Detecting Honey Bee Trophallaxis in Trajectory Data

Andreas Berg

January 7, 2018

Bachelor thesis for the studies of Computer Science at Freie Universität Berlin
Matriculation number: 4279703
a.berg.contact@gmail.com
Supervisor: Prof. Dr. Tim Landgraf, Freie Universität Berlin
Second reader: Prof. Dr. Dr. (h.c.) Raúl Rojas, Freie Universität Berlin

Social networks of honey bees can be analyzed to learn more about various topics like the characterization of the swarm, work tasks by age groups, transmission processes or even disease dynamics. Social networks can be build through contact events. One important contact event of such networks can be transmission of fluids - trophallaxis. Finding trophallactic contacts manually is time consuming. Automatic detection or at least an assisting system would be useful to be able to classify large data sets. This work attempts to develop a system to detect trophallactic contacts by filtering spatio-temporal data. The results are then discussed.

Eidesstattliche Erklärung

Ich versichere hiermit, diese Arbeit selbstständig und ausschließlich unter der Benutzung der angegebenen Hilfsmittel und Quellen verfasst zu haben. Außerdem wurde diese Arbeit weder in gleicher noch in ähnlicher Form veröffentlicht und auch keiner anderen Prüfungskommission vorgelegt.

January 7, 2018

Andreas Berg

Contents

1 Introduction

Trophallaxis of the honey bee is a topic that touches many different scientific areas. Its biological properties can be researched and help to gain understanding about how a bee colony functions in greater detail [4]. Others use trophallaxis as contact interaction for building social networks. These can be studied with respect to transmission dynamics to gain a better understanding of social transmission processess and be even related to disease dynamics like in the work of Naug [9]. In that work Naug creates a 1 hour long video per experimental treatment and analyzes it manually. This manual analysis' nature is remarked as "painstaking". The manual detection of trophallaxis is a tedious task and much time and effort has to be spend to generate a relatively bigger data sample.

If the detection of trophallaxis could be automated, researching trophallaxis related topics on a larger scale would become easier.

In this work my task is to attempt the automatic detection of trophallaxis. I take this task on by filtering spatio-temporal data obtained from the BeesBook experiment conducted in 2016 (see section 1.3). Even if this method alone turns out to not be enough to automatically detect trophallaxis, this step still has to be taken. Because of the sheer amount of data it would be a necessary preprocessor for a possible computer vision system for automatic detection of trophallaxis.

1.1 Related Work

Many publications for tracking systems of animals can be found. Most related to my thesis I want to mention a few.

In [11] the authors use a tag based tracking system to track each ant of a colony individually. These tracking data is used to automatically detect physical pairwise ant-to-ant contacts based on the method of [7]. Such physical contacts include trophallaxis, but also other interactions like antennation.

In the BeesBook project research is being conducted especially on the interactions and behavior of honey bees: In [14] a tag based tracking system of honey bees is described. It has already been used to automatically detect and decode waggle dances on high frequency data of 100 Hz [15]. [13] presents an approach for the automatic detection of waggle dancers and followers on low frequency data of 3 Hz.

[14] mentions the detection of trophallaxis briefly. There four consecutive non moving head-to-head configurations of honey bees were reported. Other than that, to my knowledge no automatic detection system of honey bee trophallaxis has been published.

1.2 Trophallaxis

In [4] trophallaxis is described in great detail. It is defined as the exchange of solution or liquid from one bee to another bee. Examples of transferred solution or liquid would be nectar and water. Trophallactic interactions are not exclusively observed in the context of the waggle dance. They occur between all adult bees of a colony. Workers participate as donors and recipients. Drones and the queen participate as recipients. Even though foragers donate food after their arrival they can receive multiple times before starting a foraging flight. The duration of a trophallaxis with a worker as recipient is mentioned as ranging from a few seconds up to some minutes. [6] report the duration of trophallaxis with a mean value ranging from 8 to 14 seconds (for a somewhat artificial scenario though - one bee was fed until satiation while the other bee was being maintained unfed). The minimal duration of trophallaxis is of great interest for this thesis. [9] counts each trophallactic contact with a duration of 5s or higher as food transfer.

[5] analyses honey bee trophallaxis with thermographic data as well as video data. One result of the work is, that many trophallactic interactions did not result in the effective transfer of food. Which is concluded by the lack of heat increase in the proboscis of the begging bee. This could mean, that food transfer is not needed to relay chemosensory information about the quality of a food source. It is also possible, that trophallaxis is more than a way to taste the quality of a food source or exchange food.

1.3 BeesBook Experiment

This work uses data from the BeesBook experiment conducted in 2016. Detailed descriptions are found in the works of Bönisch [3] and Schlegel [12]. The experiments of 2014 and 2015 are described in great detail by [14] most of the information about the experiments setup is still valid for the one conducted in 20016. I will give a brief description.

The hive of a colony of honey bees is recorded for a time period of 63 days - 19th of July 2016 until 19th of September 2016 (see figure 1.1). The bees are tagged with circular markers that carry 12 bits to identify the bee and 2 bits to show the bee's orientation (see figure 1.2). The maximum number of bees present in the hive was 1600. Overall 31931 were tagged. Four cameras with a picture resolution of 4000 x 3000 were used. Two on each side of the hive. They recorded 3 images per second. This visual data was processed through the BeesBook pipeline [14] to generate data which is easier to analyze in form of bee detections and tracks, which are described in section 2.1.

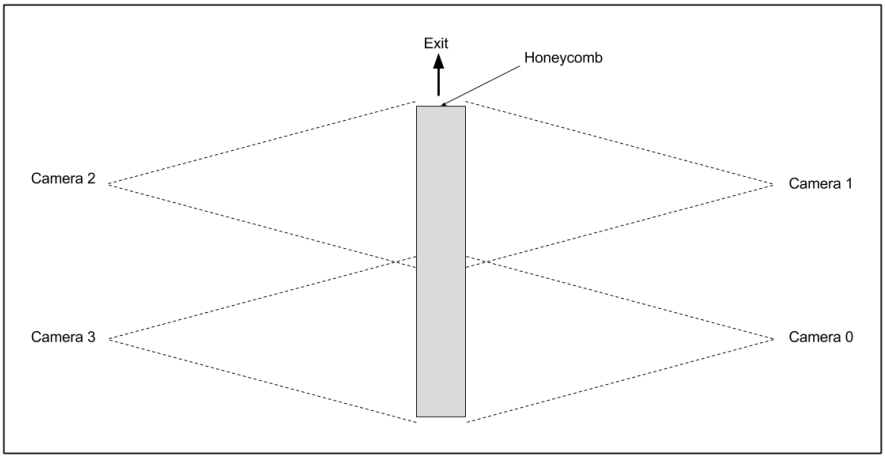

Figure 1.1: Shows the simplified setup of the cameras and the honeycomb for the BeesBook experiment conducted in 2016.

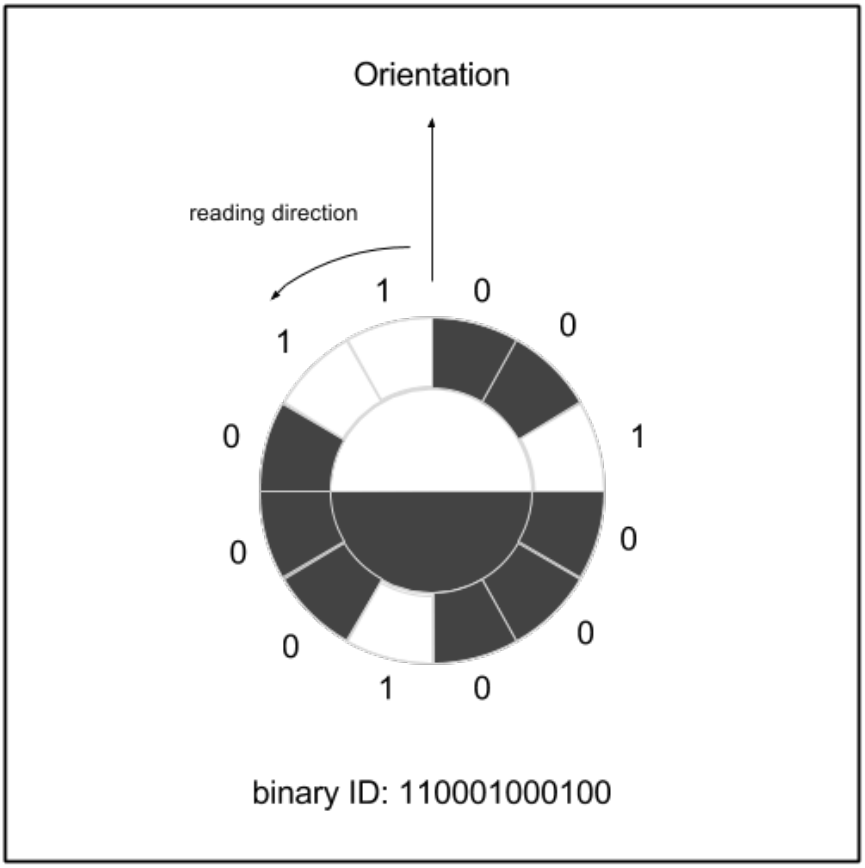

Figure 1.2: Shows the tag design with 12 bits to identify the bee and 2 bits to carry information about the bee's orientation.

2 Implementation

The implementations goal is to extract trophallaxis events from the spatio-temporal data. To look at the data of two individual bees for a specific duration, one must have means to identify each bee. The implementation deals with the problem of tags not being clearly identified per image through the use of tracks. The necessary assumption is made that one track corresponds to one specific bee. To which extend this assumption might be reasonable is discussed in detail in the works of Mischek [8] and later with further improvements to the tracking in the works of Bönisch [3].

2.1 Description of the Data

The implementation filters data of pairs of bees with respect to the time. To achieve this, multiple data structures are accessed and used. The most important ones are described in this section.

2.1.1 Detection

A *detection* is a data structure with the goal to save the observation of a bee. This is a representation of a detection written in pseudocode. (See the documentation of the bb_tracking module: [2])

```
Detection struct {
    id
    timestamp
    x
    y
    orientation
    beeId
    meta
}
```

An example detection:

```
Detection(
    id='f116787592813686709 20d56c0',
    timestamp=1468939049.378782,
    x=1686,
```

```
        y=2025,
        orientation=-1.6560965776443481,
        beeId=[

            0.06666666666666667, 0.0196078431372549, 0.11764705882352941,
            0.00784313725490196, 0.9882352941176471, 0.00392156862745098, 1.0,
            0.011764705882352941, 1.0, 0.00784313725490196, 0.9803921568627451,
            0.8784313725490196
        ],
        meta={
            'xRotation': -0.0849771499633789,
            'yRotation': -0.7539273500442505,
            'camId': 0,
            'zRotation': -1.6560965776443481
        }
    )
```

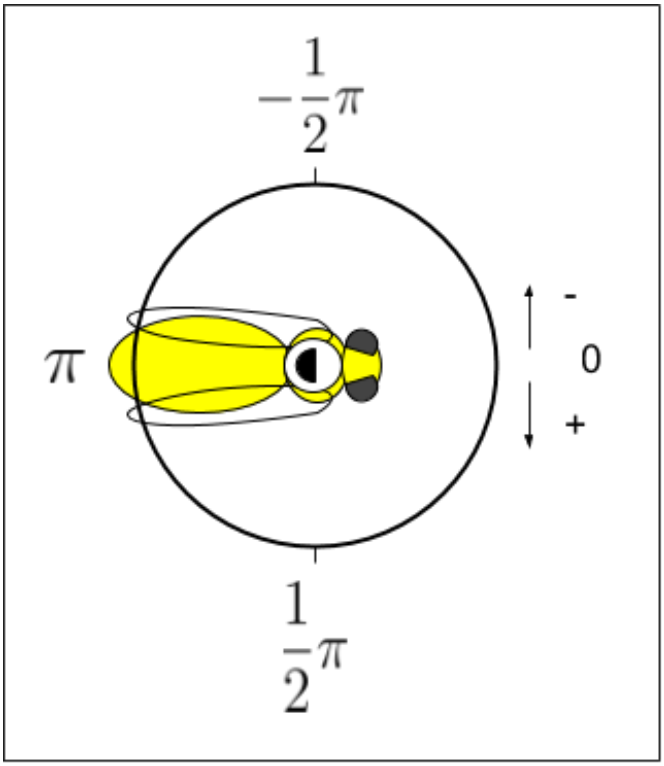

Figure 2.1: The orientation or z_rotation of a bee tag.

The id is an identifier for the detection. 'f' stands for frame and is followed by a frame id, which consists of numbers only. 'd' stands for detection and is followed by a number

that indicates the number this detection has for all detections found for this frame. 'c' stands for camera and represents the id of the camera that recorded this frame.

The timestamp represents the time of the detection in the format of Unix timestamps - which count the seconds from 1st of January 1970.

x and y represent the bee's location on the frame in pixels. x starts with the value 0 on the left side and increases to the right to a maximum of 3999. y starts with the value 0 on the top and increases to the bottom to a maximum of 2999.

The orientation represents the z_rotation of the tag aligned to the hive. See figure 2.1 for a visualization. The white half of the tags inner circle marks the direction of the z_rotation.

The beeId is a list of 12 values in [0, 1]. Each value shows the probability of the tag area being white. The tags are counted clockwise starting with the one in north north east, if the tags orientation is north. (See [8] for more information.)

meta contains additional information. In this case the camId and zRotation repeat information already contained in the Detection. The x_rotation represents the tags roll to its own side. The y_rotation represents the tags pitch to the front or back. (See [3] for more information.) Because heavily rotated tags are likely to be decoded incorrectly, they are removed in the ellipse fitting phase of the BeesBook image processing pipeline [14]. This leads to gaps in a chain of detections.

2.1.2 Frame

A *frame* is a data structure corresponding to one image of a video of one of the four cameras. It contains all detections that were found for this image. Also it points to its data source and holds a timestamp. Detailed information about a Frame can be found in the bb_binary code [1]. Schlegel [12] gives more information about the frames in the BeesBook project.

2.1.3 Track

A *track* is a data structure that groups detections with the goal that these detections reference the same tag.

This is a representation of a track written in pseudocode. (See the documentation of the bb_tracking module: [2])

```
Track struct {

    id
    ids
    timestamps
    meta

}
```

An example track:

```
Track(
    id='46f06484-b77e-4895-94ad-205a37c4aa9c',
    ids=[

        'f116787592813686709 20d56c0', 'f12386956009174622988d58c0',
        'f147054307190978556 87d57c0', 'f12369730620240256850d57c0',
        'f1037817020949088790 4d56c0'
    ],
    timestamps=[

        1468939049.378782, 1468939049.711209, 1468939050.043043,
        1468939050.375627, 1468939050.707758
    ],
    meta={
        'detections': [...],
        'median_id': (

            5, array([ 0.65882353, 0.90588235, 0.02352941, 0. ,
            1. , 0. , 0.01960784, 0.00392157, 0.99607843, 0. , 1. , 1. ])
        )
    }
)
```

The id is an identifier for the track.

ids is a list of the detections grouped by this track ordered by the values of their timestamps ascending.

timestamps is a list of the timestamps of these detections ordered ascending.

meta contains additional information. In this example it contains all the individual detection data and a calculated median_id which is a tuple of a number an a list. The number indicates how many detections were used to calculate the median values. The list is similar to the beeId except that each value here is the median of the corresponding beeId values of the detections of this track. The concept of how a track is build from detections is described in [14] page 6.

2.1.4 Detection Pairs

This is a representation of the data structure *detection_pairs* written in pseudocode.

```
DetectionPairs struct {
    [](
```

```
            detection_a,
            detection_b,
            timestamp,
        )
    }
```

The data structure *detection_pairs* is a list of triple (detection_a, detection_b, timestamp). Only two tracks a and b are used to create such a list. detection_a origins from track a and detection_b origins from track b. Both detections within one triple have the same timestamp, which is also stored as third value of the triple. The triple within the list are ordered by ascending timestamps. detection_pairs are created as part of the implementation of this thesis. The goal of detection_pairs is to connect the data of two tracks in a domain in which they are comparable - the mutual time. This enables filtering on the data.

2.1.5 Event

An *event* is a data structure that stores the one of the results of filtering a track combination for possible interactions. It is created as part of the implementation of this thesis. This is a representation of the data structure event written in pseudocode.

```
    Event struct {
        event_id
        track_id_combination
        []event_part struct {
            event_part_nr
            frame_id
            cam_id
            center_coordinates
            bee_a_coordinates
            bee_b_coordinates
            timestamp
            bee_a_detection_id
            bee_b_detection_id
        }
    }
```

event_id is an identifier for an event.

track_id_combination references the ids of the two tracks that were used to extract this event.

The list of event_parts stores data corresponding to the event ordered by ascending timestamps with the goal to visualize the event. Each event_part corresponds to one

triple in detection_pairs. But not each triple in detection_pairs has to correspond to an event_part.

event_part_nr starts with 0 per new event and increases by one per part.

frame_id references the frame in which the detections of this event_part are found.

cam_id references the id of the camera that created the image and led to a frame.

center_coordinates stores the x- and y-coordinates in the middle between both bee tags. center_coordinates of event_part_nr 0 is used to set the middle point of the video that is created from the event.

bee_a_coordinates stores the x- and y-coordinates for the tag of the bee corresponding to track a from the used track combination (a, b). They are used to mark the tag in the corresponding video.

bee_b_coordinates accordingly stores the x- and y-coordinates of the tag of the other bee corresponding to track_b.

timestamp stores the mutual timestamp of the two detections.

bee_a_detection_id and bee_b_detection_id reference the corresponding detection ids. They might be used to extract possibly omitted information or analyze the event further.

It is notable, that the filtering of one track combination might result in multiple events if the tracks in this combination are long enough.

2.2 Description of the Filter

The implementation extracts events from a list of tracks. Certain parameters are available to set up this filtering. These are the most relevant ones:

distance_min_pixels the minimum amount of pixels between two tags

distance_max_pixels the maximum amount of pixels between two tags

relative_rotation_scalar the minimum of a certain assessment of the relative orientation of one tag to another

event_time_seconds the minimum duration in seconds an event has to have

time_difference_max_seconds the maximum duration of a chain of gaps in a list of detection pairs (see section 2.1.4)

These parameters are used to execute several tests on a *track combination*. A track combination is a tuple of two tracks with the order being of no importance: (A, B) being the same as (B, A). If a track combination fails any of these tests, the combination is disregarded. If a track combination succeeds all these tests, one or more events are getting extracted from that track combination. The implementation will repeat this process with all distinct track combinations it can build from the given list of tracks.

This section describes the filter by describing the most relevant parameters and the corresponding tests.

2.2.1 Time

The first test checks wether or not the two tracks hold mutual timestamps. If they do, the duration d of the mutual timestamps must be at least the set minimum duration for an event. $d := t_n - t_0$ and $d \geq event\_time\_seconds$ with t_0 being the first mutual timestamp and t_n being the last. (See Figure 2.2.)

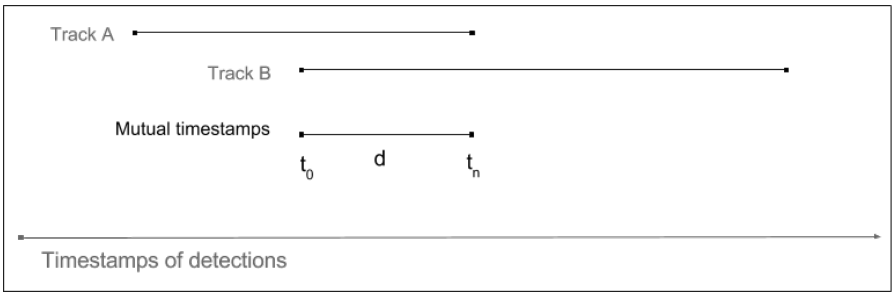

Figure 2.2: Two tracks with mutual timestamps, t_0 being the first and t_n being the last mutual timestamp

2.2.2 Distance

The next test checks wether or not the detection pairs of the track combination are within an interval I of the allowed euclidian distance $I := [distance\_min\_pixels,\ distance\_max\_pixels]$ for a duration of at least *event_time_seconds*. Figure 2.3 shows a simple example. Tag B's distance to A is below the *distance_min_pixels*. D's distance to A is above *distance_max_pixels*. Only C's distance to A is above or equal to *distance_min_pixels* and below or equal to *distance_max_pixels*. If C's distance to A stays within I for at least *event_time_seconds*, the test is passed.

For the first filtering the *distance_max_pixels* was chosen the following way:

- Some frame plots of bees in the hive were loaded.
- Bee pairs were picked by human eye when thinking that these could be able to perform trophallaxis from the distance they were standing in.
  - Some pairs were picked with an extra high distance.
- Straight lines were drawn between the middle points of the tags per bee pair.
- The pixel distances for all these lines were calculated.
- The maximum was chosen.
- The maximum was increased by approximately 10 %. This value was chosen as parameter *distance_max_pixels*.

In section 3.2 evidence is presented, showing that this value is too low to enable the filtering to catch almost all trophallaxis events. I underrated the possible length of the proboscis of the bees.

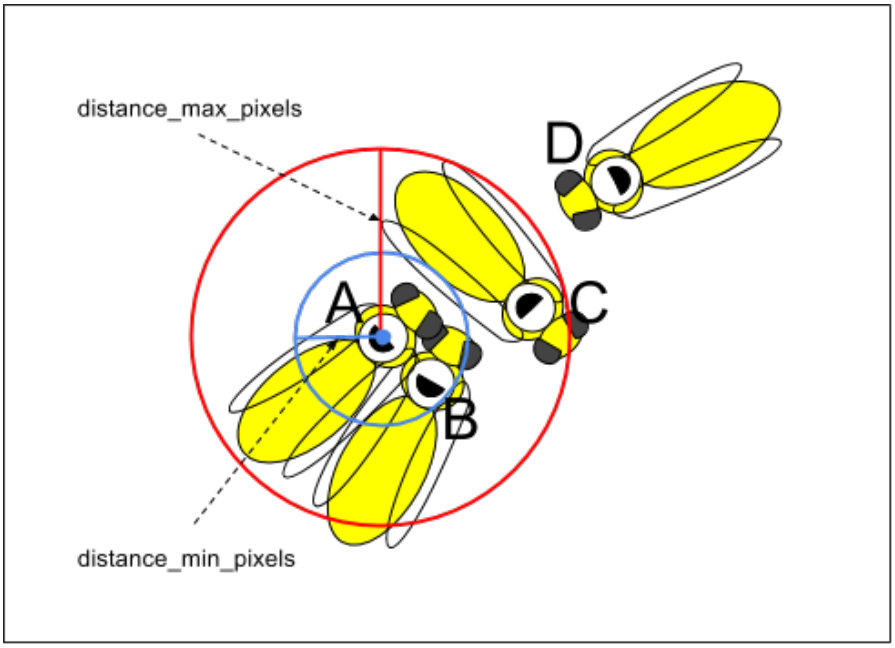

Figure 2.3: The tags A, B, C, D of bees are shown. The middle points of B, C and D are within certain euclidian distances to the middle point of A. Only C is found within the allowed distance interval.

2.2.3 Relative Rotation

The bees have to face each other in a certain range of angles to be able to perform trophallaxis. See Figure 2.3 again. While C's distance to A may enable trophallaxis its z-rotation makes it physically impossible for the C tagged bee to reach the A tagged bee with its proboscis.

Let $\vec{a}$ be the local vector of tag A, let $\vec{b}$ be the local vector of tag B, let z_rot_a be the z-rotation of tag A (see section 2.1.1 for its interpretation). Then I define the relative rotation of A to B as:

$$rel\_rot_{a,b}(a,\, z\_rot_a,\, b) := \frac{\vec{b}-\vec{a}}{\left|\vec{b}-\vec{a}\right|} \cdot \frac{\begin{pmatrix}\cos(z\_rot_a)\\ \sin(z\_rot_a)\end{pmatrix}}{\left|\begin{pmatrix}\cos(z\_rot_a)\\ \sin(z\_rot_a)\end{pmatrix}\right|}$$

This is the dot product of the normalized vector from A to B and the normalized vector with the direction of z_rot_a . The results can be in this range: $-1 \leq rel\_rot_{a,b} \leq 1$. A value near -1 means that tag A's orientation faces in the opposite direction of where tag B is located. Accordingly a value near 1 means that A's orientation faces in the direction of B. A value of 0 means that A's orientation faces in a 90° angle of the direction of where B is located. In this manner all possible relative rotations can be represented by values. This is the reason why the formula for the relative rotation is used here.

See Figure 2.4 for an example. Tag A directly faces tag B, while tag B faces 45° away from A. This results in $rel\_rot_{a,b} = 1$ and $rel\_rot_{b,a} \approx 0.707$.

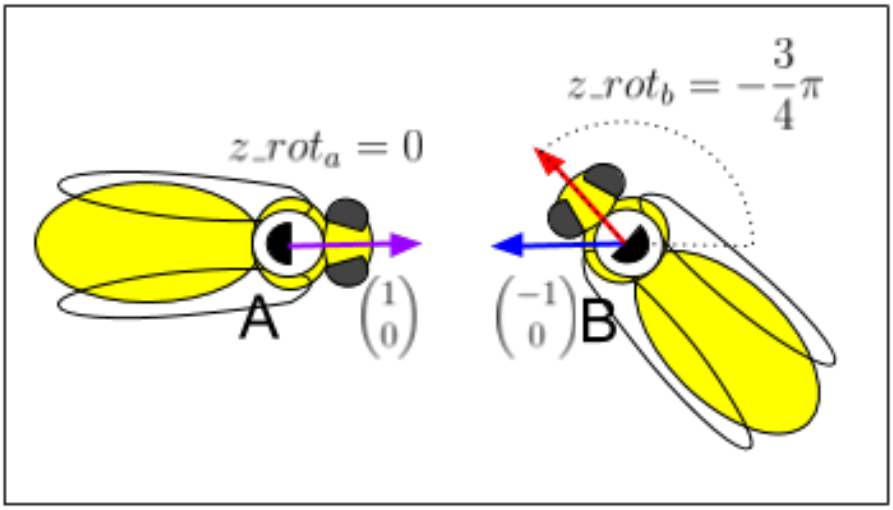

Figure 2.4: Example for visualizing the normalized vectors used to calculate the relative rotations. $rel\_rot_{a,b} = 1$ and $rel\_rot_{b,a} \approx 0.707$

This test is passed iff $min(rel\_rot_{a,b}, rel\_rot_{b,a}) \geq relative\_rotation\_scalar$ for at least *event_time_seconds*.

To choose this parameter an appraisal was done after looking at some trophallaxis situations. The bees are able to turn their heads and perform trophallaxis even though the tags face around 45° away from each other, which results in a $rel\_rot \approx 0.707$. (Higher angles might also be possible.) See figure 2.5 for an example. This fact combined with the possibility of detections having slightly wrong rotations (for example because of glare spots on the tag) suggests a value even lower than 0.7 for this parameter.

2.2.4 Duration

Each test so far also checked for a duration of at least *event_time_seconds*. But tracks might miss some detections and fail to have a complete chain of detections with respect to timestamps increasing by 0.33 seconds for each following detection in the chain. If one track misses a detection for a timestamp, the detection pair for this can not be build. In this case it can not be guaranteed, that any test passes. So a defensive approach could be to let every test fail and see if the duration until that point is at least *event_time_seconds*. Another approach is to allow gaps and assume that the bees do not move much in the unobserved time, which would mean, that if they pass the tests before the gap and after the gap, we assume they do so as well in the time of the gap. This allows to catch more trophallaxis events, which would otherwise be not found due to detection gaps in the tracks. How much time may be missing between two detection pairs is determined by the choosing of parameter *time_difference_max_seconds*.

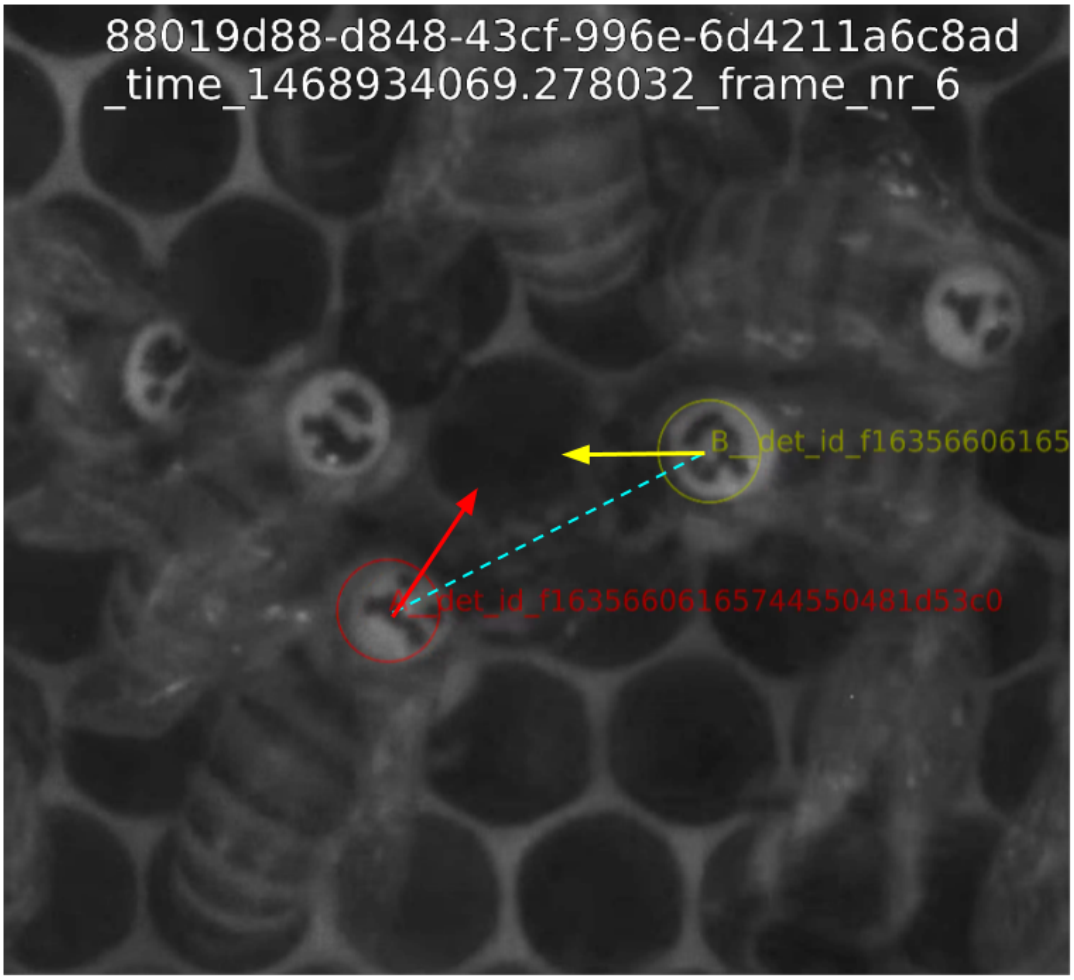

Figure 2.5: Shows an example of trophallaxis. The turquoise dashed line connects the middle points of the tags. The red vector represents the assumed orientation of tag A and the yellow vector represents the assumed orientation of tag B. Both tags are assumed face around 30° away from the other bee's tags. The proboscis indicating the trophallaxis is better visible in the video, than in a single picture.

If *time_difference_max_seconds* is set to 0 or any value smaller than 0.33 it would result in the described defensive approach. If *time_difference_max_seconds* is set to a value greater than 0.33, for each integer factor of 0.33 one frame can be omitted if the detection pair with that timestamp is missing.

2.3 Resulting Data

The result of the filtering of a list of tracks is a list of events. (Events are described in section 2.1.5.) Each event corresponds to a video, for which the event_id is used as filename.

In the first calculation a list of 11917 tracks over a two hour period were used starting at 2016-07-19T12:37:31+00:00 and ending at 2016-07-19T14:37:31+00:00 (format ISO 8601). All tracks are created from the detections within the frames of camera id 0. The honeycomb area covered by camera 0 is not connected to the exit of the hive. ([12] p. 13)

The parameters were chosen as following:

```
distance_min_pixels = 0,
distance_max_pixels = 190,
relative_rotation_scalar = 0.6,
event_time_seconds = 2,
time_difference_max_seconds = 2
```

This resulted in 3060 events found. And accordingly 3060 videos created.

Figure 2.6 add reference shows a sequence of pictures from one event.

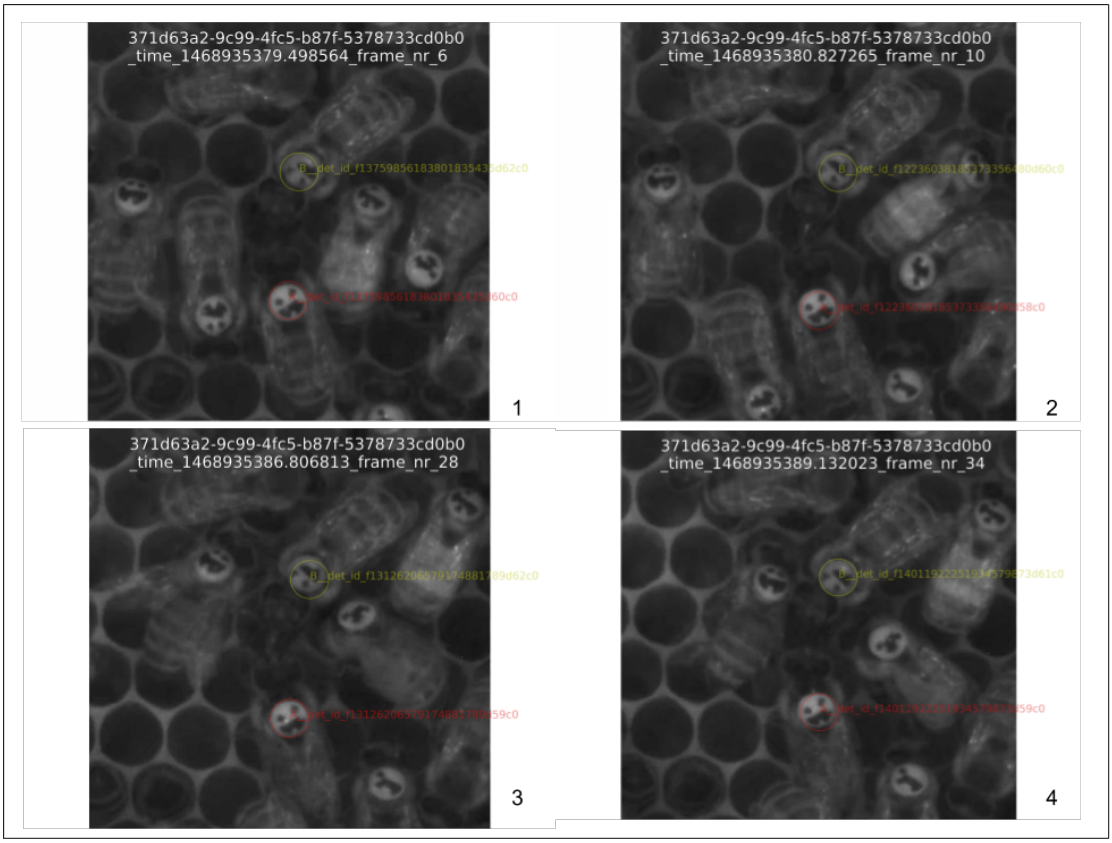

Figure 2.6: Four pictures that are part of the video corresponding to event 371d63a2-9c99-4fc5-b87f-5378733cd0b0. The frame numbers as well as the time stamps indicate that these are not consecutive pictures. The time differences are 1.329 s, 5.980 s and 2.325 s from picture 1 to 2, 2 to 3 and 3 to 4. Picture 1 does not show any visible proboscis. Bee B extends its proboscis in frame 7. In picture 2 the gloss point on the proboscis is visible. In picture 3 the whole proboscis of bee B is visible through its glossiness. Picture 4 shows the first frame after the trophallactic interaction with a duration of 8.970 s has ended. No proboscis between the bees is visible anymore. In the video the antennation can also be observed.

3 Evaluation and Discussion

This chapter presents the results of two calculation runs. After evaluating the first calculation run the parameters are discussed and adapted to generate a second set of ground truth data. This again is presented, evaluated and discussed. The data is used to train a classifier which is being evaluated and discussed. In the last two sections I draw conclusions and point out possible future work.

3.1 Ground Truth Data

After the first calculation run, I generated ground truth data for the first 1487 events, by watching the corresponding video for each event. The smallest timestamp found for these 1487 events is 2016-07-19T12:37:42+00:00. The biggest timestamp found for the events of the ground truth data is 2016-07-19T14:37:16+00:00. I did not go over all 3060 events to make more use with my time in a second or even third calculation run. Out of 1487 events, 86 were labeled as trophallactic contact. This is 5.8% of the events. Which would be the precision of the filtering. The recall can not be calculated, because it is unknown how many trophallactic contacts are really happening within the time frame.

The mean of the durations of all events is 4.18s. Figure 3.1 shows two histograms of the durations for the 86 trophallactic contact events. The durations of the trophallactic contacts are to be interpreted as minimum durations because they are measured by the time difference from the first frame with the proboscis visible until the last frame with the proboscis visible. This means, that the actual trophallactic contact might be up to 0.65s longer than observed, due to the time difference of 0.33s per frame. And it explains why a duration of an observed trophallactic contact might be 0. While an event has a minimum duration of two seconds and most events have a duration under ten seconds, almost half of the trophallactic contacts have a duration of under two seconds. The mean of the duration of the trophallactic contacts is 4.50s. Excluding the short trophallactic contacts under two seconds the mean jumps up to 7.98s. This would be more on par with the observations of [6].

The 1487 events consist of 19233 event_parts of which 1241 are event_parts labeled as trophallactic contact. To get a understanding of the observed relative rotations and distances, I am looking at all event_parts.

Figure 3.2 shows two histograms of the found relative rotations. The bulk of event_parts being labeled as trophallactic contact have a relative rotation in [0.95, 1] with the peak towards 0.99 to 1. This means that for the observed data the bees tended to *both* align themselves straight and exactly opposite of each other. Noteworthy is that only 2 of the 1241 event_parts (0.16%) show a relative rotation in the interval [0.6, 0.7].

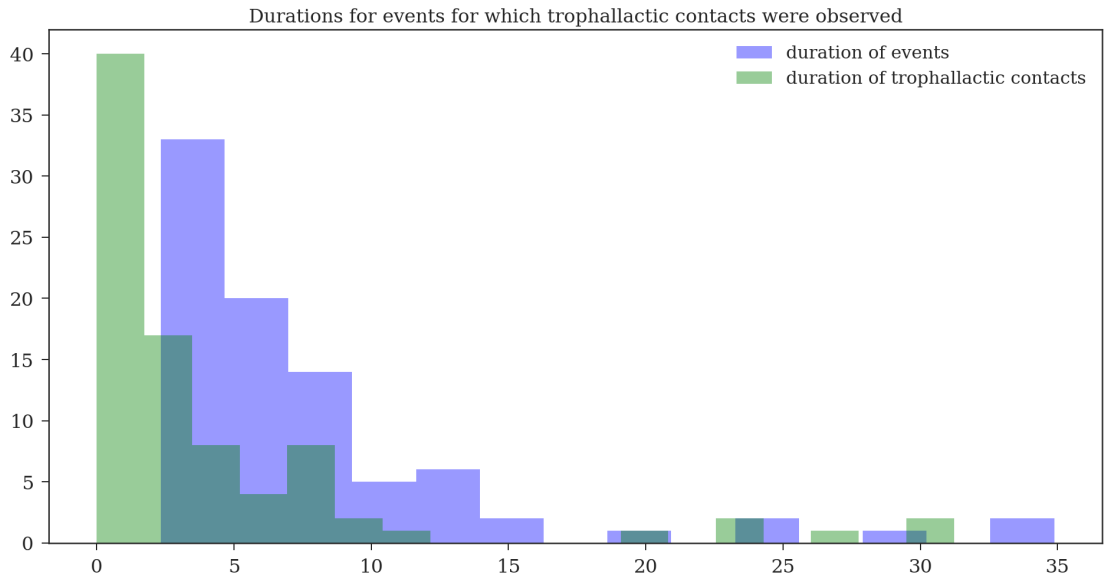

Figure 3.1: Shows two histograms of the durations for the 86 trophallactic contact events. Blue areas are build by the whole durations of the events. Green areas are build by only the part of the events, in which trophallactic contact was observed.

Figure 3.3 shows two histograms of the found distances. Most trophallactic contact event_parts show a distance in $I = [140, 190]$. The maximum of I with 190 pixels suggests that the parameter *distance_max_pixels* with its value being 190 was chosen to low to catch almost all trophallactic contacts. The minimum of I with 140 pixels and the fact that the non trophallactic contact parts show low distances under 125 pixels that are almost not at all represented in the trophallactic contacts suggest, that the parameter *distance_min_pixels* with its value being 0 can be raised quite a bit.

3.2 Adapting Parameters Manually under Assumptions

This section attempts to improve the choice of parameter values for the filtering. An understanding of how parameters can be chosen stricter without loosing too many trophallactic contacts has to be build. First distances are adapted through assuming a normal distribution. This is then used to generate another data set of ground truth data which is also evaluated.

I am adapting the distance parameters and do another run of calculations. The purpose is to find more trophallactic contacts absolutely and relative to the non trophallactic contacts found. For time management reasons only tracks starting within the first 45 minutes of my given tracks are used for this calculation run: from 2016-07-19T12:37:31 to 2016-07-19T13:23:00+00:00.

The parameters for this calculation run are:

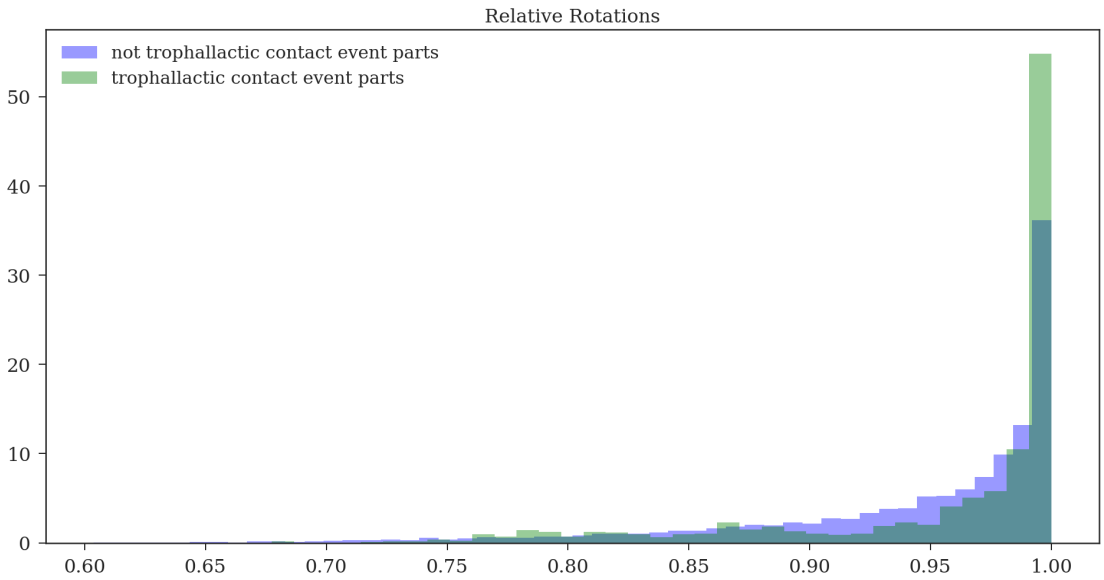

Figure 3.2: Shows two histograms of the found relative rotations for all event_parts. Both graphs are normalized, which means that the integrals under the graphs will integrate to 1. This is done just for visualization purposes: because of the high amount of event_parts labeled as not being trophallactic contacts the green graph would be not well visible. The blue graph shows the relative rotations of all event_parts which were not labeled as trophallactic contacts. The green graph shows the relative rotations of the event_parts labeled as trophallactic contact.

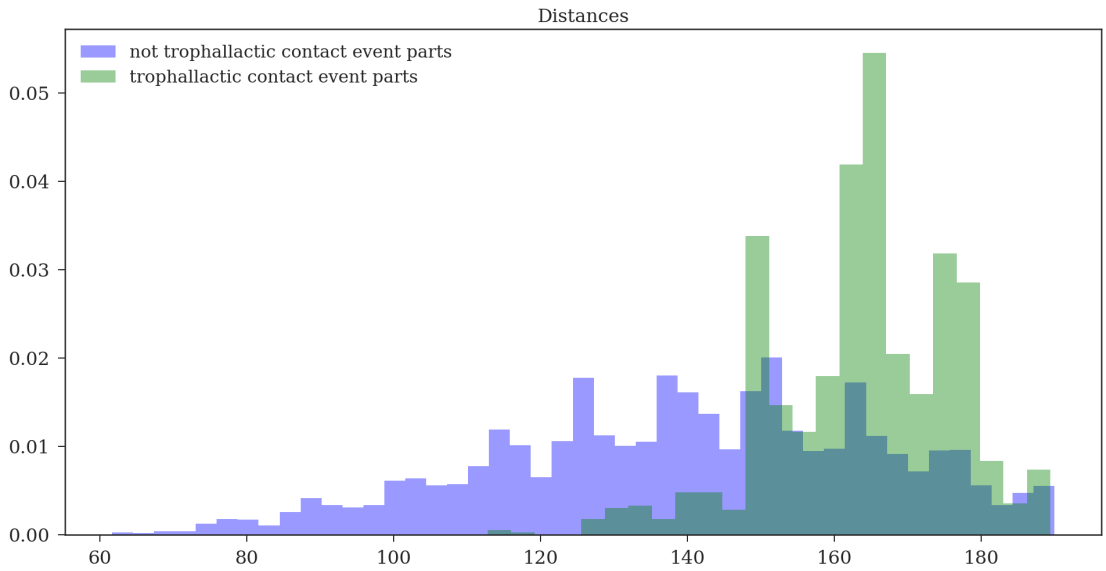

Figure 3.3: Shows two histograms of the found distances between the tags in pixels for all event_parts. Again both graphs are normalized for visualization purposes.

```
distance_min_pixels = 126,
distance_max_pixels = 201,
relative_rotation_scalar = 0.6,
event_time_seconds = 2,
time_difference_max_seconds = 2
```

relative_rotation_scalar is again 0.6 seconds because in 3.1 we found that only 0.16% of the event_parts show a relative rotation in [0.6, 0.7] with the rest showing a relative rotation in (0.7, 1]. So this parameter seems to already allow for almost all trophallactic contacts to be found with respect to the relative tag angles.

Looking at the found durations for trophallactic contact events it could well be argued to lower the parameter *event_time_seconds*. While this would allow to find more of the very short trophallactic contacts, it is very hard to not miss them. Most trophallactic contacts become visible to the human eye by multiple following images with a bee's proboscis being visible. So I can not generate good ground truth data for these very short trophallactic contacts. Therefore I am more interested in the trophallactic contacts with a duration of several seconds. This is why I decided to leave the parameter at two seconds. It also stays somewhat comparable to the first calculation run. *time_difference_max_seconds* is chosen as 2 for the same reasons as in the first calculation run.

distance_min_pixels and *distance_max_pixels* are adapted. Below a certain distance between the tags it is impossible for the bees to perform trophallactic contact, when they are aligned relatively straight and opposite of each other. Above a certain distance the length of the proboscis is just not enough to cover the distance to perform trophallaxis. In the first calculation run I neglected the first point and underestimated the possible

length of the bees' proboscis. Still not knowing these minimal and maximal distances and looking at the found data I decided for the following methodic guess: I assumed a normal distribution for the trophallactic contact event_parts and fitted a probability density function to the found distances. Figure 3.4 visualizes this.

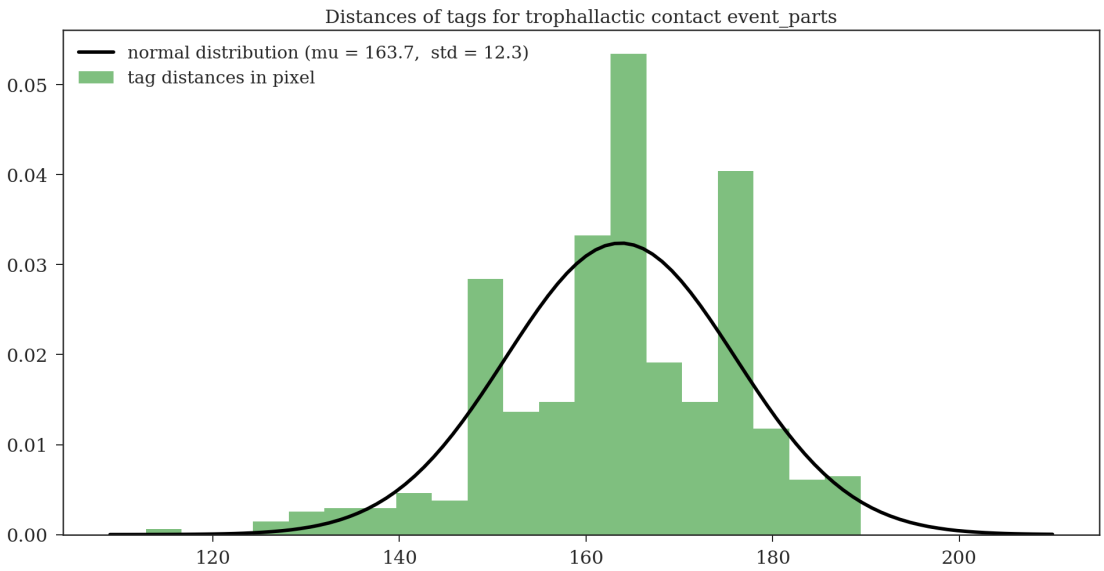

Figure 3.4: Shows the normalized histogram of only the event_parts labeled as trophallactic contacts. Assuming a normal distribution a probability density function in black is fitted to the data.

Assuming a normal distribution, 99.7% of the distances would be found in an interval of $[\mu - 3 * \sigma,\ \mu + 3 * \sigma]$ with μ being the mean and σ being the standard deviation. For $\mu = 163.7$ and $\sigma = 12.3$ the interval is [126.75, 200.65]. This concludes the reasoning for the parameters of the second calculation run.

The second run of calculations resulted in 534 events of which all videos were reviewed. 14 events had to be discarded as it was not decidable for me, what was happening there, because of obstructed view or the picture resolution. Table 3.1 gives an overview of the resulting ground truth data.

The histograms for durations and relative rotations look very similar to those for the first calculation. For that reason they are left out in this document. No event_part with relative rotations below 0.7 is found for the trophallactic contact event_parts. 10 event_parts with relative_rotation in [0.7, 0.8) are found. Setting the parameter *relative_rotation_scalar* to 0.7 might be another way to increase the relative number of trophallactic contact events found without decreasing the absolute number of trophallactic contact events found significantly. Figure 3.5 shows that some event_parts for the higher distances in the range of [190, 200] pixels were found. The mean distance for trophallactic contacts jumped up from 163.7 to 165.1 pixels.

Only after generating the second set of ground truth data I realized that the way

	trophallactic contact	non-trophallactic contact
absolute number of events	59	461
relative number of events (precision)	11.3%	88.7%
number event parts	1441	5760
mean event duration in s	10.6	3.7
mean trophallactic contact duration in s	7.8	0
number of trophallactic contacts with duration over 2 s	37	0
mean trophallactic contact duration in s for trophallactic contacts with duration over 2 s	12.1	0
mean pixel distance for trophallactic contacts	165.1	-

Table 3.1: Shows statistics for the second calculation run and the resulting ground truth data.

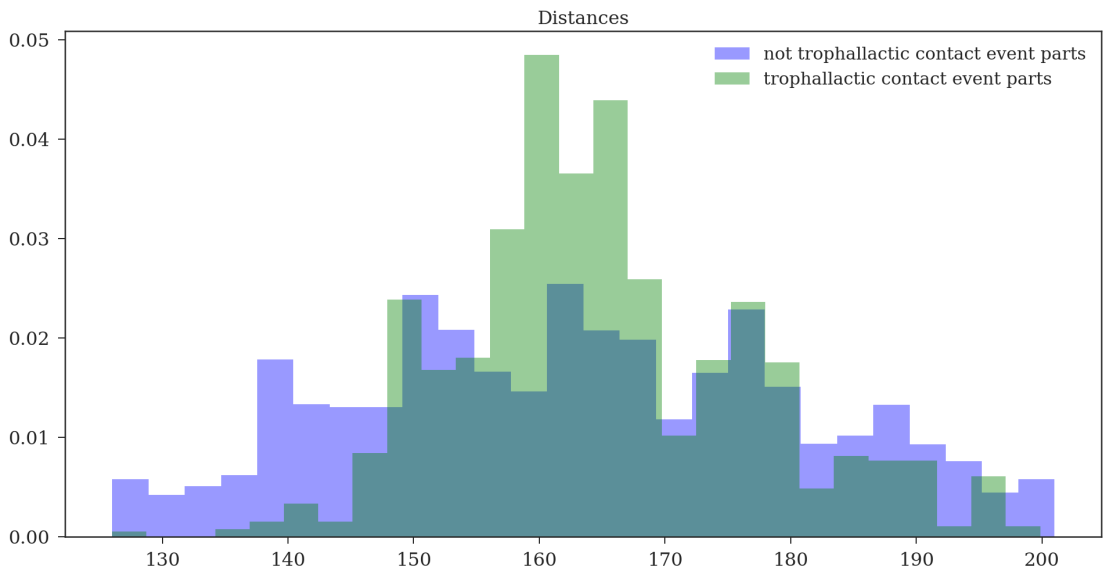

Figure 3.5: Shows two normalized histograms of the found distances between the tags in pixels for all event_parts.

the probability density function was fitted to the distances, was not a clean one. The assumed trophallactic contacts for the higher distances were not represented in the first set of ground truth data. This led to a lower mean and possible to a lower standard deviation than I would have found if I went with the following more clean approach: distribute bins (for example 60) of distances over the whole value range of distances. Count frequencies of bins by putting each distance in its corresponding bin. Choose the bin with the highest frequency. Calculate the mean distance d of that bin. Mirror the left side vertically at $x = d$. Add all values found above 190 distance to the original values of distances. Fit a probability density function to this data.

3.3 Applying Machine Learning

To predict the classification of the data into trophallactic contact events and non trophallactic contact events a random forest classifier was trained. Understanding more about the importance of the parameters could also be useful. For this purpose I calculated data per event that corresponds to the individual parameters:

trophallaxis_observed a boolean denoting wether or not trophallactic contact was found - not corresponding to a parameter

distance_min a float, the minimal found distance between the tags for all event_parts - corresponding to distance_min_pixels

distance_max a float, the maximum found distance between the tags for all event_parts - corresponding to distance_max_pixels

relative_rotation_min a float, the minimum found relative rotation for all event_parts - corresponding to relative_rotation_scalar

duration_min a float, the duration of the event in seconds - corresponding to event_time_seconds

The trophallaxis_observed data was converted to 0s and 1s and used as class labels. The rest of the data was standardized to a mean of 0 and unit variance to look like standard normally distributed data, because this a common requirement for many machine learning estimators (see scikitlearn documentation [10]. The random forest classifier was then trained and tested three times through cross validation. I am presenting the mean values of the different scores in table 3.2:

These results do not represent a well working classifier. An accuracy of 0.8980 means that about 90% of the predictions are correct. Without any context one might think, that this was a good value. The problem is that in our case 88.7% of the data is labeled as non trophallactic contacts anyways. So a classifier ignoring all information and simply always labeling as non trophallactic contact would already reach an accuracy of 0.887. Our score is just a bit better. A precision of 0.6 means that 60% of the classifiers predicted trophallactic contact events are actually labeled as trophallactic contact events. So the classifier is more often right than wrong, if it labels an event as

Accuracy	0.8980
Precision	0.6
Recall	0.3044
F1	0.4038

Table 3.2: Shows the means of the three test scores per metric obtained by cross validating a random forest classifier.

trophallactic contact. A recall of 0.3044 means that about 30% of the events labeled as trophallactic contacts were also classified as such, but the remaining 70% were wrongly classified as non trophallactic contacts. This means that the classifier does not find most of the trophallactic contacts. The F1 score connects the precision and recall values through its formula $F1 = 2 * (Recall * Precision)/(Recall + Precision)$ and indicates that either one value is very low or both are low to average. Still the classifier works better than one which would ignore all information and simply label everything as non trophallactic contact. How this is achieved can be understood a bit better by having a look at feature importances. This information can be retrieved if using a random forest classifier. Figure 3.6 shows the results.

```
Feature Ranking:
Feature 3, duration_min: 0.425036
Feature 1, distance_max: 0.210122
Feature 0, distance_min: 0.204530
Feature 2, relative_rotation_min: 0.160312
```

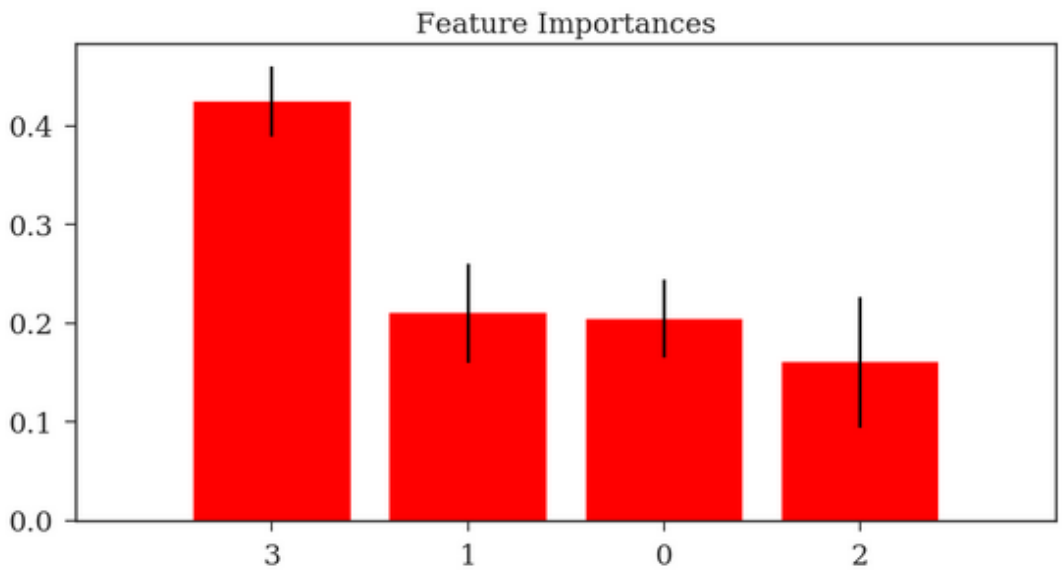

Figure 3.6: Shows the feature importances derived from the trained random forest classifier. The duration of an event influences the decision the most.

The duration of an event, influences the decision of the classifier the most. It seems to be the most informative. Still the other features' importances are not neglectable low, so they also seem to carry some information.

Does this help to understand the choice of values for the filtering parameters better? For me it is not clear how to interpret this information well for multiple reasons. First the classifier does not perform very good. Also the features do not directly translate into the filter parameters. Rather the classifier uses the features to classify the data generated by the filter through its parameters. It could support the idea that restricting the filter mostly to a higher event duration might result in a better ratio of trophallactic contact events to non trophallactic contact events - at the cost of not including the part of the trophallactic contacts which have a shorter event duration.

3.4 Automatic Detection

The goal of this thesis' implementation was to detect trophallactic contacts from the spatio-temporal data of the tracks - if possible automatically. Wether or not this goal was reached depends on the use case. While the random forest classifier (see section 3.3) would be able to give automated predictions, its precision would be low and its recall would be underperforming. The data one would get from this would not be clean. If the use case is fine with these properties it could be automatically supplied with data. If a higher precision or recall is needed, the goal is not reached.

Still, the implementation assists in the extraction of data to make it easily human processable. The evaluation gave insights and suggestions for choosing the filtering parameters values. In this way trophallactic contacts can be labeled with high precision and a significantly less timely effort than if no filtering was used to preprocess the track data or videos.

3.5 Future Work

The BeesBook project aims to automate the tracking and labeling of the bees' behavior, including the trophallactic contacts. For example trophallactic contacts could be important edges in graphs representing the social behavior of honey bees. Automatic detection of trophallactic contacts on spatio-temporal data did not show the necessary success for this use case. A classifier based on image recognition might show success and therefor could be a possible work in the future. To train such a classifier ground truth data will be needed. The generated ground truth data of this work can be used for that purpose. Also the implementation of this work can be used to generate more ground truth data if needed. If the image recognition works, the automation will need a preprocessor that feeds the classifier with events to predict labels on. If adapted this implementation could take on the role of such a preprocessor and be one part of the pipeline for the automatic detection of trophallactic contacts.

Bibliography

[1] BioroboticsLab. bb_binary documentation. available at http://bb-binary.readthedocs.io/en/latest/api/common.html (visited on 2017-11-17).

[2] BioroboticsLab. bb_tracking 0.1.0 documentation. available at http://bb-tracking.readthedocs.io/en/latest/api/data.html#module-bb_tracking.data.datastructures (visited on 2017-11-15).

[3] Franziska Boenisch. *Feature Engineering and Probabilistic Tracking on Honey Bee Trajectories*. Bachelor Thesis, Freie Universität Berlin, February 2017.

[4] Karl Crailsheim. Trophallactic interactions in the adult honeybee (Apis mellifera L.). *Apidologie*, 29(1-2):97–112, 1998.

[5] Walter M. Farina and Alejandro J. Wainselboim. Trophallaxis within the dancing context: a behavioral and thermographic analysis in honeybees (Apis mellifera). *Apidologie*, 36(1):43–47, January 2005.

[6] Joaquín Goyret and Walter M. Farina. Trophallactic chains in honeybees: a quantitative approach of the nectar circulation amongst workers. *Apidologie*, 36(4):595–600, October 2005.

[7] Danielle P. Mersch, Alessandro Crespi, and Laurent Keller. Tracking Individuals Shows Spatial Fidelity Is a Key Regulator of Ant Social Organization. *Science*, 340(6136):1090–1093, May 2013.

[8] Jakob Mischek. *Probabilistisches Tracking von Bienenpfaden*. Master Thesis, Freie Universität Berlin, September 2016.

[9] Dhruba Naug. Structure of the social network and its influence on transmission dynamics in a honeybee colony. *Behavioral Ecology and Sociobiology*, 62(11):1719–1725, September 2008.

[10] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, October 2011.

[11] Thomas O. Richardson, Jonas I. Liechti, Nathalie Stroeymeyt, Sebastian Bonhoeffer, and Laurent Keller. Short-term activity cycles impede information transmission in ant colonies. *PLOS Computational Biology*, 13(5):e1005527, May 2017.

[12] Alexa Schlegel. *Temporal Analysis of Honey Bee Interaction Networks Based on Spatial Proximity.* Master Thesis, Freie Universität Berlin, April 2017.

[13] Kadir Tugan. *Detection of Honey Bee Dancers and Followers with Model-Based Machine Learning.* Master Thesis, Freie Universität Berlin, May 2017.

[14] Fernando Wario, Benjamin Wild, Margaret Jane Couvillon, Raúl Rojas, and Tim Landgraf. Automatic methods for long-term tracking and the detection and decoding of communication dances in honeybees. *Behavioral and Evolutionary Ecology*, page 103, 2015.

[15] Fernando Wario, Benjamin Wild, Raúl Rojas, and Tim Landgraf. Automatic detection and decoding of honey bee waggle dances. *arXiv:1708.06590 [cs, q-bio]*, August 2017. arXiv: 1708.06590.