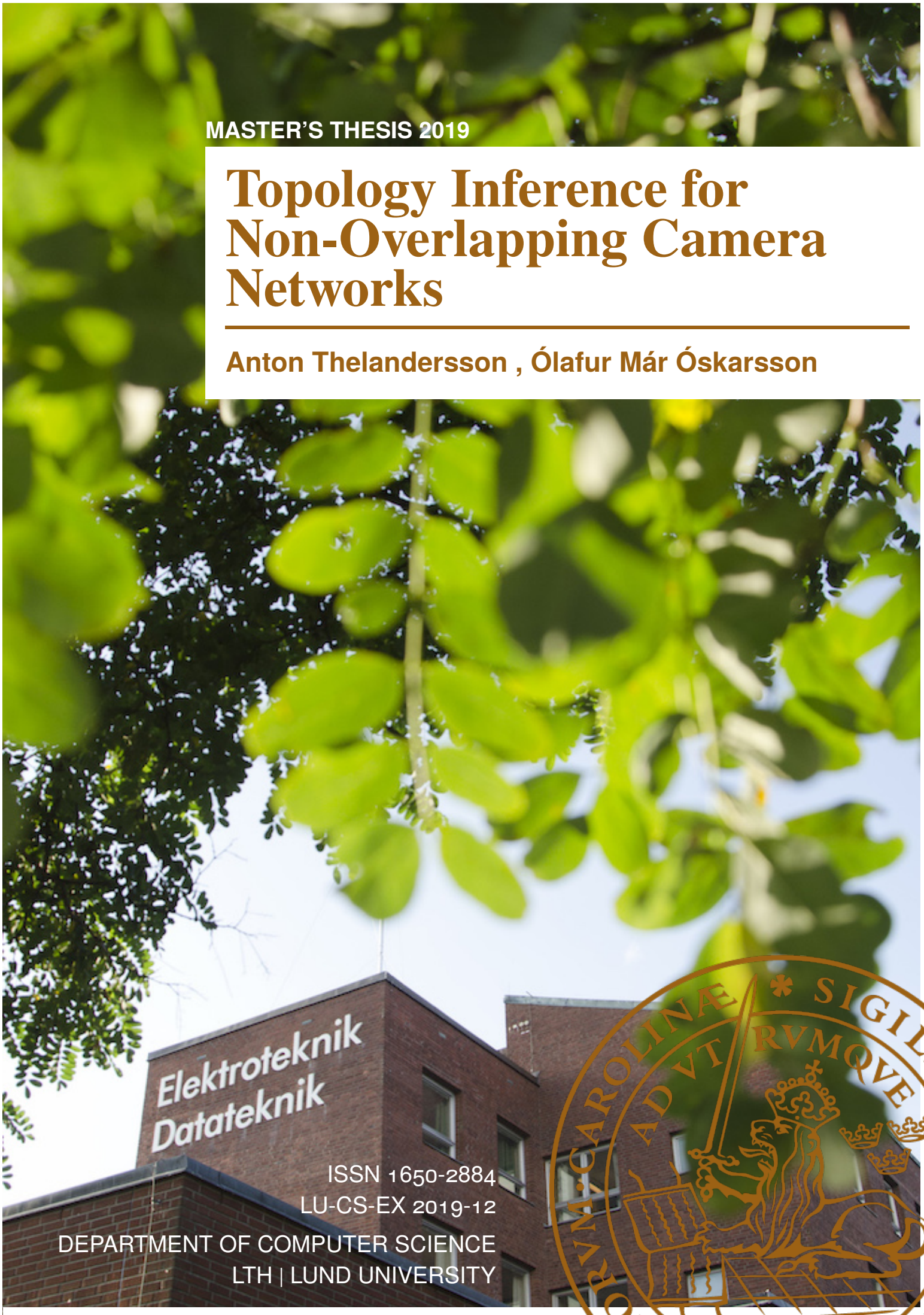


MASTER'S THESIS 2019

Topology Inference for Non-Overlapping Camera Networks

Anton Thelandersson , Ólafur Már Óskarsson

ISSN 1650-2884
LU-CS-EX 2019-12

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2019-12

Topology Inference for Non-Overlapping Camera Networks

Anton Thelandersson, Ólafur Már Óskarsson

Topology Inference for Non-Overlapping Camera Networks

Anton Thelandersson
dat14ath@cs.lth.se

Ólafur Már Óskarsson
dat14oos@cs.lth.se

June 26, 2019

Master's thesis work carried out at Axis Communications.

Supervisors: Elin Anna Topp, elin_anna.topp@cs.lth.se
Viktor Andersson, viktor.a.andersson@axis.com
Anders Krüger, anders.kruger@axis.com

Examiner: Volker Krüger, volker.krueger@cs.lth.se

Abstract

As the size of camera surveillance systems increases, the task of tracking a target becomes increasingly complex. When a target leaves a camera's view it is both time consuming and unnecessary to search for the targets reappearance in the entire camera system, since the target can only reappear in the adjacent cameras. Knowing the camera topology can therefore drastically increase the efficiency of target tracking in a camera network.

In this thesis we investigated if motion data gathered from cameras can be used to infer the camera topology. Two different approaches are evaluated to see if the camera topology can be accurately inferred without human re-identification or if human re-identification is needed.

The results show that the camera topology can be inferred without human re-identification when the traffic density in the environment is normal. However, when the traffic density is high, then human re-identification becomes essential.

Keywords: Camera topology, non-overlapping, re-identification, entry/exit zones, weak links

Acknowledgements

We would like to thank our supervisor, Elin Anna Topp, for guiding us through our master thesis. Her guidelines and feedback have been very valuable for us. We would also like to thank our supervisors at Axis Communications, Viktor Andersson and Anders Krüger, for frequent feedback and discussions.

We would also like to thank our examiner Volker Krüger and our opponent Christoffer MacFie for valuable feedback.

Finally, we would like to thank those who who mean the world to us and have made all of this possible, our wonderful moms.

Contents

Chapter 1
Introduction

In this chapter the thesis is introduced. Background information for why this thesis was performed is presented together with a detailed problem formulation. The goal of the thesis is then presented with the research questions we aim to answer. The methodology is described which consisted of four phases, literature study, development, testing and evaluation. Finally, the outline of the report is given.

1.1 Background

In recent years there has been a substantial increase in the demand of camera surveillance systems. In Skåne alone, the number of camera surveillance permits granted by the government has more than doubled in the last ten years [1]. The number of cameras that each surveillance system contains is also increasing, as well as the area that an individual system covers. This development in camera surveillance has led to rapid technological advancements in the field.

As the size of surveillance systems continues to grow, Axis Communications noticed that the task of human tracking is becoming more complex and time consuming. Axis Communications is therefore interested in the possibility of optimizing the tracking of a target in their surveillance system. In today's systems it is often quite inefficient to track a target. For instance, if the target is seen in one camera, how does the system know in which camera the target is seen in next? This uncertainty will require tracking algorithms to search the entire camera network for the target. If a target leaves a camera's view it is only necessary to search for its reappearance in the views of the adjacent cameras. It is therefore a waste of both time and computing power to search for the target's reappearance in the entire network, when it would suffice to only search the adjacent cameras.

Axis Communications have developed a video management system, VMS, that enables an operator to view the video feed from multiple cameras at the same time. One of the benefits of knowing which cameras are adjacent to each other is that it is possible to more

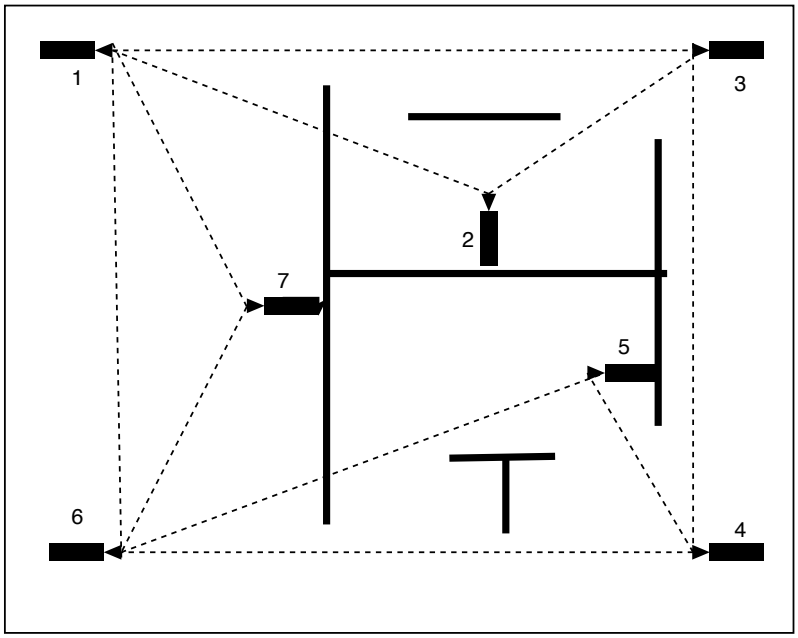


Figure 1.1: A camera network topology. The dashed lines indicate connectivity between cameras.

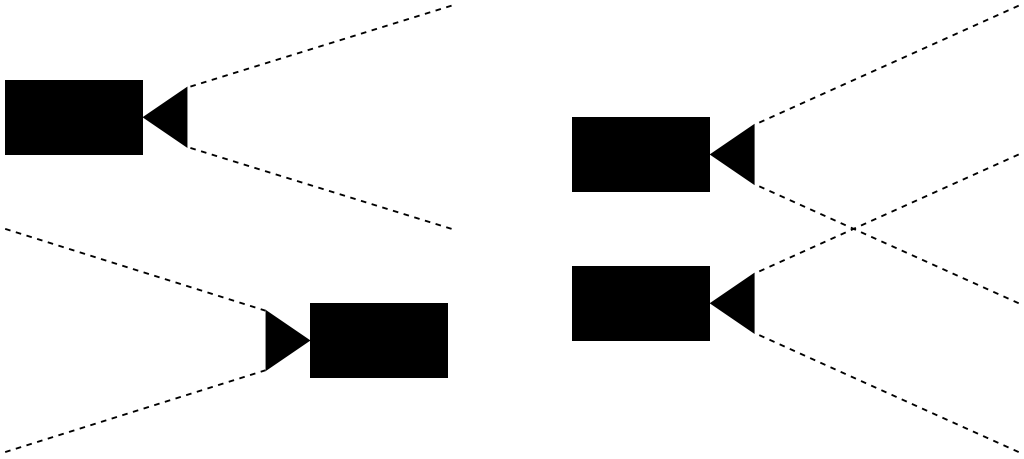
efficiently help an operator follow a target through the VMS. By limiting the number of cameras that a target can reappear in, it is possible to have the video feed from those cameras ready and show them to the operator with low latency. But how does the system know which cameras are adjacent to each other?

1.2 Problem Formulation

The problem of finding which cameras are adjacent to each other is closely related to that of finding the camera network topology. The camera network topology describes the geographical relationships that the cameras in the network have to each other. Even if two cameras are close to each other, it does not necessarily mean that they have a relation in the network topology. The topology of a camera network consists of a number of cameras. A topology consisting of seven cameras is shown in Figure 1.1. There is a link between two cameras if they a person can walk directly between them without being seen in any other camera. One possible solution to the problem of inferring the camera network topology is by doing it manually. This solution however is only suitable for very small networks and becomes cumbersome for large networks, since the number of links can quickly outgrow the number of cameras. For instance, the network shown in Figure 1.1 has only seven cameras but it has ten links between the cameras.

We will investigate if it is possible to accurately infer the camera network topology in an automatic way. We will also aim to recover some parameters about the network, such as the time it takes to walk between the cameras.

It is possible to split camera networks into two main categories: overlapping and non-overlapping, which can be seen in Figure 1.2. In an overlapping camera network, a camera's field of view, FOV, is partly covered by some other camera's FOV. A non-overlapping



(a) Non-overlapping FOVs. **(b)** Overlapping FOVs.

Figure 1.2: The dashed lines show the cameras' FOVs.

network on the other hand has blind spots, which means that there are periods when a person is not visible in any camera's FOV. These blind spots make human tracking more difficult since it is not known where and when the target will reappear and therefore we will focus on non-overlapping camera networks in this thesis. The idea is that inferring the camera network topology will "bridge" these blind spots. A camera's FOV can be split into a few zones where people are seen to enter or exit the FOV. We will find links between these entry/exit zones, instead of between cameras.

One issue when inferring the camera network topology is the weak link problem. A weak link is defined as a path between two cameras that a person cannot walk without being seen in any other camera. The link between Camera 1 and Camera 6 in Figure 1.1 is an example of a weak link. A person that leaves Camera 6 is always seen in Camera 7 before it is seen entering Camera 1. We will examine if it is possible to infer the network topology without having any weak links.

1.3 Goal of the Master Thesis

As mentioned in the previous section, the goal of this thesis is to infer the topology of a camera network consisting of non-overlapping FOV's. We will compare how well we can re-create the actual camera topology both with a correspondence free and correspondence based approach. The correspondence based approach uses human re-identification while the correspondence free does not. Human re-identification is the task of recognizing a person while it moves between cameras in a camera network. The camera topology inference method that is described in this thesis is based on several existing inference methods, that are described in Chapter 2, and is also extended with our own ideas.

The camera environment that we will use to test the camera topology inference method is similar to a supermarket or office space where there are small distances and short transition times between cameras. We will also test it on a smaller camera environment consisting of only four cameras which is tested both in a simulated environment and in a real-world environment.

The following questions will be answered about the topology inference method:

1. How does the data gathered affect the accuracy of the relation graph?
2. Does the accuracy change with the number of cameras in a network?
3. Is an inaccurate human re-identification method worse than using no re-identification?
4. When is it suitable to use human re-identification?

1.4 Methodology

Our methodology can be split into four phases. The different phases are literature study, development, testing and evaluation. Each phase has an input and results in an output that is used as input to the next phase. During the literature study phase, we investigated available topology inference methods and evaluated if they are applicable to our desired scenario. In the development phase we implemented the topology inference method, with and without re-identification, and simulations to be used for evaluation. In the test phase the topology method was applied to various simulations and one real system to measure its performance. In the evaluation phase the results of the testing phase were compared and evaluated.

1.4.1 Literature Study Phase

The literature study that we performed focused both on available methods for inferring the camera topology and the requirements that Axis Communications has of the system. In the beginning we had meetings with Axis Communications employees to get a good understanding of what they expect to get out of inferring the camera network topology. These meetings resulted in requirements and limitations on the system. We also explored in what environments Axis Communications intend on using the system to set further requirements for the system. The following are some of the criteria that we used when selecting what method to implement:

- The size of the camera network should not affect the accuracy.
- It should handle indoor environments where distances between cameras are short.
- It should be accurate.
- It should avoid weak links if possible.
- It should not take unnecessarily long time.
- It should require little to no input from the user.

All these criteria were considered when we researched available methods and we focused on the methods that were most promising and fulfilled as many of the criteria as possible. The methods that showed the most promising results are described in Chapter 2.

1.4.2 Development Phase

Our camera topology method was implemented in the Python programming language. Python was chosen for its comprehensive built-in support for mathematical functions. The implementation consists of a parser and the topology inference algorithms. The parser converts data from the cameras to a format that the topology algorithms can be applied to. The algorithms and its parameters are described in Chapter 3. The method that we implemented can be used both with and without human re-identification.

1.4.3 Testing Phase

To test the method with and without human re-identification two simulations were implemented, one simple and one complex. Both of these simulations simulate how people would have walked inside of a camera environment. The simulations are modeled after a real-world scenario to get data similar to that from a real scenario. We chose to use simulations since they can generate vast amount of data in a short period of time. One problem with simulations however is that they can never reflect a real-world scenario with 100% accuracy. We therefore also created a small real-world setup with actual cameras and people. The real-world setup had the same topology as the simple simulation to compare how well the camera topology inference worked under a real scenario versus a simulated scenario. The testing environments are described in Chapter 4.

1.4.4 Evaluation Phase

The final phase of our methodology is the evaluation phase, where we evaluated the results that we have received from the simulations and the real experiment in the testing phase. The method was evaluated on how well it fulfilled the requirements from the literature study phase. The method was evaluated on how well it performed under various circumstances and with varying traffic in the camera environment. The results of the evaluation phase are given in Chapter 5.

1.5 Outline of the Report

In the next chapter some of the concepts needed to understand this thesis are further explained. Then, previous work in the field of camera topology inference is described. First, a general description of the methods is given and then their most important features are summarized in a table. This is followed by the various theories needed to infer the camera network topology. In Chapter 3 the topology inference method and its implementation is explained. All efforts on how to improve the method is presented there. The simulations and the real-world setup that is used to test the method is described in Chapter 4. In Chapter 5 the accuracy and results are shown. This report ends with a discussion and conclusion about the topology inference method and its accuracy. Further research and improvements in this area are also suggested.

Chapter 2
Background, Related Work and Theory

This chapter contains three major parts. The three parts are background, related work and theory. The background section presents the advantages of using entry/exit zones, the reason for removing weak links and a brief explanation of human re-identification. Related work then describes previous research regarding inferring camera topology, finding entry/exit zones and walking speed of humans. Finally, the necessary theory is presented.

2.1 Background

2.1.1 Entry/Exit Zones

An entry/exit zone is a part of a camera's FOV where people are seen to walk into or walk out of the FOV. Entry/exit zones are most often located on the edges of the FOV but can also be located by a door or by some other obstacle where people can appear in the center of the FOV. Figure 2.1 shows the FOV of a camera that is positioned down a hallway. The figure shows that people can appear and disappear from the FOV both by walking in or out of the edges of the view, or in the center of the FOV by for example walking through a door.

Figure 2.1: A FOV down a hallway.

There are several advantages to be gained by finding entry/exit zones before inferring the camera topology. The primary reason for finding these zones is that it increases the accuracy of the link finding algorithm [2]. The algorithm must only be applied to discrete zones in each FOV which will reduce the noise in the data. Allowing the algorithm to focus on only the entry/exit zones enables it to disregard all movement in other parts of the FOV.

Another advantage is that a camera topology with entry/exit zones is more effective than one that only focuses on cameras [3]. By finding links between zones, instead of cameras, it is possible to more efficiently predict where a target will appear next. This is because a camera will often have many links associated with it, while a zone will only have a few. To put it more accurately, a zone can never have more links than the camera it is in, $L_{zone} \leq L_{camera}$ where L denotes the number of links. The number of links that a camera has is

$$L_{camera} = \sum_{i=1}^{N} L_{zone_i} \tag{2.1}$$

where N is the number of zones in a camera.

Figure 2.2 demonstrates a clear advantage of using entry/exit zones in a camera network topology. The FOV shows a stairwell where people can either walk up or down. There are two entry/exit zones in the FOV, one leading to the upper floor and one to the lower floor. A tracking algorithm that utilizes a topology without entry/exit zones cannot know on what floor a target that is seen exiting this FOV will reappear on. It will therefore need to search both the cameras on the upper and lower floor for the target's reappearance, which is a waste of time. If the tracking algorithm however used a topology with entry/exit

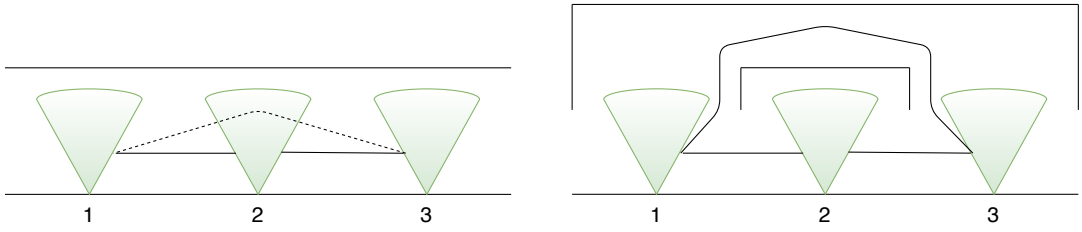
Figure 2.2: A FOV showing a stairwell. Entry/exit zones are shown with ellipses.

zones, it could know on what floor a target would reappear on. For instance, a target seen exiting the entry/exit zone on the top of the stairs can only reappear in the cameras on the upper floor. The tracking algorithm therefore only needs to search the cameras that are connected to the zone that the target exits.

Although using entry/exit zones when inferring the camera network topology has the benefit of increasing the accuracy of the topology, it also has the downside of increasing the requirements on the system. The cameras in the network must be able to know when and where a person enters and exits their FOV. They also need to be able to distinguish between individual persons in a crowded FOV to be able to follow their movement. The video data from the cameras must be processed twice to find the network topology with entry/exit zones, first to find the zones and then to infer the topology.

2.1.2 Weak Links

In a complex camera environment there are often topologically related cameras even though they are not direct neighbors. An example of this can be seen in Figure 2.3a which shows a hallway with three cameras in a row. A person can walk from Camera 1 to Camera 2 and from there to Camera 3. It is though not possible to walk from Camera 1 to Camera 3 without being seen in Camera 2 on the way. However, since a person always walks Camera 1 $\leftrightarrow$ Camera 2 $\leftrightarrow$ Camera 3 the link finding algorithm can also find a link between Camera 1 and Camera 3, since there can be a correlation between the events in Camera 1 and Camera 3. In this scenario the link from Camera 1 to Camera 3 is considered as a weak link. The definition of a weak link is that it is not possible for a person to walk



(a) One path from Camera 1 to Camera 3. **(b)** Two paths from Camera 1 to Camera 3.

Figure 2.3: Three cameras in a line.

between the two cameras that the weak link connects without being seen in one or more cameras on the way. Since it is not possible to walk from Camera 1 to Camera 3 without being seen in Camera 2 the link between them is a weak link. All cameras along a typical walking path can therefore be weakly linked.

There are some drawbacks with inferring the camera topology with weak links. Weak links can confuse an operator if they for example try to follow a target in a video management system, VMS. An example of a simple VMS where an operator can watch videos from the cameras can be seen in Figure 2.4. The left-hand side of the VMS shows the camera where the target is located, Camera 1, and the three cameras that have a link to Camera 1 are shown on the right-hand side. If the link between Camera 1 and Camera 3 is a weak link there is no possibility for the target to reappear in Camera 3 before reappearing in Camera 2 or Camera 4. This will most likely distract an operator which has to watch more video streams than necessary. Another closely related drawback of creating a camera topology with weak links is the fact that it increases the computational complexity of the tracking algorithms since it needs to search more cameras to find the target. Finally, if a camera topology consists of many weak links it is almost equal to not infer the camera topology at all since the tracking algorithm still needs to search in most cameras.

It is therefore necessary to remove the weak links, e.g. the link between Camera 1 and Camera 3 in Figure 2.3a but to keep the other links when inferring the camera topology. There is also a situation where a person can walk two different paths from one camera to another which can be seen in Figure 2.3b, where a person can walk through the FOV of Camera 2 or around it. All links in Figure 2.3b are valid links and therefore none of them should be removed.

2.1.3 Human Re-Identification

Human re-identification is widely considered one of the hardest problems in camera surveillance [4]. Human re-identification is the process of associating images of a person that are taken from different non-overlapping cameras, or from the same camera in different occasions [5]. It is not only machines that struggle with human re-identification, even humans can find it difficult to locate a specific person in a crowded scene. For machines the main difficulty with re-identification is visual ambiguity and spatiotemporal uncertainty which result in a person not having the same appearance across different cameras. Re-identification is even more challenging in video feeds that have low resolution or poor video quality. Variations in lighting across cameras can result in changes in a person appearance across cameras. For instance if a camera is located in a shaded area, the color

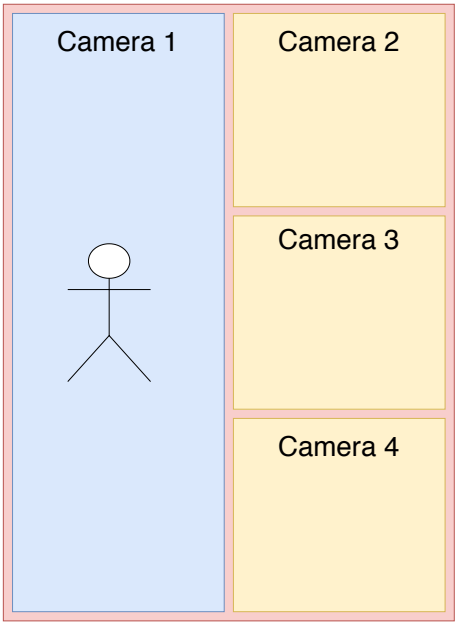


Figure 2.4: Example of a VMS.

of a person's clothing is vastly different when compared to an area with daylight [6]. A person can also be partially or even completely occluded by something in the environment or another person. Occlusion makes it harder to extract features that can be used in the re-identification.

Most of current state-of-the art re-identification methods try to find correspondences between appearance similarities in images. Low level color and clothing texture are two common features that are used [7]. For this thesis, it is not important which human re-identification method is used.

2.2 Related Work

2.2.1 Camera Topology

One of the early methods to infer camera network topology for non-overlapping networks was presented by Marinakis *et al.* [8]. They proposed a method that converts every camera in the network to a node in a graph. Areas where objects can enter/exit the camera environment are marked as source/sink nodes. Then Monte Carlo Expectation Maximization is applied on the observed data to find the valid links.

Tieu *et al.* [9] explicitly handle correspondence to infer the topology. They use Bayesian integration of the unknown correspondence and a non-parametric estimation of statistical dependence between observations in different entry/exit zones. Since correspondence is handled explicitly it is possible to handle varying object speeds in the camera environment. The method uses color transformation to match the objects seen in different cameras.

Zou *et al.* [6] introduce a layered approach that splits the camera network topology into three different layers. They state that it is not enough to use one visual cue to accurately infer the camera topology since that cue can vary substantially from one camera

to another. They therefore propose a method that uses both appearance cues and facial recognition. These cues are used to decrease the entropy of the cross correlation function between departures and arrivals in different entry/exit zones. Since the method uses both facial recognition and appearance cues it can handle varying object speeds in the network. To avoid the weak link problem, they propose a link refining method that looks at mutual information in both zones to eliminate the weak links. They are also able to identify traffic patterns over a link by normalizing the cross correlation. Their method is a continuous learning method, so it can adapt to changes in the camera network or in the camera environment.

Cho *et al.* [3] propose a method that calculates the walking speed of the objects in the camera environment to infer the camera network topology. To achieve this the cameras must first be calibrated to estimate the relative scales between cameras. The method uses the height of the objects along with other visual cues to identify different objects. The height of the objects is also used to determine their walking speed so that the distance between the cameras can be estimated. This method can handle varying speeds of the objects in the network since it calculates the speed for each individual object.

The methods that have been presented so far infer the camera topology in a centralized approach. Farrell *et al.* [10] however introduce how to find the topology in a decentralized manner where every camera is a processing agent in collectively recovering the topology. They argue that a decentralized approach will scale better than a centralized one, since centralized methods are computationally expensive. A camera finds correspondence with other cameras by looking at observation from both itself and the other cameras within a temporal window. Both appearance and time delay are used to weight the potential correspondences between the cameras. A multinomial distribution is then used to estimate the topology. This method also finds with what probability an object leaving a camera will appear in one of its neighbors.

A downside with method proposed by Marinakis *et al.* [8] is that it is very slow for large data sets and therefore only works for a small amount of data. Their approach also needs weak environmental assumptions, in that it needs to know how many objects are moving in the network. This method also differs from the others since it does not split a camera's FOV into entry/exit zones, but rather handles the FOV as a unit. Not using entry/exit zones can reduce the accuracy of the inferred camera topology, as is discussed later in this thesis. Tieu *et al.* [9] only test their method on a network that consists of two cameras, but the results indicate that it could work on a larger network as well. The method proposed by Cho *et al.* [3] needs to calibrate the cameras, to be able to calculate the height of people, before the topology can be found. This method is unsuitable since our goal is to have a method that does not need a setup phase. A decentralized approach like the one presented in Farrell *et al.* [10] requires cameras that are able to perform heavy calculations. Our goal is to infer the topology without adding additional requirements on the cameras in the network.

Makris [11] proposed a method that learns the topology in an unsupervised manner by looking at temporal correlations between departures and arrivals in entry/exit zones. The method finds the transition time between cameras by finding a peak in the temporal distribution between entry/exit zones. A fixed transition time window is utilized that makes this method unsuited for handling varying traffic speeds since that behavior will not produce a clear peak in the distribution. This method works relatively well for small networks but

for more complex networks it suffers from the weak link problem since no suggestion is made on how to remove them. It also needs a very large number of observations to get a clear peak in the temporal distribution.

X. Chen *et al.* [12] proposed a method that is intended to minimize errors for complex camera networks. They do this by first finding the cross correlation between arrival and departure times in entry/exit zone pairs, similar to [11]. After they have calculated the cross correlation they accumulate it over a small time-window to make the peak clearer. They also show that using a weighted cross correlation using appearance recognition improves the performance of their algorithm.

K. Chen *et al.* [13] proposed an adaptive learning method to infer the camera topology. They model the appearance relationship as a brightness transfer function, BTF, to find the spatio-temporal relationships between cameras. Their method consists of two phases: a batch learning phase and an incremental learning phase. The batch learning phase starts by finding entry/exit zones and then finds links between them by finding a peak in a transition time probability distribution. They also propose two methods to remove weak links after the batch learning phase. In the incremental learning phase, it is possible to find new or modify existing entry/exit zones so the system can adapt to a changing environment. Links are removed or added by incrementally updating the transition time probability distribution from the batch learning phase. They also propose a method to avoid adding weak links in the incremental phase.

The camera topology inference method presented in this thesis is partly based on the methods proposed by Makris *et al*. [11], X. Chen *et al.* [12] and K. Chen *et al.* [13]. Using cross correlation to find connections between entry/exit zones forms the basis for our method. We extend it with using the accumulated cross correlation to be able to accurately find the transition times in complex scenarios. Makris *et al*. [11] and X. Chen *et al.* [12] however do not propose a link refining method. Our link refining method is based on one of the methods presented in K. Chen *et al.* [13].

2.2.2 Entry/Exit Zones

A camera needs to support single camera tracking to be able to find entry/exit zones. Single camera tracking is the task of tracking a person while it navigates in a camera's FOV [14]. In many single camera tracking methods a rectangle or blob is constructed around a person as can be seen in Figure 2.5. A blob is constructed around every person that is seen in the FOV. The blob around a person can for example be used to find the person's position within the FOV and its relative size. By looking at a person's trajectory within a FOV it is possible to find where that person entered and exited the FOV. This is done by looking at every person's first and last point of the trajectory in the FOV.

Single camera tracking is however not flawless and errors do occur, especially in crowded or cluttered FOVs [11, p 22]. A tracking error is the result of the single camera tracking algorithms failure to track a target the entire time it is in the FOV of a camera. An error in the single camera tracking can result from how a person moves in the FOV or from an object in the environment occluding a person. If a person stands still for a long period of time, a single camera tracking algorithm might falsely think that it is a part of the environment. A crowded FOV where people meet, walk past each other or occlude each other in some way often results in tracking errors. Static occlusions caused by objects in the FOV

Table 2.1: Comparison of previous work

Paper	Method	Correspondence	Link Refinement
Marinakis et al. [8]	Monte Carlo Expectation-Maximization	No	No
Tieu et al. [9]	Mutual information, Markov Chain Monte Carlo	Color	No
Zou et al. [6]	Weighted cross correlation, Monte Carlo Expectation-Maximization	Appearance and facial	Mutual information
Cho et al. [3]	Walking speed, distance distribution estimation	Human height	No
Farrell et al. [10]	Sequential Bayesian estimation, modified multinomial distribution	Appearance	No
Makris [11]	Cross correlation, thresholding	No	No
X. Chen et al. [12]	Weighted accumulated cross correlation	Appearance	No
K. Chen et al. [13]	Transition time probability distribution, spatio-temporal information	Brightness Transfer Function	Circular path, mutual information, covariance

Figure 2.5: A blob representing a person seen in a FOV.

or lighting changes can also result in tracking errors.

These single camera tracking errors result in incorrect trajectories for a target in a FOV. An example of an incorrect trajectory is when a target's trajectory is split into smaller sub-trajectories at the locations when the tracking error occurred. A trajectory that is split into two sub-trajectories is shown in Figure 2.6. Each sub-trajectory in the figure has one valid endpoint and one invalid. The invalid endpoints will result in noise in the trajectory data.

A primitive approach to find entry/exit zones is introduced in Gilbert *et al.* [15]. Instead of finding entry/exit zones, they split up a camera's FOV in a 4x4 grid as is shown in Figure 2.7. Every section of the grid is an entry/exit block. This approach has the benefits of being easier to implement and has reduced computational time, since the blocks are fixed and do not need to be found. The disadvantages to this approach, however, outweigh the advantages. The link finding part of the topology inference algorithm will not be as accurate. As can be seen by comparing the entry/exit blocks in Figure 2.7 with the zones in Figure 2.2, the entry/exit zone on the top of the stairs is split in two blocks in the grid based representation. Another disadvantage is that there is less data for the link finding algorithm to use in each block, which will reduce the accuracy. This approach is not able to distinguish between invalid and valid endpoints. The invalid endpoints in Figure 2.6 are treated as valid endpoints, since they are located within an entry/exit block. This will also lead to reduced accuracy in the inferred topology.

Makris [11, pp 40-55] have developed an effective method to finding entry/exit zones by looking at the endpoints of trajectories in a FOV. They show that the endpoints from all trajectories have the highest density in the parts of the FOV where people enter and exit the FOV. The rest of the FOV has a low density of endpoints that are the result of

Figure 2.6: Two sub-trajectories for a person walking down the stairs. Valid endpoints are in the entry/exit zones.

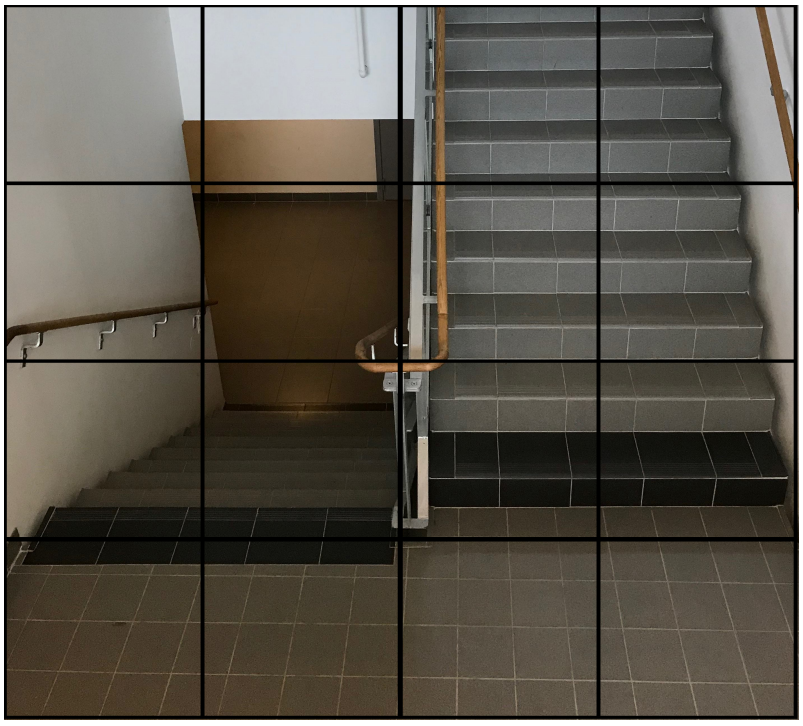


Figure 2.7: A block based representation of entry/exit zones in the FOV showing a stairwell.

tracking errors or stationary noise. Makris *et al.* show that if the endpoints are clustered together then they give an accurate representation of the entry/exit zones. They also show the importance of using clustering methods that handle noisy data efficiently, since single camera tracking errors are quite common.

The clustering method that is proposed by Makris [11, pp 42–45] to find entry/exit zones is a multi-step method that is based on the Expectation-Maximization, EM, algorithm that is described in Section 2.3.6. They also suggest using Gaussian Mixture Models, GMMs, described in Section 2.3.5, to represent the entry/exit zones since GMMs can successfully approximate the shape of most zones. The set $\theta = \{p_1, ..., p_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K\}$ from Eq. (2.6) contains the parameters for each entry/exit zone, where K is the number of zones. μ_i and Σ_i are the center position respective size of entry/exit zone i and p_i is the probability that an endpoint belongs to that zone. Makris [11] overestimates K to find all entry/exit zones, in Section 3.1 we suggest an alternative approach.

The steps in the clustering algorithm proposed in [11, p 42] can be described as follows:

1. The EM-algorithm is used to find clusters which can be characterized according to their density.

2. If a cluster from the previous step contains all the points of a trajectory, then that trajectory is considered to be semi-stationary noise and the cluster deemed invalid.

3. The EM-algorithm is used again to find clusters, but this time endpoints that belong to a semi-stationary cluster are removed from the input.

4. The density of the found clusters are used to determine if they are valid entry/exit zones or tracking failure noise.

This demonstrates the advantages of using the EM-algorithm. It can separate valid data from the noise data. Makris [11, pp 50–55] compares their clustering method with the k-means clustering algorithm by applying them both on actual motion data from a camera. Their results show that their method can successfully find all entry/exit zones and separate them from the noise generated by tracking failure and the environment, while the k-means method fails to do this.

2.2.3 Walking Speed

Previous research in the area of walking speed shows that walking speeds usually follow a Gaussian distribution. For example the study by Chandra *et al.* [16] measured the walking speed of pedestrians in seven different locations. The locations varied from open outdoor environments to a precinct in a city center. The results from each location showed that the walking speed of pedestrians can be approximated with a Gaussian distribution. Table 2.2 shows some previous research where the walking speed has been investigated in order to gather the average walking speed and the standard deviation of the speed. Table 2.2 shows that according to previous work the average walking speed is $1.37m/s$ with an average deviation of $0.24m/s$.

The location where people are walking affects the speed for example if it is a pedestrian crossing, a store or a railway station. The average walking speed in a railway station is faster than in a store since people often are in a hurry when walking in a railway station

Table 2.2: A sample of average walking speeds and standard deviations.

Paper	Mean speed (m/s)	Standard deviation (m/s)
Daamen [18]	1.41	0.22
Fruin [19]	1.40	0.15
Henderson [20]	1.44	0.23
Hoel [21]	1.50	0.20
Lam *et al.* [22]	1.19	0.26
Older [23]	1.30	0.30
Tregenza [24]	1.31	0.30
Young [25]	1.38	0.27
Estimated Average	1.37	0.24

while people in a store walk around slower to be able to look at the groceries. Walking speed is also dependent on age and gender, where men tend to walk faster than women [16]. A person's walking speed does however remain relatively unchanged between the ages of 20 and 70 [17].

2.3 Theory

This section presents the theoretical background for this thesis. Cross correlation, Dijkstra's algorithm, Gaussian distribution and F1 score is presented together with three methods that are used to find entry/exit zones. These three methods are Gaussian Mixture Model (GMM), Expectation-Maximization (EM) and Bayesian Information Criterion (BIC).

2.3.1 Cross Correlation

Cross correlation is often used to measure the similarity between two signals and can be applied to both discrete and continuous signals. One example with continuous signals is when there are two sinusoidals which are shifted by some value in the time-axis. The cross correlation can be used to find how much the sinusoidal in Figure 2.8 are shifted in respect to each other, where τ represents the shifted value. In this thesis the cross correlation is used to find similarities between discrete timestamp sequences and therefore the discrete cross correlation is further explained.

The binary sequence $D_x(t)$ represents the departure times in Zone x, where a 1 refers to an observed departure. The binary sequence $A_y(t)$ represents the arrival times in Zone y, where a 1 refers to an observed arrival. The cross correlation between these two binary sequences can be calculated using Eq. (2.2) where τ represents the time delay, which can

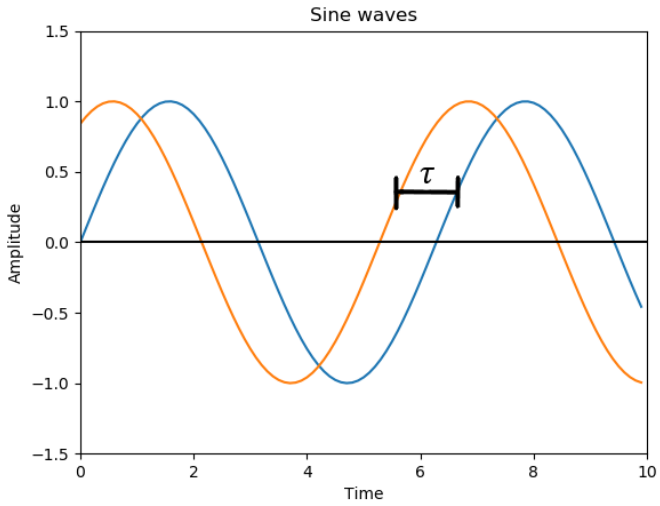


Figure 2.8: τ represents the delay between the two sinusoidals.

be explained as: a departure in D_x at time t is related to an arrival in A_y at time $t + \tau$.

$$C_{x,y}(\tau) = \sum_{t=-\infty}^{\infty} D_x(t) \cdot A_y(t + \tau) \tag{2.2}$$

If for example the two binary sequences $D_x(t) = [1, 0, 1, 0]$ and $A_y(t) = [0, 1, 0, 1]$ are used then the cross correlation is calculated as follows:

$$\begin{aligned} C_{x,y}(0) &= 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 1 = 0 \\ C_{x,y}(1) &= 1 \cdot 1 + 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 0 = 2 \\ C_{x,y}(2) &= 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 0 = 0 \\ C_{x,y}(3) &= 1 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 + 0 \cdot 0 = 1 \end{aligned}$$

This gives $C_{x,y}(\tau) = [0, 2, 0, 1]$ which shows that the cross correlation, and therefore also the similarity, between $D_x(t)$ and $A_y(t)$ is strongest when the time delay is equal to one ($\tau = 1$).

2.3.2 Dijkstra's Algorithm

Dijkstra's shortest path first algorithm, or just Dijkstra's algorithm, can be used to calculate the cost between nodes in a weighted graph. The start node which the cost is calculated from is often called source node while the end node is called sink node. One limitation with Dijkstra's algorithm is that it does not handle graphs that have negative weights on the edges. If there is no possibility to reach a specific node from the source node Dijkstra's

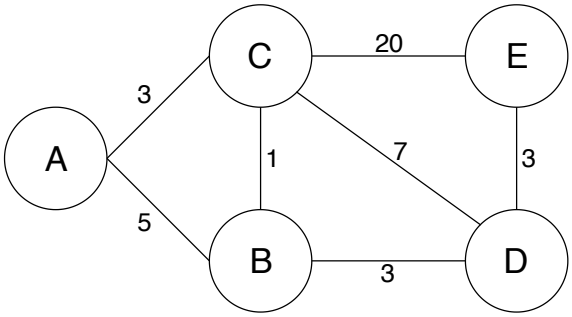


Figure 2.9: Weighted graph used in Dijkstra's algorithm.

algorithm will say that the cost to reach the node is infinity. This type of node is called an unreachable node.

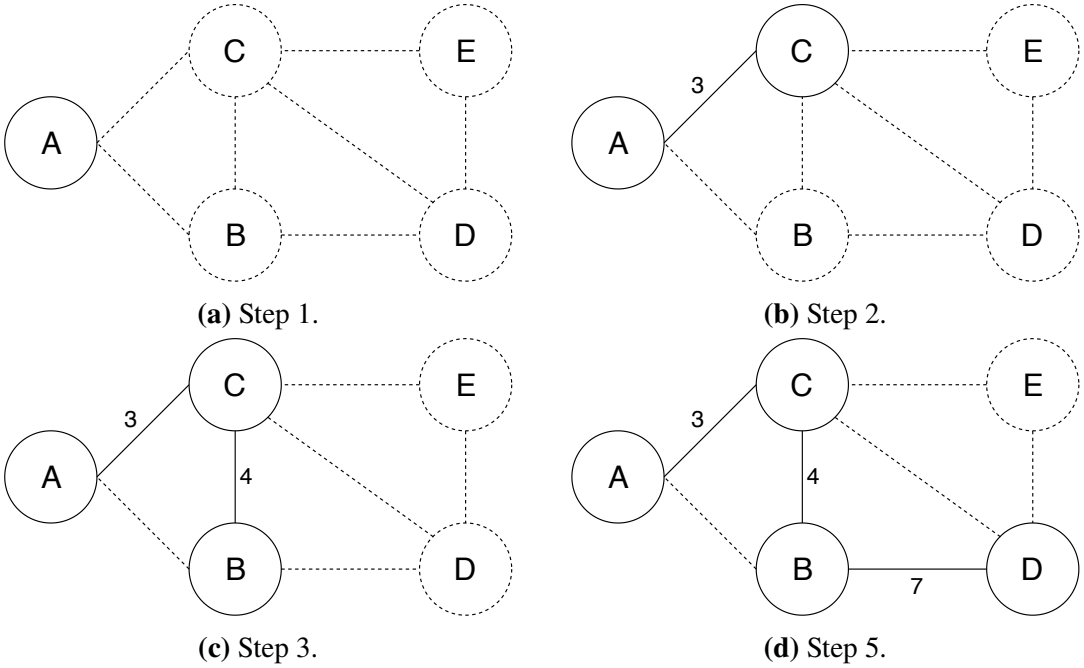
Dijkstra's algorithm is used in many different situations, e.g. when finding the shortest path from one location to another in GPS applications. Edges can, e.g. represent roads while nodes are junctions [26]. Another example for Dijkstra's algorithm is routing protocols for example Open Shortest Path First, OSPF [27].

Dijkstra's algorithm is a greedy approach that can be explained in the following way:

1. Add the source node's neighbors as possible next nodes.
2. From the source node, visit the neighbor node with lowest cost.
3. Add its neighbors as possible next nodes.
4. While there are still unvisited nodes.
   (a) Calculate the cost for all possible next nodes by summing the cost from the source node.
   (b) Visit the possible next node with lowest cost and add its neighbors as possible next nodes.

An example of a weighted graph with five nodes can be seen in Figure 2.9. The steps in Dijkstra's algorithm when calculating the cost from Node A to Node D shown below and in Figure 2.10:

1. Add Node A as source node (Figure 2.10a).
2. Node C is the possible next node with the lowest cost, so it is visited (Figure 2.10b).
3. Node B has a total cost of 4 via Node C, which is the cheapest possible option (Figure 2.10c).
4. Now the edge with the lowest cost is the one between Node A and Node B, but since Node B has already been added with a lower cost, then this edge is ignored.
5. The possible next node with lowest cost is now Node D, with a cost of 7 via Node C and Node B (Figure 2.10d)



(a) Step 1. **(b)** Step 2. **(c)** Step 3. **(d)** Step 5.

Figure 2.10: The steps in Dijkstra's algorithm when finding the shortest path between Node A and Node B.

2.3.3 Gaussian Distribution

A Gaussian distribution, also known as a Normal distribution, is a distribution which shows the probability of obtaining a specific value. A Gaussian distribution is often denoted as $N(\mu, \sigma)$ where μ represents the expected value and σ represents the standard deviation. An example of a Gaussian distribution can be seen in Figure 2.11 where the expected value is 20 and the standard deviation is 3. About *68%* of the values that are obtained from a Gaussian distribution are in the range $[\mu - \sigma, \mu + \sigma]$ while *99.73%* are in the range $[\mu - 3 \cdot \sigma, \mu + 3 \cdot \sigma]$.

2.3.4 F1 Score

F1 score is used in statistical analysis and it measures a model's accuracy. When testing a model there can be several different outcomes for every test result. The possible outcomes are true positive, false positive, false negative and true negative. For example, if a re-identification method is able to successfully recognize a person while moving in the camera environment then it is true positive, else it is false negative. If the method is able to successfully distinguish between two persons then it is true negative, else it is false positive. A perfect model would only consist of true positives and true negatives.

To calculate the F1 score the measures precision and recall are needed. Precision is the ratio between true positives and all estimated positives, which can be seen in Eq. (2.3) while recall is the ratio between the true positives and all objects that are supposed to be positive, i.e. false negatives and true positives and can be seen in Eq. (2.4). Precision can be explained as how many of the estimated objects are correct while recall on the other hand is how many of the positive objects are selected. The F1 score is a combination of

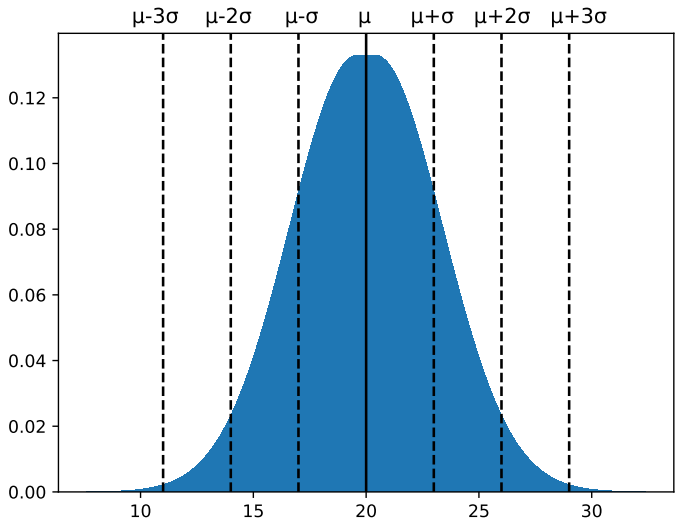


Figure 2.11: Gaussian distribution, $N(20, 3)$.

these two measures and can be seen in Eq. (2.5). The F1 score for a perfect model with no false negatives and no false positives will have the value 1 while a bad model will have a F1 score close to 0.

$$precision = \frac{true\ positives}{true\ positives + false\ positives} \tag{2.3}$$

$$recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{2.4}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.5}$$

2.3.5 Gaussian Mixture Model

Mixture models are often used in statistics to represent one or several subpopulations within the total population [28]. It is not required that the data set that contains the total population maps the individual observation to a specific subpopulation. A Gaussian Mixture Model (GMM) is able to approximate a large set of observations that are affected by several external factors which each have their own probabilities.

A GMM is defined as

$$p(x \mid \theta) = \sum_{j=1}^{K} p_j \cdot p(x \mid \mu_j, \Sigma_j) \tag{2.6}$$

where $\theta = \{p_1, ..., p_K, \mu_1, ..., \mu_K, \Sigma_1, ..., \Sigma_K\}$ represents a set of all parameters, K represents the amount of individual models, μ represents the mean vector, Σ represents the covariance

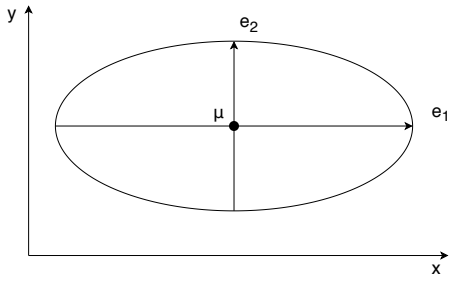


Figure 2.12: GMM ellipse in a two-dimensional space.

matrix and p_j is a set of probabilities. A requirement on p_j is that:

$$\sum_{j=1}^{K} p_j = 1 \tag{2.7}$$

The probability of each individual mixture is defined as:

$$p(i \mid x, \theta) = \frac{p_i \cdot p(x \mid \mu_i, \Sigma_i)}{\sum_{j=1}^{K} p_j \cdot p(x \mid \mu_i, \Sigma_i)} \tag{2.8}$$

where K is the number of mixtures.

Each mixture can be visualized as hyper-ellipsoidals where each point on their surface has equal probability [11, pp 154–155]. In this thesis the ellipsoidals are represented in two-dimensions. The mean vector μ gives the center position of the ellipsoidal and the covariance matrix Σ gives the orientation. An example of an two-dimensional ellipsoidal can be seen in Figure 2.12 where the first eigenvector, e_1, shows in what direction the values from the covariance matrix vary the most in the Euclidean space while the second eigenvector, e_2, shows the direction of the largest variance orthogonal to the first eigenvector.

2.3.6 Expectation-Maximization

Expectation-Maximization, EM, is an iterative algorithm that takes unlabeled data and finds the maximum likelihood estimates of the parameters for a statistical model [29]. The likelihood can be described as the probability of obtaining the data X, given a model θ. The likelihood function can be defined as

$$L(\theta, X) = p(X \mid \theta) = \prod_{i=1}^{N} p(x_i \mid \theta) \tag{2.9}$$

where $X = \{x_1, ..., x_N\}$ is a data set with N samples, θ is a set of components of a GMM as described in Section 2.3.5 and $p(x|\theta)$ is the conditional probability of x given θ. EM can therefore be used to find the parameters of GMM [30].

EM is an iterative algorithm and each iteration is divided into two steps. In the expectation step, E-step, the unlabeled data X and the parameters θ^{old}, from the previous iteration, are used to estimate the likelihood. In the maximization step, M-step, the expectation of the E-step is maximized by re-estimating the parameters θ^{new}. The algorithm will increase the likelihood with each iteration and is guaranteed to converge on a maximum of the likelihood function [31].

2.3.7 Bayesian Information Criterion

Bayesian Information Criterion, BIC, is an index often used in statistics to compare alternative statistical models [32]. A lower BIC index indicates a better model. The BIC is defined as

$$BIC = K \cdot log(N) - 2 \cdot log(L(\theta, X)) \tag{2.10}$$

where θ is the set of parameters from Section 2.3.5, K is the number of components in θ, N is the size of the data set and $L(\theta, X)$ is the likelihood of the tested model. In data fitting, an easy method to increase the likelihood is by introducing more components in θ, but this often leads to overfitting. BIC avoids overfitting by using the parameter K in the calculation of the index [33]. By doing this a penalty is given for using many parameters in θ.

Chapter 3
Approach

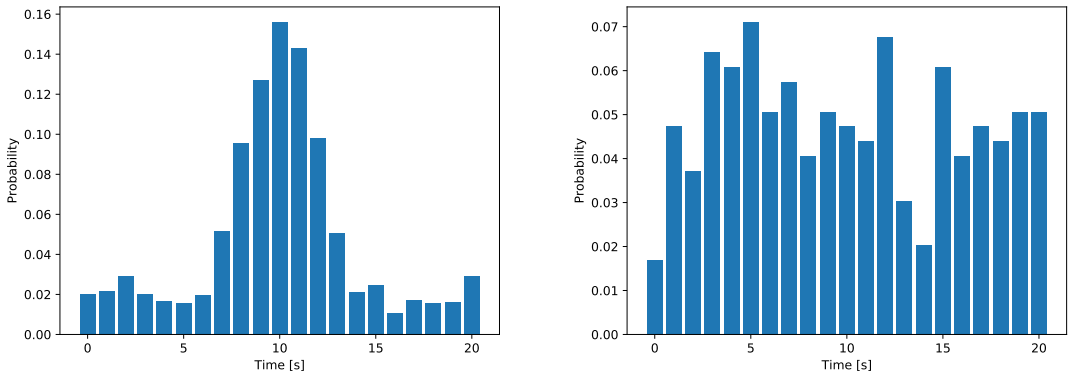
This chapter presents our method to infer the camera topology. The method can be split up into three main steps. These three steps are finding entry/exit zones, link evaluation and link refinement. Link evaluation evaluates if there is a relation between entry/exit zones. The link refinement then removes weak links without affecting valid links. The link evaluation and link refinement were implemented and tested while finding entry/exit zones was studied in a theoretical way.

3.1 Entry/Exit Zones

The first step in inferring the camera topology in a camera network with non-overlapping FOVs is to detect all entry/exit zones in the FOVs. For this we suggest the EM-based clustering algorithm suggested by Makris [11] described in Section 2.2.2. One disadvantage with the clustering algorithm presented by Makris is that it needs the number of clusters it is supposed to find as input. Makris go around this problem by overestimating how many clusters there are supposed to be. They argue that the EM-algorithm will use the extra clusters to model the noise. Their experiments however show that overestimating the number of clusters can lead to the EM-algorithm finding too many entry/exit zones. One method to avoid this problem is to use Bayesian Information Criterion to evaluate the results of the EM-algorithm [34]. This is done by extending steps (1) and (3) to apply the EM-algorithm for varying amount of clusters. The cluster size that results in the lowest BIC, defined in Eq. (2.10), is then chosen to be used in the next step.

3.2 Cross Correlation

The second step in inferring the camera topology is to find links between entry/exit zones. Makris [11] suggested calculating the cross correlation between a departure sequence in



(a) Cross correlation of a valid link.

(b) Cross correlation of an invalid link.

Figure 3.1: Two examples of a cross correlation.

one zone with the arrival sequence in another. As stated in Section 2.2.2 single camera tracking is used to find a persons trajectory in a FOV. The endpoints of the trajectories are used to create the departure and arrival sequence. The first point in a trajectory is when the person arrived in the FOV and the last point is when the person departed.

The cross correlation calculated using the departure sequence from Zone x, D_x, and an arrival sequence from Zone y, A_y, shows if there is a link between the two zones [11, pp 124–134]. Examples of cross correlations between Zone x and Zone y can be seen in Figure 3.1, both when there is a valid link and when there is no link. If there is a link between two zones, then we expect the cross correlation between the two zones to have a peak. We have defined a peak as a group of adjacent points that rise above the noise floor. An example of a peak can be seen in Figure 3.2, where the solid bars represent the peak. As can be seen in Figure 3.1a, the cross correlation has a clear peak. The time with the highest probability is $t = 10s$, which means that the "similarity" between D_x and A_y is highest when $\tau = 10$. The most probable transition time between Zone x and Zone y is therefore 10 seconds. This means that it will most likely take a person 10 seconds to walk from Zone x to Zone y. Figure 3.1b on the other hand has no clear peak which is to be expected when there is no link between the zones.

As described in Section 2.2.2, tracking errors lead to invalid trajectory endpoints that are not positioned inside an entry/exit zone. Endpoints that are not positioned inside a found entry/exit zone are rejected and not used when calculating cross correlation between zones.

3.2.1 Correspondence Free

We base our correspondence free approach on the one presented by Makris [11]. Makris represent the departure sequence as a list where 1 at a specific index represents a departure at that time, otherwise it is 0. For example, in the departure sequence $D(t) = [0, 1, 0, 0, 0, 1, 0, 0]$, there is a departure at $t = 1$ and $t = 5$. The time in this representation is relative to the start time of the video gathering. This representation of time sequences is not optimal since it will mostly be filled with zeros. We instead represent the time se-

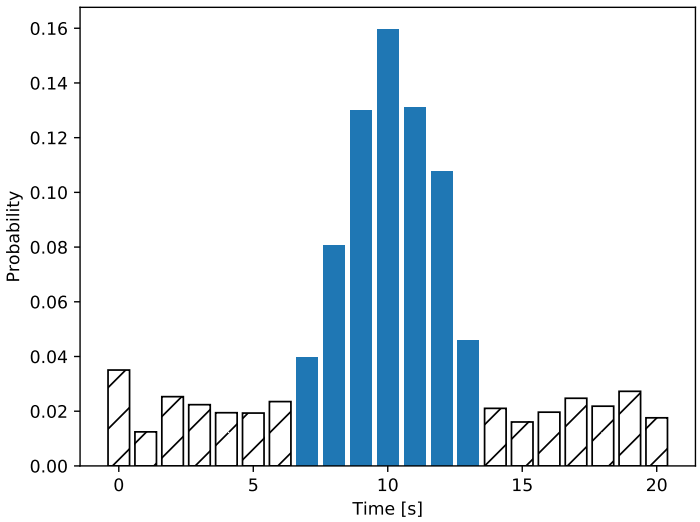


Figure 3.2: An example of a peak.

quences as lists where only the timestamps when there was an entry or exit are included, i.e. $D = [t_1, t_2, t_3]$ where t_n is the nth departure time for that zone. The departure sequence from before can therefore be rewritten as $D = [1, 5]$.

Since we have changed the representation of D_x and A_y it is no longer possible to use the definition of cross correlation, presented in Section 2.3.1, to evaluate if there is a link between Zone x and Zone y. We therefore had to adjust the calculation of cross correlation to manage our representation of D_x and A_y. The new approach to calculate the cross correlation is shown in pseudo code in Algorithm 1 on page 36. As before τ is the delay between departures and arrivals and the occurences of each value for τ, within the interval $0 \leq \tau \leq \tau_{max}$, is calculated to find the transition time distribution.

To minimize the computing time, a threshold τ_{max} was introduced that represents the maximum allowed transition time between zones. In most scenarios it takes a short time to walk between two zones, and therefore it is not necessary to consider transition times above a threshold τ_{max}. For example, if the motion data that the topology method is applied to is gathered over a long time interval, it is of no interest of us to compare departure events in the beginning of the time interval with arrivals in the end of the time interval. τ_{max} can be in the magnitude of minutes, or even hours, the only requirement is that it is larger than the longest transition time in the camera network. A positive side effect of the threshold τ_{max} is that it also minimizes the amount of weak links that are found since many weak links have a transition time above τ_{max}.

3.2.2 Correspondence Based

In the correspondence based approach it is not enough to only save the timestamp when a person entered or exited an entry/exit zone, but also the features used for re-identification. It is therefore not enough to use a time sequence to calculate the cross correlation and event sequences need to be used instead. The departure sequence then has the follow-

Algorithm 1 Correspondence Free Cross Correlation

Input: D_x, A_y, τ_{max}
Initialize $C_{x,y}(\tau) = 0$, *where* $0 \leq \tau \leq \tau_{max}$
Sort A_y *in increasing order*
for each $departure\_time \in D_x$ **do**
    for each $arrival\_time \in A_y$ **do**
        $\tau = arrival\_time - departure\_time$
        if $0 \leq \tau \leq \tau_{max}$ **then**
            $C_{x,y}(\tau)$ += 1
        else if $\tau > \tau_{max}$ **then**
            Break loop and continue to next departure_time
        end if
    end for
end for
Output: $C_{x,y}(\tau)$

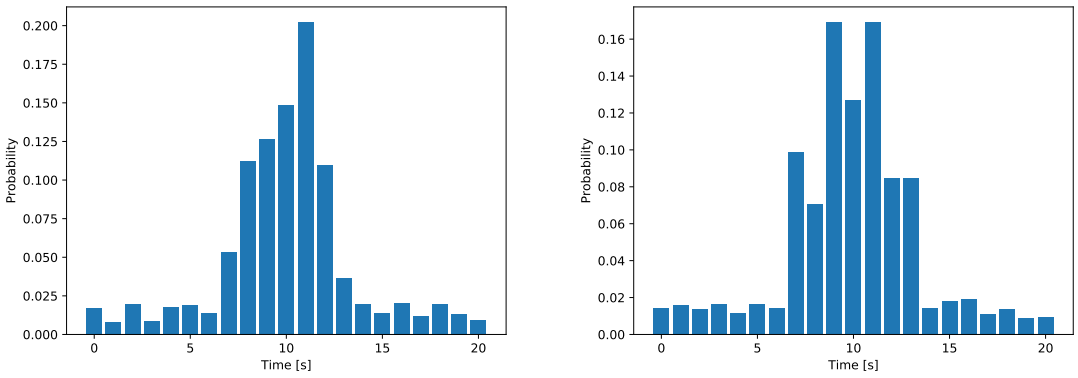
ing format: $D = [e_1, e_2, e_3]$ where e_n is the *nth* departure event in that zone. An event has information of both the departure timestamp and the re-identification features of the person that departed. The pseudo code for calculating the correspondence based cross correlation is shown in Algorithm 2 on page 36. The function $time(e_1)$ gives the timestamp of event e_1 while $feature(e_1)$ extracts the re-identification features of e_1. The function $Sim(feature_1, feature_2)$ measures the similarity in the re-identification features between $feature_1$ and $feature_2$. T_{req} is a threshold for when a similarity is high enough to be considered a re-identification of a person.

Algorithm 2 Correspondence Based Cross Correlation

Input: D_x, A_y, τ_{max}, T_{req}
Initialize $C_{x,y}(\tau) = 0$, *where* $0 \leq \tau \leq \tau_{max}$
Sort A_y *in increasing order*
for each $departure\_event \in D_x$ **do**
    for each $arrival\_event \in A_y$ **do**
        if $Sim(feature(arrival\_event), feature(departure\_event)) \geq T_{req}$ **then**
            $\tau = time(arrival\_event) - time(departure\_event)$
            $C_{x,y}(\tau)$ += 1
        end if
    end for
end for
Output: $C_{x,y}(\tau)$

3.3 Accumulated Cross Correlation

Unfortunately, the cross correlation of a valid link does not always form a perfect Gaussian distribution as in Figure 3.1a. Figure 3.3 shows two examples of cross correlations of valid



(a) **(b)**

Figure 3.3: Two examples of cross correlations.

links where the true transition time of $\tau = 10s$ does not have the highest probability. The distribution in Figure 3.3a has an inclined peak where the transition time with the highest probability is $\tau = 11s$. In Figure 3.3b $\tau = 10s$ has a low probability compared to the points around it. Judging from these distributions $\tau = 10s$ would not be a likely transition time.

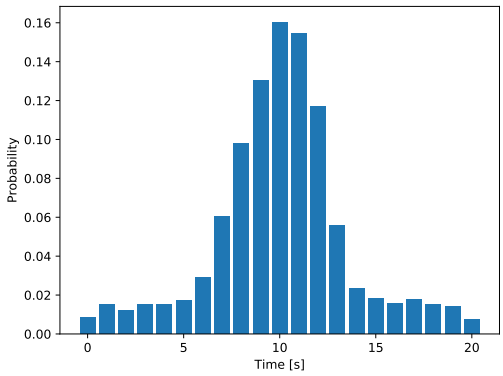
All the examples in Figure 3.3 demonstrate that the cross correlation distributions are not always accurate. We therefore needed some method that can handle distributions with imperfect peaks and still finds the true transition time. We used a method that is proposed in Chen *et al.* [12] with the goal of "smoothing" the cross correlation. We do this by calculating an accumulated cross correlation from the cross correlation. The accumulated cross correlation between Zone x and Zone y can be calculated with the following equation

$$R_{x,y}(\tau) = \sum_{\tau_0=\tau-n_1}^{\tau+n_1} C_{x,y}(\tau_0), \tau \geq n_1 \tag{3.1}$$

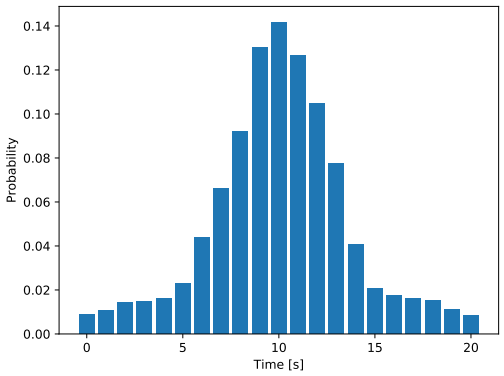
where $C_{x,y}$ is the cross correlation between Zone x and Zone y, n_1 is a "smoothing" factor and τ is the transition time. What the accumulated cross correlation does is to look at multiple points on the cross correlation at the same time instead of only focusing on individual points. The accumulated cross correlation shows the density of the cross correlation. By doing this it can generate the most steady and frequent peak from the cross correlation. Figure 3.4 shows the accumulated cross correlation of the distributions from Figure 3.3 with two different values for n_1.

3.4 Link Evaluation

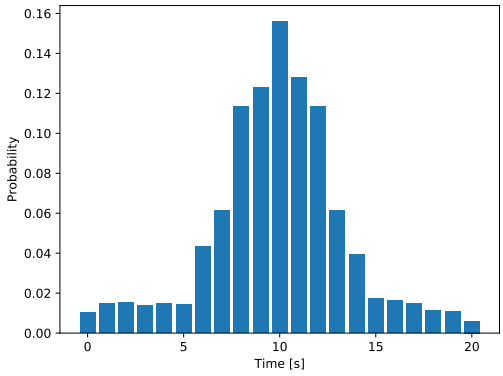
The next step after calculating the cross correlation between two zones is to evaluate if there exists a link between them. It is simple for a human to evaluate if there is a link by looking at a plot of the cross correlation. It is however very inefficient to have an operator manually evaluate every plot of the cross correlations. If there are 30 entry/exit zones in a camera network, then the operator would be required to evaluate $\frac{30 \cdot 29}{2} = 435$ cross correlations. Human error would result in many incorrect evaluations and therefore we



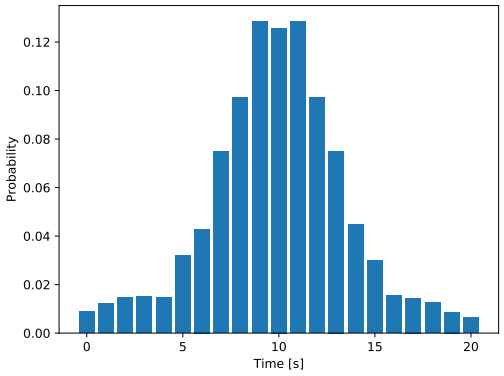
(a) Figure 3.3a with $n_1 = 1$.



(b) Figure 3.3a with $n_1 = 2$.



(c) Figure 3.3b with $n_1 = 1$.



(d) Figure 3.3b with $n_1 = 2$.

Figure 3.4: The accumulated cross correlations of the cross correlations from Figure 3.3 with $n_1 = 1$ and $n_1 = 2$.

needed to create an automatic approach to determine if the cross correlation between two zones created a peak. Evaluating the cross correlation is very complex, since it can vary substantially depending on traffic flow and distance between cameras. The accuracy of the inferred camera topology is highly dependent on the link evaluation. We used a novel approach to evaluate links. Our link evaluation can be split into two steps: finding what the transition time is and determining if the cross correlation has a peak.

3.4.1 What is the Transition Time?

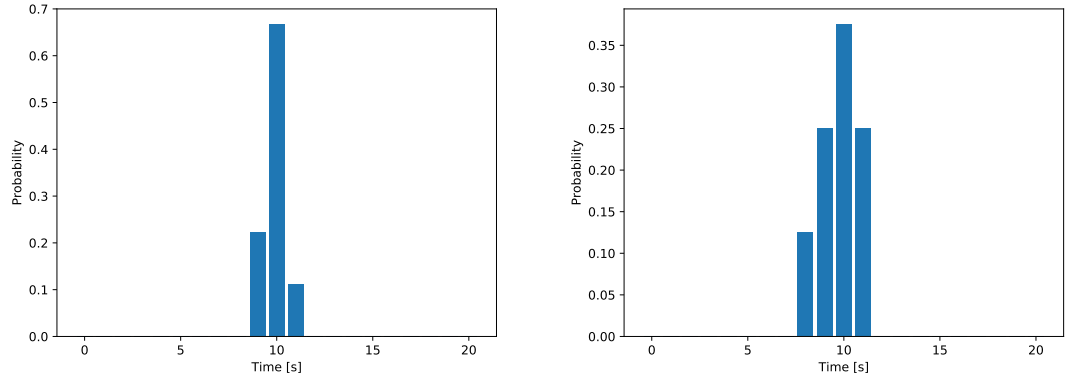
After the cross correlation has been calculated the next step is to see if the cross correlation has led to any peak. One example of a clear and steady peak can be seen in Figure 3.1 but to find peeks gets harder and harder when the noise level increases, for example when there are several different paths from one entry/exit zone to another. This can lead to a cross correlation with several peaks since the time it takes to walk the different paths is not equal. As discussed in Section 3.3, only using the cross correlation alone is not accurate enough to find the most probable transition time. We will therefore use the accumulated cross correlation, presented in Section 3.3 to find the most probable transition time.

Unfortunately, the accumulated cross correlation alone cannot always give the correct value as transition time. As can be seen in Figure 3.4c and 3.4d the points with the highest probability are not the same for the different values for n_1. We therefore need a method that can calculate the accumulated cross correlation with multiple values for n_1 and evaluate the results. We do this with the following method

Algorithm 3 Find the transition time

1: **Input:** D_x, A_y, τ_{max}
2: *Initialize* $max\_list(\tau) = 0$, *where* $0 \leq \tau \leq \tau_{max}$
3: **for** n_1 *from* 1 *to* 10 **do**
4: *Calculate* $R_{x,y}(\tau)$ *according to Eq.* (3.1) *with* n_1
5: *Denote* $\tau' = argmax\ from\ R_{x,y}(\tau)$
6: $max\_list(\tau')$ += 1
7: **end for**
8: $P_{x,y}(\tau) = \sum_{\tau_0=\tau-n_2}^{\tau+n_2} max\_list_{x,y}(\tau), \tau \geq n_2$
9: **Output:** $P_{x,y}$

where $P_{x,y}$ represents the most probable peaks and n_2 is a second "smoothing" factor. The reason for why we do this is the same as with the accumulated cross correlation, to find the steadiest peak by looking at several points on the same time instead of each individual point for itself. If Algorithm 3 on page 39 is applied to the distribution in Figure 3.3a and 3.3b, then the result of the accumulated cross correlation with multiple values for n_1, *max_list*, can be seen in Figure 3.5. If $P_{x,y}$ is calculated for Figure 3.5a, with $n_2 = 1$, then



(a) *max_list* of Figure 3.3a.

(b) *max_list* of Figure 3.3b.

Figure 3.5: Now it is clearer what the transition time is.

it gets the following values:

$$P_{x,y}(9) = C_{x,y}(8) + C_{x,y}(9) + C_{x,y}(10) = 0.89$$
$$P_{x,y}(10) = C_{x,y}(9) + C_{x,y}(10) + C_{x,y}(11) = 1$$
$$P_{x,y}(11) = C_{x,y}(10) + C_{x,y}(11) + C_{x,y}(12) = 0.77$$

$P_{x,y}(10)$ has the highest probability and should therefore be chosen as found transition time. This example shows the efficiency of this method since it was able to estimate the true transition time with 100% accuracy. There are however some cases when the method does not have 100% accuracy. In Figure 3.5b $\tau = 10s$ has the probability $P_{x,y} = 0.87$. Although it is not 100% it is still acceptable since the true transition time had the highest probability.

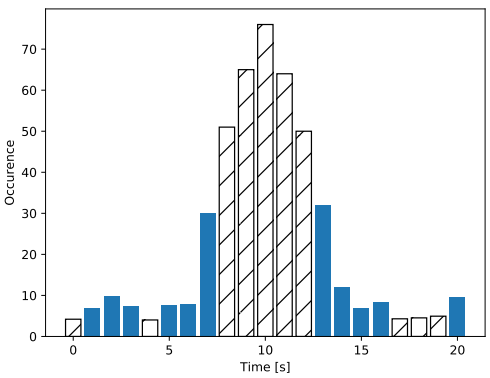
3.4.2 Are they Neighbors?

Now that we have found a candidate transition time, τ', between Zone x and Zone y, the next step is to determine if the zones are neighbors. We have three requirements that must be fulfilled for a link to be considered valid. If all three requirements are met, then we consider τ' to be the transition time.

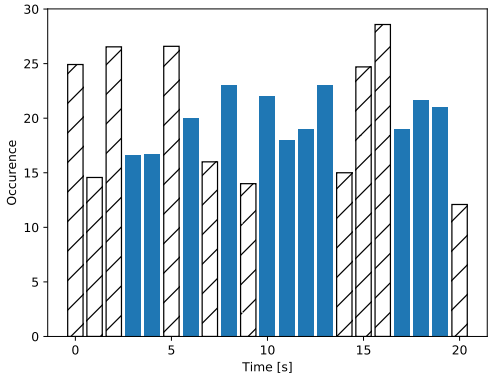
1. The mean occurrence of τ' and the points around it must be above a threshold T_{mean}.
2. The probability of the peak must be above a threshold T_{prob}.
3. The points with the highest probability must be a part of the peak.

Mean Occurrence Verification

When performing this check we use the occurrences of each transition time instead of the probability. Figure 3.6 shows the plots of two cross correlations where the occurrence of each transition time is shown. To handle cross correlations where the noise floor is



(a) Cross correlation of a valid link.



(b) Cross correlation of an invalid link.

Figure 3.6: How we calculate the normalized mean value when $m_1 = 5$.

high, we convert all occurrence values to be relative to the lowest occurrence value. When we calculate the mean occurrence of a cross correlation we do not use all points of the distribution. We instead calculate a normalized mean occurrence by ignoring m_1 points on the top and bottom of the distribution. In Figure 3.6 we only use the bars with solid filling when calculating the mean value. We do this to lower the mean occurrence in distributions that represent valid links while keeping it relatively unchanged for distributions of invalid links. For example, the regular mean value in Figure 3.6a is $mean_{regular} = 22$ while the normalized mean value is $mean_{normalized} = 8$. The distribution in Figure 3.6b has $mean_{regular} = 20$ and $mean_{normalized} = 20$.

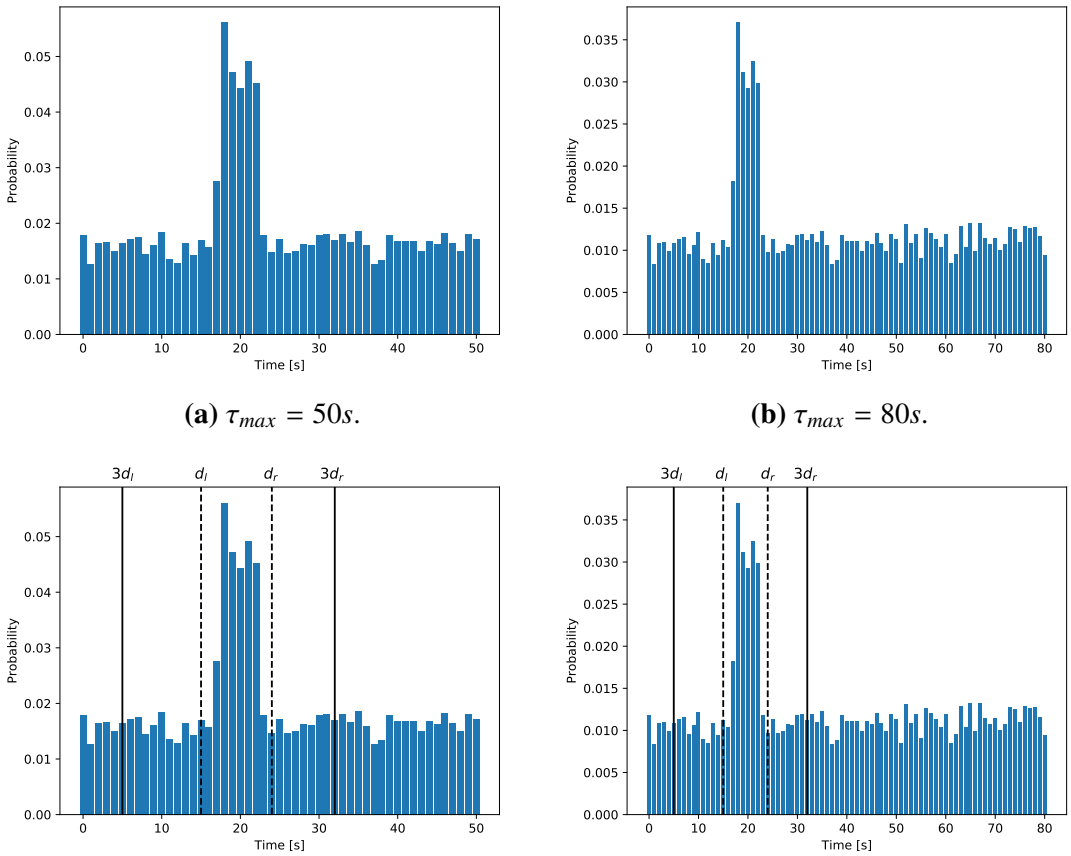
Instead of verifying that only the occurrence of τ' is above the threshold, T_{mean}, we instead verify that the average occurrence of τ' and m_2 points around it is above the threshold. For example, $m_2 = 4$ means that we calculate the mean occurrence of τ' and two points to the left and right of τ'. We do this according to the following equation:

$$\frac{1}{m_2 + 1} \sum_{\tau_0 = \tau' - \frac{m_2}{2}}^{\tau' + \frac{m_2}{2}} C_{x,y}(\tau_0) \geq T_{mean} \tag{3.2}$$

We have set $T_{mean} = k \cdot mean_{normalized}$ where k is a scaling factor. The reason we use $mean_{normalized}$ and not $mean_{regular}$ is so it becomes easier for valid links to pass this verification, while not making it easier for invalid links.

Probability Verification

The second verification checks that the probability of the peak is above a certain threshold. This check is however not as straightforward as it seems at first because the probability of the peak is highly dependent on the size of the maximum allowed transition time τ_{max}. This can be seen in Figure 3.7 where the cross correlation of a valid link is shown with two different values for τ_{max}. The input data to the cross correlations is the same in both cases, the only difference is τ_{max}. The probability for τ' is much higher in Figure 3.7a than



(a) $\tau_{max} = 50s$.

(b) $\tau_{max} = 80s$.

(c) $\tau_{max} = 50s$.

(d) $\tau_{max} = 80s$.

Figure 3.7: This demonstrates how we make the probability verification more independent of τ_{max}.

it is in Figure 3.7b. This is because there are more false correspondences when τ_{max} is large. We therefore needed a method that can accurately calculate the probability of the peak, regardless of τ_{max}. We do this by limiting the width of the time interval that is used in this verification. We denote d_l as the distance from τ' to the left end of the peak and d_r as the distance to the right end. We then calculate the probability of all points in the interval $[d_l, d_r]$ and compare it to the total probability in the interval $[3 \cdot d_l, 3 \cdot d_r]$ as shown in Eq. (3.3). This is illustrated in Figure 3.7c and 3.7d where the dashed lines represent d_l and d_r while the solid lines represent $3 \cdot d_l$ and $3 \cdot d_r$. The value of the left-hand side of Eq. (3.3) is the same for both values of τ_{max} which shows that our method generates the same result for different values of τ_{max}.

$$\frac{\sum_{\tau_0=d_l}^{d_r} C_{x,y}(\tau_0)}{\sum_{\tau_0=3\cdot d_l}^{3\cdot d_r} C_{x,y}(\tau_0)} \geq T_{prob} \tag{3.3}$$

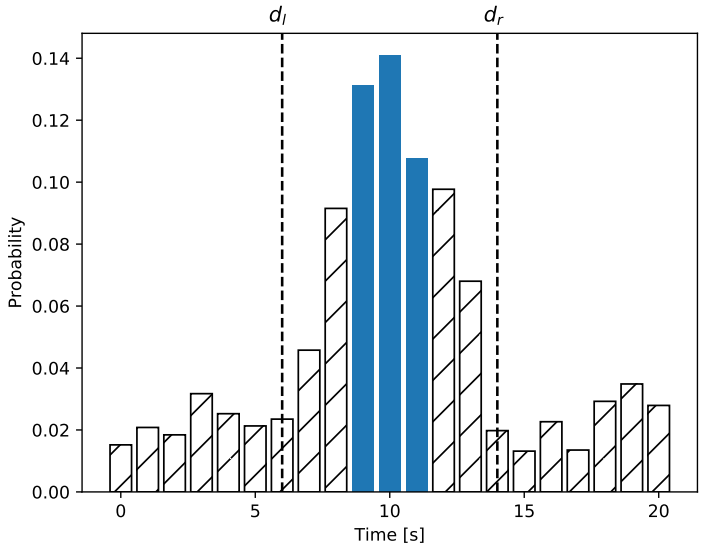


Figure 3.8: As can be seen, all the highest points are within the allowed interval.

Most Probable Points Verification

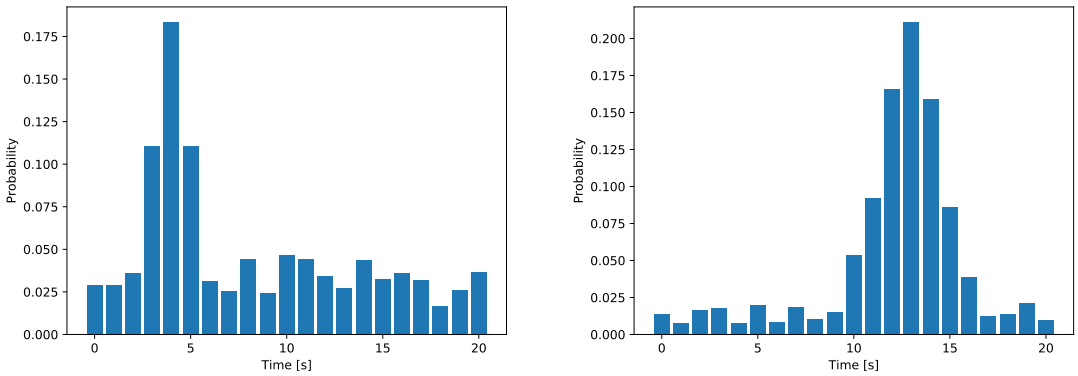
The third and final verification checks that the points which have the highest probability are part of the peak. We require that the m_3 points with the highest probability to be in the interval $[d_l, d_r]$. Figure 3.8 shows a cross correlation where $m_3 = 3$ and the points with the highest probability are shown with solid filling. The cross correlation shows a valid link and all three points with the highest probability are within the allowed interval.

3.5 Link Refinement

It is quite difficult to differentiate between a cross correlation that belongs to a valid link and one that belongs to a weak link. Figure 3.9 shows the cross correlation both for a valid and a weak link. In both cases there is a clear peak and little noise. Since the cross correlation is not efficient in recognizing weak links we will not try to use it to eliminate weak links. Our approach is to infer both valid and weak links, and then use link refinement to eliminate the weak links afterwards.

Camera network topology is often represented as an undirected weighted graph $G = (E,V)$, where the vertices, V, are cameras and links between cameras are represented by edges, E, in the graph. The cost on the edges is the transition time between cameras. An example of a camera network topology with this representation is shown in Figure 3.10, where c is an arbitrary cost.

K. Chen *et al.* [13] use link refinement where they use the found paths in the topology to identify and remove weak links. A weak link represents the same path in the camera environment as two or more valid links, therefore the cost of a weak link is similar to the accumulated cost of corresponding valid links. The graph in Figure 3.10 has two links between Node 1 and Node k, one of which is a weak link and the other consist of several



(a) Cross correlation of a valid link.

(b) Cross correlation of a weak link.

Figure 3.9: The cross correlation is not efficient to differentiate between valid and weak links.

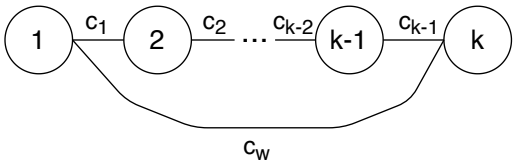


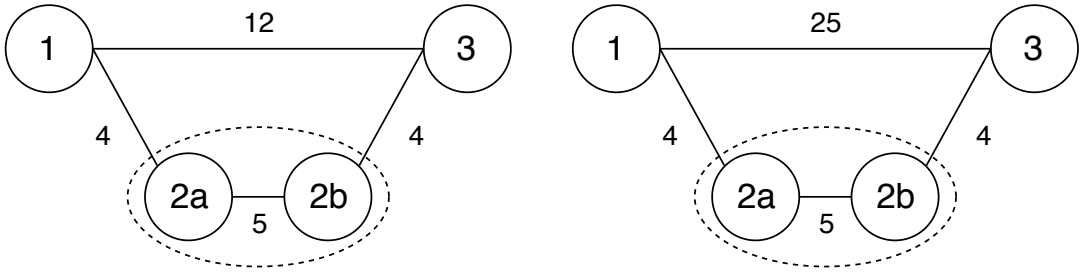
Figure 3.10: An example of a camera topology represented as a weighted graph.

valid links. We use Eq. (3.4) to evaluate if a link is a weak link. If the accumulated cost of one or more valid links between the nodes is approximately the same as the cost for the link that is being evaluated, then it is considered a weak link and is removed from the topology.

$$c_w \approx \sum_{v=1}^{k-1} c_v \tag{3.4}$$

In the rest of this section we will distinguish between two types of links, internal links and external links. Internal links are between two entry/exit zones in the same camera while external links connect two entry/exit zones in different cameras. Since we use the transition time between cameras to identify weak links in the camera topology, then it is also necessary to find the transition time on the internal links. The transition time on the internal links can be found with the same method as for the external links. In the topologies in Figure 3.11 Camera 2 has one internal link between Zone 2a and Zone 2b with a cost of five.

When removing weak links, we apply our link refinement on every entry/exit zone individually. When we evaluate the links that are connected to each zone we start by finding which links are weak link candidates. Internal links cannot be weak links, so they are ignored, and the shortest external link is also never a weak link, so that is also ignored. The remaining links are all considered to be candidate weak links and are evaluated one by one. When a link is evaluated it is removed from the topology graph and then Dijkstra's algorithm, described in Section 2.3.2, is used to find the shortest path between the two



(a) A camera topology containing one weak link. **(b)** A camera topology that has no weak links.

Figure 3.11: Two examples of camera network topologies.

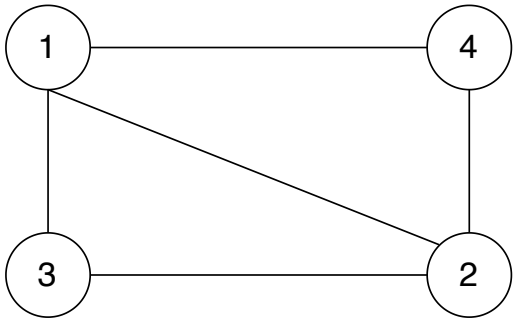
zones that the candidate weak link connects. If the accumulated cost of the shortest path is approximately the same as that of the candidate weak link according to Eq. 3.4, then the candidate link is removed from the topology. For instance, the link between Zone 1 and Zone 3 in Figure 3.11a is a weak link since it has approximately the same cost as the path: Zone 1 ↔ Zone 2a ↔ Zone 2b ↔ Zone 3. This also demonstrates that it is necessary to find the cost on internal links in order for this method to work. The link between Zone 1 and Zone 3 in Figure 3.11b is not a weak link since its cost is not similar to the cost of the shortest path.

3.6 Simulated Human Re-Identification

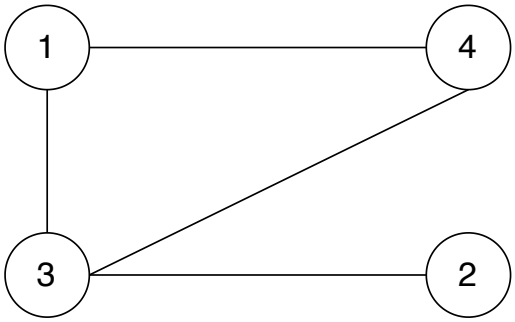
Since we used simulations to evaluate the topology inference method we also needed to simulate human re-identification to test the correspondence based approach. We did this by assigning a unique identifier to every person in the simulation. This identifier is used to identify which person entered a camera's FOV. Since we were interested in testing how well the correspondence based approach performed when using re-identification with different F1 score, we introduced some identification errors. By changing the probability of a true positive identification and false positive identification we could simulate re-identification methods with different F1 score. If the probability for true positive is increased, then the probability for false negative decreases, and vice versa. The same holds for the relationship between false positive and true negative.

3.7 Scoring System

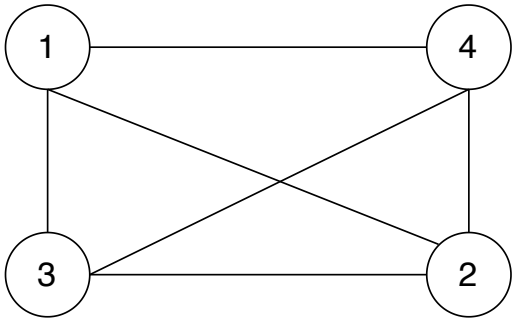
We created a scoring system to compare how well the camera topology is recreated. We set the maximum score for a camera topology to be 1 which represents that all valid links are found, and no incorrect links are found. An incorrect link can either be a weak link or a link that is not possible. The score is reduced if links that exist in the ground truth are missing or if incorrect links are added. A higher penalty is given for missing a valid link than finding an incorrect link. This is because if a valid link is missing then the tracking algorithm could completely lose track of the target, which would force it to search the entire camera network to re-locate it. An incorrect link would however only force the tracking algorithm to search in one more camera, which is not as serious. Therefore, one point is



(a) Ground truth topology.



(b) Two links missing and one incorrect link.



(c) One incorrect link.

Figure 3.12: Ground truth topology and two inferred topologies.

deducted for each incorrect link and two points for each missing link. Internal links are not part of the scoring system since they do not affect the tracking of a target.

The scoring system that we created to evaluate the accuracy of the inferred camera topology is calculated as follows

$$score = \frac{N - 2 \cdot i - j}{N} \tag{3.5}$$

where N represents the number of links in the ground truth, i represents the total number of missing links and j represents the number of incorrect links that are not part of the ground truth. The score is a measure on how accurately the camera topology was inferred.

Figure 3.12a shows a camera topology consisting of four cameras with a total of five links which is used as a ground truth value for the following examples. Figure 3.12b shows an example of an inferred camera topology. There are two correct links missing in this topology which are the links from Camera 2 to Camera 1 and from Camera 4 to Camera 2. The topology also contains one incorrect link, which is not part of the ground truth topology, between Camera 4 and Camera 3. The score of this inferred topology is $\frac{5-2\cdot2-1}{5} = 0$ according to Eq. (3.5). Another example of an inferred camera topology can be seen in Figure 3.12c which contains all links from the ground truth and one incorrect link. The score of this topology is $\frac{5-2\cdot0-1}{5} = 0.8$, according to Eq. (3.5) and has a higher accuracy than the topology in Figure 3.12b.

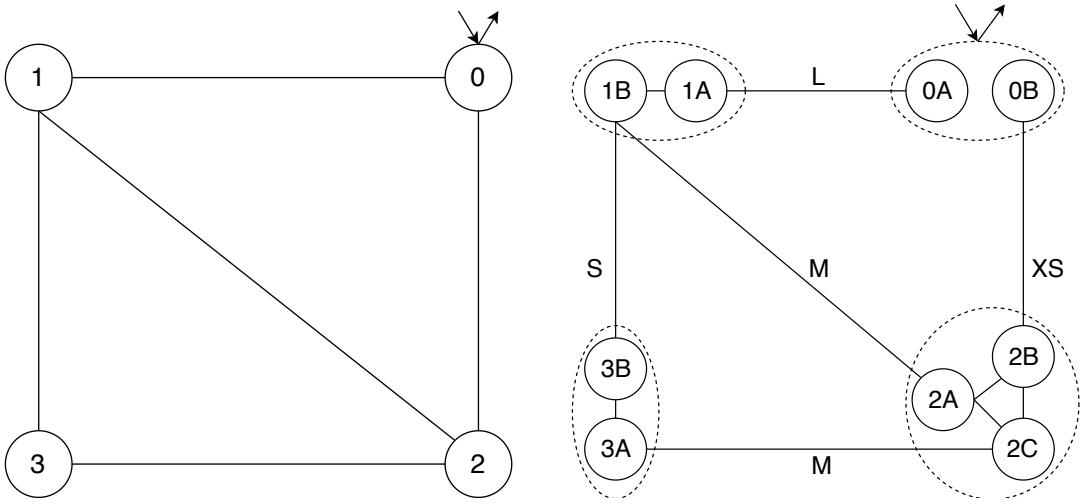
Chapter 4
Testing Environments

This chapter presents the testing environments that was used to evaluate the camera topology inference method. The camera topology inference method was tested with two simulations, one small and one complex, and one real scenario. The small simulation consisted of four cameras while the complex was a camera network that was based on a real camera network inside a store consisting of 35 cameras. The real scenario was based on the small simulation but with real cameras and persons to test the method in a real-world scenario.

4.1 People Behaviour

To make the simulations as similar to the real-world as possible some people behavior was incorporated into the simulations. Bennewitz *et al.* [35] show that people do not move randomly as they walk in an environment. Their motion follows a certain pattern that is often connected to specific locations that they are interested in and they follow specific trajectories while approaching that location.

The time it takes for a person to walk between two entry/exit zones was simulated as a normal distribution. This is because the speed of persons is almost normal distributed as mentioned in Section 2.2.3. Therefore the values from Table 2.2 are used to calculate the different normal distributions according to the length of the path. If for example the distance between two cameras is 10 meters then the time it takes to walk that distance is $\frac{10}{1.37} \approx 7.3$ seconds which is then used as the expected value for the normal distribution. The standard deviation is supposed to be 5.7 times lower than the mean value, since the ratio between the expected walk time and standard deviation from Table 2.2 is 5.7, and therefore the standard deviation is $\frac{7.3}{5.7} \approx 1.3$ seconds when walking a distance of 10 meters.



(a) The simple simulation shown only with cameras.

(b) A more detailed view of the simple simulation.

Figure 4.1: Visualization of the simple simulation.

4.2 Data Gathered from Simulations

The information gathered from the simulations is a text file containing all entry and exit events from all cameras. Each event contains the following information: the time when the event occurred, type of event (enter or exit), what camera it occurred in, which entry/exit zone and the person's unique identifier.

4.3 Simple Simulation

The simple simulation used in evaluating the camera topology inference method has a setup of four cameras and a total of nine entry/exit zones. This camera network has the same size as the one that X. Chen *et al.* [12] use to evaluate their method. The setup contains four different distances which can be seen in Figure 4.1b. Each camera has multiple entry/exit zones which are connected to exactly one entry/exit zone in another camera except Zone 1B that is connected to Zone 2A and Zone 3B.

In this camera network there can be up to 30 links between entry/exit zones. Of these 30 links only five are valid and this shows how important it is to infer the camera network topology if the goal is to track a person in a camera environment in an effective way.

A person can only enter or depart the camera environment via Camera 0. This means that the path of every person in the simulation starts and ends in Camera 0.

4.4 Complex Simulation

The complex simulation is based on a real setup of cameras in a store and contains 35 cameras. A visualization of the complex simulation can be seen in Figure 4.2. The 35 cameras all have multiple entry/exit zones so there is a total of 90 zones. The total amount

Camera Name	Model
Camera 0	AXIS P3364
Camera 1	AXIS M1143-L
Camera 2	AXIS P1425-E
Camera 3	AXIS P3367

Table 4.1: Cameras used in the real experiment.

of possible links in the simulation is 4100 links but only 60 of these are valid links. Figure 4.2 shows all valid links in the simulation. Multiple edges connected to a camera at the same position means that they are connected to the same entry/exit zone in that camera.

A person can walk different paths in a grocery store and therefore two cameras can be connected through several different entry/exit zones. An example of this type of connection is the relationship between Camera 17 and Camera 18.

A person can enter the camera environment through Camera 1 and Camera 20, but it is only possible to exit the camera environment through Camera 1. There are also some paths that people can only walk in one direction to test how well the system handles that case and to make the simulation as close to the reality as possible. For example, when a person enters a store they often must walk through "gates" and it is only possible to walk in one direction through those gates. There are also paths that people walk very seldom and that is simulated by that only a few percent of people choose to take those paths. This can for example be a path that leads from the store to the staff room.

A link between two cameras can have one of four different lengths. These lengths are small, medium, long and extra-long. There are 22 small, 19 medium, 16 long and 3 extra-long links in the simulation.

4.5 Real Experiment

An experiment consisting of four cameras was set up to test if the method worked on real data and not just on simulated data. The cameras that were used are four Axis Communications cameras and their respective model name can be seen in Table 4.1. The relations between the different cameras were the same as in the simple simulation which can be seen in Figure 4.1. The camera FOVs and entry/exit zones can be seen in Figure 4.3. The entry/exit zones were the only location where persons could enter or leave the cameras' FOV. As can be seen in the figure most entry/exit zones were located at the edges of the FOV's except Zone 2C which was in the middle of Camera 2's FOV.

To generate data two persons walked around in the camera environment for a total of 60 minutes. The two test persons were two 24-year-old males and they entered the camera environment a total of 345 times. To generate varying data the two test persons changed their walking behavior with regular intervals. They for example changed their walking speed, started running, changed direction while outside of a FOV and they also stood still for a short period of time inside or outside of a FOV.

To get the necessary information needed to infer the network topology for real cameras we first had to parse metadata from the cameras. The metadata contained the information from the single camera tracking, such as the timestamp and coordinates where a person

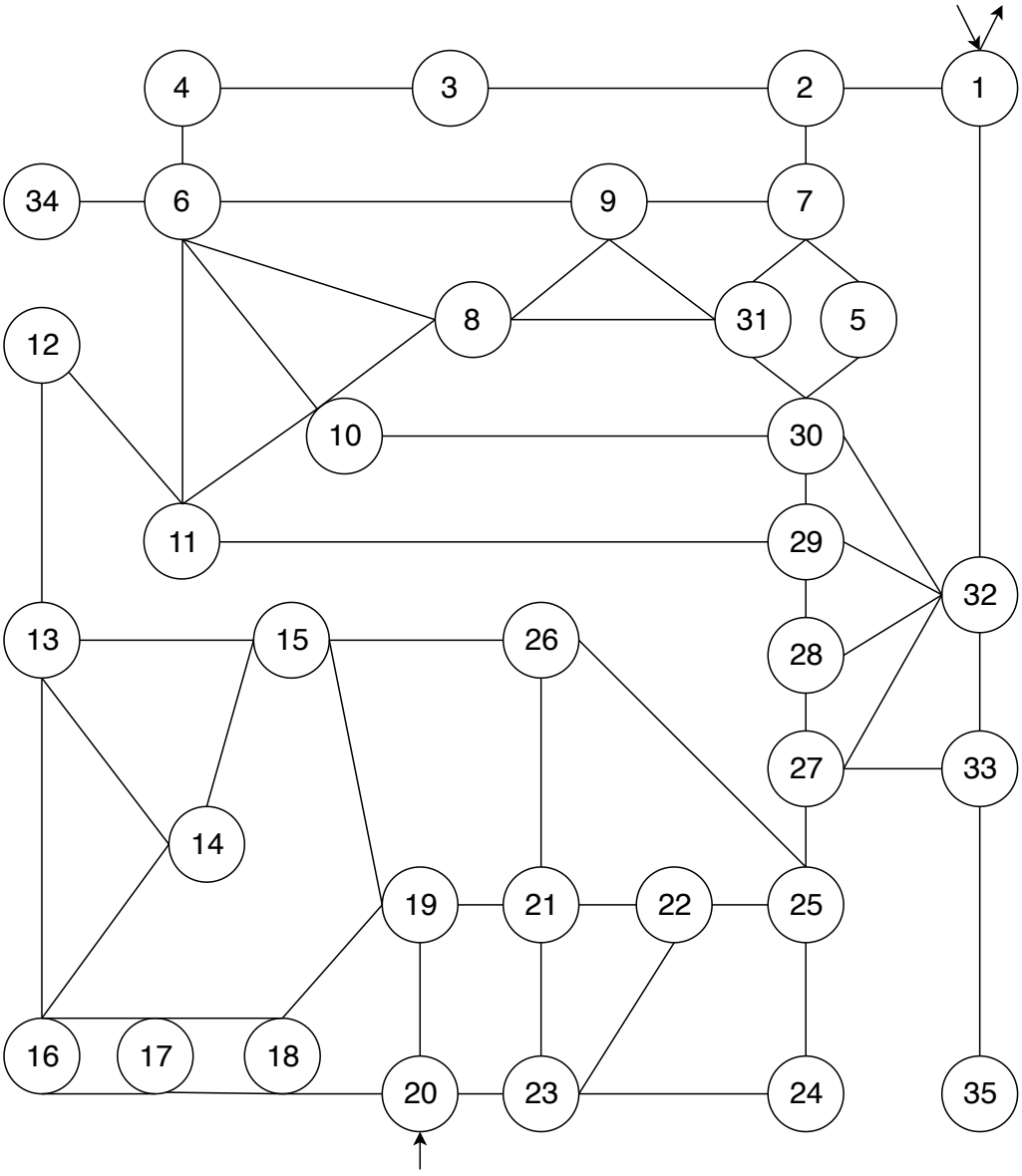


Figure 4.2: Visualization of the complex simulation.

Figure 4.3: FOV's and entry/exit zones in the real experiment.

entered or exited the FOV. Since the parsing of metadata is not the focus of this thesis a few limitations were introduced to keep the parsing simple. These limitations were that multiple people could not be in the same FOV at the same time and it was only allowed to enter the FOVs in the predefined entry/exit zones shown in Figure 4.3. The metadata did not contain any information that could be used for human re-identification and therefore the correspondence based approach could not be tested in the real experiment.

Chapter 5
Results

This chapter presents the result gathered from our tests. First all variables and their respective values are given. The results for the correspondence free approach are then presented for the two simulations. Then, the results for the correspondence based approach are presented for the complex simulation. The results for the simple simulation can be seen in Appendix A. Finally, the results from the real test are presented.

5.1 Information about the Tests

As mentioned in the previous chapter walking speed can be approximated as a Gaussian distribution. Six different distances were used in the simulations. The distances and their respective values for their Gaussian distribution can be seen in Table 5.1. The standard deviation of the internal-camera transition time was increased to simulate that people stop inside a camera's FOV.

Table 5.1: Different walking times depending on distance.

Type of distance	Length [m]	Mean [s]	Standard deviation [s]
Extra-small	0.7	0.5	0.1
Small	2	1.5	0.3
Medium	5	3.7	0.7
Long	7	5.1	0.9
Extra-long	10	7.3	1.3
Internal-camera	-	4.5	1.0

When testing the method some variables were predefined. The variables and their respective values can be seen in Table 5.2. The values were chosen after empirical testing since they gave the best results. When performing link refinement we allow the error margin in Eq. (3.4) to be within 20%.

Table 5.2: The different variables used to infer the camera topology.

Variable	Value
Max transition time, τ_{max}	20s
Smoothing factor, n_2	1
Values to ignore for normalized mean, m_1	10
Number of points for mean, m_2	4
Threshold for mean occurrence, k	1.8
Threshold for probability, T_{prob}	0.4
Highest points part of peak, m_3	3
Probability of true positives, TP	0 - 1
Probability of false positives, FP	0 - 1

The tests that we performed to evaluate the topology inference method had varying amounts of people and frequency, f. The frequency describes how many people entered the simulation every unit of time. We used a tenth of a second as the unit of time in all simulations. Therefore, $f = 1$ means that one person entered the simulation every tenth of a second, i.e. 10 persons per second. We used three different frequencies in our tests to simulate different traffic densities. A high frequency means that there are many people in the camera environment at the same time, similar to rush traffic, while a low frequency means that there are few people in the camera environment at the same time. We used $f = 1$ to simulate rush traffic, $f = 0.1$ for normal traffic and $f = 0.01$ for low traffic. The reason for why we tested with different number of people and frequency was to evaluate how traffic density and the number of people in the system affects the topology accuracy. To evaluate how a camera networks size affects the accuracy of the inferred topology, all tests were performed on both the simple simulation and the complex simulation.

The tests that we performed had six different number of people. Each of the number of people was tested with the three different frequencies. Each test was performed three times and the average of which is shown below.

5.2 Correspondence Free

During these tests the correspondence free algorithm, which can be seen in Algorithm 1 on page 36, was used.

5.2.1 Simple Simulation

The result of the test performed with the simple simulation with varying number of people and $f = 0.1$ can be seen in Table 5.3. The tables for the two other frequencies, $f = 0.01$ and $f = 1$ can be seen in Appendix A.1.1. The score with the three different frequencies for different number of people can be seen in Figure 5.1.

Table 5.3: Results from the simple simulation, $f = 0.1$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	1	0	0	0.60
1000	0	0	0	1.00
2000	0	0	2	1.00
4000	0	0	5	1.00
8000	0	0	9	1.00
16000	0	0	12	1.00

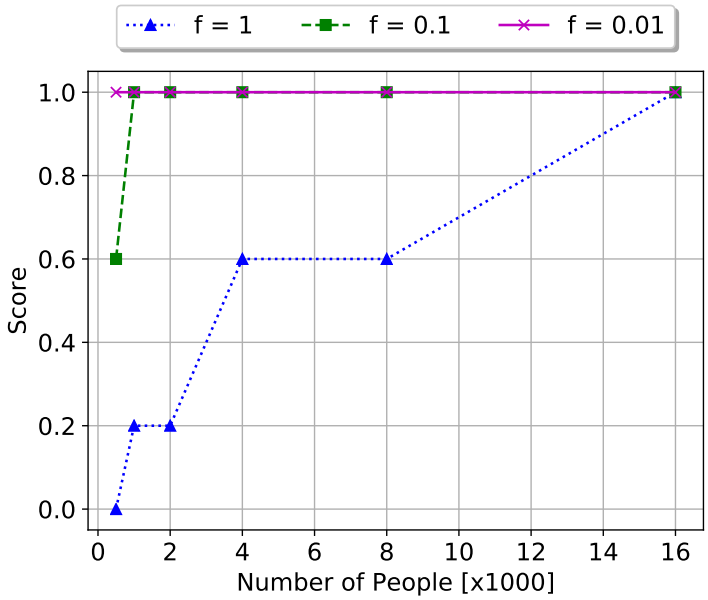


Figure 5.1: Results from the simple simulation with different frequencies.

5.2.2 Complex Simulation

The result of the test performed with the complex simulation with varying number of people and $f = 0.1$ can be seen in Table 5.4. The tables for the two other frequencies, $f = 0.01$ and $f = 1$ can be seen in Appendix A.1.2. The score with the three different frequencies for different number of people can be seen in Figure 5.2.

Table 5.4: Results from the complex simulation, $f = 0.1$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	15	1	5	0.48
1000	10	0	9	0.67
2000	5	0	27	0.83
4000	2	0	60	0.93
8000	1	0	120	0.97
16000	0	0	174	1.00

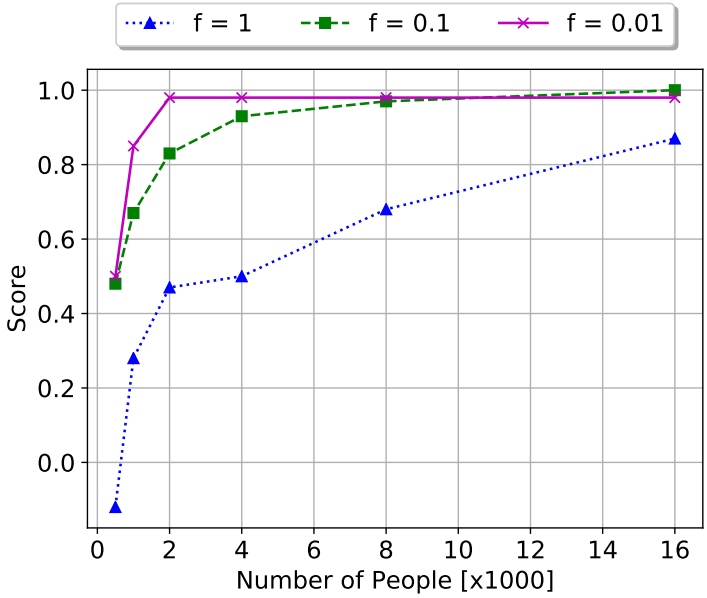


Figure 5.2: Results from the complex simulation with different frequencies.

5.3 Correspondence Based

During these tests the correspondence based algorithm, which can be seen in Algorithm 2 on page 36, was used. Therefore, the probability of true and false positives was changed in the different tests. The results for the simple simulation can be seen in Appendix A.2.

5.3.1 Complex Simulation

Two different tests were performed on the complex simulation were human re-identification was used. The results from the first test can be seen in Figure 5.3 where $f = 1$ and $number\ of\ people = 2000$. The first test was performed to see if human re-identification could infer the camera topology when the correspondence free approach failed. The horizontal dashed line shows the score for the correspondence free approach.

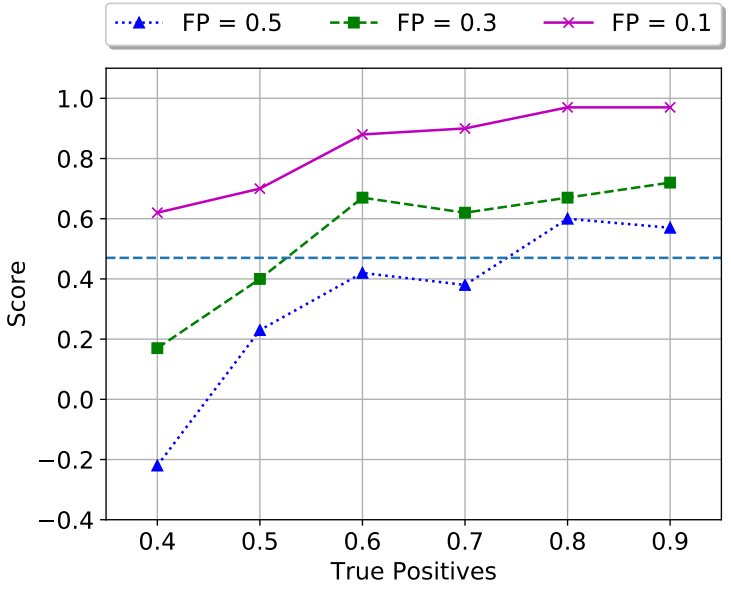


Figure 5.3: Can human re-id increase the accuracy?

The results from the second test can be seen in Figure 5.4 where $f = 0.1$ and $number\ of\ people = 2000$. This test was performed to see if human re-identification could decrease the accuracy. The horizontal dashed line shows the score for the correspondence free approach.

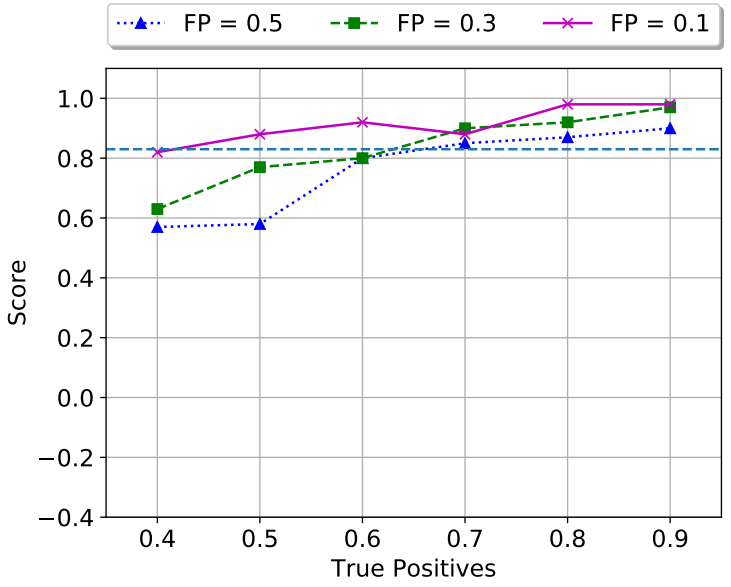


Figure 5.4: Can human re-id decrease the accuracy?

The score of the inferred camera topology for every F1 score from Figure 5.3 can be seen in Figure 5.5.

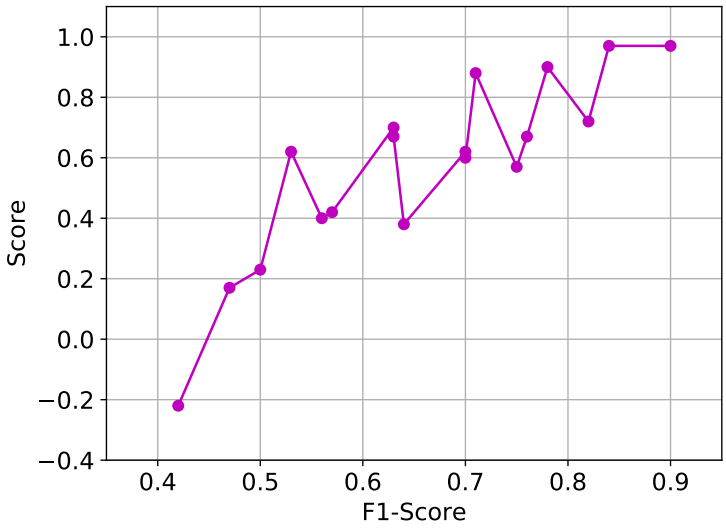
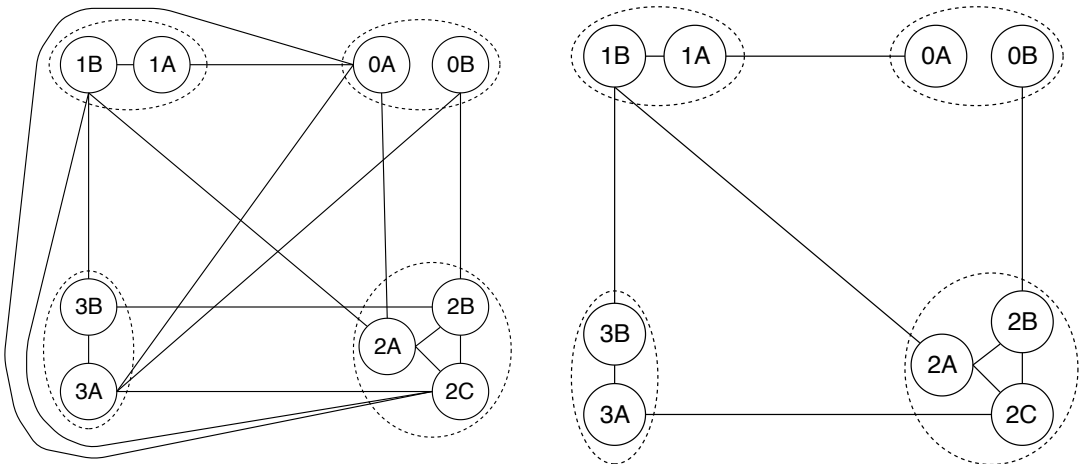


Figure 5.5: Score vs F1 score.

5.4 Real Experiment

During these tests human re-identification was not used as mentioned in the previous chapter and therefore Algorithm 1 on page 36 was used.

Figure 5.6 shows the camera topology after our method has been applied to the data from the cameras. As can be seen in Figure 5.6a 11 links were found after the link evaluation. Figure 5.6b shows the topology after the link refinement has been applied. As can be seen in the figure, the link refinement was able to correctly identify the six weak links and remove them. The transition times from the real experiment can be seen in Table 5.5. As can be seen in the table different transition times were found depending on direction.



(a) The camera topology before link refinement. **(b)** The camera topology after link refinement.

Figure 5.6: The inferred topology in the real experiment.

Table 5.5: Transition times for all links found in link evaluation.

Link	Distance	Time one direction [s]	Time other direction [s]
0A - 1A	long	7.3	7.3
0B - 2B	extra-small	0.0	2.1
1B - 2A	medium	2.2	4.5
1B - 3B	small	1.3	1.6
2C - 3A	medium	5.1	3.3
0A - 2A	-	12.0	14.7
0A - 2C	-	16.8	18.8
0A - 3B	-	11.7	11.6
0B - 3A	-	7.8	7.8
1B - 2C	-	6.9	8.3
2B - 3B	-	17.0	15.0

Chapter 6
Discussion

In this chapter, the outcomes of the tests that were performed are discussed. The answers to the questions in the thesis goal are presented throughout this section. A more general discussion about the entire method is then performed together with suggestions for future work in this area.

6.1 Correspondence Free

The results that we received after testing our correspondence free approach show that it performs well overall as can be seen in Section 5.2 and in Appendix A.1. When the frequency is normal or low, most valid links and few incorrect links are found. An example of a cross correlation from the complex simulation can be seen in Appendix B where there is a clear peak when the frequency is low. When the frequency is higher, then the peak is not as clear since the noise floor is very high. The reason for the noise floor being higher for high frequencies is that there are more false correlations in the cross correlation, because every departure from one zone can be correlated with a higher number of arrivals in the other zone. The results show that the accuracy of the inferred topology is not only dependent on the frequency, but also on the number of people that are in the camera environment. The figures in Section 5.2 show that the method can infer an accurate topology when the frequency is high, if the number of people in the camera environment is high enough. When the frequency is high the amount of false correlations is very high, but when the number of people is high, then the true correlations can form a peak that rises above the noise floor. As can be seen in the figures in Section 5.2 the frequency has a high impact on the accuracy when the number of people is low, but when the number of people increases then the frequency does not affect the result as much. The answer to Question 1 from Section 1.3 is therefore that both the traffic density and the number of people in the camera environment affect the accuracy of the topology.

The results in Section 5.2 show that the correspondence free approach performs better

on the simple simulation than on the complex simulation. This was unexpected, but it can be easily explained. The simple simulation only has a total of five links while the complex simulation has 60 links. This means that in the complex simulation the traffic is divided between more links and therefore each link has fewer people that walk it. An example of this can be seen in Appendix B.2 which shows the cross correlation for two links with length *medium* from both simulations. The complex simulation also has a few links that have low probability to be taken. We did this to simulate certain paths in the camera environment that are not in the main traffic patterns. This resulted in too few people walking those links, so a clear peak could not be formed. The accuracy of the inferred topology is therefore more dependent on how many people that walk each link rather than the total number of people in the environment. It does not matter how many people there are in the environment, if there are few people that walk a certain link it is less likely that it can be found. The answer to Question 2 in Section 1.3 is therefore that the size of a camera network can affect the accuracy of the topology if it results in too few people walking certain paths.

The links that were missed in the complex simulation were either links that had a low probability of being taken, or links that are connected to entry/exit zones that have multiple links connected to them. For example, Camera 6 in the complex simulation, has three links connected to one of its entry/exit zones. If an entry/exit zone has multiple links connected to it then the number of false correlations in the cross correlation increases substantially as can be seen in Appendix B.3.

6.2 Correspondence Based

The results in Section 5.3 and Appendix A.2 show that the accuracy of the topology inference method can be increased with human re-identification. We performed 36 tests with human re-identification with varying accuracy in the complex simulation with different traffic density. The tests show that human re-identification improved the accuracy of the inferred topology in 72% of the test cases. The figures in Section 5.3 show that both the value for true positives and false positives affect the accuracy of the topology but also that the value for false positives has a higher affect. The explanation behind this is that decreasing false positives lowers the amount of false correlations in the cross correlation, as can be seen in Appendix B.4. As is discussed in the previous section, reducing the amount of false correlations leads to a clearer peak in the cross correlation. A human re-identification method with a low probability for false positives will therefore be able to accurately infer the camera topology even when there is high traffic density in the environment. The answer to Question 3 in Section 1.3 is therefore that human re-identification with a high probability for false positives can be worse than using no re-identification. In environments where the traffic flow is normal or low it is not necessary to use human re-identification to accurately infer the topology, as is shown in the previous section. If, however human re-identification is used in those scenarios then it will most often increase the accuracy of the topology. The answer to Question 4 in Section 1.3 is therefore that it is suitable to use human re-identification when the traffic density is high.

Since the probability of false positives affects the accuracy score more than the probability of true positives, the F1 score is not an accurate method for evaluating the perfor-

mance of human re-identification when inferring the topology, as can be seen in Figure 5.5. A higher F1 score is not guaranteed to increase the accuracy of the topology. The F1 score is a harmonic mean between precision and recall meaning that equal weight is given to both. As has been discussed it is more important to only select the relevant objects, i.e. few false positives, and therefore higher weight should be given to precision.

6.3 Link Refinement

Link refinement is an important part of our suggested topology inference method since the link evaluation finds both valid and weak links as can be seen in the tables in Section 5.2. The tables, and those in Appendix A, show that the amount of weak links often is larger than the amount of valid links, especially when the number of people is high and the traffic density is low. The results also show that the number of weak links increases when the number of people increases and decreases when the traffic density increases. The reason for this is the same as why more valid links are found in the same circumstances. The peak in the weak links becomes clearer, the same as for the valid links. The amount of weak links found increases even further when human re-identification is used. Human re-identification lowers the false correlations in the cross correlation, so the noise floor is lower, and the link evaluation will therefore find more weak links.

The incorrect links in Table 5.3 and Table 5.4 show how many weak links were not removed by link refinement. The main reason for why a weak link is not removed in the link refinement is because a valid link is missing. If a valid link is missing, there is no possibility for the link refinement to identify the weak link as a weak link. This scenario can be seen in Figure 6.1 where the valid link between Camera 2 and Camera 3 is not found. Therefore, the link refinement will not be able to find the shortest path from Camera 1 to Camera 3 via Camera 2 and assumes that the link between Camera 1 and Camera 3 is a valid link.

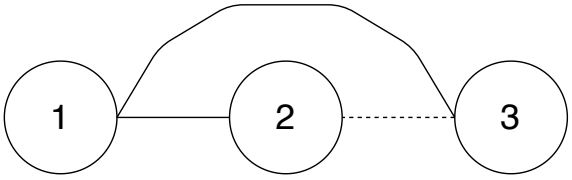


Figure 6.1: Camera topology with one missed link and one weak link.

The results show that there is one scenario where our link refinement will not be able to identify a weak link as a weak link. This occurs when there are multiple paths between two entry/exit zones that have different transition times. Figure 6.2 shows a more detailed view of the links between Camera 7 and Camera 30 from Figure 4.2. The shorter path consisting of Camera 7 ↔ Camera 31 ↔ Camera 30 has a transition time of 7.5s while the longer path consisting of Camera 7 ↔ Camera 5 ↔ Camera 30 has a transition time of 9.7s. In the simulation the longer path has higher probability of being taken, since not all people that enter Camera 31 continue to Camera 30. Therefore, the cross correlation between the two zones is likely to be skewed to the right, as can be seen in Appendix B.5.

The link evaluation will therefore infer a transition time for the weak link that is closer to that of the longer path, than that of the shorter path. When link refinement then evaluates if the link between Camera 7 and Camera 30 is a weak link it compares the transition time of the weak link with that of the shortest path. Since those two transition times are not similar the weak link is falsely believed to be a valid link. A solution to this problem is to compare the transition time of the weak link with that of all possible paths between the two cameras. This approach is however not feasible since finding all possible paths from one node to another in a graph is an NP-hard problem.

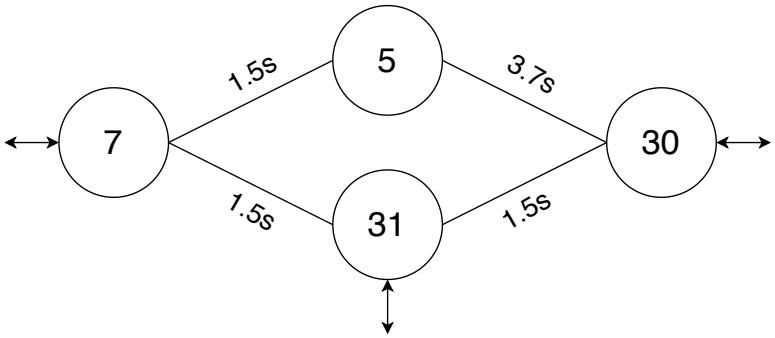


Figure 6.2: The internal link cost in all cameras is 4.5s.

6.4 Real Experiment

The results from the real experiment show that the transition times found between entry/exit zones differed depending on direction, as mentioned before. Since the transition time depends on what direction is walked in the camera environment it would have been more suitable to use directed graphs instead of undirected graphs in the simulations.

There are two explanations for why the transition time is not the same in both directions. The first one is that the two test persons walked with different speeds in the two directions. Although this can happen in other camera environments, for example for cameras that are located in a stairwell, it was not the case in our environment. A more likely explanation is that the internal clocks in the cameras were not synchronized. We parsed the metadata from each individual camera to find when and where a person entered or exited the FOV and it is the internal clock in each camera that is used as timestamp in the metadata. If the internal clocks in two cameras are sufficiently asynchronous this can lead to the link evaluation inferring a negative transition time between the cameras. This only happens if the latency in the clocks, Δt, is larger than the true transition time, τ, between the cameras, i.e. $\tau < \Delta t$. This is especially a problem since our link refinement is based on Dijkstra's shortest path algorithm, and as mentioned in Section 2.3.2 the algorithm does not handle graphs with negative costs. The internal clocks do however not need to be exactly synchronized. As we show in Appendix C there can be differences in the internal clocks without affecting the link refinement, as long as there are no negative transition times.

The real experiment shows the proof of concept of the topology inference method since it was able to accurately infer the camera topology for an actual camera network. The method was able to correctly infer the links in the topology and then eliminate them with

link refinement. Further tests are needed with large and complex camera environments with a high traffic flow before it is possible to say if this method works as well for real networks as for our simulations.

6.5 General Discussion

As we discussed in Section 6.1 the correspondence free approach has problems with finding links for entry/exit zones that have multiple links connected to them. This problem would have been even more prominent if links were found between cameras instead of zones. If the link evaluation is applied to cameras the number of false correlations would be even higher and even lower traffic density would be required to accurately infer the topology. From this follows that it is important to be able to accurately find all entry/exit zones in a camera's FOV. If too few zones are found or if the zones give an inaccurate representation of reality, they will not have as big of a positive effect.

Our results from both the simulations and the real experiment show that the link evaluation can accurately find the transition time between zones, see Appendix B.6. Finding an accurate transition time between zones is a crucial aspect of our method, since the link refinement is based on finding paths in the resulting topology of equal length. If the transition time found is inaccurate then it could result in several weak links in the final topology.

6.6 Future Work

Before our topology inference method can be integrated into a Video Management System, VMS, some additional work and testing must be performed. We have tested our method with a small real camera network with a low traffic density and it performs well in that scenario. The other tests that we did were only performed on simulations and it remains to test if the method performs equally well for a real network in those situations. The experiment that we performed had only two 24-year-old males as test persons, further testing can be done with people of different ages, genders and physical status to evaluate if the diversity of people affects the accuracy.

As mentioned, our method does not consider camera networks that contain cameras with overlapping FOVs. Therefore, future research must be performed to evaluate if it is possible to use our method in a camera network with overlapping FOVs or combine it with another method that handles overlapping FOVs.

The accuracy of our method decreases when the traffic density is high. Future work could consist of an algorithm that detects when the traffic flow is too high in the camera environment and dismisses that data from the camera topology inference method. Such an algorithm would make it possible to run our topology inference method on data that contains high traffic density without reducing the accuracy of the method.

Chapter 7
Conclusion

In this thesis we have examined if it is possible to accurately infer camera network topology using motion data gathered from cameras. Two topology inference approaches were constructed to achieve this. Both approaches follow the same three steps: finding entry/exit zones, link evaluation and link refinement. It is only the link evaluation part that differs between the two approaches, where the correspondence based approach uses human re-identification while the correspondence free does not. To evaluate the accuracy of the approaches two simulations of different sizes were created. The correspondence free approach was also tested in a small real camera network consisting of Axis Communication cameras. The aim of the simulations was to evaluate how the size of the camera network, amount of people in the camera environment and the traffic flow affect the accuracy of the inferred topology.

The results from the tests show that both approaches can accurately infer the camera topology under most circumstances. However, when traffic density becomes high then the correspondence free approach fails but the correspondence based approach is still able to accurately infer the topology. The simulation results further show that if the traffic in the camera environment is widespread, then the size of the camera network does not have a large impact on the accuracy. The results from the real camera network show that it is possible to infer the topology for real camera networks as well without losing accuracy, even if the internal clocks in the cameras are not synchronized. Before the topology inference method can be fully integrated into a video management system, it needs to be tested in a larger real camera network with a more diverse group of test persons.

Bibliography

[1] W. Gravlund, "Vi blir allt mer övervakade av kameror." https://www.svt.se/nyheter/lokalt/helsingborg/vi-blir-allt-mer-overvakade-av-kameror, 2018. Accessed 6 February 2019.

[2] T. J. Ellis, D. Makris, and J. K. Black, "Bridging the gaps between cameras," *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[3] Y. Cho and K. Yoon, "Distance-based camera network topology inference for person re-identification," *Pattern Recognition Letters*, vol. 125, 2019.

[4] Z. Wu, *Human Re-Identification*. Springer, 2016.

[5] A. Gala and S. K. Shah, "A survey of approaches and trends in person re-identification," *Image Vision Comput.*, vol. 32, pp. 270–286, 2014.

[6] X. Zou, B. Bhanu, and A. K. Roy-Chowdhury, "Continuous learning of a multilayered network topology in a video camera network," *EURASIP Journal on Image and Video Processing*, Nov 2009.

[7] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2360–2367, June 2010.

[8] D. Marinakis and G. Dudek, "Topology inference for a vision-based sensor network," in *The 2nd Canadian Conference on Computer and Robot Vision (CRV'05)*, IEEE, May 2005.

[9] K. Tieu, G. Dalley, and W. E. L. Grimson, "Inference of non-overlapping camera network topology by measuring statistical dependence," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, IEEE, October 2005.

[10] R. Farrel and L. S. Davis, "Decentralized discovery of camera network topology," *2008 Second ACM/IEEE International Conference on Distributed Smart Cameras*, 2008.

[11] D. Makris, *Learning an activity-based semantic scene model*. PhD thesis, City University, London, 2004.

[12] X. Chen, K. Huang, and T. Tan, "Learning the three factors of a non-overlapping multi-camera network topology," *Chinese Conference on Pattern Recognition*, pp. 104–112, 2012.

[13] K. Chen, C. Lai, P. Lee, C. Chen, and Y. Hung, "Adaptive learning for target tracking and true linking discovering across multiple non-overlapping cameras," *IEEE Transactions on Multimedia*, vol. 13, no. 4, pp. 625–638, 2011.

[14] O. Masoud and N. P. Papanikolopoulos, "A novel method for tracking and counting pedestrians in real-time using a single camera," *IEEE Transactions on Vehicular Technology*, vol. 50, pp. 1267–1278, Sep. 2001.

[15] A. Gilbert and R. Bowden, "Incremental, scalable tracking of objects inter camera," *Computer Vision and Image Understanding*, vol. 111, pp. 43–58, 07 2008.

[16] S. Chandra and A. Bharti, "Speed distribution curves for pedestrians during walking and crossing," *Procedia - Social and Behavioral Sciences*, vol. 104, 12 2013.

[17] R. Bohannon and A. Andrews, "Normal walking speed: A descriptive meta-analysis," *Physiotherapy*, vol. 97, pp. 182–189, 09 2011.

[18] W. Daamen, *Modelling Passenger Flows in Public Transport Facilities*. PhD thesis, Delft University of Technology, 2004.

[19] J. J. Fruin, "Modelling passenger flows in public transport facilities," *Metropolitan Association of Urban Designers and Environmental Planners*, 1971.

[20] L. F. Henderson, "The statistics of crowd fluids," *Nature*, vol. 229, pp. 381–383, 1971.

[21] L. A. Hoel, "Pedestrian travel rates in central business districts," *Traffic Engineering*, vol. 38, pp. 10–13, 1968.

[22] W. Lam, J. F. Morrall, and H. Ho, "Pedestrian flow characteristics in hong kong," *Transportation Research Record*, vol. 1487, pp. 56–62, 07 1995.

[23] S. J. Older, "Movement of pedestrians on footways in shopping streets," *Traffic Engineering and Control*, vol. 10, no. 4, pp. 160–163, 1968.

[24] P. R. Tregenza, *The Design of Interior Circulation*. Van Nostrand Reinhold Company, 1976.

[25] S. Young, "Evaluation of pedestrian walking speeds in airport terminals," *Transportation Research Record*, vol. 1674, pp. 20–26, 10 1999.

[26] P. Singal and R.S.Chhillar, "Dijkstra shortest path algorithm using global position system," *International Journal of Computer Applications*, vol. 101, pp. 12–18, 09 2014.

[27] J. Moy, "OSPF version 2," *RFC 2178*, July 1997.

[28] D. Reynolds, *Gaussian Mixture Models*, pp. 827–832. Boston, MA: Springer US, 2015.

[29] A. Dempster and D. R. N. Laird, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society*, vol. 38, no. 1, pp. 1–38, 1997.

[30] J. Blimes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126, 1998.

[31] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, pp. 47–60, Nov 1996.

[32] D. L. WEAKLIEM, "A critique of the bayesian information criterion for model selection," *Sociological Methods & Research*, vol. 27, no. 3, pp. 359–397, 1999.

[33] C. Biernacki, G. Celeux, and G. Govaert, "Assessing a mixture model for clustering with the integrated completed likelihood," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 719–725, July 2000.

[34] A. E. Raftery and C. Fraley, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis," *The Computer Journal*, vol. 41, pp. 578–588, 01 1998.

[35] M. Bennewitz, W. Burgard, and S. Thrun, "Using EM to learn motion behaviors of persons with mobile robots," vol. 1, pp. 502–507, Sep. 2002.

Appendices

Appendix A
Extra Results

A.1 Correspondence Free

A.1.1 Simple Simulation

Table A.1: Results from the simple simulation, $f = 0.01$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	0	0	5	1.00
1000	0	0	10	1.00
2000	0	0	15	1.00
4000	0	0	18	1.00
8000	0	0	20	1.00
16000	0	0	25	1.00

Table A.2: Results from the simple simulation, $f = 1$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	2	1	3	0.00
1000	2	0	0	0.20
2000	2	0	0	0.20
4000	1	0	0	0.60
8000	1	0	1	0.60
16000	0	0	1	1.00

A.1.2 Complex Simulation

Table A.3: Results from the complex simulation, $f = 0.01$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	15	0	24	0.50
1000	4	1	110	0.85
2000	0	1	215	0.98
4000	0	1	303	0.98
8000	0	1	411	0.98
16000	0	1	533	0.98

Table A.4: Results from the complex simulation, $f = 1$.

Number of People	Missed Links	Incorrect Links	Weak Links Removed	Score
500	23	21	13	-0.12
1000	17	9	5	0.28
2000	15	2	3	0.47
4000	14	2	5	0.50
8000	9	1	9	0.68
16000	4	0	19	0.87

A.2 Correspondence Based

A.2.1 Simple Simulation

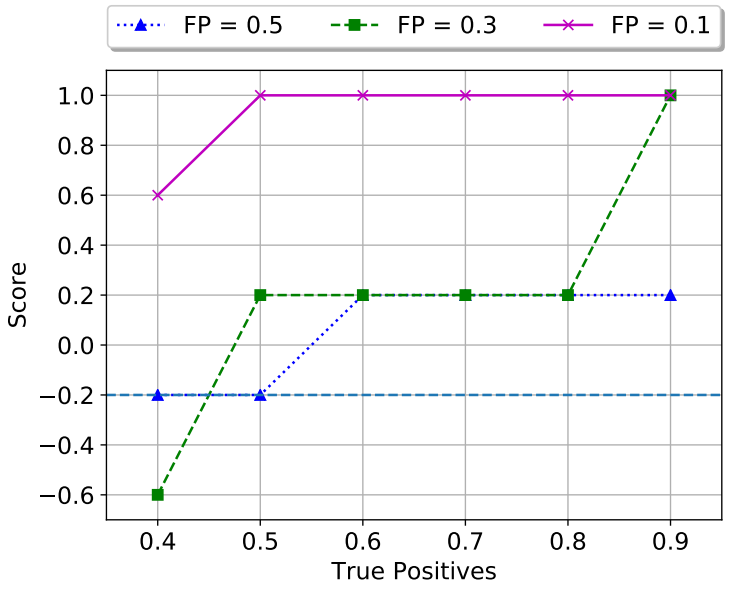


Figure A.1: Can human re-id increase the accuracy?

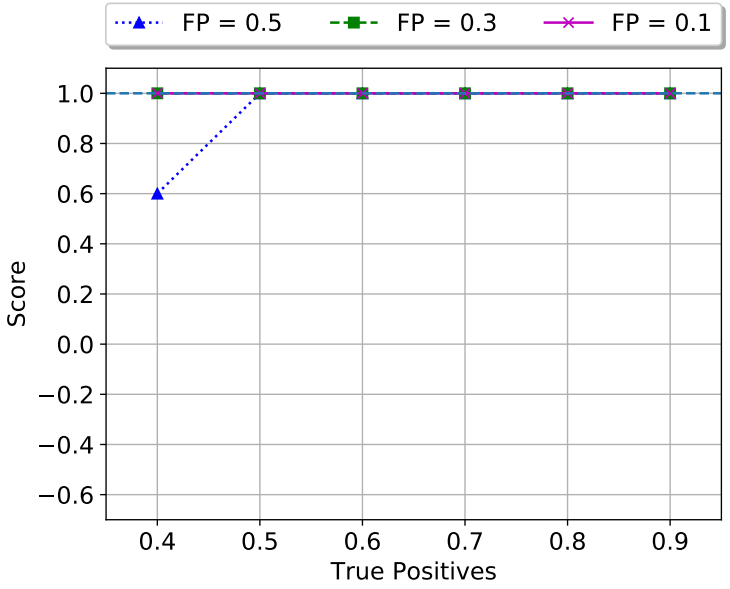


Figure A.2: Can human re-id decrease the accuracy?

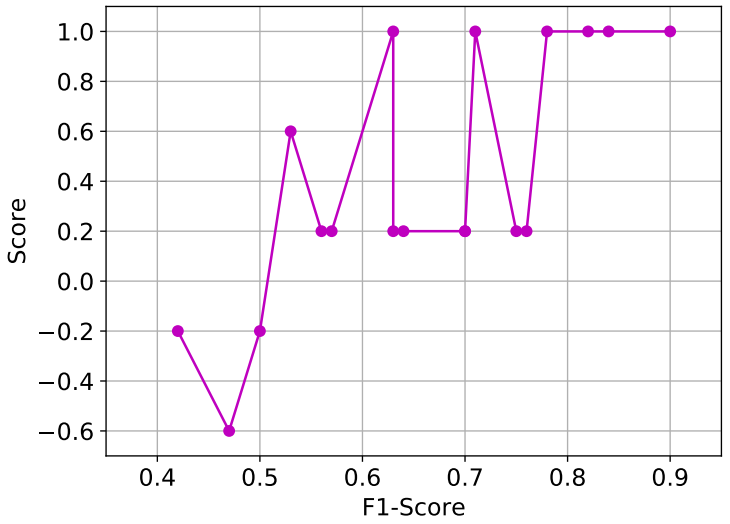
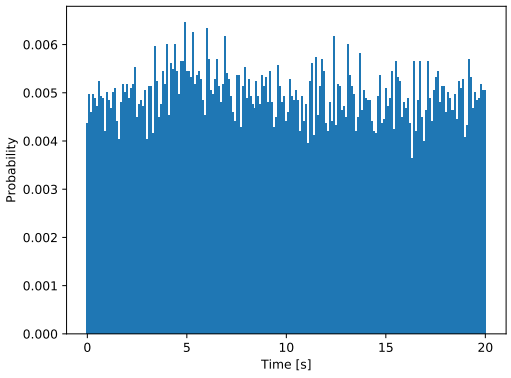


Figure A.3: Score vs F1 score.

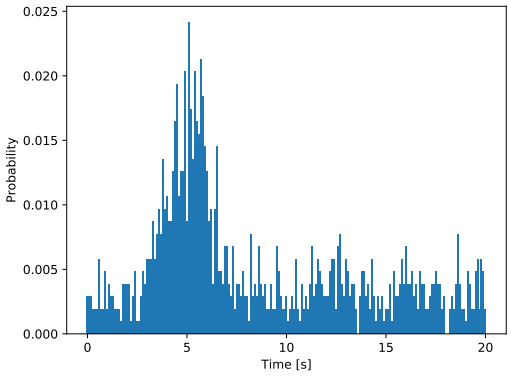
Appendix B
Cross correlation

B.1 The Effects of Traffic Density

The cross correlations in Figure B.1 and B.2 shows the cross correlation for the link 8 – 31 in the complex simulation from Figure 4.2. Since the noise floor is so high in Figure B.1a and B.2a, we also plot the same cross correlation, but we have lowered the noise to make the upper part of the cross correlation more clear which can be seen in Figure B.3.

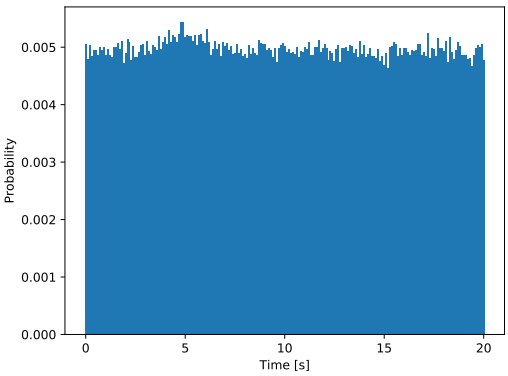


(a) Cross correlation with $f = 1$.

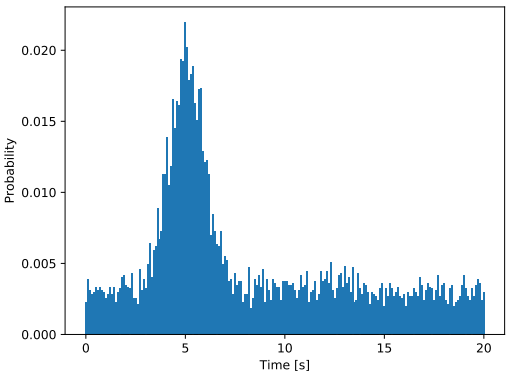


(b) Cross correlation with $f = 0.01$.

Figure B.1: $Number\ of\ people = 2000$ for both cross correlations.

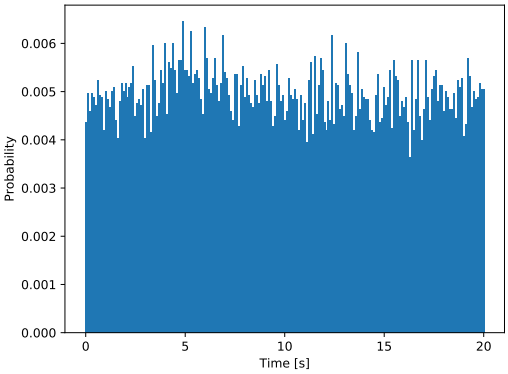


(a) Cross correlation with $f = 1$.

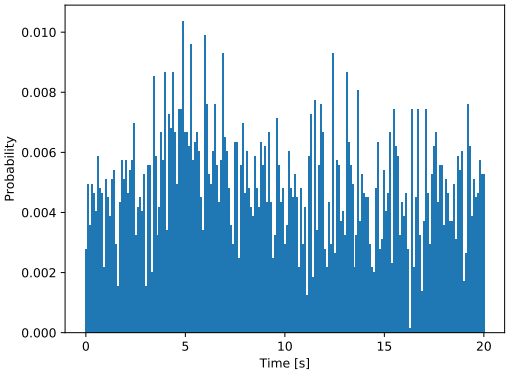


(b) Cross correlation with $f = 0.01$.

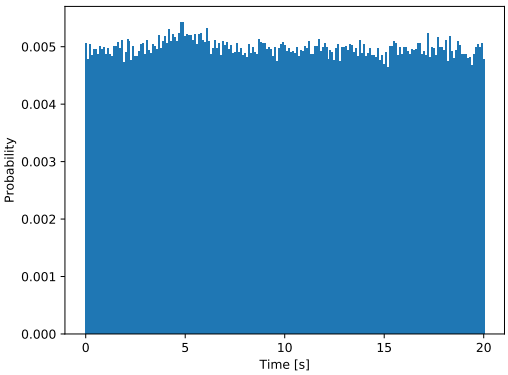
Figure B.2: $Number\ of\ people = 16000$ for both cross correlations.



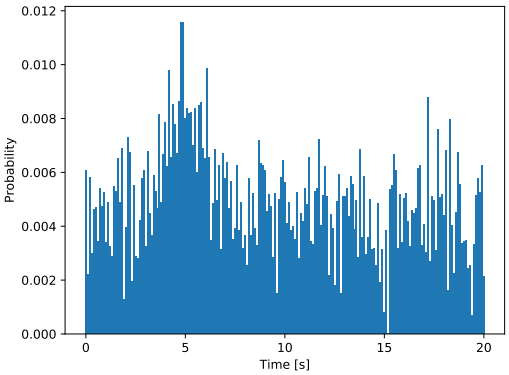
(a) Cross correlation with $number\ of\ people = 2000$.



(b) Cross correlation with $number\ of\ people = 2000$.



(c) Cross correlation with $number\ of\ people = 16000$.

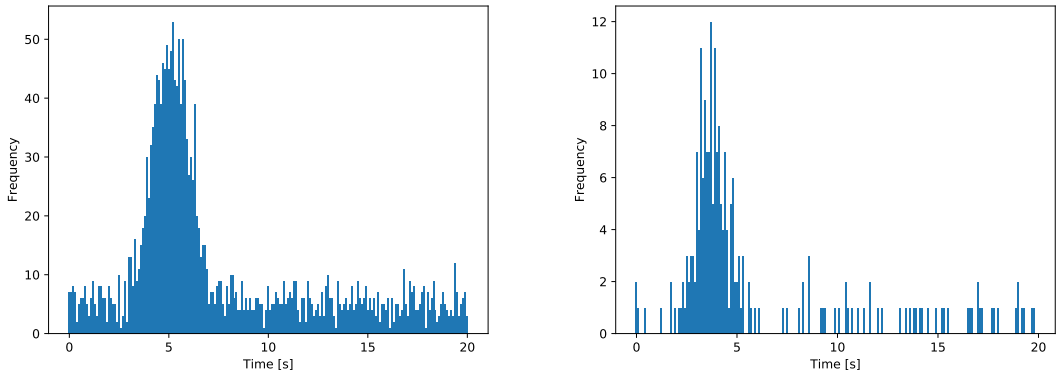


(d) Cross correlation with $number\ of\ people = 16000$.

Figure B.3: $f = 1$ for both cross correlations.

B.2 The Effects of Network Size

In the simple simulation there are more people that walk each link since there are fewer links in the simulation. This can be seen in Figure B.4 where the frequency is much higher on the link from the simple simulation.



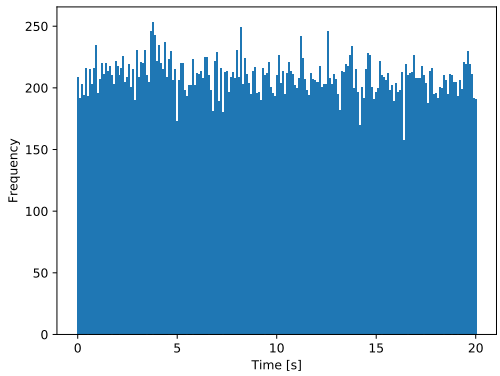
(a) Simple simulation.

(b) Complex simulation.

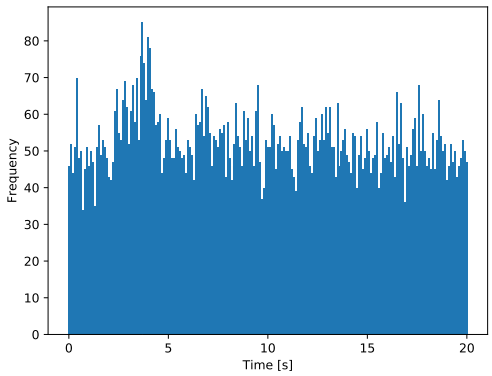
Figure B.4: Cross correlations of a *medium* link when $f = 0.01$ and $number\ of\ people = 2000$.

B.3 The Effects of Multiple Links

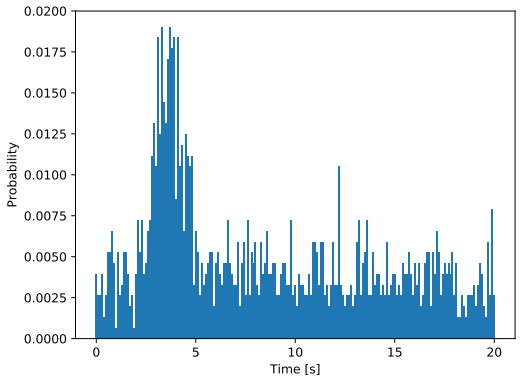
Figure B.5 shows the cross correlations for the link 6 – 10 in the complex simulation. The entry/exit zones in both cameras have multiple links connected to them which increases the amount of false correlations and increases the noise. As can be seen in Figure B.5a there is no clear peak when the frequency is high, but the peak becomes clearer when the frequency lowers, as can be seen in Figures B.5b and B.5c. Our link evaluation is correctly able to identify the cross correlations in Figure B.5b and B.5c as a valid link.



(a) Cross correlation with $f = 1$.



(b) Cross correlation with $f = 0.1$.

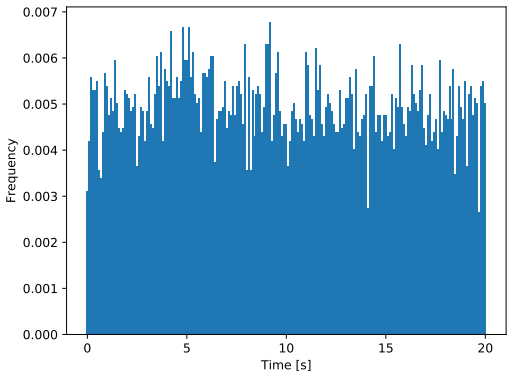


(c) Cross correlation with $f = 0.01$.

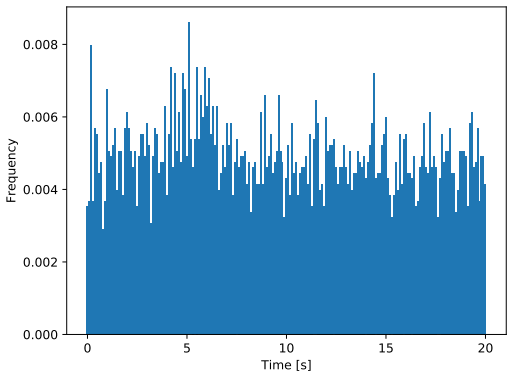
Figure B.5: $Number of people = 2000$ in all cross correlations.

B.4 The Effects of Human Re-identification

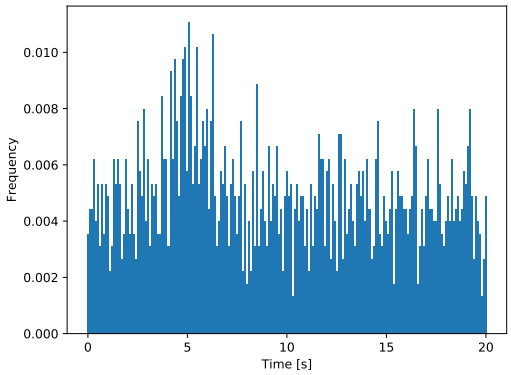
Figure B.6 shows three cross correlations for a link with equal probability of true positives and with different probabilities for false positives. As can be seen in the figure, the noise is reduced when the probability of false positives decreases and the peak becomes clearer.



(a) Cross correlation with $FP = 0.5$.



(b) Cross correlation with $FP = 0.3$.



(c) Cross correlation with $FP = 0.1$.

Figure B.6: $Number\ of\ people = 2000$ and $TP = 0.6$ in all cross correlations.

B.5 The Effects of Multiple Paths

Figure B.7 shows the cross correlation between Camera 7 and Camera 30 in the complex simulation. The cross correlation is skewed to the right since the path via Camera 5 is more probable.

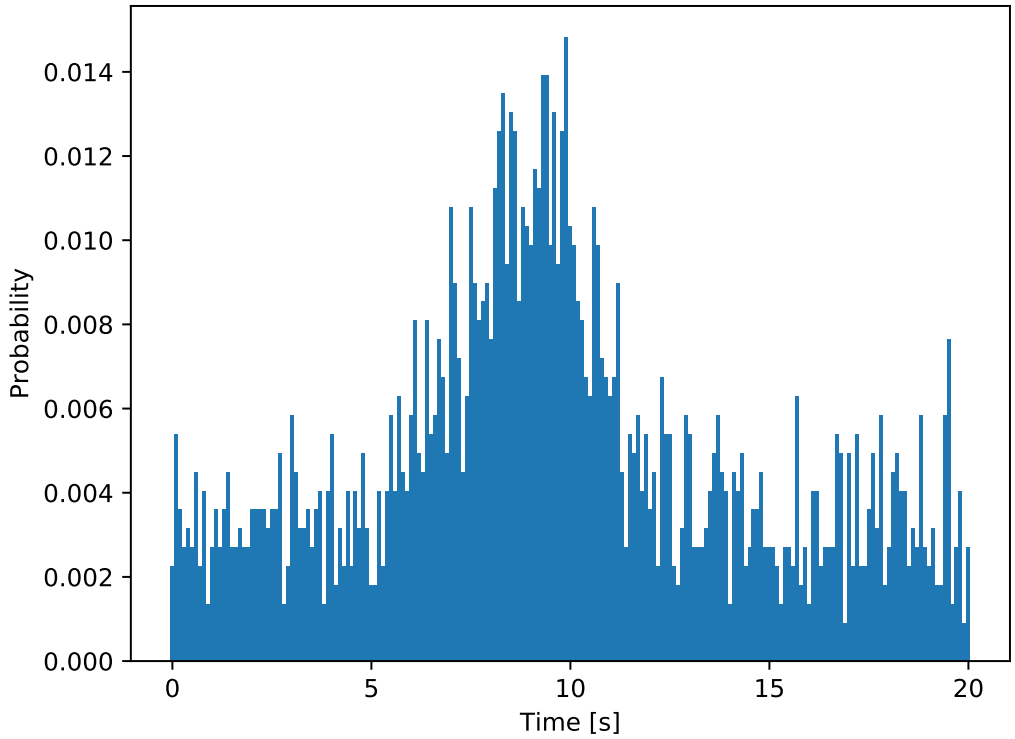


Figure B.7: $Number\ of\ people = 2000$ and $f = 0.01$.

B.6 Link Evaluation Accuracy

Figures B.8 and B.9 show that it is not always obvious from the cross correlations what the true transition time is. The true transition time is often not the point with the highest probability, which is true for all the cross correlations in Figures B.8 and B.9. Figure B.8 shows the cross correlation of link $0A - 1A$ from the real experiment. The link has a true transition time of 7s but that is not clear from the figure. The time with the highest probability is 7.9s, but our link evaluation selects 7.3s as the transition time. Although the link evaluation does not find the true transition time it finds a value that is very close. The cross correlations in Figure B.9 are from valid links in the complex simulation. In Figure B.9a the true transition time is 5.1s, the time with the highest probability is 4.7s but our link evaluation selects 5.2s as transition time. Figure B.9b shows the same link as Figure B.9a but this time 6.1s has the highest probability but our method selects 5.3s as the transition time. The cross correlation in Figure B.9c shows a link that has the true transition time of 3.7s, the time with the highest probability is 4.3s but the link evaluation selects 3.8s as transition time. Figure B.9d shows the cross correlation of a link with the true transition time when the traffic density is high. In the figure the time 10.4s has the highest probability from pure chance. Our method is however able to select 3.9s as the transition time. All of these examples show the efficiency of our method.

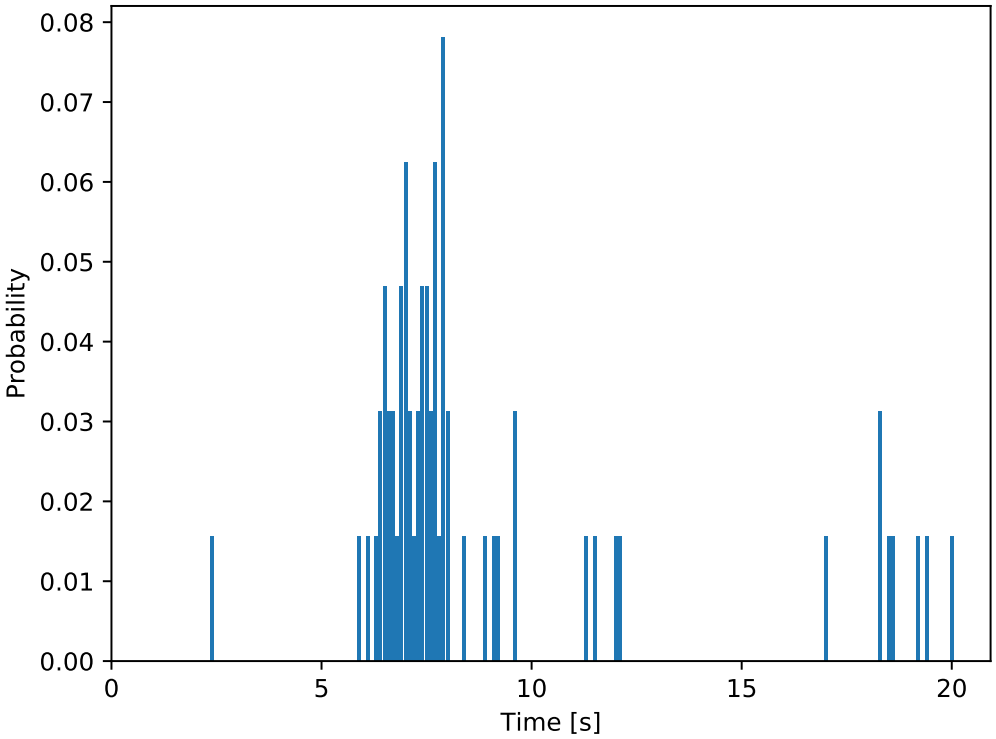
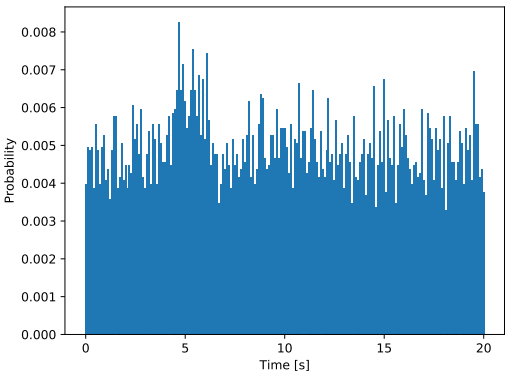
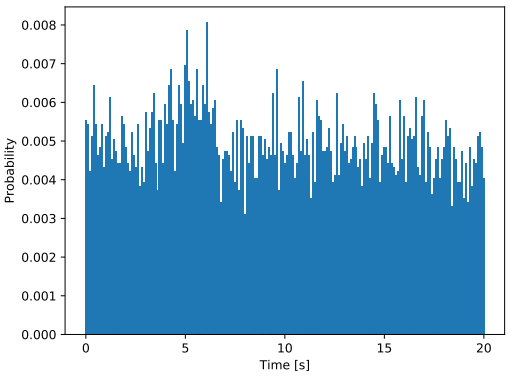


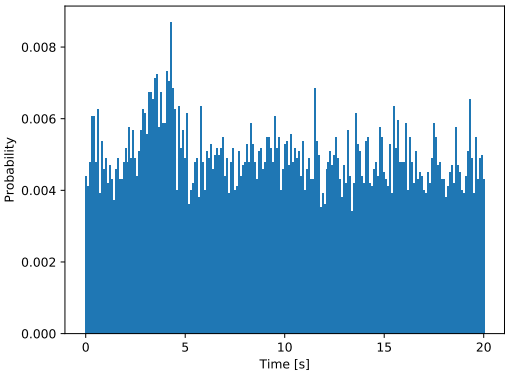
Figure B.8: Link 0A - 1A.



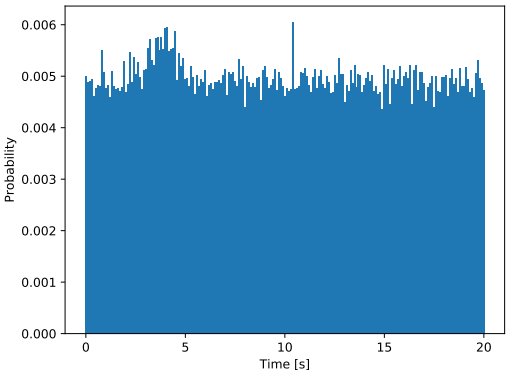
(a) Link 6 - 11.



(b) Link 6 - 11.



(c) Link 6 - 10.



(d) Link 2 - 7.

Figure B.9: Four cross correlations that show the efficiency of our link evaluation.

Appendix C
Link Refinement with Clock Latency

In this appendix we will show that our link refinement method does work even when there is a small latency between the internal clocks of the cameras in the network. Figure C.1 shows an example of a camera topology with four cameras. The topology shown in Figure C.1 could represent an entire camera network or just a small part of a larger topology. In the graph c represents the transition time of the links and t represents the time in the cameras. The internal links, i.e. the time it takes to move inside a cameras FOV, can be ignored since it is the same internal clock that sets the time when a person enters and exits the FOV and therefore there is no latency between those two times. The proof below shows that latency between the cameras internal clocks does not affect the identification of a weak link. In the proof the cost cl represents the transition time between cameras accounted for latency. As the proof shows time latency cancels each other out so it does not affect the link refinement. However, as is mentioned in Section 6.4 the link refinement will not work if cl is negative.

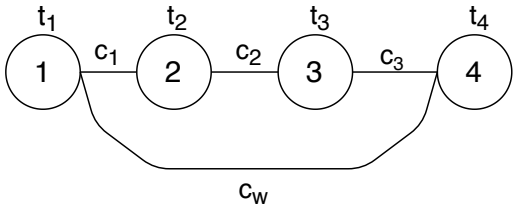


Figure C.1: A camera network topology represented as a weighted graph.

Proof. Let $cl_1 = \Delta t_{1,2} + c_1$, $cl_2 = \Delta t_{2,3} + c_2$, $cl_3 = \Delta t_{3,4} + c_3$ and $cl_w = \Delta t_{4,1} + c_w$ where $\Delta t_{1,2} = t_2 - t_1$, $\Delta t_{2,3} = t_3 - t_2$, $\Delta t_{4,3} = t_4 - t_3$ and $\Delta t_{4,1} = t_4 - t_1$ So,

$$
\begin{aligned}
cl_w &\approx \sum_{v=1}^{3} cl_v \Leftrightarrow \\
\Delta t_{4,1} + c_w &\approx \Delta t_{1,2} + c_1 + \Delta t_{2,3} + c_2 + \Delta t_{3,4} + c_3 \Leftrightarrow \\
(t_4 - t_1) + c_w &\approx (t_2 - t_1) + c_1 + (t_3 - t_2) + c_2 + (t_4 - t_3) + c_3 \Leftrightarrow \\
c_w &\approx c_1 + c_2 + c_3
\end{aligned}
$$

□

INSTITUTIONEN FÖR DATAVETENSKAP | LUNDS TEKNISKA HÖGSKOLA | PRESENTERAD 2019-06-14

EXAMENSARBETE Topology Inference for Non-Overlapping Camera Networks
STUDENTER Anton Thelandersson, Ólafur Már Óskarsson
HANDLEDARE Elin Anna Topp (LTH), Viktor Andersson (Axis), Anders Krüger (Axis)
EXAMINATOR Volker Krüger (LTH)

Automatisk generering av relationer mellan kamerors synfält

POPULÄRVETENSKAPLIG SAMMANFATTNING **Anton Thelandersson, Ólafur Már Óskarsson**

Storleken på övervakningssystem ökar och därmed även svårigheten att förutspå var en människa kommer befinna sig inom övervakningssystemet. Detta arbete har gått ut på att skapa en topologi bestående av relationer mellan kameror för att enklare kunna förutspå var en person kommer gå.

Axis Communications är ett världsledande företag inom övervakningssystem. Axis Communications har märkt att storleken på kundernas övervakningssytemen ökar. De ville därför undersöka möjligheten för att automatiskt hitta relationer mellan kameror eller mer specifikt, mellan olika zoner i kamerorna. Kameror har ett synfält och personer kan gå in eller ut ur dessa synfält på flera olika ställen, så kallade zoner. Kamerasystemen som utvärderades bestod av kameror som inte hade överlappande synfält. Med detta menas att en person endast kan synas i en kameras synfält åt gången.

Det specifika användningsfallet som användes som grund i detta arbete var en mataffär där avstånden mellan kameror är korta. Att avstånden är korta gör att metoden som används måste vara precis för att inte återskapa felaktiga relationer. Vårt mål är att kameror endast skall ha relationer med sina närmaste grannar i färdriktningen. En relation som inte är mellan två närliggande kameror är en så kallad svag relation. Vårt mål är att vår metod inte skall föreslå svaga relationer.

Metoden som utvecklades hade två olika användningsfall. Den ena metoden använde sig av igenkänning för att evaluera relationer medans den andra inte tog någon hänsyn till vem som gick ut eller in i en kameras synfält. Våra resultat visar att det fungerar väldigt bra att återskapa topologin genom att inte använda sig av igenkänning om trafiken i övervakningssystemet inte är för högt. Detta gör det enklare att integrera vår lösning i befintliga system då det inte behövs någon metod för att matcha personer.

Metoden som utvecklades testades på två olika simuleringar samt på ett riktigt nätverk. Resultatet visar att vår metod kunde noggrant återskapa topologin vid både simuleringar och i det riktiga systemet. Det riktiga systemet bestod dock endast av fyra kameror så innan vår metod kan bli integrerad till ett befintligt system krävs utförligare tester på större riktiga system.