formed, each multiplied with a weight function $w()$. This function is equal to frequency score sort position. Higher scores are linearly weighted higher. Finally, this sum is multiplied by amplitude $A_j^n = \max_j^n - \min_j^n$. The function $\mathrm{potential}$ can be written as:

$$\mathrm{potential}(D_j) = A_j^n * \sum_{r \in \text{FREQS}} w(\mathrm{score}(\bar{B}_j^n, r)) * \mathrm{score}(\bar{B}_j^n, r)$$

The last step is to evaluate the $\text{signal}()$ function, which involves the DotDetector signal threshold θ:

$$\text{signal}(D_j) = \begin{cases} 1 & \text{if potential}(D_j) > \theta \\ 0 & \text{else} \end{cases}$$

An example view of the process is placed in appendix figure A.1. Once this layer finishes, next layer 2 continues to process positive DotDetector signals.

Dynamic Number of DotDetectors

In a naive implementation one can expect to have the same number of DotDetectors as target frame pixels:

$$N_{DotDetectors} = f_{height} * f_{width}$$

However, as described in section 3.2.2, the arena shrinks detection area and therefore DotDetectors are only placed within that limits.

A lot of calculation time can be saved with another small improvement. The number of *active* DotDetectors is dynamic and depends on the value of A_j^n. Because a waggle pattern is expected to be accompanied with a strong amplitude it is safe to skip further processing when a certain limit is not reached.

3.2.6 Waggle Dance Decoder Layer 2: Signals

The goal for layer 2 is to determine clusters of signals received from layer 1. Outcome of decoder layer 2 is mainly controlled by two parameters.

In order to associate single layer 1 signals with each other a maximum distance $d^2_{\max}$ is introduced. With respect to subsampling settings, this value has to be set approximately to the scaled height of a honey bee. If two DotDetectors D_i and D_j have an euclidean distance lower then $d^2_{\max}$, they belong to the same cluster and therefore to the same honey bee.

With the intention to filter noisy signals minimal cluster size $c_{\min}$ is used. This value has to be adjusted according to the potential threshold value θ used at layer 1, which regulates the sensibility of detection. Higher values of θ implicate lower values of $c_{\min}$. In the opposite case, lower values of θ need higher values of $c_{\min}$. However, a big value of $c_{\min}$ will impact calculation time and therefore endanger the realtime capability.

At the end of layer 2 for each cluster, that at least connects $c_{\min}$ positive DotDetectors, a signal is created. Waggle position is calculated as the average of clustered DotDetector positions. In physical terms, layer 2 signals are equivalent to a waggle movement detection at the time of processing frame f^n. Cluster signals are passed on with their position information to layer 3.

3.2.7 Waggle Dance Decoder Layer 3: Waggle Runs

Layer 3 concatenates separate, consecutive waggle movement detections over time. Three parameters regulate the outcome of this processing step.

In order to determine if a layer 2 signal can be concatenated to an existing layer 3 detection, the spatial property is of interest. Similar to layer 2, a maximal distance for signals is defined as $d^3_{\max}$. The value has to respect scaled bee size in target frame format. Waggle movement signals are matched against existing waggle run assumptions. If no waggle run assumption is matched, the system initiates a new one. Otherwise, the layer 2 signal is added to the fitting waggle run assumption.

For the purpose of smoothing layer 2 signals a maximum frame gap parameter $g_{\max}$ is used. It is possible that waggle movement detection experiences some gaps, i.e. a drop beneath $c_{\min}$. On the other side, if any waggle run assumption does not receive a fresh signal for more then $g_{\max}$ frames, it receives a timeout and is checked at the final following step.

Any timeout waggle run assumption, that has less consecutive frames then parameter $l_{\min}$, is dropped. This parameter is useful to set the minimal duration of waggle run detections. Finally, if an assumption is declared to be a waggle run detection the information package is passed to the subsequent, last decoder system step. There, detected waggle runs are decoded and extracted to harddrive.

3.2.8 Decoding and Saving Waggle Runs

Once a waggle run is passed to the extraction layer, information collected during detection is used to decode dance properties.

Since position information is part of waggle run detection, it's decoding is a simple copy. The timestamp of detection is saved within layer 2, when a new waggle run assumption is created. Duration information is decoded as the number of frames, which are concatenated at layer 3. Finally, first and last honey bee positions are used to decode waggle run orientation.

Once this data is extracted and collected, it is written within a specific folder structure into a file. Additionally, VideoFrameBuffer is used to retrieve a cropped frame sequence of the waggle run. This image sequence is also saved into the detection folder and can be used for later review.

3.3 Summary

In this part formal requirements of a software implementation are shown. The goal is to implement an *automatic* algorithm, that can *detect* and *decode* waggle runs in *realtime*. Hardware requirements are rather ordinary in the area of honey bee observation experiments. Main element of the decoder system is the concept of a DotDetector, which detects characteristic 13 Hz patterns. This pixel based detectors can determine a frequency score for a history of observed grayscale pixel values and decide to produce a positive signal. The 2nd layer puts those signal together with respect to spatial distribution. Aspect of signal time and waggle run termination is processed in the 3rd layer. Finally, for each positive waggle run detection appropriate information about time of occurrence, position, duration, and orientation is written into a file. An image sequence, which depicts the detection area, is additionally written to the folder.

Chapter 4

Evaluation

Evaluation results of the realtime honey bee waggle dance decoding system are examined in this section. Table 4.1 gives an overview of available evaluation material. First the decoder system performance is measured against recorded RoboBee

data name	**time** [h]	**size**($\widetilde{f}$) [px]	s_R [Hz]	**size**(f) [px]	**kind**
RoboBee	0.732	640x480	100	40x30	recorded
Beehouse	~1150	320x240	102	160x120	live
BeesBook	~740	320x240	102	160x120	live

Table 4.1: Overview of evaluation data sources.

data. Then live detection results for the Beehouse and the BeesBook experiments are reported. Finally, results of a honey bee conditioning experiment are presented.

Each following evaluation part has the same structure. First the origin of used material is explained, then the method of evaluation is declared, and finally results are reported.

4.1 RoboBee Evaluation

4.1.1 RoboBee Source

The RoboBee data set consists of 16 videos and additional metadata files. This includes trajectories for 207 waggle runs. The records date to 2008-08-22 starting from 11:55 until 13:01. Metafiles and video recordings are part of data Landgraf et. al.[10] used. The source of this evaluation part is a data set with precise waggle dance trajectories and high quality video recordings. Those videos capture honey bees within an observation hive under weak red light illumination. The input frame format is equivalent to:

$$\mathrm{format}(\widetilde{f}_{\text{RoboBee}}) = [640, 480, \text{GRAYSCALE}]$$

During one movie, the camera position remains fixed. My measurements determine an input video resolution of approximately $7.78\,$dpmm. For each dancing bee an associated metadata file provides beginnings and ends of dances, which are stored as video frame numbers. With frame rate s_R one can calculate dance **duration** information. Additionally, metadata files store **position** and body **orientation** within dance phases. This trajectory data is updated for each video frame and consequently has a $10\,$ms time resolution.

Landgraf et. al.[10] manually reviewed and published this data. For the purpose of my evaluation I declare the available RoboBee data set as ground truth.

4.1.2 Evaluation Method

Decoder system's decoding quality is evaluated by contrasting it's results with RoboBee data. For each processed video a response file with decoded waggle run information is created. It's content is similar to information available in RoboBee metadata files. The realtime honey bee waggle dance decoder system uses the same set of parameters for all videos, which in detail are listed in appendix table A.1.

A semi automated approach is used to match waggle runs between RoboBee data and decoder system output. For one video, each RoboBee waggle run is transformed into a distribution of it's positions and it's frame numbers. Then for all decoded waggle runs a Mahalanobis distance towards all distributions is calculated. If this distance is below a certain limit, the RoboBee waggle run is matched with the shortest distant decoded waggle run. Otherwise RoboBee data remains unmatched and becomes a false negative.

In the end, all automatic assignments are manually reviewed and correctness is verified using manual visual feedback. For this purpose, video segments of dance phases are replayed while metadata and decoder data is drawn simultaneously on top. As depicted in figure 4.1 matched waggle runs are recognized by sharing the same symbol. Those matched waggle runs are used to determine the realtime honey bee waggle dance decoder system's decoding quality.

4.1.3 RoboBee Evaluation Results

Out of 207 available RoboBee waggle runs the realtime honey bee waggle dance decoding system detects 199 and reports additional 25 false positives. Counting

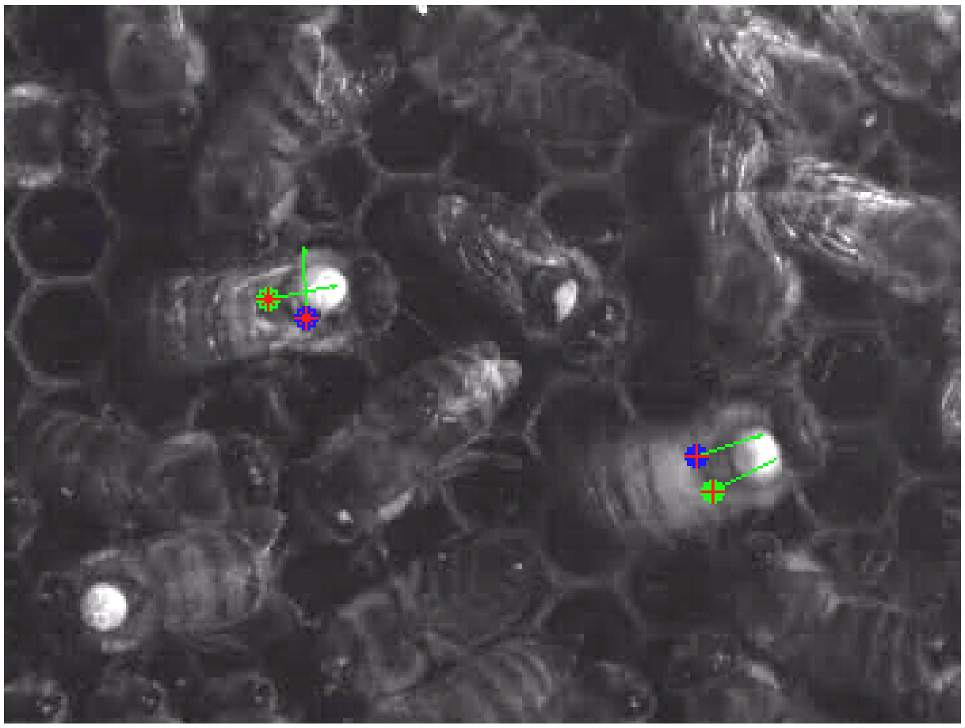

Figure 4.1: Screenshot of visual feed, back which is used to manually review automatic matches of waggle runs. Green circles represent matched RoboBee data and blue circles depict decoder system data. Inside the circles red '*' and '+' symbols indicate matched pairs. Arrows indicate the according honey bee dance orientations.

only RoboBee data, the number of false negatives equals 8 and decoder systems's recall rate is $96.1\,\%$. Consequently, $88.8\,\%$ precision rate is achieved.

However, the decoder system also detects 14 additional waggle runs, which are not included within RoboBee data. One can also present results which include all 221 real waggle runs. In this case, the decoder system reports 238 detections, with 213 true positives and 25 false positives. Hereby recall rate equals $96.4\,\%$ and precision rate equals $89.5\,\%$. A detailed overview of detection performance is listed in appendix table A.2. Subsequent decoding quality results are based on the associated data of 199 waggle runs.

Time Decoding Results

The decoder system's time decoding quality is evaluated with 199 samples, which are equal to the 199 matched waggle runs. Waggle run phases are coded as video frame numbers. A translation from frame numbers into time units is done using the frame rate. Signed gaps for start and end frame numbers are collected. A distinction between start and end timings allows a richer view on time decoding quality. The total frame counts of waggle runs are also compared.

Results show that the decoder system delays decoded waggle run start timings. On average the decoder system lags 111 ms behind, with a standard deviation of 101 ms. Decoding of waggle run ends lags more. Hereby the average delay is 209 ms, with a standard deviation of 79 ms. Both data distributions are depicted in figure 4.2.a and 4.2.b. Final results show that on average the decoder system overestimates decoded waggle run durations by 98 ms, with a standard deviation of 139 ms[1].

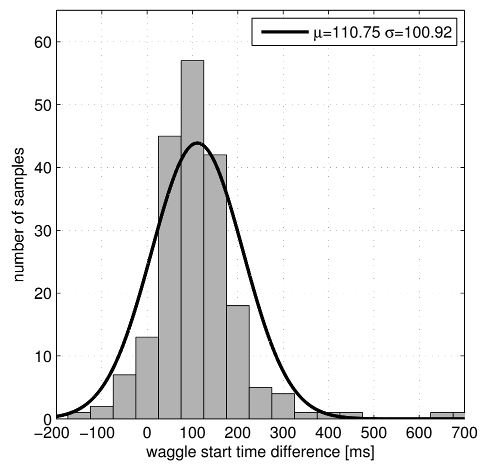

Figure 4.2.a

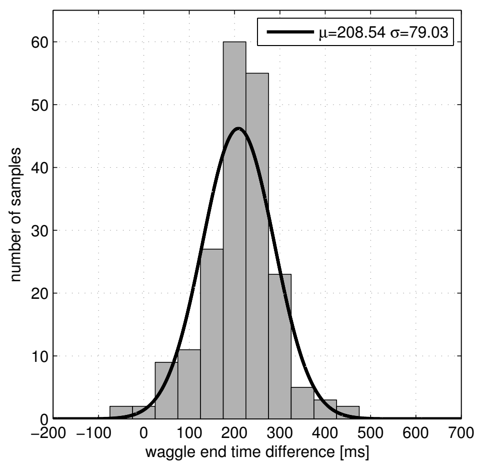

Figure 4.2.b

Figure 4.2: Distributions of decoded time errors for 199 waggle run samples. Decoded waggle run start time error is plotted in 4.2.a and decoded end time error in 4.2.b respectively.

[1]Evaluated waggle run duration decoding error distribution is depicted in appendix figure A.2

Position Decoding Results

The decoded positions of dancing bees are updated with every frame. In order to compare RoboBee position data with decoder system's decoded positions sets of intersecting frames are calculated. In total 5861 comparable position pairs are available out of this intersection sets. The RoboBee video resolution is used to transform pixel distances into millimeter units. For each position pair an offset vector is computed, which points from RoboBee position towards decoder system position. Hereby realtime honey bee waggle dance decoder positions are scaled from initial f size to $\widetilde{f}$ size with factor 16. The frame based body orientation of RoboBee data is used to project offset vectors into the bees body axis. This way, video frame based coordinates are translated into relative bee coordinates. A clear picture of decoded position errors, with reference to the bee's body, is achieved. The 3d histogram in figure 4.3.a shows high sample counts near 0 mm error. In

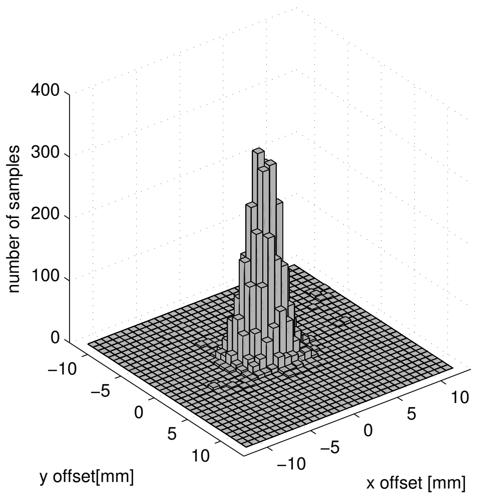

Figure 4.3.a

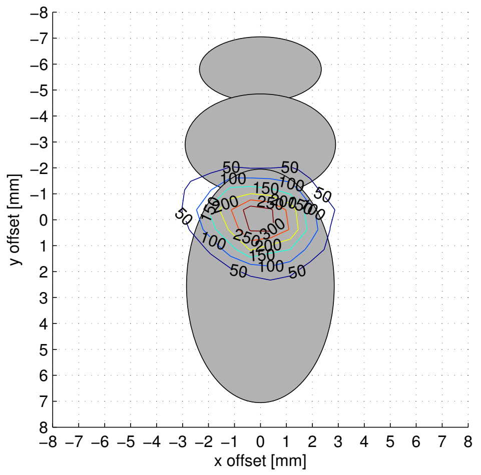

Figure 4.3.b

Figure 4.3: Distributions of decoded position errors for 5861 location samples. A 3d histogram in figure 4.3.a shows the highest number of samples at the center region. In figure 4.3.b the same data is viewed from above and contour lines refer to the size of bins within that area. A gray bee body shape is used as size reference. Note that the standard deviation error of x axis is visibly higher then for the y axis.

figure 4.3.b the previous histogram is translated into contour lines and projected on top of a bee's body shape. Contour line values indicate the size of bins within it's enclosed areas. With a 2d view from above one can see an approximately concentric distribution of decoded position errors.

One can research normal distribution parameters of decoded position errors separately for the two axes. Hereby the bee's body orientation, which equals y axis, has it's mean error value at $0.02\,\text{mm}$ with a standard deviation of $1.20\,\text{mm}$. The orthogonal x axis shows a mean error at $0\,\text{mm}$, but $1.7\,\text{mm}$ standard deviation. Plots of the decoded position error distributions are depicted in appendix figures A.3.a and A.3.b.

Orientation Decoding Results

The realtime honey bee waggle dance decoder calculates only one final orientation of a waggle run. In order to compare decoder system's quality the frame based RoboBee orientation data is averaged. All orientations are temporarily transformed into unity circle vectors. Then deviations of 199 waggle run samples are collected. This values are distributed between $-180°$ to $180°$.

On average evaluation results show that decoded orientations estimate the real dance direction. As depicted in figure 4.4, the mean decoded orientation error is $-1°$. Results also show a standard deviation error of $36°$.

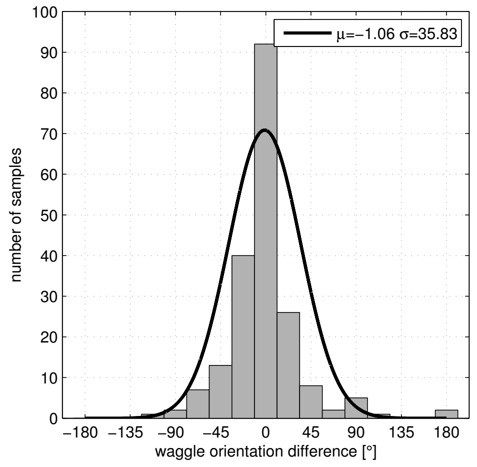

Figure 4.4: Distribution of decoded orientation errors for 199 waggle run samples. On average the decoded orientation is near $0°$ error.

4.2 Beehouse and BeesBook Evaluation

4.2.1 Beehouse and BeesBook Source

I collect Beehouse and BeesBook data at two separate experiment sites, where observation conditions share some similarities. At both locations, a honey bee comb is observed from two sides 24 h a day. PlayStation Eye Cameras, with removed infrared filters, are utilized. The camera pairs are connected over an USB 2.0 interface with local computers, which are technically identical. For each camera those machines run a realtime honey bee waggle dance decoding process. Decoded waggle run results are locally stored within a defined folder structure. Beehouse camera1 detects with an image resolution of 0.43 dpmm and camera2 with 0.42 dpmm. Both cameras at BeesBook detect with 0.43 dpmm resolution. The resolution values are calculated with arena configurations and the comb frame size, which is 37 cm width and 21 cm height. For all four comb frame edges, the projected number of pixels is calculated. Then the mean of it results in an approximate resolution value. A frame rate of 102 Hz is used, because I experienced PlayStation Eye Camera driver issues at 100 Hz.

However, experiment sites differ in the aspect of red light illumination conditions and laboratory access. The Beehouse experiment is executed as a proof of concept before BeesBook. Hence camera placement and illumination is executed

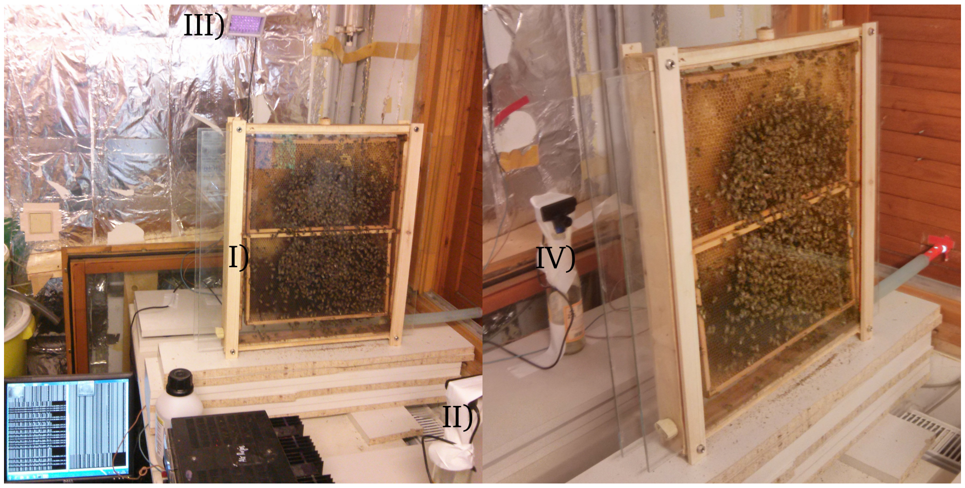

Figure 4.5: Beehouse experiment site setup. Marked items are: I) observed honey bee comb, II) PlayStation Exe Camera fixed on prototypical mounting, III) red light device, IV) second camera.

fast and simple in a prototypical manner, as shown in figure 4.5. Access to the observation room is shared among an undefined group of other people.

The Beesbook data is collected within a collaborative experiment setup. At each honey comb side one PlayStation Eye Camera is integrated next to two high resolution cameras, as seen in figure 4.6. I customized items to provide mounting and fixing capabilities within the observation cage. Red light illumination is adjusted with four distinct light sources by another team member. His goal is to provide a highly homogenous illumination with less reflection, because readability of high resolution images shall be maximized. Access to the laboratory is restricted to team members.

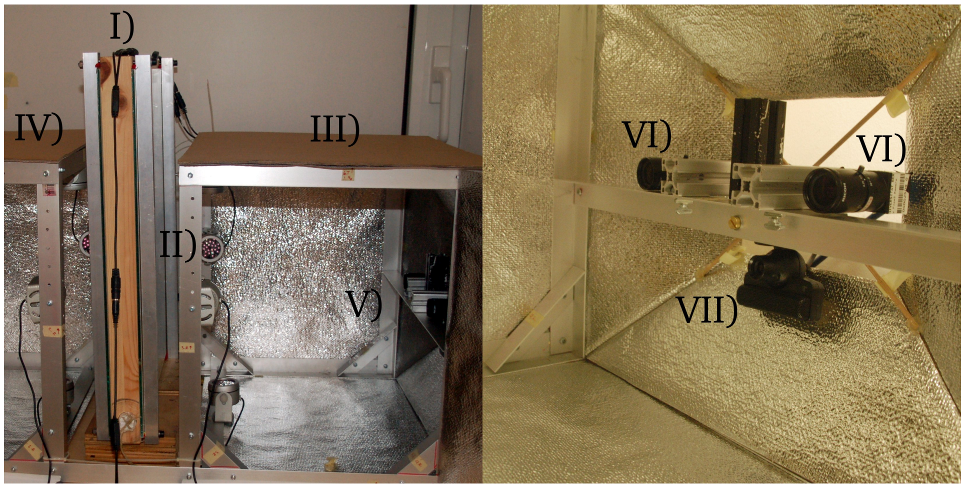

Figure 4.6: Beesbook experiment site setup. Marked items are: I) observed honey bee comb, II) one red light device, III) right side of observation cage, IV) mirrored left side of observation cage, V) right side camera array, VI) high resolution cameras, VII) PlayStation Eye Camera.

4.2.2 Evaluation Method

The results of Beehouse experiment include 48 days observation time, which starts from 2014-07-15 and ends 2014-08-31. Within this time 220127 waggle run detections are available. Evaluated Beesbook data starts from 2014-08-01 and ends 2014-08-31, with 132153 waggle run detections. For the purpose of evaluation 1000 separate detection samples are selected randomly. Saved detection image sequences are reviewed in order to determine the number of true positives. In this context **true positive** means one can observe typical waggle movement of a honey bee. Additionally, following categories of false positives are distinguished:

bee moth detections of lesser bee moths flying or roaming in the hive

blinking bee blinking signal of a bee at a fixed position

falling bee falling or walking honey bee which passes a comb frame

illumination switching on/off room light or red light illumination

false positive other detections that do not fit in any previous category

This method is applied to both data sets and results are presented in the following.

4.2.3 Beehouse and BeesBook Evaluation Results

The Beehouse evaluation shows that out of 1000 random waggle run detections 78.6 % are true positives. Details of false positives are listed in table 4.2. The true positive rate of evaluated BeesBook data equals 67.4 % and details are listed in 4.3. Between the two experiments a deviation of approximately 11 % true positive rate

type	total	ratio in %
true positive	786	78.6
bee moth	95	9.5
blinking bee	46	4.6
illumination	35	3.5
false positive	26	2.6
falling bee	12	1.2

Table 4.2: Beehouse evaluation results.

type	total	ratio in %
true positive	674	67.4
false positive	302	30.2
falling bee	20	2.0
illumination	4	0.4
bee moth	0	-
blinking bee	0	-

Table 4.3: Beesbook evaluation results.

is observed. But the Beehouse experiment site is influenced by external factors of *lesser bee moth*, and *illumination* issues due to unknown room access. One can perfectly assume that such factors, under stricter observation circumstances, can be avoided. If this false positives are removed from data, then Beehouse true positive rate equals 90.3 %, while BeesBook data remains at 67.7 %. The false positive rate at BeesBook experiment definitely is significant higher. Reasons for this divergence are discussed in section 5.1.5.

4.3 Conditioned Bees Evaluation

4.3.1 Conditioned Bees Source

The source for conditioned bees evaluation is a subset of the Beehouse data. Honey bees from the Beehouse colony are trained to a certain feeder location. At the beginning, the initial feeder location is next to the hives exit. Foraging bees are marked with a specific color at the feeder. After some time the food source is moved away. It remains at a new location until enough previously marked bees rediscover it. Then again the feeder is displaced further away. This process is

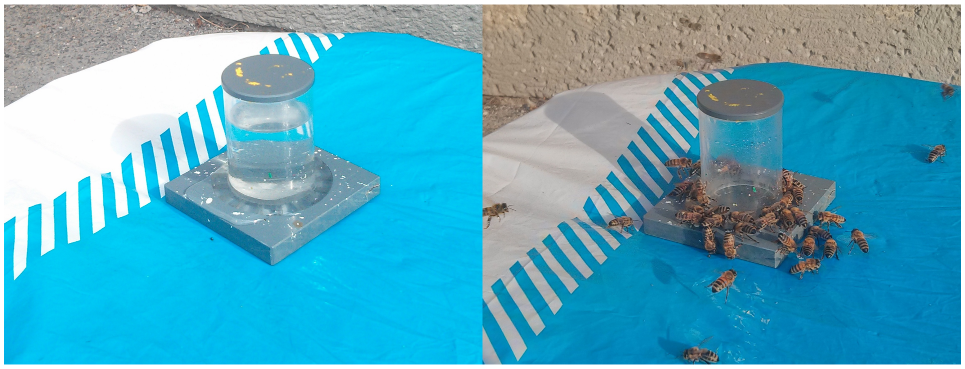

Figure 4.7: Pictures of feeder target location on 2014-08-24. On left side, the feeder is just placed and opened at 15:45. On right side, after 1.5 h one can see a number of foraging honey bees.

repeated several days until the target location is reached and the trained bees still follow. Even if nutrition is provided only a few hours honey bees find the feeder, as shown in figure 4.7. After almost 90 min one can see many honey bees and a depleted feeder. Precise feeder timings at the final location are listed in table 4.4. In total, the food source remains open for 355 min, distributed over three days. Within this time 4651 Beehouse waggle run detections are stored.

date	**open time**	**close time**	**number of waggle dances**
2014-08-21	13:20	15:50	31
2014-08-22	16:15	18:10	77
2014-08-24	15:45	17:15	84

Table 4.4: Details of Beehouse artificial feeder timings.

4.3.2 Evaluation Method

The goal is to map decoded waggle dances, which occur during feeder allocation, into real world coordinates. It is expected that a significant amount of waggle dances advertise the previously trained food source. Therefore results are better if decoded map positions are distributed closer to the target location. Using Google Maps service I define Beehouse hive's location at 52.457 075 lat and 13.296 055 lon and feeder position at 52.455 563 lat and 13.298 462 lon. With an online calculator[2] I estimate 234 m distance and a 135.9° bearing (north equals 0°, ascending clockwise until 360°).

In order to smooth waggle run detection results they are assembled into waggle dances. My approach simply connects detections that occur within a defined area and a maximum time delay. Finally, only waggle dances which consists of at least four waggle runs are kept. This way 192 waggle dances are extracted out of 4651 detections. With manual review I find two false positive waggle dances; one due to a dancing honey moth and one due to frequent illumination issues.

For each waggle dance detection time, the averaged duration and rectified orientation are saved. Arena coordinates are utilized to create a homomorphic transformation matrix. I assume that the comb frame is set to be aligned horizontally. This way for any rectified point the gravity direction equals a straight line towards the bottom. Rectified frame positions therefore can be used to restore real world orientations.

The extracted honey bee dance orientation values range from $-180°$ to $180°$. While $0°$ equals opposite gravity direction, negative orientation values are arranged counter clockwise.

Azimuth data of 2014-08-22[3] translates dance orientations to real world bearings. Dance durations are translated into distance using a linear mapping. Values are provided by Landgraf et. al.[10]. He determines that 215 m distance equals 440 ms duration, when using same hive and feeder location with his experiments. In the end, real world distance and orientation informations are transformed to lat/lon coordinates.

4.3.3 Conditioned Bees Evaluation Results

Waggle dances, which occur during allocation of artificial feeder, are decoded and visualized in figure 4.8. Most of the decoded dances advertise spots around the feeder location. The average of all positions is at 52.455 874 lat and 13.298 265 lon, next to the food source. Decoded position color refers to it's absolute difference be-

[2]http://www.movable-type.co.uk/scripts/latlong.html

[3]http://aa.usno.navy.mil/data/docs/AltAz.php

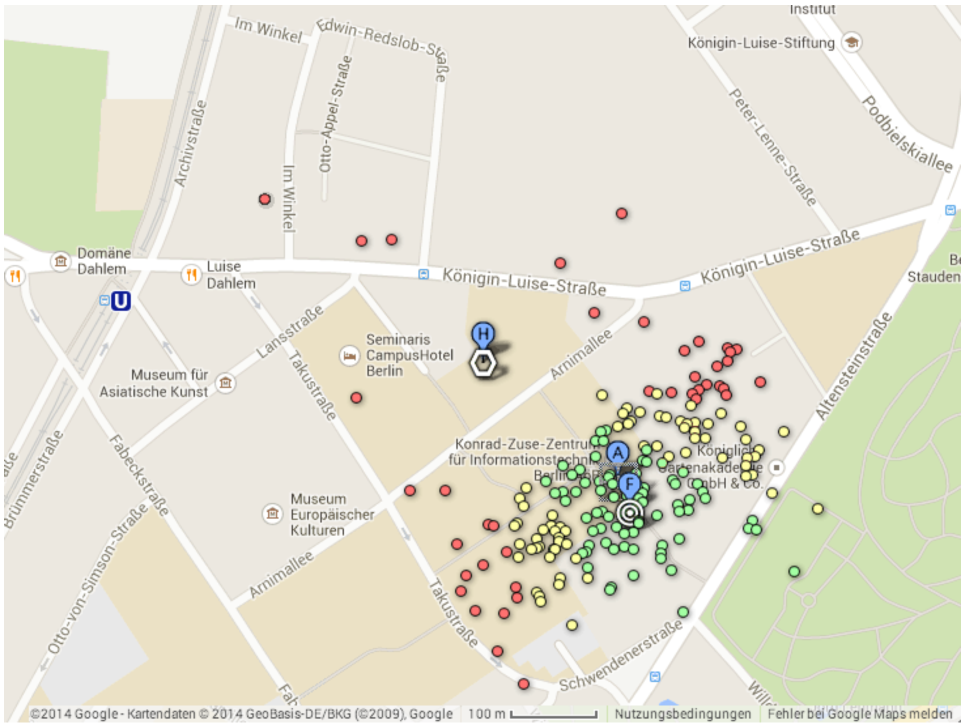

Figure 4.8: Results of 192 decoded waggle dances at Beehouse during feeder experiments as listed in table 4.4. Blue marker 'H' corresponds to honey bee hive, marker 'F' to feeder, marker 'A' to average decoded position. Red (21.4 %), yellow (34.9 %), and green (43.7 %) dots indicate single waggle dance positions. Color is assigned as listed in table 4.5. Concentration of waggle dance positions near feeder location is clearly visible.

tween the expected hive-to-feeder orientation of $135.9°$. In table 4.5 color division and statistics are listed. For comparison three other decoded maps are presented in appendix figure A.4, figure A.5 and figure A.6.

4.4 Summary

In this part different evaluation results are presented. Using RoboBee data a detailed analysis is conducted. It is shown that the realtime honey bee waggle dance decoding system achieves 96.4 % precision rate and 89.5 % recall rate. Using 5861 samples, on average decoded dance positions have 0 mm error and a standard deviation error smaller then 1.68 mm. For 199 waggle run durations, the decoding

color	**max deviation**	**number**	**percent**
green	$\leq \frac{\sigma_{ori}}{2}$	84	43.75
yellow	$\leq \sigma_{ori}$	67	34.90
red	other	41	21.35

Table 4.5: Statistics for evaluation of trained bees and categorization of orientation error. Color assignment depends on the offset from $135.9°$, where $\sigma_{ori} = 35.83°$ (see 4.1.3).

exhibits 97.79 ms overestimation in the mean and has a standard deviation error of 138.52 ms. On average the evaluation of decoded honey bees dance direction shows $-1.06°$ error and a standard deviation error of $35.83°$. With 1000 random samples, the live observation at the Beehouse is estimated to have 78.6 % precision rate; the BeesBook live experiment shows 67.4 % precision rate. Finally, it is shown that waggle runs, which are decoded during artificial feeder placement, can be used to estimate the advertised food location.

Chapter 5

Discussion and Outlook

In this section previously presented evaluation results are discussed. All aspects of the automatic realtime honey bee waggle dance decoding system are inspected. Then reasons for different performance results of the Beehouse and BeesBook experiment are shown. Next, flaws in the current decoder system design are outlined. Finally, an outlook and brief description for some directions of future work is presented.

5.1 Evaluation Results Interpretation

In the following, separate decoder system aspects are subsequently addressed. The discussion starts with the decoder system's major ability to detect honey bee waggle runs. Then quality of position, duration and orientation decoding is examined. Interpretation of different Beehouse and Beesbook evaluation results is presented. Finally, some problems of the current realtime honey bee waggle dance decoder system are addressed.

5.1.1 Waggle Run Detection

The decoder system shows a high capability to detect honey bee waggle runs. It is tested against RoboBee data with a recall rate of $96.4\,\%$ and precision rate of $89.5\,\%$. Results for the Beehouse observation show that the decoder system can achieve $90.3\,\%$ true positive rate, if external factors are extracted. But for the Beehouse or Beesbook observations there truly does not exist any information about missed waggle runs (*false negatives*). For online observations, it is unlikely to achieve a precision rate comparable to offline RoboBee data. In the case of live observations, illumination conditions and the input image quality are more chal-

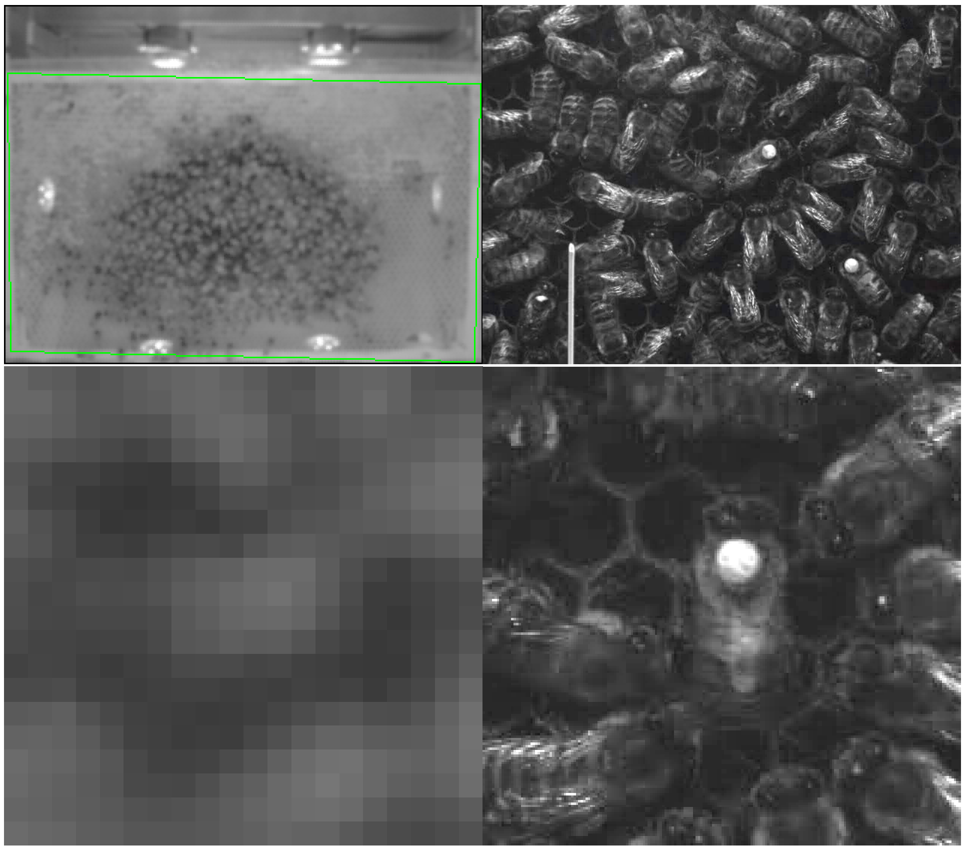

Figure 5.1: Video quality comparison of different sources. On the top left, an original input frame from BeesBook experiment is depicted. A green line on top of the comb frame shows the configured arena. An original input frame from RoboBee experiment is shown on the top right. For each video source the second row shows an example cropped detection image, which are stored in the VideoFrameBuffer.

lenging. As depicted in figure 5.1 the difference of video qualities is too high to be ignored.

In order to determine decoder system's *precision rate* at the Beehouse or BeesBooks experiments, additional recordings would be necessary. Afterwards, those videos could be manually analyzed. For this purpose video records should have a high frame rate and resolution. Optical sensors additionally must work under red light illumination.

But camera equipment, which fulfills this properties, is quite costly. Maybe renting an appropriate device could help. Even then, correct manual detection of waggle movements would take extraordinary hours of working time.

In the end, it is desirable to determine the numbers of false negatives for live experiments and consequently calculate precision rates. But the implied efforts are out of scope for this master's thesis.

5.1.2 Waggle Run Position Decoding

The position decoding performance of RoboBee data is satisfactory. But the evaluated data belongs to a rather small set and it's video quality is not comparable to live observations. Further proof of position decoding quality within the live data set is needed, particularly because the subsequent orientation decoding relies on correct positions.

On one hand, my experience shows that live position decoding works quite well. For each waggle run detection a sequence of images is written to hard disk. The first decoded position determines the center view for those sequences. I manually reviewed many of this detection footage and decoded positions often pointed at the actually waggling honey bee.

But on the other hand, I can not present any quantified data for live observations. Again, additional hardware and time to manually extract precise honey bee positions is needed. Without precise ground truth data it is impossible to measure live position decoding performance.

5.1.3 Waggle Run Duration Decoding

The evaluation data shows a non trivial error for the decoding system's duration decoding quality. This time span information is important, because it defines the distance towards an advertised food location.

The realtime honey bee waggle dance decoder first needs to observe some waggle movements before a signal is created. This is due to implementation details of a DotDetector, that are explained in section 3.2.5. However, according to measurement conventions a waggle run commences at the resting position, *before* a first swing. When utilizing DotDetector signals a delay in waggle run start decoding therefore is unavoidable.

If a waggle run end is decoded I observe a related problem. Even after a honey bee stops the waggle movement, the DotDetector buffers will still contain former sinus waves. Only after enough new frames, and therefore pixel values, are pushed trough, the detection signal will go off.

Finally, I want to mention that other approaches to decode waggle run duration exist. However, it was not possible to successfully research and implement them in the scope of this master's thesis. Some hints are given in section 5.2.

5.1.4 Waggle Run Orientation Decoding

The reliability of waggle run orientation decoding leaves room for improvement. Evaluation results show an unacceptable high standard deviation error for orientation decoding.

The current decoder system simply decodes directions as a vector form the first towards the last decoded position. If the start and end position are the same, then orientation decoding completely fails. Waggle runs with little or no forward movement usually are poorly decoded. Orientation decoding depends on correct start and end position decoding.

With this drawbacks in mind results that are shown for RoboBee evaluation in section 4.1.3 and conditioned bee evaluation in section 4.3.3 are remarkable. The mean values of decoded waggle dance orientations indeed point towards the expected directions. Nonetheless, I investigated other approaches for orientation extraction.

One idea is to utilize more waggle run positions, because previous evaluation results show an overall well decoding performance. I try to fit a line trough the 2d point space with different methods. However, results are worse then the current simple implementation.

When a first live observation in the Beehouse was established another idea was implemented. Back then, orientation extraction was based on a sophisticated image processing chain. It used temporal differences of cropped image sequences and calculated properties for move fragments. The solution was fast enough to remain realtime capabilities. Although this image based approach seemed promising when tested with the RoboBee movies, in live environments it failed.

From a physical point of view, the orientation extraction is definitely the most difficult part of waggle run decoding. Honey bees rarely perfectly waggle towards a fixed orientation. More likely they tend to have some deviation within waggle swings of one run. Even with high resolution videos it is difficult to extract the *correct orientation*. It depends on the definition of how to calculate it.

The RoboBee data for instance captures honey bee body orientations for each video frame during a waggle run. Therefore it is possible to determine the dance orientation by averaging over all those angles. But my decoder system can not use this approach, because such detailed information is not accessible. And for the moment, I do not see a practical solution to gather frame based body orientations in realtime.

This observation leads to the conclusion that one either has to use an approximative realtime algorithm or a post process orientation decoding. Due to my goal to achieve realtime capability I only explored online solutions for orientation decoding.

5.1.5 Beehouse and BeesBook Evaluation Difference

The difference between the Beehouse and BeesBook evaluation results needs further investigation. Cameras and computers are technically the same models and the decoder system software is identical. Both experiments operate at approximately the same target frame resolution of 0.43 dpmm. One definite distinction is the red light illumination condition.

At the BeesBook laboratory a compound of two high resolution cameras is added on each side of the hive. Red light exposure is configured to maximize quality of megapixel images. After some configuration time a homogenous illumination across the comb surface is achieved. This leads to the loss of contrast in video frames. Consequently, smaller amplitude values result in lower DotDetector potentials. During this period, the number of detections per day drops near zero. As a countermeasure I have to decline the DotDetector signal threshold by 69.3 % of the Beehouse value and set $c_{\text{min}} = 5$ (for full parameter overview see appendix table A.1). But this step makes the decoder system more sensitive and therefore false positives more likely to occur.

There is another difference between experiments that is related to the subjects of observation: honey bees. The Beehouse colony experiences almost natural conditions, except for some room light exposure over short periods of time. In contrast, each BeesBook honey bee runs through the process of being tagged with a round plastic code on it's thorax. Furthermore, honey bees are moved from their former hive into another observation hive.

It is very hypothetical, but this process may lead to some of the *abnormal* movement behavior I observe while evaluating the BeesBook detections. Many of false positive detections are due to inharmonious shaking movements of bees towards all directions. I use the term abnormal, because I have not seen any equivalent detections within the Beehouse or RoboBee data.

Of course one can argue that the described shaking movements are only detected due to a very sensitive threshold. I have no proof that such movements do not occur at the Beehouse colony.

In sum influences like illumination conditions, decoder system parameters and conditions of tagged honey bees differ from one to the other experiment site. Research to determine which aspect really affects successful decoder system detection rates remains open.

5.1.6 Decoder System Design Drawbacks

Finally, I discuss problems in the current realtime honey bee waggle dance decoding system design of layer 1 and layer 2. All decoding processes (position, duration, orientation) rely on DotDetector signals. Consequently, those decoding modules inherit problems of the DotDetector design.

One flaw is the use of amplitude A_j as a scaling factor in layer 1, as pointed out in section 3.2.5. On one side, amplitude is important in order to distinguish between high frequency scores created either by random noise or real waggle movements. On the other side, false positives due to a dominant amplitude value may occur. Those amplitudes are generated through a big contrast change. Honey moth tendentially appear almost white on footage, as seen in figure 5.2. Moving over a

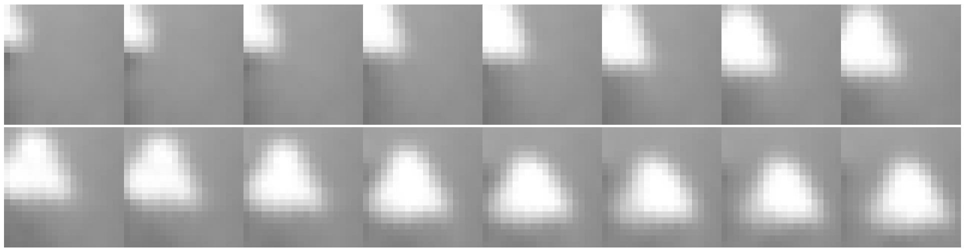
Figure 5.2: Cropped image sequence of a false positive detection due to honey moth. Sequence order is from top left towards bottom right. Because honey moth appear almost white the huge contrast may lead to false positive detections.

rather dark comb surface they create amplitude spikes which turn to false positives. The same mechanism can be observed with a rather dark honey bee body, which runs or falls over a bright comb frame. I initially try to set a global maximum amplitude value, in order to limit spikes to a determined level. In the end I fail to find a balanced value that reduces false positives and keeps the previous detection rate.

Another flaw in the current system design concerns the absence of a physical honey bee model. Indeed there are some implicit physical assumptions, i.e. the maximum distance of DotDetector signals to form one cluster. However, there is no upper bound for the number of positive DotDetectors that can make up one single honey bee. Also no restrictions apply to the area and shape a signal cluster on layer 2 can have. Therefore two simultaneously dancing honey bees, located right next to each other, are merged into a single waggle run. Another problem is an unrestricted position decoding. It can locate a honey bee on any consecutive frame at arbitrary distance without considering physical maximum movement speed.

This problems are not final but due to resource limitation better approaches are not feasible in the scope of my master's thesis.

5.2 Outlook

In the end I can say that the presented realtime honey bee waggle dance decoding system is the first publication of its kind. The current implementation meets the overall system requirements set in section 3.1.1. The results of evaluation data evidentially approve the capacities of the decoder system. Nevertheless, many improvements can be made in order to advance the automatic honey bee waggle dance detection and decoding. I want to outline future development possibilities and open research topics.

In my opinion, the core concept of detection using DotDetectors needs to be translated from discrete value thresholding into a probabilistic model. This model could enforce laws of physic, like maximum movement speed and body shape restrictions and introduce a notion of *waggle energy*. Rather then rely on binary decisions, which are thresholded at a fixed minimum cluster size, this probabilistic model could smooth detections and make them more robust. Furthermore, it would eradicate most of the flaws pointed out in section 5.1.6.

I relate most of the encountered decoding problems to the fact that DotDetectors currently are utilized not only for their intended use of detection but also decoding. One can observe that if decoding gets heavier (position $<$ duration $<$ orientation), then less decoding quality is achieved. It would be interesting to research an approach which splits decoding and detection into separate processes.

For example, one could pass dance packages to a parallel process, which can run decoding without real time deadline limitations. Dance packages could contain cropped image sequence provided by the VideoFrameBuffer. With image analysis one can improve duration decoding by measuring some sort of movement energy as sum of pixel value deviations. This movement energy also could be used to filter out blinking false positives. However, it remains an open question if some image process can be found to stably decode orientations if low resolution images are used.

Another question concerns an efficient way to determine decoder system's performance in live observation environments. The establishment of precise data comes with high expense of time and labour. How this can be overcome remains an open question for the future.

List of Figures

1.1 Ideal waggle dance path 5

3.1 Decoder system process overview 12

4.1 Automated match visual feed back example 23
4.2 Decoded start and end time error distributions 24
4.3 Decoded position error distributions 25
4.4 Decoded orientation error distribution 26
4.5 Beehouse experiment site setup 27
4.6 BeesBook experiment site setup 28
4.7 Target feeder location before and after placement 30
4.8 Visualization of 192 decoded waggle dances 2014-08-21 32

5.1 Video quality comparison between RoboBee and live data 35
5.2 Image sequence of false positive detection due to honey moth . . . 39

A.1 Development view on DotDetector signal processing 45
A.2 Decoded duration error distribution 47
A.3 Decoded x and y position error distributions 47
A.4 Visualization of 1138 decoded waggle dances 2014-07-15 48
A.5 Visualization of 214 decoded waggle dances 2014-08-15 49
A.6 Visualization of 72 decoded waggle dances 2014-08-29 50

Bibliography

[1] Margaret J. Couvillon, Roger Schürch, and Francis L. W. Ratnieks. Waggle dance distances as integrative indicators of seasonal foraging challenges. *PLoS ONE*, 9(4):e93495, 2014. doi: 10.1371/journal.pone.0093495. URL http://dx.doi.org/10.1371%2Fjournal.pone.0093495.

[2] Harald E. Esch, Shaowu Zhang, Mandyan V. Srinivasan, and Juergen Tautz. Honeybee dances communicate distances measured by optic flow. *Nature*, 411(6837):581–583, 05 2001. URL http://dx.doi.org/10.1038/35079072.

[3] Mariana Gil and Rodrigo J. De Marco. Decoding information in the honeybee dance: revisiting the tactile hypothesis. *Animal Behaviour*, 80(5):887 – 894, 2010. ISSN 0003-3472. doi: 10.1016/j.anbehav.2010.08.012. URL http://dx.doi.org/10.1016/j.anbehav.2010.08.012.

[4] Christoph Grüter and Walter M. Farina. The honeybee waggle dance: can we follow the steps? *Trends in Ecology & Evolution*, 24(5):242–247, 2009. doi: 10.1016/j.tree.2008.12.007. URL http://dx.doi.org/10.1016/j.tree.2008.12.007.

[5] Tina Heidborn and Tania Munz. Dancing with bees. *MaxPlanckResearch*, 4 (2):75–80, 2010. ISSN 1616-4172.

[6] 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. ISSN 0044-8435. doi: 10.1007/s13592-011-0060-6. URL http://dx.doi.org/10.1007/s13592-011-0060-6.

[7] Toshifumi Kimura, Mizue Ohashi, Karl Crailsheim, Thomas Schmickl, Ryuichi Okada, Gerald Radspieler, and Hidetoshi Ikeno. Development of a new method to track multiple honey bees with complex behaviors on a flat

laboratory arena. *PLoS ONE*, 9(1):e84656, 01 2014. doi: 10.1371/journal.pone.0084656. URL `http://dx.doi.org/10.1371%2Fjournal.pone.0084656`.

[8] Barrett Anthony Klein, Martin Stiegler, Arno Klein, and Jürgen Tautz. Mapping sleeping bees within their nest: Spatial and temporal analysis of worker honey bee sleep. *PLoS ONE*, 9(7):e102316, 2014. doi: 10.1371/journal.pone.0102316. URL `http://dx.doi.org/10.1371%2Fjournal.pone.0102316`.

[9] Uwe Knauer, Fred Zautke, Kaspar Bienefeld, and Beate Meffert. A comparison of classifiers for prescreening of honeybee brood cells. In *Proc. International Conference on Computer Vision Systems*. ICVS, 2007.

[10] 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. doi: 10.1371/journal.pone.0021354. URL `http://dx.doi.org/10.1371%2Fjournal.pone.0021354`.

[11] K. Rohrseitz and J. Tautz. Honey bee dance communication: waggle run direction coded in antennal contacts? *Journal of Comparative Physiology A*, 184(4):463–470, 1999. ISSN 0340-7594. doi: 10.1007/s003590050346. URL `http://dx.doi.org/10.1007/s003590050346`.

[12] Thomas D. Seeley and Susannah C. Buhrman. Group decision making in swarms of honey bees. *Behavioral Ecology and Sociobiology*, 45(1):19–31, 1999. ISSN 0340-5443. doi: 10.1007/s002650050536. URL `http://dx.doi.org/10.1007/s002650050536`.

[13] David R. Tarpy and North Carolina Cooperative Extension Service. *The Honey Bee Dance Language*, volume 646 of *AG (Series)*. N.C. Cooperative Extension Service, 2004.

Appendix A

Appendix

Parameter	Short	Robobee	Beehouse	BeesBook
VideoFrameBuffer History	-	600	600	600
Video Frame Rate	s_R	100	102	102
Input Frame Width	$\widetilde{f}_{width}$	640	320	320
Input Frame Height	$\widetilde{f}_{height}$	480	240	240
Input Frame Colorspace	$\widetilde{f}_{colorspace}$	GRAY	GRAY	GRAY
Target Frame Width	f_{width}	40	160	160
Target Frame Height	f_{height}	30	120	120
Target Frame Colorspace	$f_{colorspace}$	GRAY	GRAY	GRAY
Frequency Minimum	FREQ_{min}	11	11	11
Frequency Maximum	FREQ_{max}	17	17	17
Frequency Step	FREQ_{step}	1	1	1
DotDetector Buffer Size	b	32	32	32
DotDetector Threshold	θ	19500	32888	10088
Layer 2 Max Signal Dist.	d^2_{max}	2.3	2.3	2.3
Layer 2 Min Cluster Size	c_{min}	5	6	5
Layer 3 Max Signal Dist.	d^3_{max}	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$
Layer 3 Max Frame Gap	g_{max}	3	3	3
Layer 3 Min Frame Number	l_{min}	20	20	20

Table A.1: Details of decoder system parameters used in all different setups.

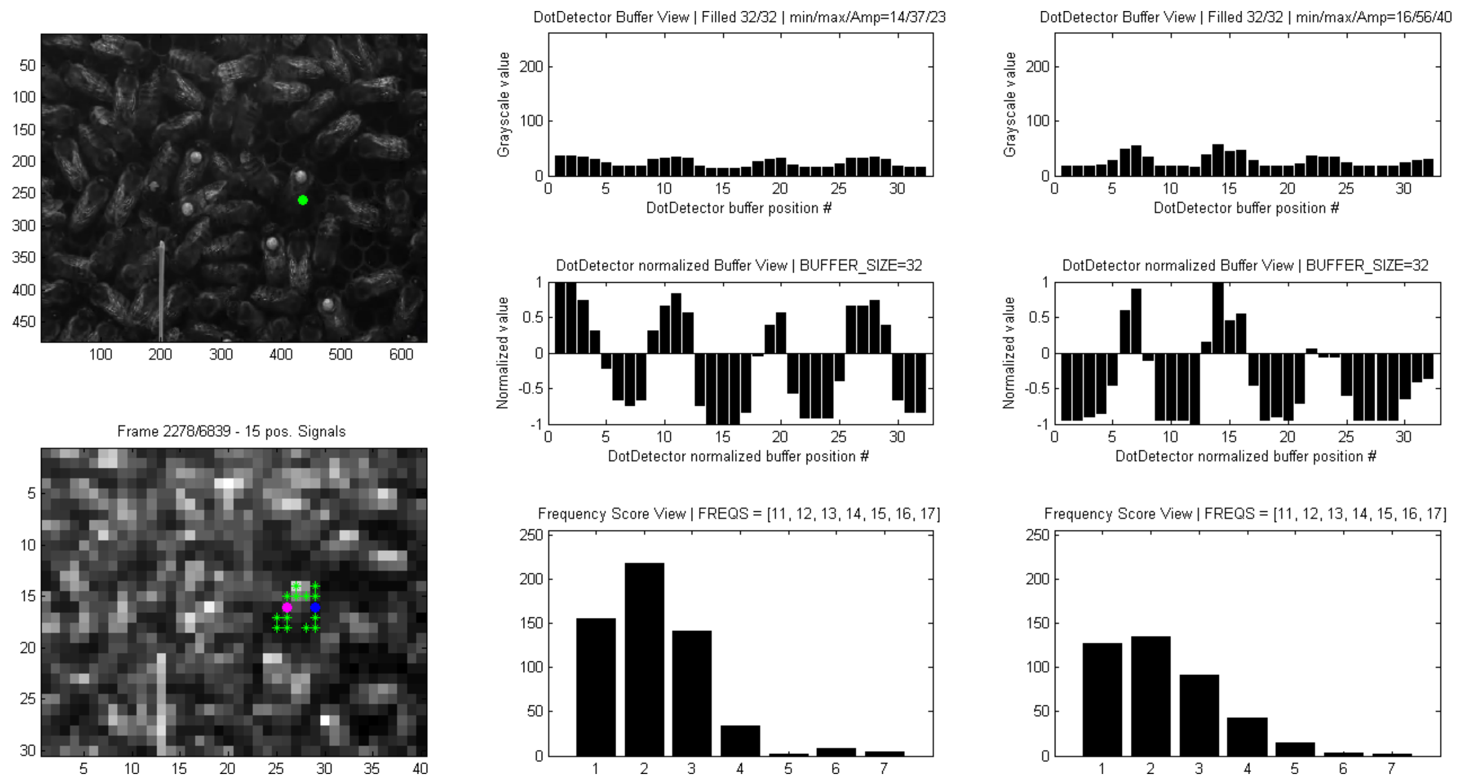

Figure A.1: Example view of DotDetector function in process of video detection. Top left image shows original input video frame. Below is the transformed target video frame. Green '*' symbols indicate DotDetectors with positive signal. Above a green circle indicates the decoded position. Blue and pink circle are selected signaling DotDetectors. Their inspection view is plotted on the right in two separate rows. First column shows the buffer content Bj. Second column shows the normalized buffer content $\bar{B}$. Last column shows the frequency scores. Waggle patterns are clearly visible in all buffer plots.

Video File	**Detection Rate**	**Total WDD**	**Add/True/False**
1155-SINGLE.raw.avi	$17/17$	23	$6/2/4$
1156-SINGLE00.raw.avi	$15/17$	19	$4/3^1/1$
1157-SINGLE.raw.avi	$3/3$	3	$-/-/-$
1159-SINGLE.raw.avi	$8/8$	10	$2/2^2/0$
1203-SINGLE.raw.avi	$18/18$	21	$3/1^1/2$
1204-SINGLE00.raw.avi	$23/24$	25	$2/2^1/0$
1205-SINGLE.raw.avi	$10/10$	10	$-/-/-$
1206-SINGLE.raw.avi	$8/9$	10	$2/2^1/0$
1227-SINGLE.raw.avi	$11/11$	13	$2/1/1$
1229-SINGLE.raw.avi	$3/3$	5	$2/1/1$
1234-SINGLE.raw.avi	$6/7$	9	$3/1/2$
1237-SINGLE.raw.avi	$8/8$	11	$3/2/1$
1244-SINGLE.raw.avi	$6/6$	7	$1/0/1$
1249-SINGLE.raw.avi	$25/27$	30	$5/4^4/1$
1251-SINGLE.raw.avi	$17/17$	18	$1/1/0$
1301-SINGLE.raw.avi	$21/22$	24	$3/2/1$
$\sum$	$199/207$	238	$39/24^{10}/15$

Table A.2: Details of RoboBee evaluation. Second column shows how many known waggle runs are correctly detected. Next column shows the total number of decoder system detections. In the last column, number of "additional" waggle detections *vs.* number of true waggles *vs.* number of false positives is shown. Potent numbers indicate the number of *double detections* due to an unknown bug. Results show 14 extra waggle run detections and 25 false positives.

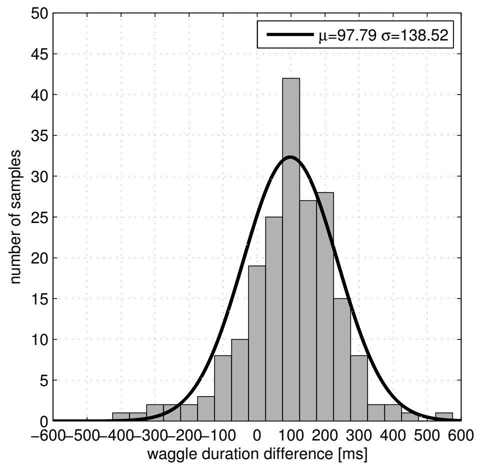

Figure A.2: Distribution of decoded duration errors for 199 waggle run samples.

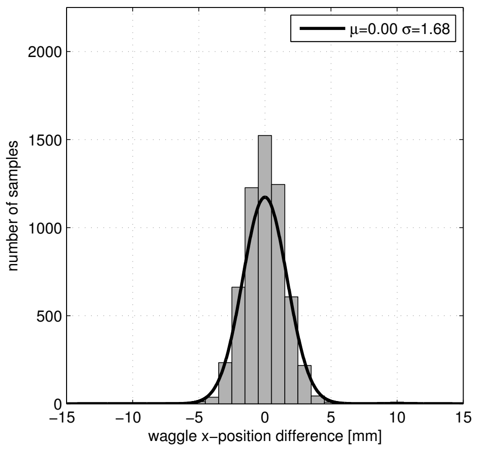

Figure A.3.a: position x difference

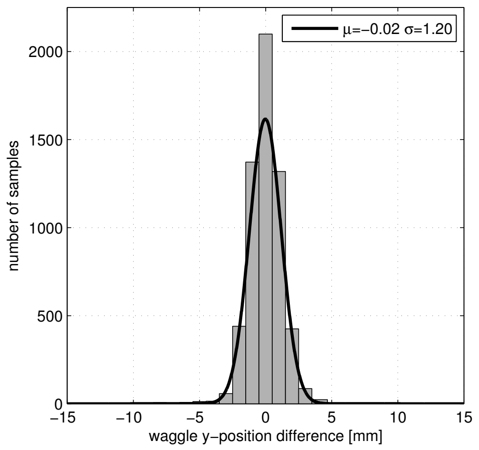

Figure A.3.b: position y difference

Figure A.3: Distributions of decoded position errors for 5861 location samples. In figure A.3.a values for the x axis are plotted and figure A.3.b depicts y axis, which is the honey bee body axis.

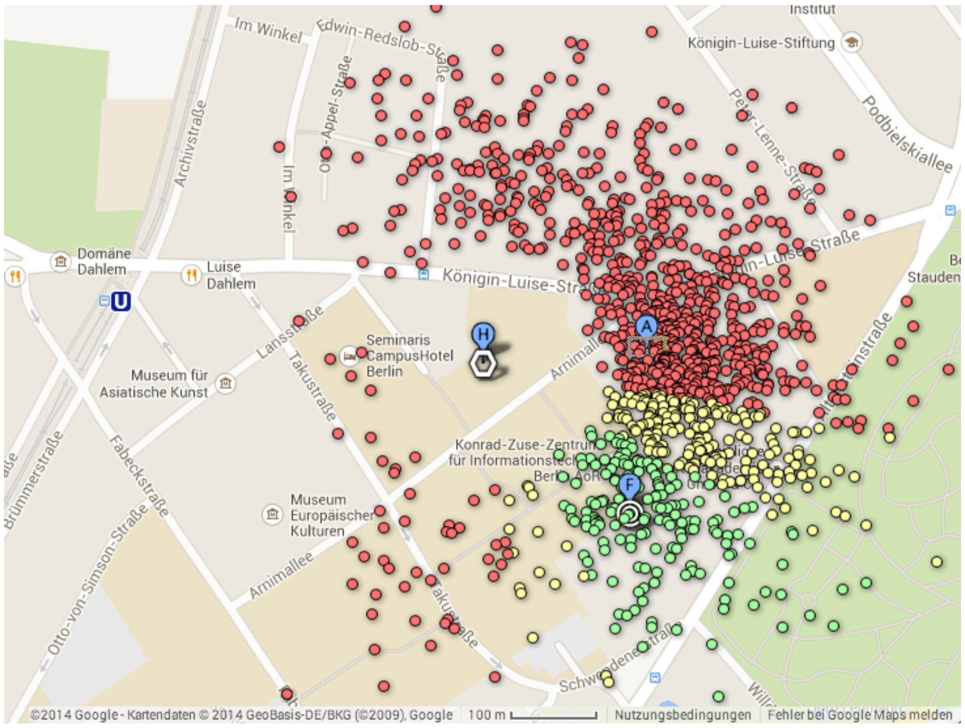

Figure A.4: Results of 1138 decoded waggle dances at Beehouse from 2014-07-15 0:00 until 2014-07-17 23:59. Blue marker 'H' corresponds to honey bee hive, marker 'F' to feeder, marker 'A' to average decoded position. Red (67.8 %), yellow (17.3 %), and green (14.9 %) dots indicate single decoded waggle dance positions. Color is assigned as listed in table 4.5.

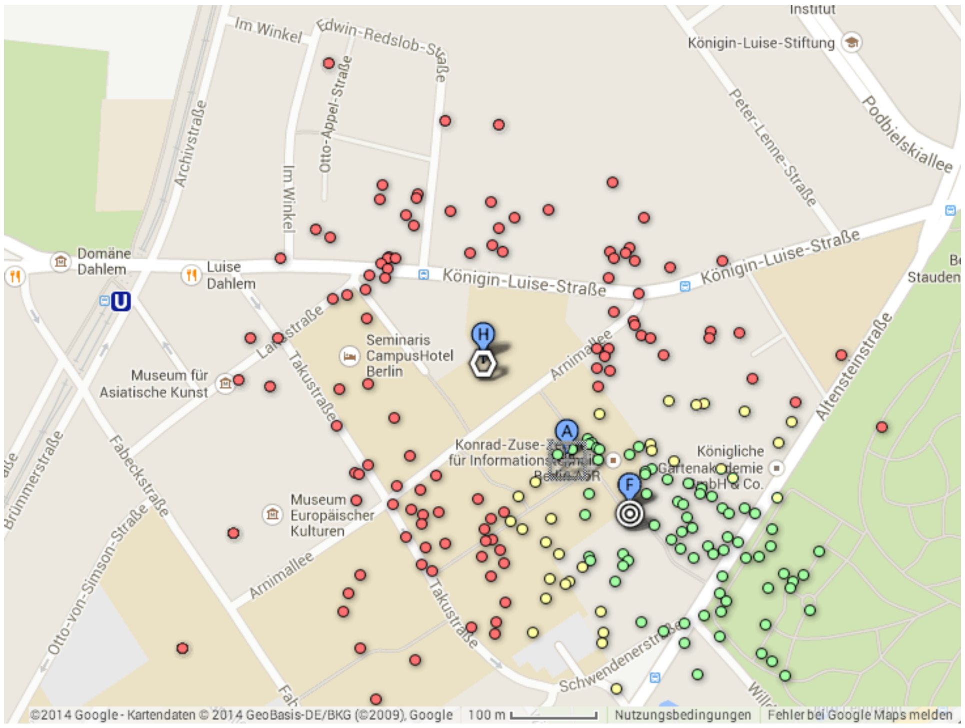

Figure A.5: Results of 214 decoded waggle dances at Beehouse from 2014-08-15 0:00 until 2014-08-17 23:59. Blue marker 'H' corresponds to honey bee hive, marker 'F' to feeder, marker 'A' to average decoded position. Red (52.3 %), yellow (14.0 %), and green (33.7 %) dots indicate single decoded waggle dance positions. Color is assigned as listed in table 4.5.

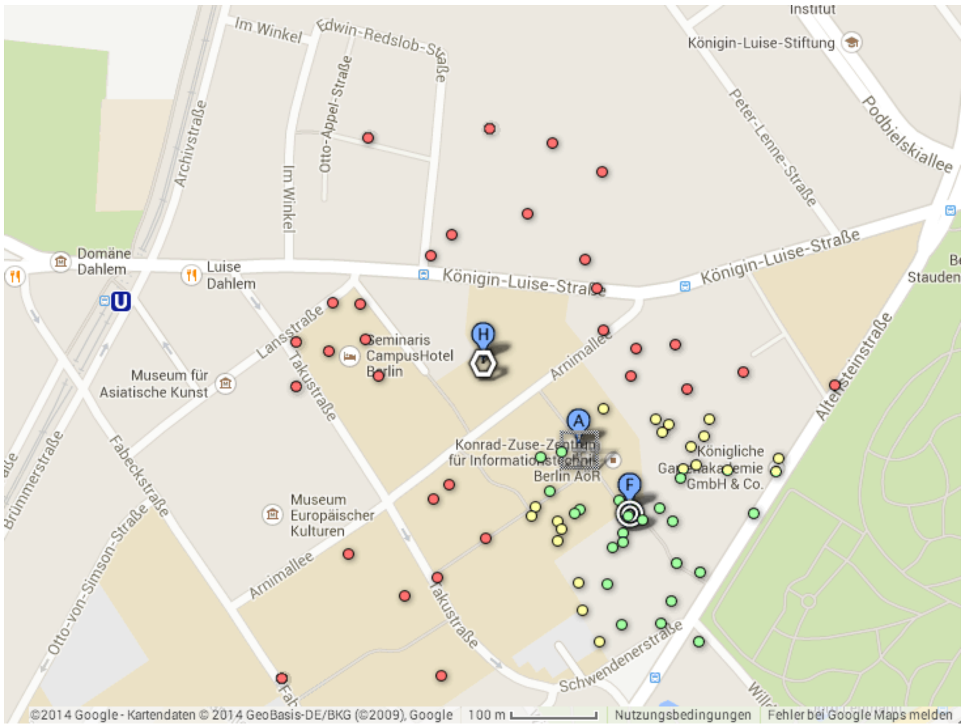

Figure A.6: Results of 74 decoded waggle dances at Beehouse from 2014-08-29 0:00 until 2014-08-31 23:59. Blue marker 'H' corresponds to honey bee hive, marker 'F' to feeder, marker 'A' to average decoded position. Red (43.2 %), yellow (27.0 %), and green (29.8 %) dots indicate single decoded waggle dance positions. Color is assigned as listed in table 4.5.