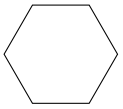

**Master Thesis, Institute of Computer Science, Freie Universität Berlin**

**Biorobotics Lab, Intelligent Systems and Robotics**

# Temporal Analysis of Honeybee Interaction Networks

## based on Spatial Proximity



*Alexa Schlegel*

Matriculation number: 4292909

alexa.schlegel@fu-berlin.de

Supervisor: Prof. Dr. Tim Landgraf, Freie Universität Berlin

Second Supervisor: Dr. Philipp Hövel, Technische Universität Berlin

Berlin, March 20, 2017

## Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Berlin, den March 20, 2017

Alexa Schlegel

Everything we hear is an opinion, not a fact.
Everything we see is a perspective, not the truth.

— Marcus Aurelius

Dedicated to my parents and my brother.

## Abstract

TODO

## Zusammenfassung

TODO

# Contents

# Chapter 1

# Introduction

Social insect societies are formed by thousands of individuals, which continuously move and interact with each other inside a dark nest. Honey bee colonies are thus organized complex and dynamical systems, which form a collective intelligence. Observing individual honey bees is, therefore, vital for understanding collective behavior, decision making, and organization of tasks within the colony.

Within the BeesBook[1] project of the Biorobotics Lab of Freie Universität Berlin Wario et al. [26] developed technologies to automatically track all individuals of a honey bee (Apis mellifera) colony. Shortly after birth, each bee has been marked on their thorax using circular 12-bit tags (figure 1.1) and then added to the colony. The comb is observed by four cameras over a period of nine weeks and each picture is evaluated automatically. The resulting data set contains, for each point in time, the exact position of each bee on the honey comb, its decoded id and its age.

In this thesis, temporal worker-worker interaction networks, based on spatial proximity, are derived from the described data set. Each node in the network is a bee and a link between two individuals is created if they share a position close to each other. The temporal networks are aggregated for several points in time. Social network analysis methods are applied to determine the usefulness and the characteristics of the resulting networks and its communities.

Until now, social network analysis has been applied to only a subset of a honey bee colonies live, simply because the data were not available to this extent and quality so far.

## 1.1 Motivation

Most of the studies in the field of animal social network analysis, especially when analyzing the behaviour of social insects colonies, only include a reduced subset of the colonies life, due to a large number of individuals. Usually, the reduction is carried out on three levels (1) time and resolution, (2) space, and (3) number of individuals.

Labeling only a subset of the colonies individuals, a short observation period, low res-

[1] `http://beesbook.mi.fu-berlin.de/wordpress/`; Last accessed: 16.03.2016, 12:05 p.m.

**Figure 1.1:** Tagged bees inside the hive.

olution and manually extracting information from photos or videos is very common in behavioral sciences [18, 24].

Baracchi and Cini [3] observed 300 bees out of a colony with 4000 individuals over a period of ten hours. One of the two sides of the hive was observed by taking a photo each minute. Coloured numbered discs were used for individually marking bees. The analysis of a weighted and undirected worker-worker interaction network revealed a highly compartmentalized structure inside the honey bee colony. Depending on the age, bees occupy different areas of the comb and correspond to different tasks. Also, the contact is limited within groups. Blonder and Dornhaus [5] color painted all individuals of ant colonies (size 6-90 for each colony) and filmed the colonies for 30 minutes. Interactions between individuals were manually extracted by watching the videos. Using time-ordered (dynamic) networks they analyzed the temporal and spatial dynamics of information flow.

Recently, automated tracking of insects has become technically feasible [26, 8, 11]. Using automated high resolution tracking data, which includes all individuals of the complete comb over a long time period allows for more advanced analysis focusing on temporal dynamics. Mersch et al. [17] automatically tracked all individuals of six ant colonies over a period of 41 days. Applying the Infomap community detection algorithm to the physical contact networks for each day, revealed three distinct and robust groups. Each group represents a functional behavioral unit, with individuals changing groups as they age.

The majority of social insect interaction networks studies, due to previously technical limitations, aggregate temporal tracking data into a single static network [16, Chapter 15]. Automatic tracking allows shifting more towards the temporal and dynamic investigation.

## 1.2 Research Goal

The aim of this thesis is to investigate whether the provided data set of tracked honey bees is useful for creating worker-worker interaction networks using spatial proximity as a proxy for interactions between bees. Thus, I need to implement a pipeline to infer networks out of the data set. Furthermore I want to find out if the resulting networks are suitable for social network analysis.

I want to achieve my research goals by answering the following questions:

1. *Is it possible to infer networks with the provided honey bee tracking data?*
   What challenges and limitations does the data set imply? What pipeline parameters are necessary?
2. *What kind of worker-worker interaction networks emerge and how are they structured?*
   How are those networks characterized and are they different from a random network?
3. *Does the network display a meaningful community structure?*
   Are those communities robust in terms of pipeline parameters?
4. *How are communities characterized?*
   Do they reflect known colony behavior in terms of age and spatial distribution?
5. *How do the communities emerge over time?*
   Are they stable regarding their properties? How do members move between communities?

This work is meant to be the foundation for further more hypothesis-driven research using a network science approach to study the complex system of honey bee colonies and their collective behavior.

## 1.3 Methodology

The methodology of this thesis follows in the first part a classical explorative data analysis procedure to understand the given data set and estimate its quality. The elaborated characteristics of the dataset are then used to define parameters and steps for the network extraction pipeline. The pipeline is tested, reviewed and refined in an iterative process. The resulting networks are evaluated after each refinement step of the pipeline using the following approaches: (1) check network properties by investigating the effects of pipeline parameters and features (2) check network quality using the provided age information of bees, (3) comparing to a random graph model (Erdös-Renyi), and (4) repeatability of known results concerning community structures.

## 1.4 Outline

This thesis is organized as follows. Chapter 2, gives a short introduction into social network analysis (SNA) and defines network measures, terms, and algorithms used throughout this work. In chapter 3 a brief summary about the current state of research concerning social insect networks, temporal (dynamic) networks and community detection in animal social networks is given. How the networks are derived from the given dataset is described in chapter 4. Also, the implementation of the network pipeline and its parameters are presented, as well as steps and decisions during the network analyses and community detection process. The results of the network analysis and the characteristics of the extracted communities are presented in chapter 5. Finally, in chapter 6 I explore the results and discuss limitations. I conclude with directions for future work.

# Chapter 2

# Theoretical Background

The following chapter gives a short introduction into social network analysis (SNA). It introduces animal interaction networks as a special type of network. It defines terms and concepts used throughout this work and explains networks metrics and algorithms of which we will make use of.

A *social network* is a representation of a social structure comprising actors such as individuals, affiliations, as well as their social interactions. The network model conceptualizes social, economic, or political structures as lasting patterns of interactions between actors [27]. In mathematical terms, networks are graphs, and thus consist of *nodes* (vertex, representing individuals), and *links* (edges, relationships or interactions). Social network analysis provides a set of methods, measures and theories, borrowed from network and graph theory, to investigate social structures and its dynamics.

## 2.1 Animal Interaction Networks

This work is focusing on the special case of animal social networks. Networks where individuals are nodes and edges are defined as interaction events between individuals are called *interaction networks*, sometimes also contact networks. According to Charbonneau et al. [7] those interactions used as an edge can be of four different types when looking at animal networks: spatial proximity, physical contact (usually with antennae, "antennation"), a food exchange event (trophallaxis), or specific communication signals.

I decided to use spatial proximity as the interaction type for edges. Those edges can be directed (e.g. trophallaxis) or undirected, weighted or unweighted. As edge weights the frequency and duration of interactions are commonly used.

## 2.2 Network Measure, Metrics and Algorithms

The definitions are mainly taken from Barabási [2] for general definitions and from Wey et al. [28] for more animal specific definitions, if not stated otherwise. The measures are for undirected and weighted graphs.

**Network size** $N$ is the total number of nodes (animals).

**Number of links** $L$ is the total number of links (social interaction).

**Edge weight** $w_i$ of an edge $l_i$ is an indicator of how important that edge is.

**Density** $D$ is the number of realized links divided by the number of theoretically possible links is defined as $D = \frac{2L}{N(N-1)}$.

**Component** is a subnet of nodes in a network, so that there is a path between any two nodes that belong to the component.

**Degree** $k_i$ of a node $n_i$ represents the number of edges a node has; so the number of other animals this animal interacts with.

**Average Degree** $\langle k \rangle$

**Strength** $s_i$ of a node is also called the weighted degree. It measures the total weight of edges connected to a node $n_i$ and is definded as $s_i = \sum_{j=1}^{n} w_{ij}$ according to Barrat et al. [4]

**Average Strength** $\langle s \rangle$

**Path length** $d$ the shortest number of links between two nodes.

**Average path length** $\langle d \rangle$ is the average of all shortest path between all pairs of nodes.

**Diameter** $d_{\texttt{max}}$ is the longest of all path length. The distance between the two furthest nodes, the longest possible path length in the network.

**Global clustering coefficient** $C_\Delta$ also called transitivity. According to [27] is is defined as $C_\Delta = \frac{3 \times \texttt{number of triangles}}{\texttt{number of connected triples}}$.

**Local clustering coefficient** $c_i$ of a node $n_i$ quantifies how close its neighbours are to being a clique (complete graph).

### 2.2.1 Network Centrality and Centralization

When looking at the networks local structure (node level), it is possible to identify nodes, which are important or central to the network, regarding different aspects. This concept is called *centrality* and measures the influence of a node in a network. [19]

In the course of analysing networks and their local node level structures, you will find and encounter the most important (central) nodes and vertices by indicators of centrality. These indicators give values to the nodes and therefore they can be listed in a way of importance.

**Degree Centrality** Degree centrality $C_D^i$ of a node $n_i$ is the normalized degree $k_i$ in relation to the whole network, it is calculated as follows:

$$C_D^i = \frac{k_i}{N-1}$$

**Eigenvector Centrality** The eigenvector centrality $x_i$ of a node $n_i$ is the sum of its connections to other nodes, weighted by their centrality.

$$x_i = \frac{1}{\lambda} \sum_j A_{ij} x_j$$

It is like a recursive version of degree centrality. So a nodes importance (centrality) is inceased by having neighbours that are themselves important. Eigenvector centrality gives each vertex a score proportional to the sum of the scores of its neghbours. [19]

**Closeness Centrality** Is is the average length of the shortes path between node $n_i$ to all other nodes in the network. The more central a node is the closer it is to all other nodes. Mean distance from a node to other nodes. [19]

$$C_C^i = \frac{N}{\sum_j d_{ij}}$$

**Betweenness Centrality** It measures the extend to which a node lies on paths between other nodes. Nodes that occur on many shortest paths between other nodes have higher betweenness that those that do not.

## 2.3 Temporal Networks

[TODO]

## 2.4 Community Detection

To understand the large-scale structure of networks, one can look at the network's community structure. Communities are naturally occurring groups within a network, usually also called clusters, cohesive groups or modules and have no widely accepted, unique definition [22]. For my work, I adapt the definition according to Barabási [2]: "In network science, we call a community a group of nodes that have a higher likelihood of connecting to each other than to nodes from other communities." [2, p. TODO]. In contrast to a simple graph partition, the number and size of communities is not predetermined or set in advance.

Communities in animal social networks refer to groups of individuals that are associated more with each other than they are with the rest of the population. These

communities reflect an intermediate level of social organization, which is located between the individual and population level [9].

There are a lot of different approaches and algorithms who adress the detection of communities. Fortunato [12] gives an extended overview of the various types of community detection algorithms. Explaining any of those would be beyond the scope of this work. For example, traditional methods include algorithms based on graph partitioning, hierarchical clustering, and spectral clustering. There are also divisive and agglomerative algorithms.

The algorithms used in this work are described in the following sections and include the leading eigenvector [20] and walktrap [23] algorithm.

### 2.4.1 Modularity

Modularity is a quantity, that measures the quality of a partitioning. It can be used to compare a community partition to another and decide for the better one. Modularity optimization is also used for community detection algorithms.

A high modularity of a network indicates more connection between nodes within a community and fewer connections between nodes of different communities. The basic idea is: If the fraction of links inside the community is higher, than expected in the same community of a related random graph having the same degree distribution, then it is a community in the sense of modularity.

### 2.4.2 Leading Eigenvector

This algorithm was proposed by Newman [20]. It uses the eigenvectors of matrices for finding community structures in networks. It is a top-down hierarchical approach that optomizes modularity.

"In each step, the graph is split into two parts in a way that the separation itself yields a significant increase in the modularity. The split is determined by evaluating the leading eigenvector of the so-called modularity matrix, and there is also a stopping condition which prevents tightly connected groups to be split further. (stackoverflow)

"The heart of this algorithm is the spectral optimisation of modularity by using the eigenvalues and eigenvectors of the modularity matrix. First, the leading eigenvector of the modularity matrix is calculated, and then the graph is split into two parts in a way that modularity improvement is maximised based on the leading eigenvector. After that, the modularity contribution is calculated at each step in the subdivision of a network. It stops once the value of the modularity contribution is not positive." [29]

### 2.4.3 Walktrap

[TODO: make own text] This algorithm is based on random walks. The general idea is that if you perform random walks on the graph, then the walks are more likely to stay within the same community because there are only a few edges that lead outside a given community. Walktrap runs short random walks (depending on one of its parameters, default is 4) and uses the results of these random walks to merge separate communities in a bottom-up manner. Again, you can use the modularity score to select where to cut the dendrogram. (stackoverflow)

"It is a hierarchical clustering algorithm. The basic idea of this method is that short distance random walks tend to stay in the same community. Starting from a totally non-clustered partition, the distances between all adjacent nodes are computed. Then, two adjacent communities are chosen, they are merged into a new one and the distances between communities are updated. This step is repeated $(N-1)$ times." [29]

### 2.4.4 Communities in Evolving Networks

According to Aynaud et al. [1] and Bródka et al. [6] there are three main approaches for community detection in temporal networks (also called community tracking): (1) using a static community detection algorithm on several snapshots and then solving a matchig problem, (2) using algorithms who are directly suited for temporal networks and (3) using incremental or online algorithms when processing data streams. For each of the three approaches, several mehods already exist.

As community tracking is not the main focus of this work, I chose to apply the most intuitive approach aout of approach (1): detecting static communities for each snapshot and then matching those communities using set theory. Two communities at successive timesteps are matched if they share enough nodes. The *match value* (between 1 and 0) between two communities $C$ and $D$ according to [14] is defined as:

$$\texttt{match}(C, D) = \texttt{min}\left(\frac{|C \cap D|}{|C|}, \frac{|C \cap D|}{|D|}\right) \tag{2.1}$$

A high match value accurse, when two communities share a lot of nodes and are of a similar size. Communities with the highest value are matched. A threshold should be applied to more precicley define what "share enough nodes" means.

# Chapter 3

# Related Work

Look at [24] for static network analysis stuff they measured (ants, only small amount with observed interaction using 20 Minute videos).

## 3.1 Networks in social Insects and Honey Bees

In *Social Insects: A Model System for Network Dynamics* [7] a good overview is given about Role and Types of Networks in Social Insects.

## 3.2 Spatial Proximity Networks

TODO short summary, also on network sience methods:
*Long-term dynamics in proximity networks in ants* [15]

*Contact networks and transmission of an intestinal pathogen in bumble bee (Bombus impatiens) colonies* [21]

## 3.3 Temporal Aspects in Networks

## 3.4 Network Analysis Tools

# Chapter 4

# Approach and Implementation

In this chapter the basic work flow is described in detail. The process is mainly driven by an exploratory approach, but follows primarily Farines and Whiteheads [10] primary steps and key considerations for social network analysis to non-human animal data. The adapted and resulting process is visualized in figure 4.1.

The dataset was first analysed regarding data quality and to form an understanding of the dataset and the behaviour of bees in general. Those findings were used to define nodes and infer associations to build the network, respectively derive the parameters for the network pipeline. The static and temporal networks are analysed using network science tools and methods. For testing hypothesis the networks are combined with spatial and age information. Each step is explained within the following sections.

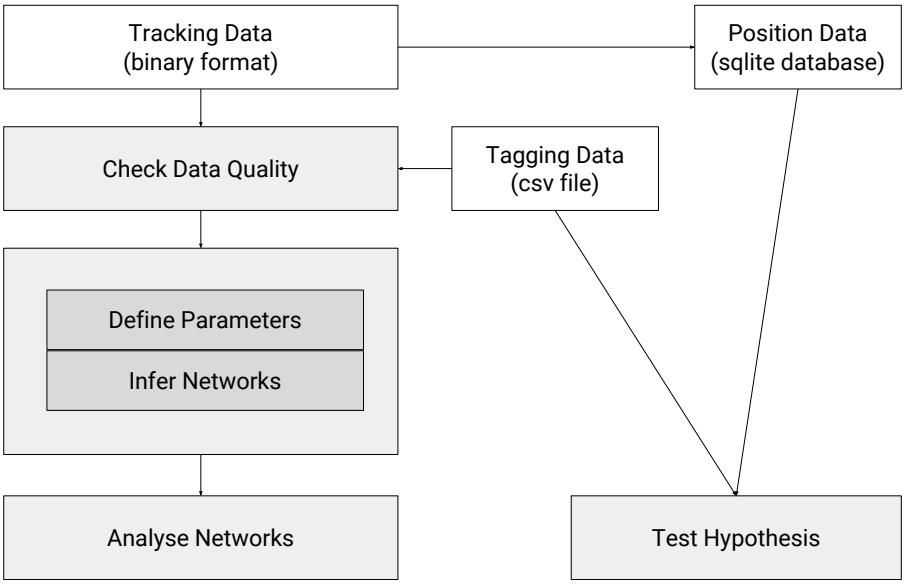



**Figure 4.1:** Steps of the Research Approach

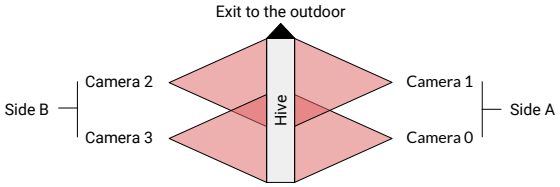



**Figure 4.2:** Camera Setup in 2016

## 4.1 The Dataset

The dataset derives from video files, that capture tagged honey bees of one colony in an observation hive. Each individual of the colony, including about 3000 bees, were tagged with circular 12-bit markers (figure 1.1, section 1). Four cameras were used to film the hive. The setup of the cameras is illustrated in figure 4.2.

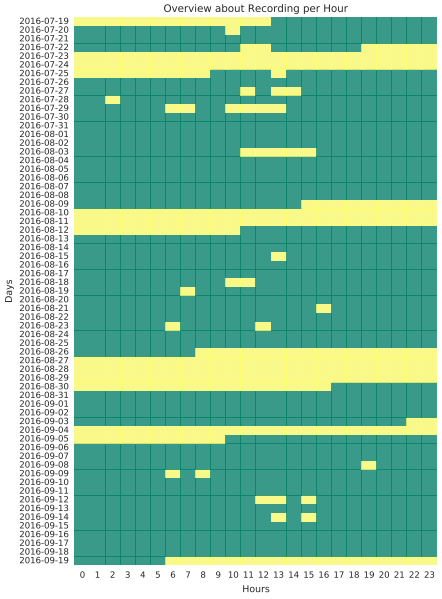



**Figure 4.3:** Recording Season with maintainance and failures: *Green* indicates recording went without any big interruption; *Yellow* indicates maintainance work or technical failures of one or all cameras. This is calculated using the expected number of files produced by each camera per hour.

The recording season lastet nine weeks (63 days), around the clock, from 19.07.2016 until 19.09.2016, with some interruptions due to maintainance and technical failures. I chose four consecutive days (30.07, 31.07, 1.8,2.8), marked in figure 4.3 for further analysis and the 26.07. for testing purpose.

The recording resolution of each camera is three frames per second, with 1024 frames per video file. For each frame, bees were detected by using an image analysis pipeline, which is explained in detail in [26]. The resulting detection data is stored in a binary file format. A python library called *bb-binary*[1] provides easy access to the binary

[1] `https://github.com/BioroboticsLab/bb_binary`; Last accesed: 2106-02-16, 04:28PM

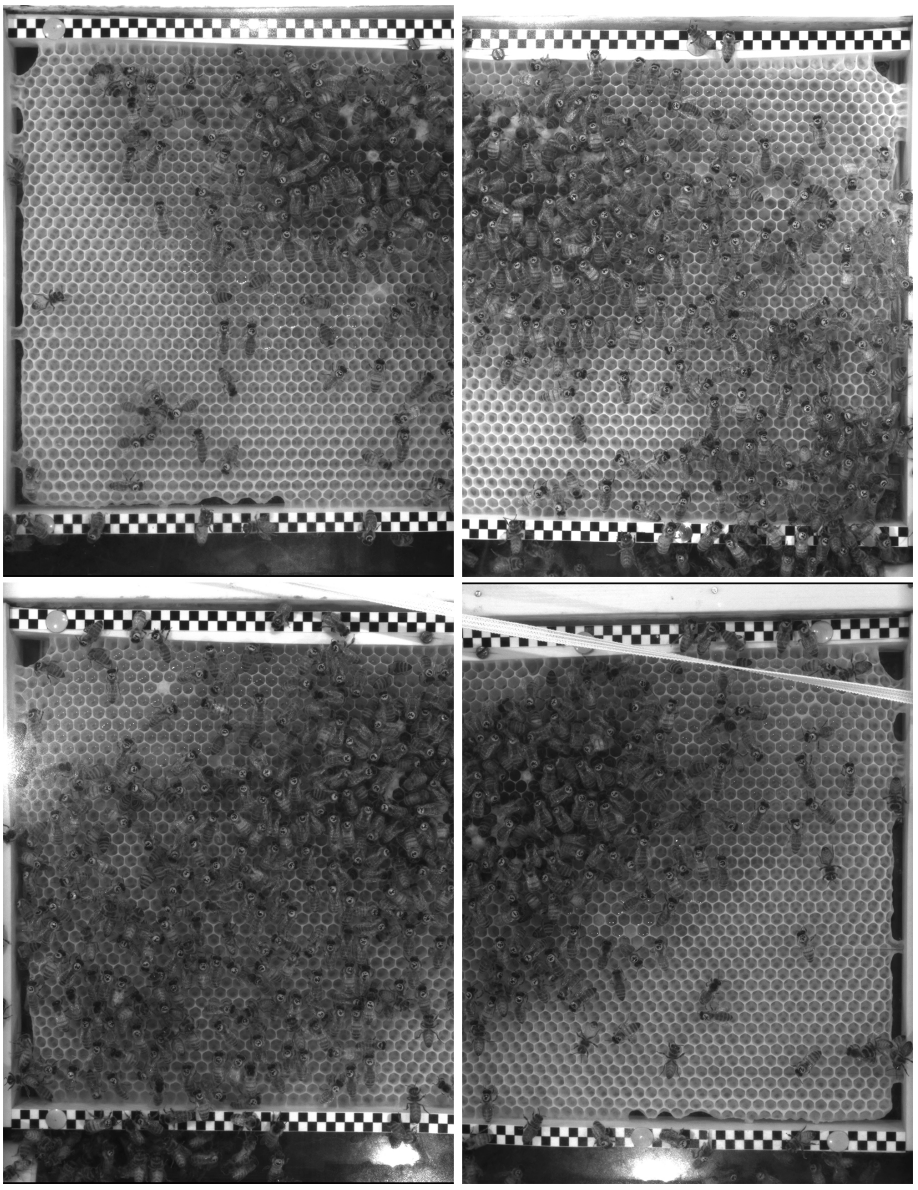


**Figure 4.4:** xxx

files. Each file in bb_binary file format corresponds to a video file of a single camera. The size of the complete dataset for 2016 is 470 GB, about 7.5 GB of binary data per day.

Exactly 3.191 bees were tagged. The tagging period was 67 days long. The tagging started on 28.06.2016 (22 days before the recording started) and lasted until 02.09.2016 (17 days before the recording ended). The young bees were tagged and then added to the hive, about noon each day. The overall tagging frequency is shown in figure 4.5. The hatching day for each bee was documented. Therefore the age of each bee at a certain point in time can be calculated.

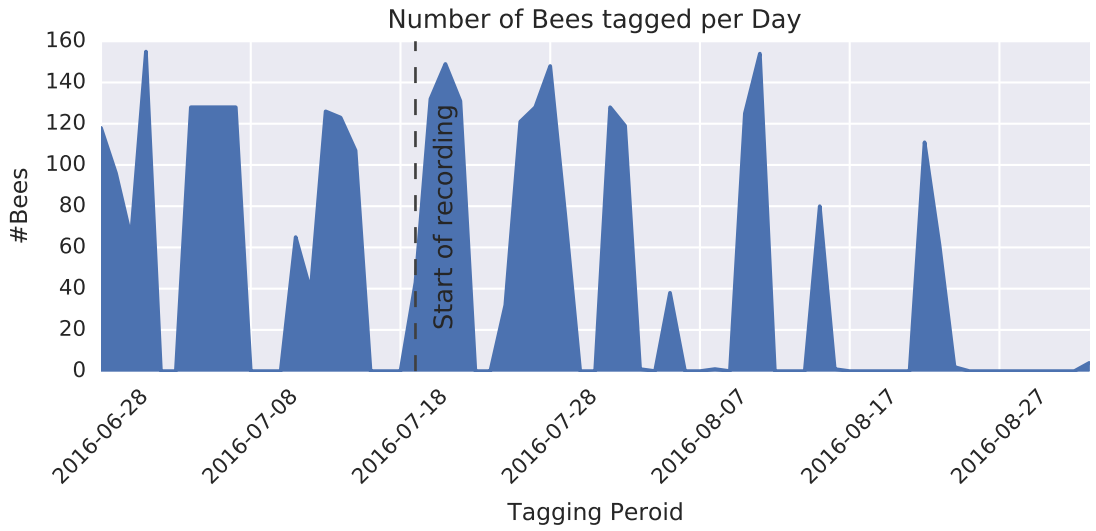



**Figure 4.5:** Tagging frequency of bees: The bees were primarily tagged during the week. On average 48 bees were tagged each day, considering only tagging days, the average is about 91 (±50) bees (median 118).

### 4.1.1 Structure of the Dataset

The data is organised in *frame container*, wich corresponds to a video file of a single camera. A frame container holds all *frames* for that specific video. Each frame has a list of all detected bees. A *detection* has the following attributes, which are relevant to this project:

- **xpos**: $x$ coordinate of bee with respect to the image in pixel
- **ypos**: $y$ coordinate of bee with respect to the image in pixel
- **radius**: of the circular 12-bit tag
- **decodedId**: decoded 12-bit id

Besides further information, the frame container specifies the camera, which took the video. A frame is also attributed with a timestamp. The data can be accessed iterating on the frame level, using two timestamps (start and end) for specifying the time interval. The complete data scheme can be found on github[2].

### 4.1.2 ID Probabilities and Confidence Level

With a 12-bit ID, 4096 bees can be tagged. Each bit of the decodedId is not a 1 or 0, but represents a probability between 0 and 255. It indicates the reliablility of the image analysis pipeline for that specific bit. A number closer to 255 represents a 1, a number closer to 0 a 0. A *confidence value* can be calculated for each ID. The

[2] `https://github.com/BioroboticsLab/bb_binary/blob/master/bb_binary/bb_binary_schema.capnp`; Last accessed: 2106-02-16, 04:46PM

confidence value of a single bit is its distance to 128, discretized to a value between 0 and 1. The confidence value of an ID is therefore the minimum of all bit confidences. [TODO: in leons Arbeit ist das gut erklärt]

The amount of data that remains for further processing and its quality depends heavily on the chosen *confidence level*. Figure **??** shows the relationship between confidence level, the amount of remainig data and the number of unique IDs.

### 4.1.3 Time Series of Bees

The original dataset (a number of frames containing bee detections), is transformed to binary *time series of bees*, depicted in figure 4.7 (left and middle). A time series of a bee is a sequence of zeros and ones indicating the absence and presence of a bee over a specified time interval. As expected the confidence level has an effect on the resulting time series of a bee. A high confidence leads to more gaps in the series and also to more shorter gaps (see figure 4.8).

### 4.1.4 Data Quality

The quality of the data could be indirectly checked using the age information of bees. On 26.07.2016, about half of the bee tags (2014 of 4096) were assigned to a bee. This day was chosen to determine the effects of the confidence level on data quality. At first, for each detection the age of that bee was calculated, a negative age was counted as a wrong detection. I assumed that the number of wrong detections indicated by negative age also occured in the positive half, but remained unseen, therefore I doubled the error. Secondly, the number of wrong unique IDs is also determinde using the age test. Figure 4.6 shows that even though the number of wrong detections decreases steadily with an increasing confidence level, but the number of wrong IDs only starts to decrease with a very high level.

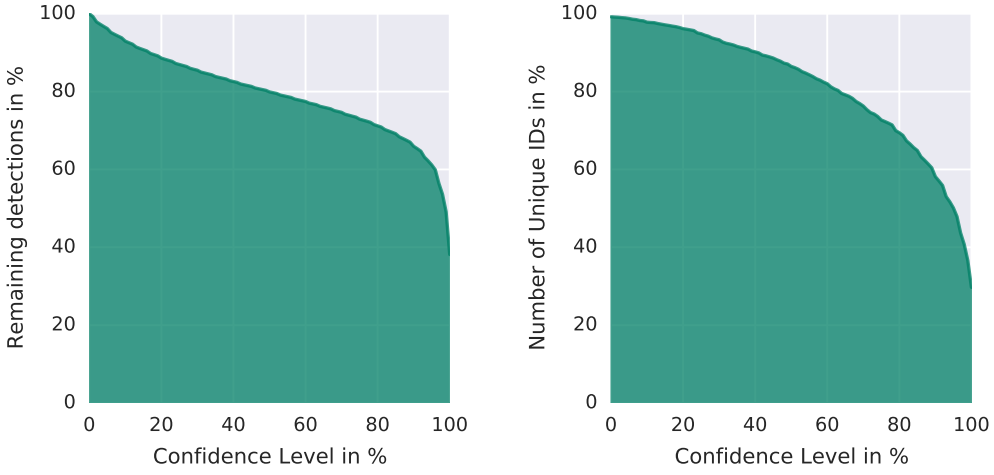
Even with an confidence level of 100%, 30.2% of the unique IDs are wrong (have a negative age), corresponding to only 2.5% of detections. Therefore wrong IDs need to be filtered out anyway (independent of the confidence value), to obtain a more reliable dataset. A good indicator is the detection frequency of IDs. IDs with a negative age are on average less detected than IDs with a positive age.

#### Frequency Filter

IDs who definately exists, but their age can to be determinded are excluded from the analysis completely. These are bees, who were tagged later ($n = 10$)[3] and IDs whos detection frequency is absurd high but there age is unknown ($n = 7$)[4]

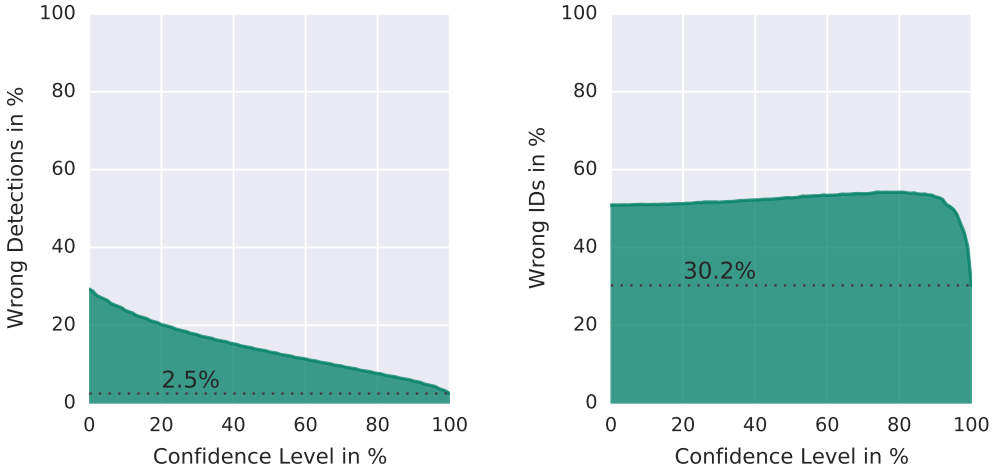
For each analysis day the number of detections per ID is obtained, excluding the mentioned IDs above. The frequency distribution tells that, IDs with a negative

[3] id= [2, 74, 2045, 3172, 3764, 3796, 3827, 3836, 3844, 3940]
[4] id=[has changed! 17, 168, 801, 888, 2045, 2357, 2607]




**(a)** Remaining Detections depending on the level of confidence.

**(b)** Number of unique IDs depending on the level of confidence.




**(c)** Wrong detections in relation to the level of confidence.

**(d)** Number of wrong IDs depending on the level of confidence.

**Figure 4.6:** The number of wrong detections decreases with an increasing level of confidence. In contrast, the number of false IDS becomes noticeably less only with a very high confidence level. The amount of remaining data decreases with an increasing confidence level. The number of unique IDs behave similar. (Dataset: 26.07.2016, 16:00-16:05) (Dataset: 26.07.2016, 16:00-16:05)

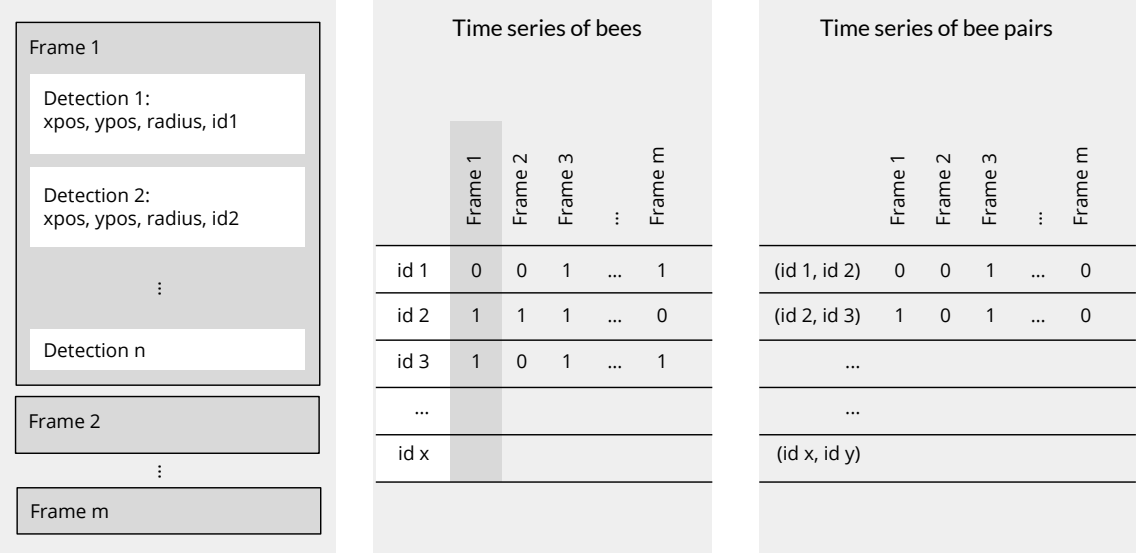



**Figure 4.7: Left:** original dataset - bb_binary data containing frames and detections; **Middle:** transformation to time series - zero indicating absence of the bee, one indicating presence of the bee; **Right:** transformation to bee pairs - zero indicating eighter one or both bees are not present at the same time or not close to each other, one indicating bees are present at the same time and nearby.

age are detected less often (704 frames ±65) than bees with a positive age (36.603 frames ±2.345). The cutoff is 99% of the negative IDs distribution. All IDs with a detection frequency below $4737 \pm 644$ frames are discarded. A list with possible (valid) IDs is kept for each day. Using this list, faulty detections can be filtered out beforehand.

### 4.1.5 Implications

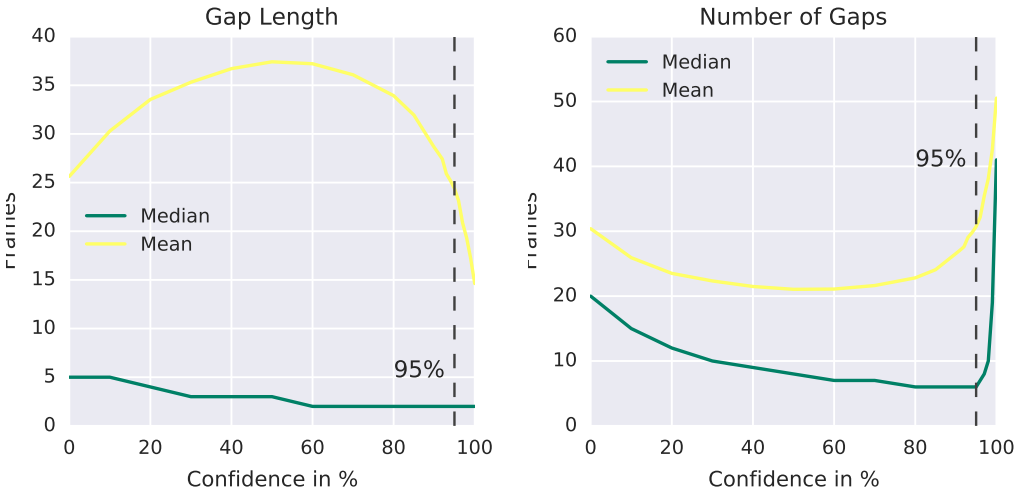
For further analysis I use the following days: 14.08.2016, 17.08.2016, 20.08.2016, 02.09.2016 with a confidence level of 95%. This period is chose because of XY.

Because bee time series contain a lot of short gaps (mean = 3, 95% confidence), the inferring of edges (bees who are close to each other at the same time), should be not that strict, or at least variable. This has to be taken into acoount, when looking at spatialy close bees.

## 4.2 Inferring Networks

The following part describes the pipeline for generating spatial proximity networks out of honey bee tracking data. A node in the network is a bee. They are distinguished by IDs. Only bees are in the network who interact at least once with another bee.

undirected and weighted, aggregated networks




**(a)** Length of Gaps depending on the level of confidence.

**(b)** Number of Gaps depending on the level of confidence.

**Figure 4.8:** Influence of the confidence level of the length of gaps and number of gaps: With an increasing level of confidence the average gap length decreases and the number of gaps per bee series increases. (Dataset: 26.07.2016, 16:00-16:05)

Two bees are associated (spatially close to each other), if their distance is minor to a *maximum distance*. As everything is very close in a bee hive this value is hard to choose. Only this criteria is very week, meaning having a resolution of three frames per seconds results in interactions which could only last for 0.33 seconds. So an additional parameter the *minimum contact duration* is introduced, it is the minimum time they have to spend at least nearby to be called associated.

Taking the fragmentation of tracks into account, it is obvious that two bees could be nearby but not at the very same time, but slightly shifted. So the minimum contact duration would be too errow prone. To overcome this issue one could correct the bee tracks, by filling gaps of varius sizes and interpolating the position of that bee accordingly. This is rather time consuming for this amount of tracking data (TODO: naja so doll auch nicht, scipy.ndimage.morphology.binary_dilation) and also considering, that the tracking data is going to be improved in the future, then manipulating the raw data seems senseless. I rather perform a gap filling (maybe similar to binary dilation) on the time series of pairs, but not on the bee tracks, because this is independent of the input data.

Edges are attributed with two parameters. The first one is the frequency of contacts, so how often they share a close position. The second parameter is the total duration of contact, how many time frames in total they spend close by.

### 4.2.1 Network Pipeline

The network pipeline takes as input a path to the bb-binary data and outputs a graph in graphML file format. The pipeline takes the following parameters:

- path to data
- confidence in percent
- gap size in frames - this is used to corret the time series of bee pairs
- maximum distance in px - define what close means (spatial proximity)
- minimum contact duration in frames - how many frames bees need to spend nearby
- cutoff in percent - IDs with a number of total detections below X percent of the mean frequency are discarded
- start timestamp - start of network slice
- window size in minutes - size of time window for aggregating the network
- number of used CPUs for parallelization
- year - calculate IDs and set camera setup for 2015 or 2016

The pipeline is parallelized on frame level, that means, each process gets a portion (frames for a timeinterval of five minutes) of the data and extracts interactions/edges. The main process adds everything up and creates a network. The steps are the following:

1. **Filter detections by confidence**
   For each of the four camera the detections are filtered by the confidence level.
2. **Simple stitching**
   Each side of the hive consists of two cameras. The $x$-coordinates of each detection (of the right cameras) is moved further to the right, also adding an offset of $2 \times$ `maximum distance`. So the left and the right detection of each side of the hive are move into one reference system.
3. **Syncronize Cameras**
   For each side of the hive the cameras need to be syncronized. In the normal case the difference between consecutive frames should be about 0.332 seconds, due to technical problem this value can be lower (0.003 ) and higher (2.932) at certain times. Cameras 3 and 2 and cameras 1 and 0 are matched, frames without a match are dropped (shorter number of frames, matchen, threshold 0.33/2, minimum).
4. **Discard Detections with certain IDs**
   All detections whos ID is in a list are keept, other detections are discarded. (see frequency filter)
5. **Extract close pairs**
   For each side of the hive, all close pairs according to the maximum distance

parameter are calculated and then joined together.

6. **Generate time series of bee pairs**
   The data structure (frames and detection) is transformed to time series of bee pairs.
7. **Correct pair time series.**
   The time series of bees are corrected by filling in the gaps of length `gap size`.
8. **Extract edges**
   The edges and its attributes (frequency and duration) are extracted from the time series of bees using the minimum contact duration parameter. A sequence of at least X ones counts as one interaction. The frequency of those series adn the total duration (number of ones) are the attributes.

### 4.2.2 Pipeline Parameters

For performing the network analysis, I chose the pipeline parameters as follows:

**Confidence** As explained in section4.1.2, the confidence is set to 95%.

**Maximum Distance** I chose the length of a bee body, according to Baracchi and Cini [3], as the maximum distance between two bees (figure 4.9a). The average bee length of 212px (±16px) was determinded by manually measuring the length of all bees ($n = 337$) in four images (one for each camera, 21.07.2016, 03:00PM) using the tool ImageJ[5].

**Gap Size** The gap size is set to two frames. This value corresponds to the median gap length in the time series of pairs (`mode` = 1, `mean` = 27). [TODO: what dataset was used (95% confidence, XXX% cutOff, XXXpx maximal distance, date, camera)]

**Minimum Contact Duration** This is set to three frames (one second). This corresponds to Mersch et al. [17], they as well exclude interactions below one second. Looking at the frequency distribution of chains of ones (1, 11, 111, and so on) of the pair time series (after filling the gaps), then: `mode` = 1, `median` = 2 and `mean` = 4. Three frames corresponds to 57% of all chains, this seem to be reasonable. [TODO: what dataset was used (95% confidence, XXX% cutOff, XXXpx maximal distance, date, camera)]

The networks are not thresholded (according to [10]).

## 4.3 Static and Temporal Analysis

Despite the possibility of generating networks of different granularity (resolution is minutes), here for further analysis daily networks (10h, two hours after sunrise until zwo hours before sunset) are aggregated.

[5] `http://imagej.net/Welcome`; Last accessed: 22.02.2016

### 4.3.1 Static Network Analysis

The following network properties were analysed for a static day and hour network.
TODO: list of properties. (similar to what others have done) nodes, edges, density, diameter

### 4.3.2 Temporal Analysis

three day networks (2 days gap)
one network 2 weeks later

### 4.3.3 Community Detection

I tested all community detection algorithms implemented in python, to find an algorithm, which works well for my case of animal social networks. The three most common python libraries for network analysis were reviewed: NetworkX[6], igraph[7], and graph-tool[8])

The algorithm needs to fulfill the following criteria:

- Support for large and very dense networks ($N > 1000$, $D > 50$ %)
- Support weighted edges
- Fast runtime

Table 4.1 gives an overview about the twelve algorithms reviewed. Five algorithms did not terminate after 15 minutes and were therefore excluded from further investigations. Infomap and label propagation tend to partition all nodes into a single community, this is known especially in dense graphs [29, 12]. The Louvain algorithm is the same as multilevel, but takes longer producing almost the same communities and therefore was also excluded. Walktrap was tested for different step size parameters, as suggested in [23], the communities remained almost the same (only a few nodes switched communities).

I had a closer look at fastgreedy, leading eigenvector, multilevel, and walktrap regarding the number of detected communities and community size for all three networks. Table 4.2 shows the results. All algorithms found at least two communities. Except for leading eigenvector, there is a tendency that a third community exists. I decided to use two algorithms for community detection: leading eigenvector and walktrap. Farine and Whitehead [10] explains that leading eigenvector is often used with animal social networks and works well. Walktrap is chosen for also examining the possible third community.

[6] `https://networkx.github.io/`; Last accessed: 16.03.2016, 6:36 p.m.
[7] `http://igraph.org/python/`; Last accessed: 16.03.2016, 6:38 p.m.
[8] `https://graph-tool.skewed.de/`; Last accessed: 16:03.2016, 6:39 p.m.

There are comparative analysis of community detection algorithms, e.g. [29, 13]. They seem to be promising, but assume eighter a power law degree distribution or evaluate networks with a low density, which is not applicable here.

**Table 4.1: Comparing community detection algorithms** Comparison of algorithms implemented in python. Criterias are the support of weighted edges, runtime and number of communities. A runtime indicated by − mean no termination after 15 minutes.

| | fastgreedy[1] | leading eigenvector[1] | louvain[2] | multilevel[1] | walktrap[1] | infomap[1] | label propagation[1] | edge betweenness[1] | k-clique communities[2] | optimal modularity[1] | spinglass[1] | statistical inference[3] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Edge weights | × | × | × | × | × | × | × | | × | × | × | |
| Runtime in sec | 3.6 | 6.3 | 11.7 | 0.7 | 19.4 | 13.2 | 0.2 | − | − | − | − | − |
| Communities | 3 | 2 | 2 | 3 | 2 | 1 | 1 | − | − | − | − | − |
| | 473<br>434<br>15 | 488<br>434 | 469<br>453 | 462<br>427<br>33 | 490<br>431<br>(1) | 922 | 922 | | | | | |

[1] igraph, [2] NetworkX, [3] graph-tool

## 4.4 Attributed Data and Hypothesis Testing

Hypothesis
(1) Communities reflect groups of bees working in different areas of the hive and
(2) Communities reflect different age groups

The data which was used to test the hypothesis (1) is saved in a sqlite database for faster access, because using bb_binary (parsing the data over and over again) was to slow. For testing if lists of positions (spatial ditribution) are different the test XY was used [TODO: what to use here]

For hypothesis (2) the data is stored as a csv file of birth dates of each bee. For testing if age goups are different the Kolmogorov Smirnov Test was used.

**Table 4.2: X** X

| | fastgreedy | leading eigenvector | multilevel | walktrap |
|---|---|---|---|---|
| Network 1 | 473 | 488 | 462 | 490 |
| | 434 | 434 | 427 | 431 |
| | 15 | | 33 | (1) |
| Network 2 | 504 | 503 | 481 | 372 |
| | 467 | 475 | 439 | 311 |
| | 7 | | 58 | 294 |
| | | | | (1) |
| Network 3 | 534 | 537 | 505 | 310 |
| | 388 | 385 | 415 | 390 |
| | | | (2) | 231 |

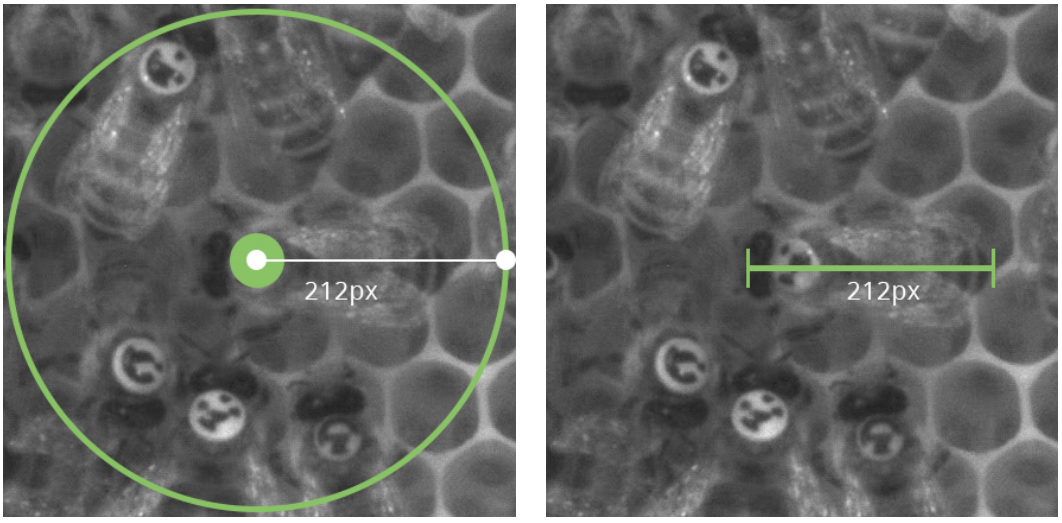
## 4.5 Implementation, Runtime and Complexity

For implementing the network pipeline python, with pandas and numpy, are used, because the bb_binary library, for accessing the tracking, data is only available in python. The networks, in graphML format, are created using the python library *NetworkX*[9] in version 1.11. iGraph for community detection
some bash scrips for generating multiple networks

bottleneck is reading bb_binary data into pandas dataframes
using multithreading for distribution on frame level (a process gets X frames for processing)

maybe some table with how long nees what with how many cores (hom much RAM and so on)

[9] `https://networkx.github.io/` ; Last accessed: 2017-02-17, 08:07PM




**(a)** Contact Radius **(b)** Bee Length

**Figure 4.9:** Distance Between Bees: A length of a bee is chosen as the maximal distance between bees.

# Chapter 5

# Results

I analyzed three daily networks, these are referred to below as network 1 ($N = 922$), network 2 ($N = 978$) and network 3 ($N = 922$), or $g_1$, $g_2$, and $g_3$. Each network is aggregated for ten hours (108 000 frames) starting at 8 a.m. and lasting until 6 p.m. I chose the 20.08.2016, 22.08.2016, and 24.08.2016, because at this point in time the hive also contained tagged foragers, thus older bees. At the same time young bees were added to the colony, see table 5.1 for details. The age distribution for each network is shown in figure 5.1. The match value, according to(2.1) between network 1 and 2 is 85% (833 bees), between network 2 and 3, it is 84% (823 bees) and between network 1 and 3, it is 78% (716 bees).

**Table 5.1: Sampling period** Overview of the chosen day networks including the number of added bees and the time they were added to the hive.

| | 20.08.16 | 21.08.16 | 22.08.16 | 23.08.16 | 24.08.16 |
|---|---|---|---|---|---|
| Network ID | 1 | - | 2 | - | 3 |
| Number of added bees | 0 | 0 | 110 | 60 | 0 |
| Time added | - | - | 2 p.m. | 6 p.m. | - |

**Table 5.2: Global network properties** $N$ is the number of nodes, $L$ the number of edges, $D$ is the diameter, $\langle d_{\texttt{max}} \rangle$ is the average path length, $C_\Delta$ the global clustering coefficient, $\langle k \rangle$ the average degree and $\langle s \rangle$ represents the average strength, as introduced in section 2.2.

| | $N$ | $L$ | $D$ | $\langle d_{\texttt{max}} \rangle$ | $\langle d \rangle$ | $C_\Delta$ | $\langle k \rangle$ | $\langle s \rangle$ |
|---|---|---|---|---|---|---|---|---|
| Network 1 | 922 | 291179 | 0.69 | 3 | 1.32 | 0.79 | 631.62 | 5680.17 |
| Network 2 | 978 | 256066 | 0.54 | 3 | 1.46 | 0.72 | 523.65 | 3977.94 |
| Network 3 | 922 | 259421 | 0.61 | 3 | 1.39 | 0.75 | 562.74 | 4205.99 |

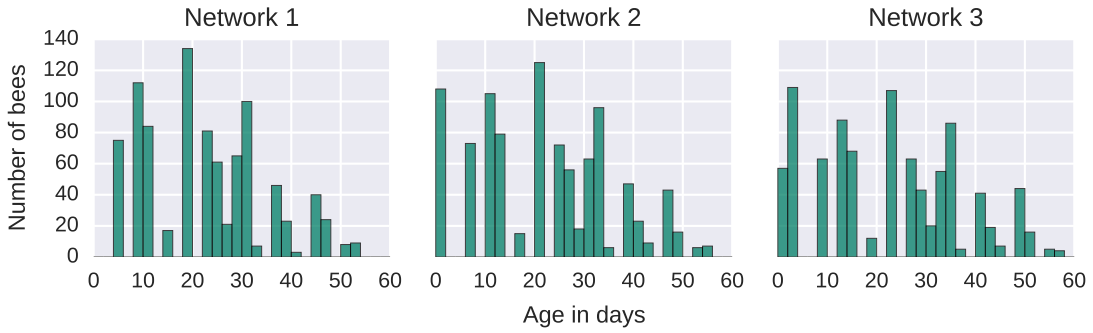



**Figure 5.1: Age distribution per network** The width of a bar corresponds to two days. For each network bees with a negative age and the queen were removed (11, 10, and 9 bees).

## 5.1 Network Topology and Type

TODO

### 5.1.1 Global Structure - Network Level

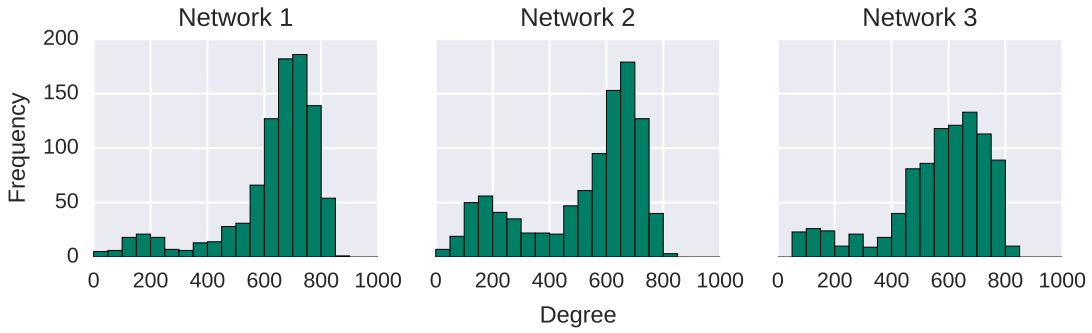
Each network consists of one giant component. Table 5.2 summarizes basic network properties. The network densities are over 50%. The diameter is for all three networks 3 and the average path length is below two. Compared to an Erdös-Renyi random graph the global clustering coefficient is higher: 0.68 for $g_1$, 0.54 for $g_2$, and 0.61 for $g_3$, with an SD $< 0.00005$ averaged over 100 runs. The average strength is ther average number of times two bees are spatially close to each other. On average a bee is connected to 68% for $g_1$, 54% for $g_2$ and 61% for $g_3$ percent of all bees in the network. The degree distribution is bimodal (see figure 5.2a. The distribution of strength is shown in figure 5.2b and the distribution of edge weights in figure 5.2c.

### 5.1.2 Local Structure - Node Level

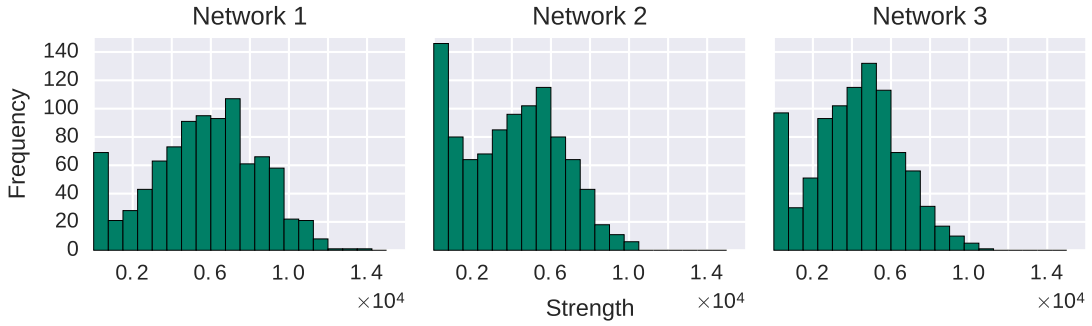
[TODO: PLOT closeness, betweenness, local clustering coefficient, eigenvector centrality, weighted and unweighted]

### 5.1.3 Network type

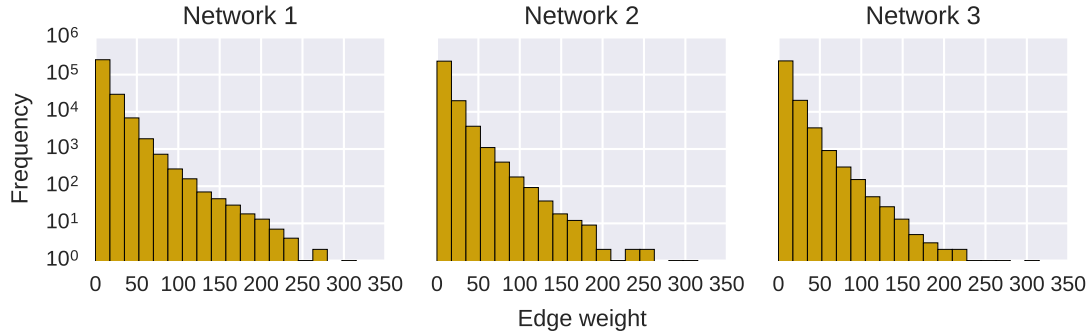
[TODO: compare to random graph]
degree distribution random poisson or binominal, not random power law
giant component, connectedness
average path length and diameter very small, small world phenomenon
higher clustering coefficient than in random




**(a)** Degree distribution




**(b)** Strength distribution




**(c)** Edge weight distribution

**Figure 5.2: Degree, strength and edge weight distribution** for all three networks.

## 5.2 Network Metrics in Relation to Age of Bees

[TODO: all plot in relation to age]

**Table 5.3: Overview about communities per network** Communities marked with * contain the queen. Age and standard deviation (SD) are measured in days.

| | ID | Members | Proportion | Age | SD |
|---|---|---|---|---|---|
| Leading eigenvector | | | | | |
| Network 1 | C1 | ∗434 | 47.07% | 16.81 | ±17.91 |
| | C2 | 488 | 52.93% | 25.15 | ±19.49 |
| Network 2 | C1 | ∗503 | 51.43% | 15.44 | ±19.54 |
| | C2 | 475 | 48.57% | 26.37 | ±18.01 |
| Network 3 | C1 | ∗385 | 41.76% | 12.85 | ±20.24 |
| | C2 | 537 | 58.24% | 27.26 | ±17.84 |
| Walktrap | | | | | |
| Network 1 | C1 | ∗431 | 46.75% | 16.76 | ±17.92 |
| | C2 | 490 | 53.15% | 25.16 | ±19.48 |
| Network 2 | C1 | ∗372 | 38.04% | 12.98 | ±19.00 |
| | C2 | 311 | 31.80% | 23.11 | ±19.48 |
| | C3 | 294 | 30.06% | 28.15 | ±16.77 |
| Network 3 | C1 | ∗231 | 25.05% | 7.09 | ±19.60 |
| | C2 | 301 | 32.65% | 23.83 | ±17.22 |
| | C3 | 390 | 42.30% | 27.63 | ±18.48 |

## 5.3 Community Detection

Using the leading eigenvector (LE) algorithm in all three networks two communities with about the same number of nodes are detected. With walktrap in network 1 two communities and in network 2 and 3, three communities. The exact number of community members for algorithm is shown in table 5.1.

The first community (LE-C1, WT-C1) contains the queen and bees who are on average younger than the second community (LE-C2, WT-C3). For walktraps third community it is a middle-aged community (WT-C2). The age difference for network 1 is 8.4 days, for network 2 10.9 days, and for network 3 14.4 days on average for leading eigenvector communities.

The age distribution for each community and network is depicted in figure 5.3.

A two sample Kolmogorov–Smirnov test showed, that for leading eigenvector communities, the age distributions are significantly different ($p < 0.001$). For walktrap C1 with C2 and C3 are significantly different, but C2 and C3 are not that much significant. Exact $p$-values are shown in table 5.4

**Table 5.4: Kolmogorov-Smirnov test** $p$-values for leading eigenvector (LE) and walktrap (WT) for each network and its communities.

| | | LE p-value | WT p-value |
|---|---|---|---|
| Network 1 | C1, C2 | 5.1e-32 | 3.3e-31 |
| Network 2 | C1, C2 | 7.6e-38 | 2.3e-32 |
| | C1, C3 | | 1.3e-44 |
| | C2, C3 | | 6.6e-05 |
| Network 3 | C1, C2 | 1.4e-64 | 1.8e-65 |
| | C1, C3 | | 3.9e-93 |
| | C2, C3 | | 2.6e-05 |

### 5.3.1 Spatial Distribution of Communities

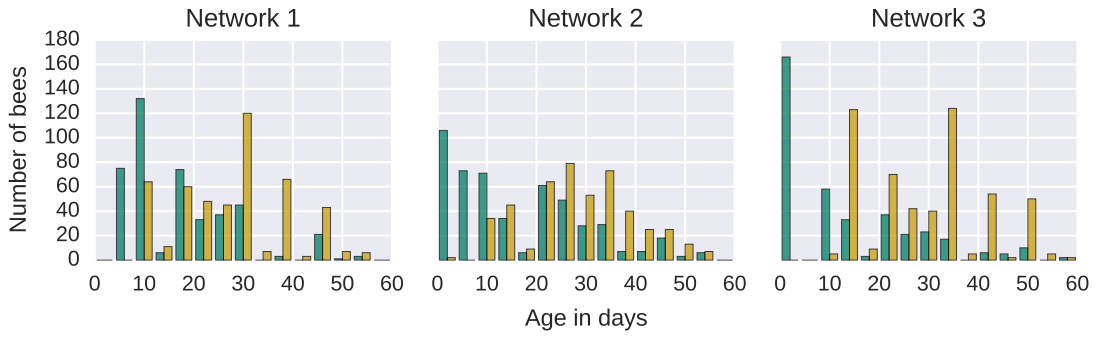
The two communities detected by leading eigenvector are located in two different regions of the honeycomb. The older community (*orange* in figure 5.4)) is in all three networks closer to the hive exit and the younger community (*green* in figure 5.4)) is situated in the comb center. Walktrap revealed the same two communities like leading eigenvector for all three networks, with the same spatial distribution. For network 2 and 3, a third community (*gray* in figure 5.5)) is located between the young and old community.

### 5.3.2 Community members over time

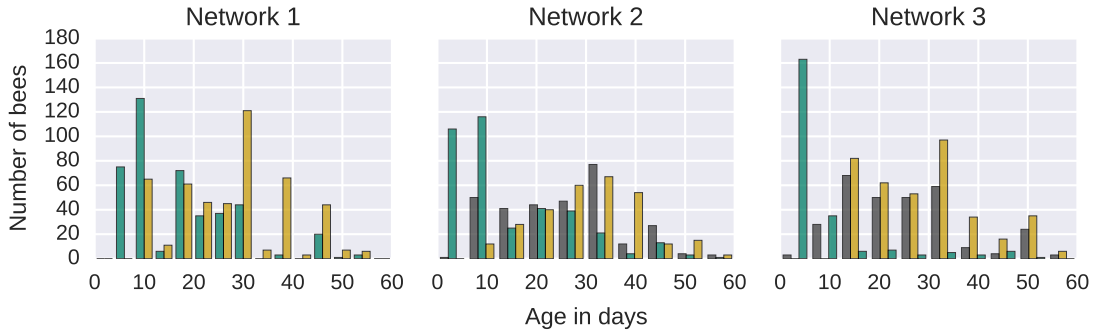
The match value between the two communities in successive time steps are calculated with formula (2.1) and presented in figure 5.6a for the resulting communities using the leading eigenvector algorithm and figure 5.6b for the communities detected with walktrap.

## 5.4 Summary

TODO

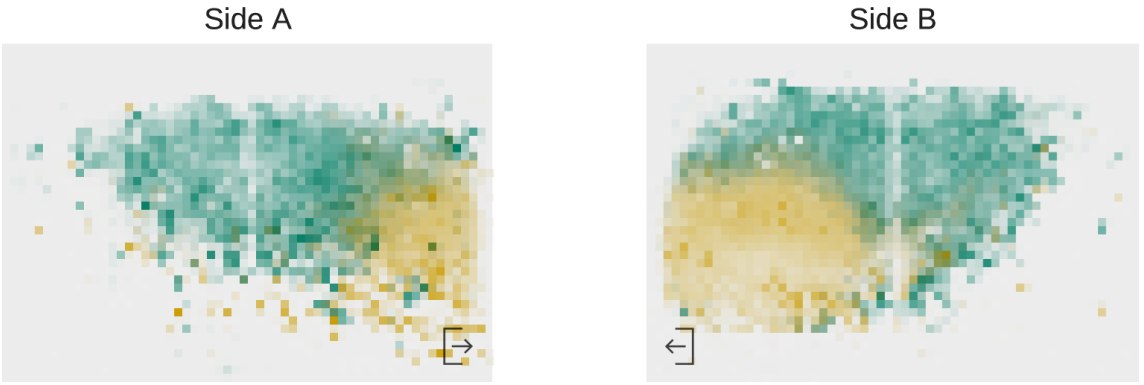



**(a)** Leading eigenvector

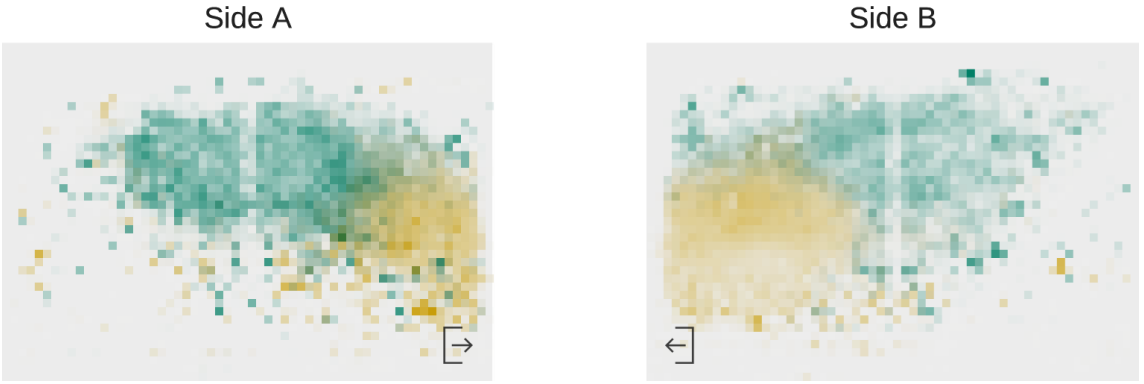



**(b)** Walktrap

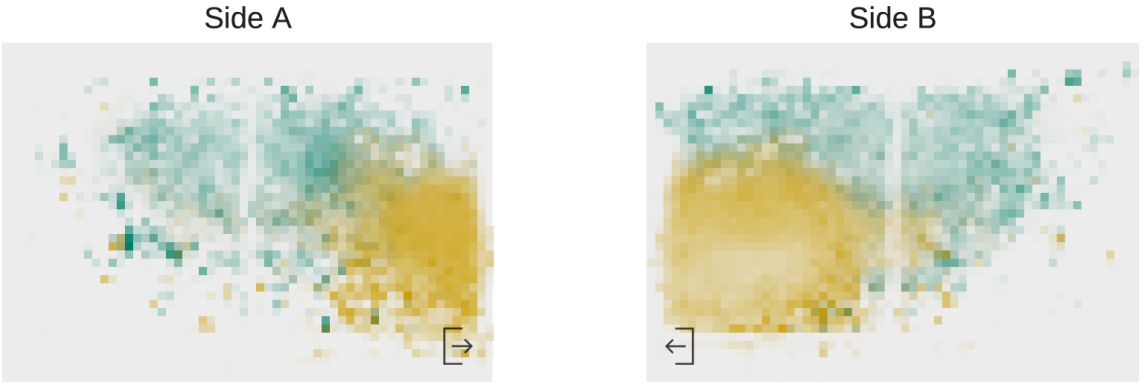
**Figure 5.3: Age distribution for each community and network** The *green* bar is the community containing the queen. The queens age is not included in the statistic. The *orange* bars coresspond to the second community, containing older bees. The *gray* bars is a third community only revealed by walktrap and contains middle-aged bees.




**(a)** Network 1

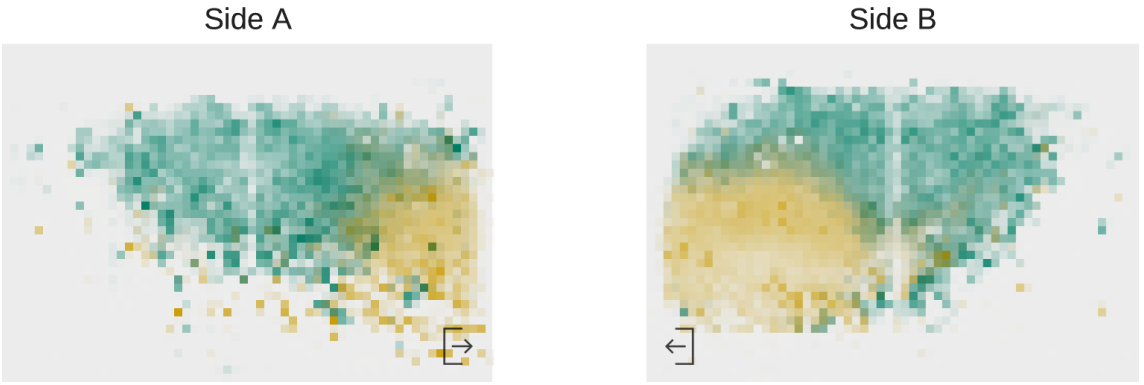



**(b)** Network 2

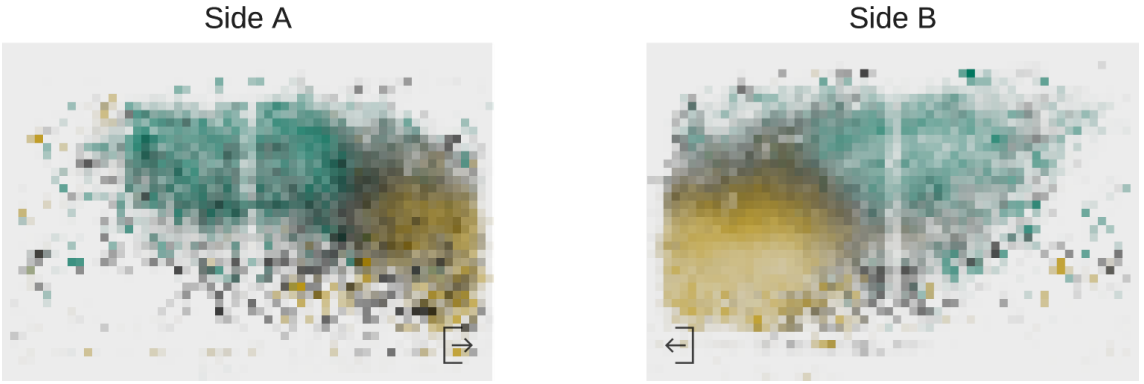



**(c)** Network 3

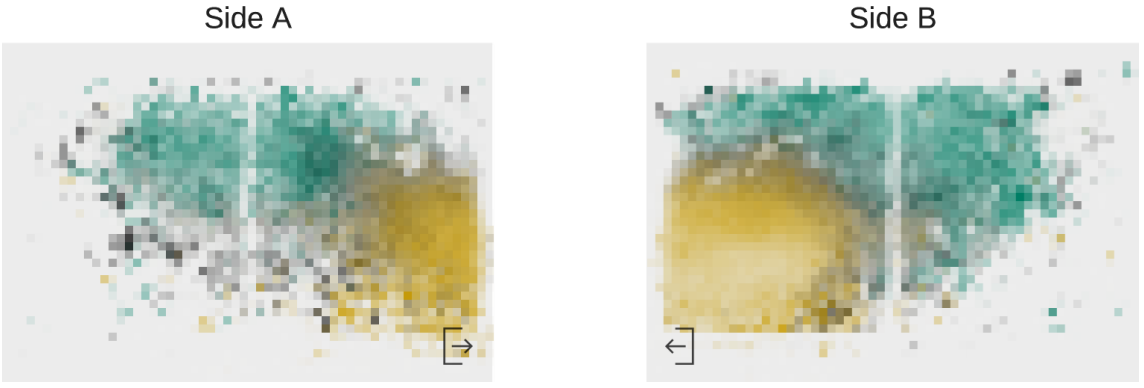
**Figure 5.4: Communities per network - leading eigenvector** The *green* colour represents the younger community, containing the queen. The *orange* color represents the older community. The hive exit on side A is on the bottom right and on side B on the bottom left. The data is aggredated for the complete timeframe of ten hours.




**(a)** Network 1

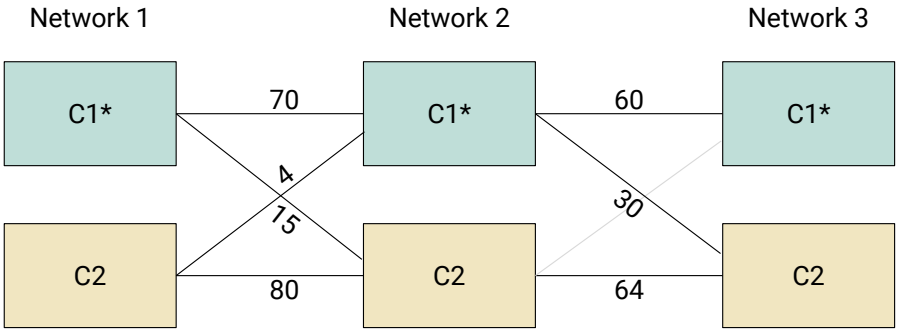



**(b)** Network 2

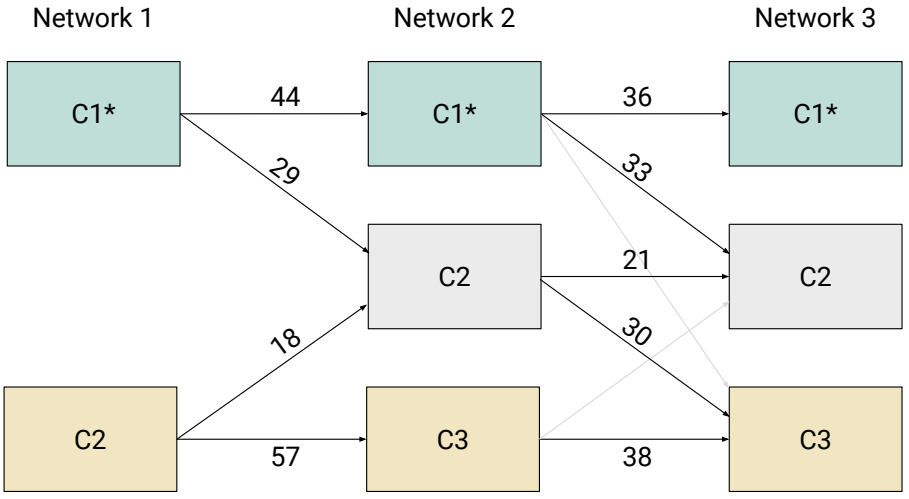



**(c)** Network 3

**Figure 5.5: Communities per network - walktrap** The *green* colour represents the younger community, containing the queen. The *orange* color represents the older community. The *gray* represents the middle-age community. The hive exit on side A is on the bottom right and on side B on the bottom left. The data is aggredated for the complete timeframe of ten hours.




**(a)** Leading eigenvector communities




**(b)** Walktrap communities

**Figure 5.6: Community matching** The numbers indicate the match values in percent. The community marked with * contains the queen. *Green* represents the younger community *orange* the older community, and gray the middle-aged community. The light gray arrow represents match values below three percent.

# Chapter 6

# Conclusion

## 6.1 Discussion

http://mathoverflow.net/questions/126704/the-probability-distribution-for-vertex-

## 6.2 Limitation

contact duration and contact frequency
distances is are hard to measure, because hive is very small and dense (is spatial proximity a good proxy in this case)

nur weil man mehr daten hat, muss es nicht unbedingt besser werden
aber mehr daten (aufbereiteter guter datensatz), evtl. fragen stellen, an die man vorher noch nicht gedacht hat.

## 6.3 Summary

## 6.4 Future Work

Measuring the robustness of network community structure using assortativity [25]

# Bibliography


[1] Thomas Aynaud et al. “Communities in evolving networks: definitions, detection, and analysis techniques”. In: *Dynamics On and Of Complex Networks, Volume 2*. Springer, 2013, pp. 159–200.

[2] Albert-László Barabási. *Network science*. Cambridge University Press, 2016.

[3] David Baracchi and Alessandro Cini. “A Socio-Spatial Combined Approach Confirms a Highly Compartmentalised Structure in Honeybees”. In: *Ethology* 120.12 (2014), pp. 1167–1176.

[4] Alain Barrat et al. “The architecture of complex weighted networks”. In: *Proceedings of the National Academy of Sciences of the United States of America* 101.11 (2004), pp. 3747–3752.

[5] Benjamin Blonder and Anna Dornhaus. “Time-ordered networks reveal limitations to information flow in ant colonies”. In: *PloS one* 6.5 (2011), e20298.

[6] Piotr Bródka, Stanisław Saganowski, and Przemysław Kazienko. “Community Evolution”. In: *Encyclopedia of Social Network Analysis and Mining* (2014), pp. 220–232.

[7] Daniel Charbonneau, Benjamin Blonder, and Anna Dornhaus. “Social insects: a model system for network dynamics”. In: *Temporal Networks*. Springer, 2013, pp. 217–244.

[8] James D Crall et al. “BEEtag: a low-cost, image-based tracking system for the study of animal behavior and locomotion”. In: *PloS one* 10.9 (2015), e0136487.

[9] Darren P Croft, Richard James, and Jens Krause. *Exploring animal social networks*. Princeton University Press, 2008.

[10] Damien R Farine and Hal Whitehead. “Constructing, conducting and interpreting animal social network analysis”. In: *Journal of Animal Ecology* 84.5 (2015), pp. 1144–1163.

[11] Mark Fiala. “Comparing artag and artoolkit plus fiducial marker systems”. In: *Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on*. IEEE. 2005, 6–pp.

[12] Santo Fortunato. “Community detection in graphs”. In: *Physics reports* 486.3 (2010), pp. 75–174.

[13] Steve Harenberg et al. “Community detection in large-scale networks: a survey and empirical evaluation”. In: *Wiley Interdisciplinary Reviews: Computational Statistics* 6.6 (2014), pp. 426–439.

[14] John Hopcroft et al. “Tracking evolving communities in large linked networks”. In: *Proceedings of the National Academy of Sciences* 101.suppl 1 (2004), pp. 5249–5253.

[15] Raphaël Jeanson. “Long-term dynamics in proximity networks in ants”. In: *Animal Behaviour* 83.4 (2012), pp. 915–923.

[16] Jens Krause et al. *Animal social networks*. Oxford University Press, USA, 2014.

[17] Danielle P Mersch, Alessandro Crespi, and Laurent Keller. “Tracking individuals shows spatial fidelity is a key regulator of ant social organization”. In: *Science* 340.6136 (2013), pp. 1090–1093.

[18] Dhruba Naug. “Structure of the social network and its influence on transmission dynamics in a honeybee colony”. In: *Behavioral Ecology and Sociobiology* 62.11 (2008), pp. 1719–1725.

[19] M. Newman. *Networks: An Introduction*. OUP Oxford, 2010.

[20] Mark EJ Newman. “Finding community structure in networks using the eigenvectors of matrices”. In: *Physical review E* 74.3 (2006), p. 036104.

[21] Michael C Otterstatter and James D Thomson. “Contact networks and transmission of an intestinal pathogen in bumble bee (Bombus impatiens) colonies”. In: *Oecologia* 154.2 (2007), pp. 411–421.

[22] Gergely Palla et al. “Uncovering the overlapping community structure of complex networks in nature and society”. In: *Nature* 435.7043 (2005), pp. 814–818.

[23] Pascal Pons and Matthieu Latapy. “Computing communities in large networks using random walks”. In: *International Symposium on Computer and Information Sciences*. Springer. 2005, pp. 284–293.

[24] Lauren E Quevillon et al. “Social, spatial, and temporal organization in a complex insect society”. In: *Scientific reports* 5 (2015).

[25] Daizaburo Shizuka and Damien R Farine. “Measuring the robustness of network community structure using assortativity”. In: *Animal behaviour* 112 (2016), pp. 237–246.

[26] Fernando Wario et al. “Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees”. In: (2015).

[27] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*. Vol. 8. Cambridge university press, 1994.

[28] Tina Wey et al. “Social network analysis of animal behaviour: a promising tool for the study of sociality”. In: *Animal behaviour* 75.2 (2008), pp. 333–344.

[29] Zhao Yang, René Algesheimer, and Claudio J Tessone. “A Comparative Analysis of Community Detection Algorithms on Artificial Networks”. In: *Scientific Reports* 6 (2016).

# List of Figures

# List of Tables

# Appendix A

# Appendix Stuff