Diploma Thesis IST-77

Distributed multi-camera synchronization for smart-intruder detection

Markus Spindler

Supervisor: Fabio Pasqualetti
Francesco Bullo

University of Stuttgart
Institute for Systems Theory and Automatic Control
Prof. Dr.–Ing. F. Allgöwer

10. September 2011

Contents

1 Introduction **9**
- 1.1 Problem description 9
- 1.2 Related work 10
- 1.3 Contribution 11
- 1.4 Organization 12

2 Problem formulation **13**

3 Main results **17**

4 An illustrative example **23**
- 4.1 Numerical verification of the performance bound . . . 23
- 4.2 Distributed feedback algorithm 25

5 Proof of Theorem 4.0.1 **29**
- 5.1 Properties of an optimal trajectory 29
- 5.2 Performance bounds 34

6 Partial homogenous field of view **53**
- 6.1 Parameterized model 54
- 6.2 Cost function 55

7 Conclusion **59**
- 7.1 Summary 59
- 7.2 Outlook 60

8 Appendix **61**
8.1 Method of Lagrange multipliers 61

List of Figures

1.1 Setup example. Five cameras ($c_1 \dots c_5$) are installed along an open path Γ, which is partitioned into 5 non overlapping clusters. Each camera observes its dedicated cluster, which is described with the extremes l_i and r_i and the length d_i. The dots on the clusters represent the point f.o.v. of the cameras. 10

2.1 Camera model. In the left figure the installation of the camera is illustrated and in the right figure the more detailed description of the f.o.v. and its reduction is shown. 14

2.2 Synchronization versus no synchronization. Illustrated is the movement of 3 cameras. The smart intruder e_1 between the synchronized cameras 1 and 2 get detected latest at the time when the f.o.v. of camera 1 and camera 2 are at the same position (black dot). Whereas, a smart intruder e_2 between the not synchronized cameras 2 and 3 gets never detected. 16

3.1 Equal-waiting trajectories. Illustration of an equal-waiting trajectory, formally described in Trajectory 1, for an $n = 4$ cameras example. The black dots illustrate the points where two f.o.v. meet or a f.o.v. reaches a wall. 19

4.1 Simulation of bound $(3+\sqrt{n})/4$. The red dots illustrate the calculated performance bound $(3+\sqrt{n})/4$ of Theorem 4.0.2, and each of the blue dots illustrates a calculated ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$ for a specific setup, which is specified in the subfigures 24

4.2 Simulation of bound $1/2 + d_{\max}/2d_{\min}$. The green dots illustrate the calculated performance bound $1/2+ d_{\max}/2d_{\min}$ of Theorem 4.0.2, and each of the blue dots illustrates a calculated ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$ for a setup described by $n = 50$, $d_1 = 1$ and d_i with $i = 2, \ldots, 50$ is uniformly distributed within interval $[\frac{d_{\min}}{d_{\max}}, 1]$. 25

4.3 Simulation of Algorithm 2. Shown is the initialization procedure from random starting positions between time $t = [0, 150]$, a temporary camera failure of camera 4 between $t = [340, 440]$, and how the algorithm works under the influence of noise ($t > 700$). 27

5.1 Performance calculation. Illustration of the performance separation into a right and left part, ADT_i^r and ADT_i^l respectively, of the trajectory. The black line illustrates the trajectory $x_i(t)$, the black dots at time t_1, t_2, t_3 catch-points, and the red dot at t_0, p_0 an intruder. The smart intruder at t_0, p_0, like all other intruders on the left side of $x_i(t)$ at this time, gets detected at t_3, whereas intruders appearing at time t_0 left of $x_i(t)$ get detected at t_2 32

5.2 Possible trajectory shapes. The figures illustrate the trajectories between two catch-points t_j and t_{j+1}. In figures a), b) and c) $\theta_i^l(t) - \theta_i^r(t) > 0$, therefore $x_i(t)$ is as small as possible. In figures d), e), and f) $\theta_i^l(t) - \theta_i^r(t) < 0$, here the trajectories are such that $x_i(t)$ is as big as possible. All trajectories are unique, because they are obtained with camera speed either zero or maximum. 33

5.3 Sequence of catch-points. Case 1 and case 2 illustrate the two possible cases of $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}^l$. The yellow cases in row a) show trajectories with three consecutive catch-points on one boundary. The blue cases in row b) show, how these trajectories can be improved. 35
5.4 Parameterized lower bound model. Fig a) shows the parameterized model for a single cluster. Fig b) explains why the catch-points on opposite boundaries must be in distance d_i. 39
5.5 Parameterized model for a section f. The model shows the catch-points (black dots), whose position is described by parameters, and the trajectory between the catch-points. 40
5.6 Performance calculation of the equal-waiting trajectory for a cluster d_i. The bold black line illustrates the trajectory and the black dots the catch-points. . . 49

6.1 Partial homogenous field of view model. The model of 4 cameras with a partial homogenous field of view, that means $d_{\max}/d_{\min} < 2$, is illustrated. The color boxes illustrate a period for each cluster, and the black dots the catch-points of the cameras. 55

Chapter 1

Introduction

Coordinated teams of autonomous agents have recently been used for many tasks requiring repetitive execution, including the monitoring of oil spills [1], the detection of forest fires [2], the track of border changes [3], and the patrol (surveillance) of an environment [4]. Especially the use of autonomous systems for surveillance is getting more and more interesting, since the systems got much better and cheaper in recent years. The surveillance of an area of interest requires the agents to continuously and repeatedly sweep the environment, and the challenging problem consists in scheduling the agents trajectories so as to optimize a certain performance criteria.

1.1 Problem description

In this work we consider a network of identical Pan-Tilt-Zoom (PTZ) cameras for videosurveillance, and we focus on the development of distributed and autonomous surveillance strategies for the detection of moving intruders. We make the following combined assumptions on the environment to be monitored, the cameras, and the intruders. We assume the environment to be one dimensional, in the sense that it can be completely observed by a chain of cameras by using the panning motion only. The problem of perimeter surveillance is a special case of this framework. We assume the cameras to be subject to physical constraints, e.g., limited field of view (f.o.v.) and panning speed,

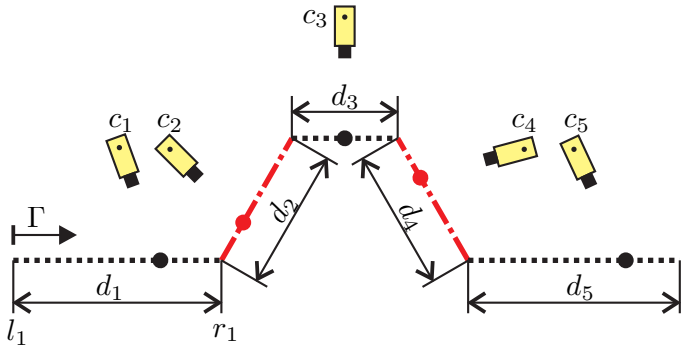


Figure 1.1: Setup example. Five cameras ($c_1 \dots c_5$) are installed along an open path Γ, which is partitioned into 5 non overlapping clusters. Each camera observes its dedicated cluster, which is described with the extremes l_i and r_i and the length d_i. The dots on the clusters represent the point f.o.v. of the cameras.

and to be equipped with a low-level routine to detect intruders that fall within the f.o.v. of a camera. Regarding intruders, we assume them to be *smart*, in the sense that they have access to the cameras configuration at every time instant, and schedule their trajectory to avoid detection, if possible. We study the problem of scheduling the cameras trajectory as to minimize the *worst-case detection time* and the *average detection time* of a smart intruder.

1.2 Related work

The problem of patrolling an environment by means of a team of autonomous robots has received attention from scientists interested in mobile robotics. Typically, (i) the environment is represented by a graph on which the agents motion is constrained, and (ii) the patrolling performance is given by the *worst-case* detection time of a

static event. In [5, 6] an empirical evaluation of certain patrolling heuristics is performed. In [4] and [2], an efficient and distributed solution to the (worst-case) perimeter patrolling problem for robots with zero communication range is proposed. In [7] the computational complexity of the patrolling problem is studied as a function of the environment topology, and optimal strategies as well as constant-factor approximations are proposed. With respect to these works, we consider smart intruders, as opposed to static ones, and we focus on the worst-case and average detection time, as opposed to only the worst-case detection time.

In the context of camera networks, the perimeter patrolling problem has recently been studied in [8, 9]. In these works, distributed algorithms are proposed for the cameras to partition a one-dimensional environment, and to synchronize along a trajectory with minimum worst-case detection time of static events. We improve the results along this direction by showing that the strategies proposed in [8, 9] generally fail at detecting smart intruders, and by focusing on the average detection time of smart intruders while preserve the optimal worst-case detection time.

1.3 Contribution

The contribution of this work consists of four parts. First, we mathematically formalize the concept of cameras trajectory and smart intruder. Then, we prove a lower bound for the worst-case and average detection time of smart intruders. Second, we propose a cameras trajectory, called *equal-waiting* trajectory, with minimum worst-case detection time and constant-factor optimal average detection time. Third, we develop a distributed synchronization algorithm to steer the cameras towards an equal-waiting trajectory, for which only neighboring cameras need to communicate when they reach the end of their observed cluster. This algorithm converges in finite time, which we characterize, and it requires only minimal information to be implemented. Moreover, we perform a simulation study to show that our synchronization algorithm is robust against cameras failure and motion uncertainties. Fourth and finally, we investigate a special case, which has restrictions on the observed cluster lengths. For

this case we derive a parameterized model and cost function, which minimization leads to the optimal cameras trajectory with respect to the average detection time criteria.

1.4 Organization

The remainder of this work is organized as follows. In Section 2 we introduce our notation and we formulate the considered problem. In Section 3 we present and characterize our two main results, the equal-waiting trajectory and the synchronization algorithm. Section 4 contains an illustrative example for the stated constant factor approximation and the proposed distributed feedback algorithm. In the next section, Section 5, a proof of our theoretical results follow. In section 6 we present for a special case a parameterized model and cost function. Finally, our conclusion and an outlook of the work is found in Section 7.

Chapter 2

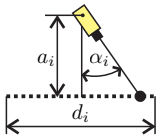
Problem formulation

Consider a set of $n \in \mathbb{N}$ identical cameras installed along a one dimensional open path Γ (cf. Fig 1.1) in distance a_i, $i = 1, \ldots, n$, from the path. Assume that (i) the field of view (f.o.v.) of each camera is a point on Γ, and that (ii) the motion of each f.o.v. is uniquely determined by the pan movement of the corresponding camera. To obtain a sufficient image quality, which allows to process the images, the movement of the f.o.v. along the path must never exceed a maximum speed $\dot{x}_{\max}$. This is obtained by controlling the movement with the control law $\dot{\alpha_i} \leq \dot{x}_{\max}/(a_i(\tan \alpha_i)^2 + a_i)$, where α_i and $\dot{\alpha_i}$ are the pan angel and pan movement of camera i respectively (cf. Fig 2.1(a)).

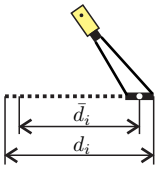
Let Γ_0 and Γ_f be the two extremes of Γ. For simplicity, we label the cameras in increasing order from c_1 to c_n according to their distance from Γ_0 on Γ. Let $l_i \leq x_i(t) \leq r_i$, where l_i and r_i are points of Γ, and $\dot{x}_i(t) \in [-1, 1]$ denote the position of f.o.v. of camera i and its velocity at time $t \in \mathbb{R}_{\geq 0}$, respectively. We refer to $s_i = [l_i, r_i]$ as to the i-th cluster, and we let d_i be the distance on Γ between l_i and r_i. We additionally assume that $l_i = r_{i-1}$, with $i = 2, \ldots, n$, so that the clusters are a partition of Γ. A *(cameras) trajectory* is an array $X(t) = \{x_1(t), \ldots, x_n(t)\}$ of n continuous and *periodic* functions describing the motion of the cameras f.o.v. on Γ. Notice that, for each cameras trajectory, there exists $T \in \mathbb{R}_{\geq 0}$ such that $X(t + T) = X(t)$, i.e., $x_i(t + T) = x_i(t)$ for $i \in \{1, \ldots, n\}$.

The assumption, that the f.o.v. of the cameras is a point is not

valid for actual cameras. If looking at a one dimensional path, an real camera has always a f.o.v. that has a dimension along the path and therefore can be described by a line with a midpoint. Considering the f.o.v. as a line with a midpoint and just observing the motion of the midpoint for a modified cluster length (cf. Fig 2.1(b)) the same result as a point f.o.v. is obtained. Therefore, we can consider the point f.o.v without loosing generality.



(a) The camera is installed in distance a_i from the path and observes the cluster at an angle α_i.



(b) Reduction of a line f.o.v. to a point f.o.v. by only observing the midpoint (white dot on the line) and changing the cluster length from d_i to $\bar{d}_i$.

Figure 2.1: Camera model. In the left figure the installation of the camera is illustrated and in the right figure the more detailed description of the f.o.v. and its reduction is shown.

In this work, we focus on the problem of detecting moving objects, referred here to as *intruders*, by means of a set of cameras. We consider the case of intruders with speed greater or equal than the cameras speed. We represent the intruder as a point on Γ, and we let $t_0 \in \mathbb{R}_{\geq 0}$ be the time at which the intruder appears on Γ. Moreover, we let the continuous map $p : \mathbb{R}_{\geq t_0} \mapsto \Gamma$ describe the position of the intruder at a certain time $t \geq t_0$. We say that an intruder is detected at time $t_d \in \mathbb{R}_{\geq t_0}$ if $p(t_d) \in X(t_d)$. We focus on *smart* intruders, which have full knowledge of the cameras trajectory and choose their trajectory $p(t)$ to avoid detection as long as possible. More formally, given a time $t_0 \in \mathbb{R}_{\geq 0}$, a point $p_0 \in \Gamma$ and a cameras trajectory X, the trajectory of a smart intruder $p^*_{t_0,p_0}(t)$ is such that

$$p^*_{t_0,p_0}(t) = \arg\max\{t^*_d(p) - t_0 \mid p \in \Phi(t_0, p_0)\},$$

where $\Phi(t_0, p_0)$ is the set of continuous maps $p : \mathbb{R}_{\geq t_0} \mapsto \Gamma$ with $p(0) = p_0$, and

$$t_d^*(p) = \min\{t \mid t \geq t_0, p(t) \in X(t)\}.$$

Notice that the trajectory $p^*_{t_0,p_0}(t)$ is, in general, not unique. In the following sections, we design cameras trajectories that minimize *the average detection time* ADT of a smart intruder. In particular, we define the performance of a cameras trajectory as

$$\mathrm{ADT}(X(t)) = \frac{1}{TL}\int_0^T \int_\Gamma (t_d^*(p^*_{\tau,\gamma}) - \tau) d\gamma \, d\tau, \tag{2.1}$$

where $T = 2\max\{d_1, \ldots, d_n\}$, and $L = \sum_{i=1}^{n} d_i$. We consider the following problem.

Problem 1 (Trajectory design)**.** *For a set of n cameras on an open path, design a cameras trajectory $X^*(t)$ such that*

$$X^*(t) = \arg\min_{X(t)} \ \mathrm{ADT}(X(t)).$$

With the above definition of $X^*(t)$, we let

$$\mathrm{ADT}^* = \mathrm{ADT}(X^*(t)).$$

Finally, we say that a cameras trajectory is *synchronized*, if, for each pair of neighboring cameras c_i and c_{i+1}, there exists $t \in [0, T]$ such that $x_i(t) = x_{i+1}(t)$, where T denotes the periodicity of $X(t)$.

Remark 1 (Worst-case detection time)**.** *For a cameras trajectory $X(t)$, the worst-case detection time of a smart intruder is given by* $\mathrm{WDT}(X(t)) = \max_{p_0,t_0} t_d^*(p) - t_0$. *It can be shown that the minimum value of* $\mathrm{WDT}(X(t))$ *equals* $2d_{\max}$*, where* $d_{\max}$ *is the largest among the clusters lengths [7, 8]. Observe that any* $2d_{\max}$*-periodic synchronized cameras trajectory attains minimum worst-case detection time of smart intruders (cf. Fig. 2.2). In the next section we propose a particular* $2d_{\max}$*-periodic synchronized cameras trajectory as solution to Problem 1. Hence, our proposed trajectory is also optimal with respect to the worst-case detection time of smart intruders. Finally note that, for the trajectories described in [8], the worst-case detection time of a smart intruder is unbounded (cf. Fig. 2.2).*

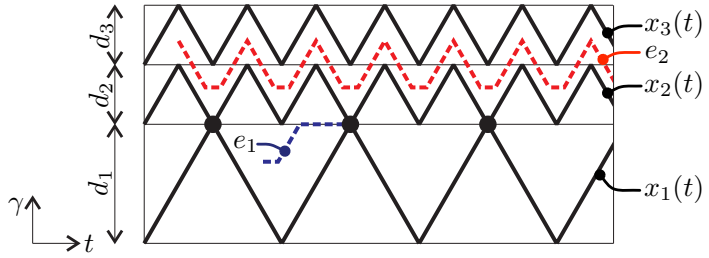


Figure 2.2: Synchronization versus no synchronization. Illustrated is the movement of 3 cameras. The smart intruder e_1 between the synchronized cameras 1 and 2 get detected latest at the time when the f.o.v. of camera 1 and camera 2 are at the same position (black dot). Whereas, a smart intruder e_2 between the not synchronized cameras 2 and 3 gets never detected.

Besides of identifying the trajectory that optimizes the average detection time, we also want to develop an distributed feedback algorithm, which steers the cameras towards this optimal trajectory. In order to make this distributed algorithm scalable, the algorithm should include only communications between neighboring cameras. This leads us to the second problem, where we write $X(t \geq \bar{t})$ to denote the restriction of the trajectory $X(t)$ to the interval $t \in [\bar{t}, \infty)$.

Problem 2 (Distributed Algorithm design)**.** *Let for a set of cameras installed along an open path $X(t)$ be the cameras trajectory and $X^*(t)$ be the trajectory that yields the optimal average detection time. Design a distributed algorithm, with the communication constraint that only neighboring cameras need to communicate, that steers the cameras in a finite time $\bar{t}$ to the optimal cameras trajectory, i.e.*

$$X(t \geq \bar{t}) = X^*(t).$$

In the next section we present a constant-factor approximation for Problem 1 and an algorithm, which steers the cameras towards this solution.

Chapter 3

Main results

In this section we describe an approximate solution to Problem 1, and we design a distributed algorithm for the cameras to converge to such a trajectory. We remark that, for some cases, an exact solution to Problem 1 could be computed through standard optimization techniques, cf. Section 6. Such computation, however, is not scalable with the number of cameras, and it is not amenable to distributed implementation. Our approximate solution, instead, is extremely simple and efficient to compute, and its performance is shown to be within a certain bound of the optimum. Moreover, our approximate solution is valid for every number of cameras and environment configuration.

The cameras trajectory we propose can informally be described as follows.

> Each camera continuously sweep its cluster at maximum speed, and it stops for a certain time when its f.o.v. reaches a boundary. The waiting interval at each boundary is chosen such that each boundary point is observed simultaneously by two neighboring cameras. In other words, the waiting interval at the boundary point r_i is chosen such that $x_i(t) = r_i$ as soon as $x_{i+1}(t) = l_{i+1}$, and, analogously, the waiting interval at the boundary point l_i is chosen such that $x_i(t) = l_i$ as soon as $x_{i-1}(t) = r_{i-1}$.

Since we let each camera wait the same interval at its two boundaries, we call this cameras trajectory *equal-waiting trajectory*. An example

of equal-waiting trajectory is in Fig. 3.1, and a formal description is in Trajectory 1.

It should be noticed that, as discussed in Remark 1, the equal-waiting cameras trajectory is optimal with respect to the worst-case detection time criterium. Indeed, by construction, the equal-waiting camera trajectory is synchronized and $2d_{\max}$-periodic. Next we show that the equal-waiting cameras trajectory is constant factor optimal with respect to the average detection time criterium. A proof of this result is postponed to Section 5.

Theorem 3.0.1 (Performance of equal-waiting trajectories)**.** *For a set of n cameras with clusters lengths $d_1, \ldots, d_n$, let $X(t)$ be equal-waiting trajectory defined in Trajectory 1.*

1. *The optimal average detection time for a smart intruder satisfies the lower bound:*

$$\mathrm{ADT}^* \geq \frac{\sum_{i=1}^{n} d_i^2}{L}, \tag{3.1}$$

   where $L = \sum_{i=1}^{n} d_i$.

2. *The equal-waiting trajectory $X(t)$ has performance*

$$\mathrm{ADT}(X(t)) = \frac{1}{2} d_{\max} + \frac{1}{2} \frac{\sum_{i=1}^{n} d_i^2}{L}, \tag{3.2}$$

   where $d_{\max} = \max\{d_1, \ldots, d_n\}$.

3. *The equal-waiting trajectory $X(t)$ has performance within a constant factor of the optimum, that is,*

$$\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*} \leq \min\left\{ \frac{1}{2} + \frac{d_{\max}}{2d_{\min}}, \frac{3+\sqrt{n}}{4} \right\}, \tag{3.3}$$

   where $d_{\min} = \min\{d_1, \ldots, d_n\}$.

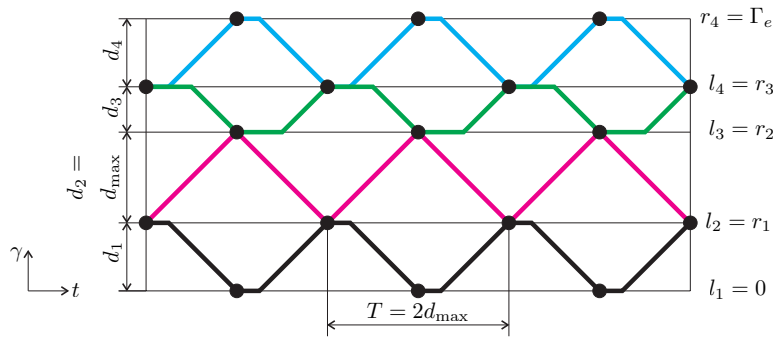


Figure 3.1: Equal-waiting trajectories. Illustration of an equal-waiting trajectory, formally described in Trajectory 1, for an $n = 4$ cameras example. The black dots illustrate the points where two f.o.v. meet or a f.o.v. reaches a wall.

Trajectory 1: *Equal-waiting trajectory (camera i)*

Input : $d_{\max}, r_i, l_i, d_i$;
Set : $t_{i,w} = d_{\max} - d_i, c_1 = t_{i,w} + kd_{\max}, c_2 = (1+k)d_{\max}$;

if *i is odd* **then**
- $x_i(t) := r_i$ for $kd_{\max} \le t \le c_1$, $k = 0, 2 \ldots$;
- $x_i(t) := -t + c_2$ for $c_1 \le t \le c_2$, $k = 0, 2 \ldots$;
- $x_i(t) := l_i$ for $kd_{\max} \le t \le c_1$, $k = 1, 3, \ldots$;
- $x_i(t) := t - c_1$ for $c_1 \le t \le c_2$, $k = 1, 3, \ldots$;

else if *i is even* **then**
- $x_i(t) := l_i$ for $kd_{\max} \le t \le c_1$, $k = 0, 2 \ldots$;
- $x_i(t) := t - c_1$ for $c_1 \le t \le c_2$, $k = 0, 2 \ldots$;
- $x_i(t) : r_i$ for $kd_{\max} \le t \le c_2$, $k = 1, 3, \ldots$;
- $x_i(t) := -t + c_2$ for $c_1 \le t \le c_2$, $k = 1, 3, \ldots$;

The following facts follow from Theorem 3.0.1. First, if all clusters have the same length, i.e. $d_{\max} = d_{\min}$, then Trajectory 1 is an optimal solution to Problem 1. Second, the lower bound is independent of the clusters arrangement.

We now design a distributed feedback algorithm that steers the cameras towards an equal-waiting trajectory. The algorithm is informally described as follows.

> First each camera moves to its left boundary. Then it waits until the f.o.v. of its left neighboring camera occupies the same position. As an exception, camera 1, which has no left neighbor, does not wait after it arrives at its left boundary. As soon as the f.o.v. of two neighboring cameras occupy the same position, both cameras wait for an interval of time as specified in Trajectory 1 and then move towards the opposite boundary. Clearly, the resulting cameras trajectory is synchronized and equal-waiting. Finally, in order for the cameras to maintain synchronization in the face of failures and motion uncertainties, we stop each camera at its boundary until the neighboring camera arrives.

A related example is in Section 4. Our distributed algorithm is formally described in Algorithm 2. Two comments are in order. First, we set $x_0(t) := l_1$ (resp. $x_{n+1}(t) := r_n$) for all times t since l_1 (resp. r_n) is the left (resp. right) extreme of the path Γ. Second, for the implementation of the proposed distributed algorithm, each camera is required to know only the endpoints of its cluster, the length of cluster, and the longest clusters length, and to be able of communicating with a neighboring camera. The following theorem characterizes the convergence properties of Algorithm 2, where we write $X(t \geq \bar{t})$ to denote the restriction of the trajectory $X(t)$ to the interval $t \in [\bar{t}, \infty)$.

Theorem 3.0.2 (Convergence of Algorithm 2)**.** *For a set of n cameras with clusters lengths $d_1, \ldots, d_n$, let $X(t)$ be the cameras trajectory generated by Algorithm 2. Let $\bar{t} = d_1 + (n-1)d_{\max}$. Then, $X(t \geq \bar{t})$ is an equal waiting trajectory.*

Proof. Notice that the f.o.v. of camera 1 coincides with the f.o.v. of camera 2 within time $d_1 + d_{\max}$. Then, the f.o.v. of camera c_i coincides with the f.o.v. of camera c_{i+i} within time $d_1 + id_{\max}$. Hence, within time $d_1 + (n-1)d_{\max}$ the cameras trajectory coincides with the equal-waiting trajectory in Trajectory 1. The claimed statement follows. □

It should be observed that in line 1 of Algorithm 2 the cameras could equivalently move to their right boundary instead of moving to the left one. In this case the convergence time is bounded by $d_n + (n-1)d_{\max}$. The cameras would then converge to a equal-waiting trajectory which is shifted in time, but its performance is still given by (3.3).

Algorithm 2: *Distributed camera synchronization along a equal-waiting trajectory (camera i)*

Input : $d_{\max}, l_i, r_i, d_i$;
Set : $t_{i,w} := d_{\max} - d_i$, $x_0(t) := l_1$ and $x_{n+1}(t) := r_n \ \forall t$;
Move towards l_i with $|\dot{x}_i(t)| = 1$;
while *True* **do**
    if $x_i(t) = x_{i-1}(t)$ *or* $x_i(t) = x_{i+1}(t)$ **then**
        wait until time $t + t_{i,w}$;
        move towards the opposite boundary with $|\dot{x}_i(t)| = 1$;
    else wait;

Chapter 4

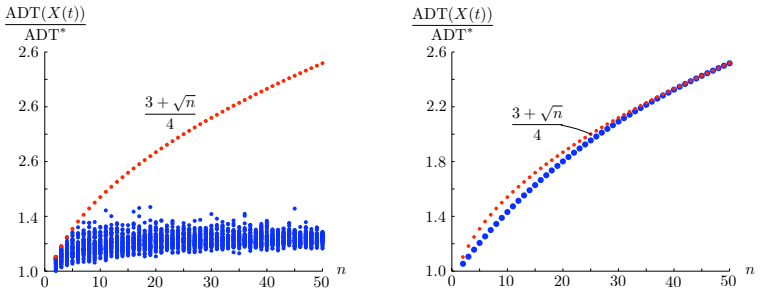
An illustrative example

Three simulation studies are presented in this section. In subsection 4.1 we present two studies, which numerically verify the bound of Equation (3.3) in Theorem 3.0.2. In subsection 4.2 our third simulation is presented. This simulation shows the convergence and robustness of Algorithm 2, which steers the cameras towards the equal-waiting trajectory.

4.1 Numerical verification of the performance bound

In our fist simulation, we want to verify the correctness of the bound $(3 + \sqrt{n})/4$. For this simulation study, we let the number of cameras n vary from 2 to 50. For each value of n, we generate 50 sets of clusters lengths $\{d_1, \ldots, d_n\}$, where $d_1 = \max_i d_i = 1$, and d_i, with $i = 2, \ldots, n$, is uniformly distributed within the interval $(0, 1]$. For each configuration, we compute the corresponding equal waiting trajectory $X(t)$ and evaluate the cost $\mathrm{ADT}(X(t))$. Additionally, for each configuration we compute the lower bound in Equation (3.1) and then calculate the ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$. Fig. 4.1(a) shows the result of this study. It should be observed that, for large values of n and uniformly distributed cluster lengths, the expected average detection time is much smaller than the bound described in Theorem

3.0.2. However, it is possible to create specific configurations of cluster lengths, such that the performance of the equal-waiting trajectory gets arbitrarily close to the lower bound performance. This means the bound in Theorem 3.0.2 is very tight. These configurations are that one cluster has a a long cluster length and all the other clusters have a much shorter cluster length. This implies, for these configurations the equal-waiting trajectory has almost optimal performance. A simulation for one of these special cases is shown in Fig. 4.1(b).



(a) The cluster lengths are $d_1 = 1$ and d_i for $i = 2, \ldots, n$ are uniformly distributed within $0 < d_i \leq 1$.

(b) Configuration where the equal-waiting trajectory is very close to the optimal performance. The cluster lengths are $d_1 = 1$ and $d_i = 0.125, i = 2, \ldots, n$.

Figure 4.1: Simulation of bound $(3 + \sqrt{n})/4$. The red dots illustrate the calculated performance bound $(3 + \sqrt{n})/4$ of Theorem 3.0.2, and each of the blue dots illustrates a calculated ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$ for a specific setup, which is specified in the subfigures

In our second simulation we want to show the correctness of the second bound, that is $1/2 + d_{\max}/2d_{\min}$. For this simulation study, we let the number of cameras be fixed (50 cameras), and we vary the parameter $d_{\min}$. Specifically, we let $d_1 = \max_i d_i = 1$ and d_i, with $i = 2, \ldots, 50$ be uniformly distributed within the interval $[d_{\min}/d_{\max}, 1]$. For these configurations we compute the corresponding equal waiting trajectory $X(t)$, evaluate the performance $\mathrm{ADT}(X(t))$, and then compute the lower bound in Equation (3.1).

Fig. 4.2 shows the result of this study. Notice that, for large values of $d_{\max}/d_{\min}$ and uniformly distributed cluster lengths, the bound in Theorem 3.0.2 is again conservative.

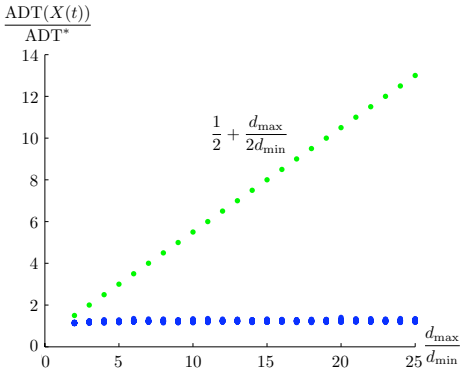


Figure 4.2: Simulation of bound $1/2 + d_{\max}/2d_{\min}$. The green dots illustrate the calculated performance bound $1/2 + d_{\max}/2d_{\min}$ of Theorem 3.0.2, and each of the blue dots illustrates a calculated ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$ for a setup described by $n = 50$, $d_1 = 1$ and d_i with $i = 2, \ldots, 50$ is uniformly distributed within interval $[\frac{d_{\min}}{d_{\max}}, 1]$.

4.2 Distributed feedback algorithm

In our third simulation we want to show the properties of the distributed feedback algorithm, described in Algorithm 2. For this simulation study, we consider a set of 4 cameras, and a fixed set of clusters with lengths $\{d_1, \ldots, d_4\}$. In this simulation we show three properties of the algorithm. First, the algorithm steers the cameras from random starting positions to the equal-waiting trajectory. Second, it is robust against camera failure, and third, the algorithm is robust against noise in the camera speed. This simulation is illustrated in Fig. 4.3 and an explanation follows.

The cameras start at time $t = 0$ at random starting positions. At

the beginning, all cameras move to the left boundary of their cluster and wait there for the arrival of the neighbor. Camera 1, which has no left neighbor, starts immediately after arriving at its left boundary with performing the equal-waiting trajectory. At about time $t = 50$ camera 1 meets camera 2 and therefore camera 2 starts performing the equal-waiting trajectory. With this procedure, all the neighboring cameras meet. After time $t = 150$, all neighboring cameras have met and the initialization phase is completed. The end of the initialization phase results in a uniform execution of the equal-waiting trajectory.

Between time $t = 340$ and time $t = 440$ camera 4 fails. Because of that, camera 3 cannot meet camera 4 at their shared boundary, and therefore camera 3 stops its movement and waits for camera 4 at its right boundary. All cameras cannot meet their neighbors any more, and because of that, all cameras wait at their right boundary. At time $t = 440$ camera 4 starts to work again and the camera performs again the initialization procedure, which results in the movement to the left boundary. As soon as camera 4 arrives at its left boundary, it meets camera 3. After that, all cameras meet again their neighbor and the equal-waiting trajectory is performed again. The algorithm could easily be changed such that if one camera fails, it sends a signal to its neighbors and the neighbors continue working like the shared boundary to the failed camera would be a wall. This would mean, that the remaining cameras continue working.

The third property of the algorithm, the robustness against noise in the movement is shown for time $t > 700$. After this time, noise is added to the movement of the cameras. The noise is normally distributed with mean 0.2 and standard derivation 1.0, while the maximum camera speed is 1.0. Because of this noise, the cameras trajectories are not straight any more and they arrive at the shared boundary at different times. However, since the cameras wait at their shared boundary for their neighbor, the synchronization of the cameras, which is essential to detect all intruders, is still guaranteed. Hence, the algorithm is robust against noise in the movement.

Following on the numerical validations in this chapter, we present an analytical proof of Theorem 3.0.2 in the next chapter. This proof consists of characterizing the performance of the lower bound and the equal-waiting trajectory, and of showing the constant factor approximation.

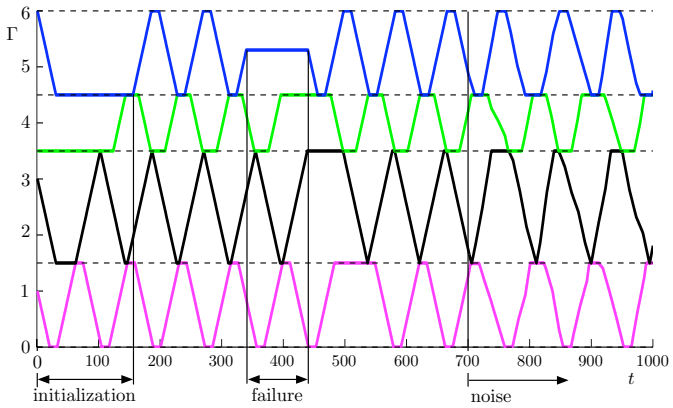


Figure 4.3: Simulation of Algorithm 2. Shown is the initialization procedure from random starting positions between time $t = [0, 150]$, a temporary camera failure of camera 4 between $t = [340, 440]$, and how the algorithm works under the influence of noise ($t > 700$).

Chapter 5

Proof of Theorem 3.0.1

This section contains a proof of Theorem 3.0.1, and it is organized as follows. In subsection 5.1 we characterize some useful properties of a minimum average detection time cameras trajectory. In subsection 5.2 we prove a lower bound for the minimum average detection time, we characterize the performance of the equal-waiting cameras trajectory, and we conclude the proof. The following definition will be used in the proof. For a cameras trajectory $X(t)$, the sets of *left catch-points* are defined as

$$\mathrm{CP}_i^l = \begin{cases} \{t \mid x_i(t) = x_{i-1}(t)\}, & \text{if } 2 \le i \le n \\ \{t \mid x_i(t) = \Gamma_0\}, & \text{if } i = 1, \end{cases}$$

and the sets of *right catch-points* as

$$\mathrm{CP}_i^r = \begin{cases} \{t \mid x_i(t) = x_{i+1}(t)\}, & \text{if } 1 \le i \le n-1 \\ \{t \mid x_i(t) = \Gamma_{\mathrm{f}}\}, & \text{if } i = n. \end{cases}$$

5.1 Properties of an optimal trajectory

In this section we characterize an optimal cameras trajectory. We start by rewriting the performance function (2.1) with the above defined catch-point in a more convenient form.

Lemma 5.1.1 (Modification of the performance function)**.** *Let $X(t) = \{x_1(t), \ldots, x_n(t)\}$ be a T-periodic cameras trajectory. Let CP_i^l and CP_i^r denote, respectively, the sets of left and right catch points of $X(t)$. We have*

$$\mathrm{ADT}(X(t)) = \frac{1}{T\sum\limits_{i=1}^{n} d_i} \sum_{i=1}^{n} \mathrm{ADT}_i^l(x_i(t)) + \mathrm{ADT}_i^r(x_i(t)), \qquad (5.1)$$

where,

$$\mathrm{ADT}_i^l(x_i(t)) = \int_0^T (x_i(t) - l_i)(\theta_i^l(t) - t)dt,$$
$$\mathrm{ADT}_i^r(x_i(t)) = \int_0^T (r_i - x_i(t))(\theta_i^r(t) - t)dt,$$

and

$$\theta_i^l(t) = \min_{q \in \mathrm{CP}_i^l} q \geq t, \quad \theta_i^r(t) = \min_{q \in \mathrm{CP}_i^r} q \geq t.$$

Proof. Notice that our defined performance function $\mathrm{ADT}(X(t))$ from chapter 2 is

$$\mathrm{ADT}(X(t)) = \frac{1}{T\sum\limits_{i=1}^{n} d_i} \int_0^T \int_\Gamma (t_d^*(p_{t,\gamma}^*) - t)d\gamma dt.$$

By spitting up the integral over the path Γ into integrals over the clusters s_i and summing them up over all i, this can be written as

$$\mathrm{ADT}(X(t)) = \frac{1}{T\sum\limits_{i=1}^{n} d_i} \sum_{i=1}^{n} \int_0^T \int_{s_i} (t_d^*(p_{t,\gamma}^*) - t)d\gamma dt.$$

Consider an intruder that appears at time t_0 at position $p_0 \in s_i$. Let $p_0 \leq x_i(t)$, i.e., the intruder appears to the left of the cameras f.o.v. (cf. Fig. 5.1). Since the intruder is smart, this means the intruder hides as long as possible, it is detected by camera c_i at time $\theta_i^l(t)$. It

follows that the detection time $t_d^*(p_{t_0,p_0}^*)$ equals the time of the next left catch-point, which is given by $\theta_i^l(t_0)$, this means

$$t_d^*(p_{t_0,p_0}^*) = \theta_i^l(t_0).$$

In particular, every intruder appearing at time t_0 to the left of the cameras f.o.v. will be detected at time $\theta_i^l(t_0)$. Analogously, every intruder appearing at time t_0 to the right of the cameras f.o.v. will be detected at time $\theta_i^r(t_0)$. Hence we have

$$\int_0^T \int_{s_i} (t_d^*(p_{t,\gamma}^*) - t) d\gamma dt = \int_0^T \int_{l_i}^{x_i(t)} (\theta_i^l(t) - t) d\gamma$$
$$+ \int_0^T \int_{x_i(t)}^{r_i} (\theta_i^r(t) - t) d\gamma = \mathrm{ADT}_i^l(x_i(t)) + \mathrm{ADT}_i^r(x_i(t)).$$

The claimed statement follows. □

Notice that a cameras trajectory is synchronized if and only if all cameras have at least one left and right catch-point, that means the sets CP_i^l and CP_i^r must be nonempty. It is now clear from Lemma 5.1.1 that a cameras trajectory $X(t)$ must be synchronized to obtaine a finite performance $\mathrm{ADT}(X(t))$. This is stated in the following Lemma.

Lemma 5.1.2 (Necessity of a synchronized trajectory)**.** *Let* $X(t) = \{x_1(t), \ldots, x_n(t)\}$ *be a periodic trajectory. Then*

$$\mathrm{ADT}(X(t)) < \infty$$

if and only if $\mathrm{CP}_i^l \neq \emptyset$ *and* $\mathrm{CP}_i^r \neq \emptyset$ *for all* $i = 1, \ldots, n$.

Proof. Notice that if $\mathrm{CP}_i^l = \emptyset$ (resp. $\mathrm{CP}_i^r = \emptyset$), then there exists a trajectory for a smart intruder to evade detection (cf. Fig. 2.2). Consequently, for some time $t \in [0, T]$, we have $\theta_i^l(t) = \infty$ (resp. $\theta_i^r(t) = \infty$). The performance calculation with (5.1) from Lemma 5.1.1 leads to $\mathrm{ADT}(X(t)) = \infty$. □

Following Lemma 5.1.1 it is possible to determine the shape of an optimal trajectory between any two consecutive catch points. Given the sets of right and left catch-points CP_i^l and CP_i^r of a camera. Let

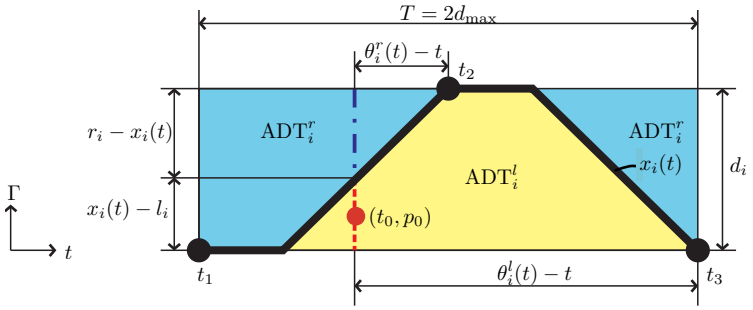


Figure 5.1: Performance calculation. Illustration of the performance separation into a right and left part, ADT_i^r and ADT_i^l respectively, of the trajectory. The black line illustrates the trajectory $x_i(t)$, the black dots at time t_1, t_2, t_3 catch-points, and the red dot at t_0, p_0 an intruder. The smart intruder at t_0, p_0, like all other intruders on the left side of $x_i(t)$ at this time, gets detected at t_3, whereas intruders appearing at time t_0 left of $x_i(t)$ get detected at t_2

$\mathrm{CP}_i = (t_0, \ldots, t_k)$ be the tuple of all catch-points of a camera i, where $t_i \in \mathrm{CP}_i^l \cup \mathrm{CP}_i^r$, $t_i < t_{i+1}$, and $t_k \leq t_0 + T$. Notice that CP_i is an ordered sequence of catch-points.

Lemma 5.1.3 (Trajectory shape)**.** *Let* $\mathrm{CP} = \{\mathrm{CP}_1, \ldots, \mathrm{CP}_n\}$ *be a given sequence of catch-points. A cameras trajectory* $X(t)$ *with* CP *as catch-points satisfies* $\mathrm{ADT}(X(t)) = \mathrm{ADT}^*$ *if and only if the trajectory of each camera* i *between any two consecutive points* $t_j, t_{j+1} \in \mathrm{CP}_i$ *is as in Fig. 5.2.*

Proof. Let the set of catch-points be fixed. Observe from Lemma 5.1.1 that the performance $\mathrm{ADT}(X(t))$ is computed by summing the performance of each cluster, and the performance of two different clusters are independent from each other. Then, in order to minimize the sum over all clusters, the performance of each cluster needs to be minimized. Notice that, because of Lemma 5.1.1, each term ADT_i

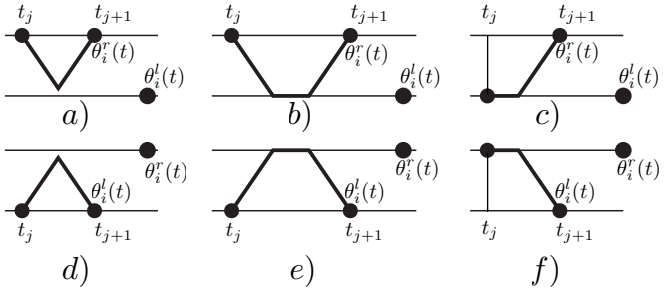


Figure 5.2: Possible trajectory shapes. The figures illustrate the trajectories between two catch-points t_j and t_{j+1}. In figures a), b) and c) $\theta_i^l(t) - \theta_i^r(t) > 0$, therefore $x_i(t)$ is as small as possible. In figures d), e), and f) $\theta_i^l(t) - \theta_i^r(t) < 0$, here the trajectories are such that $x_i(t)$ is as big as possible. All trajectories are unique, because they are obtained with camera speed either zero or maximum.

can be written as

$$\begin{aligned} \mathrm{ADT}_i &= \mathrm{ADT}_i^l + \mathrm{ADT}_i^r \\ &= \sum_{j=0}^{k-1} \int_{t_j}^{t_{j+1}} (x_i(t) - l_i)(\theta_i^l(t) - t) + (r_i - x_i(t))(\theta_i^r(t) - t)dt. \end{aligned} \tag{5.2}$$

Since the summands of Equation (5.2) are independent of each other, every term can be minimized on its own. Therefore this leads to the optimization problem

$$\min_{x_i(t)} \int_{t_j}^{t_{j+1}} (x_i(t) - l_i)(\theta_i^l(t) - t) + (r_i - x_i(t))(\theta_i^r(t) - t)dt. \tag{5.3}$$

Since Equation (5.3) shall be minimized subject to $x_i(t)$, it can be

rewritten to

$$\min_{x_i(t)}(\theta_i^l(t) - \theta_i^r(t)) \int_{t_j}^{t_{j+1}} x_i(t)dt + c, \tag{5.4}$$

where c is independent of $x_i(t)$, and $\theta_i^l(t)$, $\theta_i^r(t)$ are constant within the interval $t \in [t_j, t_{j+1}]$. Two cases can occur:

1. $\theta_i^l(t) - \theta_i^r(t) > 0$. This means, starting from time t the first occurring catch-point is an upper catch-point. The integral has positive sign and x_i must be minimized in order to minimize (5.4). These cases are illustrated in Fig. 5.2 a), b), and c);

2. $\theta_i^l(t) - \theta_i^r(t) < 0$. This means, starting from time t the first occurring catch-point is a left catch-point. The integral has negative sign and $x_i(t)$ must maximize the integral in (5.4). These cases are illustrated in Fig. 5.2 d), e) and f).

The statement follows. □

As a consequence of Lemma 5.1.2, if the catch-points are given, then an optimal cameras trajectory can be obtained from the trajectories in Fig. 5.2, where the speed of each camera is either zero or maximum. Hence, the problem of designing optimal cameras trajectory reduces to the problem of finding a set of catch-points yielding optimal performance. This insight is used in the next subsection to obtain the performance bounds.

5.2 Performance bounds

In this subsection we state three things. First we derive a lower bound for the average detection time. Second we characterize the average detection time of the equal-waiting trajectory and third and last, we characterize the performance of the equal-waiting trajectory by evaluating the performance ratio of the equal-waiting trajectory and the lower bound.

The lower bound performance is obtained as the sum of the lower bound performances for the detection time of each single cluster.

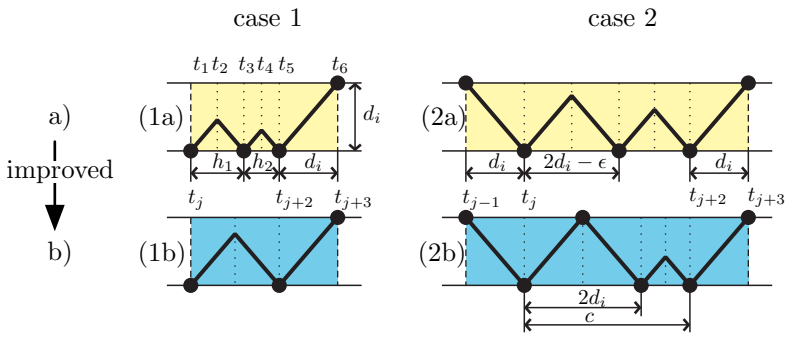


Figure 5.3: Sequence of catch-points. Case 1 and case 2 illustrate the two possible cases of $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}^l$. The yellow cases in row a) show trajectories with three consecutive catch-points on one boundary. The blue cases in row b) show, how these trajectories can be improved.

To calculate the lower bound for a single cluster we assume that the neighboring cameras of this cluster behave exactly like it is needed for this cluster. That means, if a camera c_i reaches the left boundary, the left neighbor is at this shared boundary, and if the camera c_i reaches the right boundary, the right neighbor is at the shared boundary. More formally, that means the neighboring cameras c_{i-1} and c_{i+1} are such that $x_{i-1}(t) = l_i$ and $x_{i+1}(t) = r_i$ at all times. Since this cannot be guaranteed in general, the so calculated lower performance bound is in general not achieved.

To calculate the lower performance bound we first state another property of the trajectory, which is necessary for an optimal average detection time. This property is, that for the lower bound scenario three consecutive catch-points on the right or left boundary never lead to an optimal average detection time. The performance of a trajectory with three consecutive catch-points on one boundary can always be improved by a trajectory that has at most two consecutive catch-points on one boundary. This property is stated in the following Lemma.

Lemma 5.2.1 (Sequence of catch-points)**.** *For a cameras trajectory*

$X(t)$, assume that $\mathrm{CP}_i^l = \{t \mid x_i(t) = l_i\}$ and $\mathrm{CP}_i^r = \{t \mid x_i(t) = r_i\}$ for each camera i. Let CP_i be the ordered sequence of catch-points within time $[0, T]$. Then, $\mathrm{ADT}(X(t)) = \mathrm{ADT}^*$ only if there exists no consecutive catch-points $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}_i$ such that $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}_i^l$ or $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}_i^r$, for each camera i.

Proof. We proceed by contradiction. Suppose that there exist three consecutive catch-points $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}_i$ such that $t_j, t_{j+1}, t_{j+2} \in \mathrm{CP}_i^l$ for some camera i. Assume that $t_{j+2} - t_j \leq 2d_i$, and consider the cost ADT_i as defined in Lemma 5.1.3. Referring to Fig. 5.3 (1a), it can be verified that the trajectory in Fig. 5.3 (1b) has a smaller average detection time. This is done by calculating $\mathrm{ADT}_i(1a)$ and $\mathrm{ADT}_i(1b)$, which denotes the performance of the cases (1a) and (1b) respectively. Then it is shown that $\mathrm{ADT}_i(1a) - \mathrm{ADT}_i(1b) > 0$, which means that case (1a) has always a greater average detection time than case (1b).

The performance of a cluster can be calculated with the modified performance Equation (5.1) as

$$\mathrm{ADT}_i = \int_0^T (x_i(t) - l_i)(\theta_i^l(t) - t) + (r_i - x_i(t))(\theta_i^r(t) - t)dt.$$

Since the trajectory $x_i(t)$ is not smooth, it is helpful, to split up the integration into different intervals, such that in each interval the trajectory can be expressed as a smooth function. This modification simplifies the evaluation of the integrals. The time integral is split up into $\bar{j}$ intervals $[t_j, t_{j+1}]$, where $j = 1, \ldots, \bar{j}$. In each interval $[t_j, t_{j+1}]$ the function $x_{i,j}(t)$ describes the function $x_i(t)$. With this modification, the performance calculation can be expressed as

$$\mathrm{ADT}_i = \sum_{j=1}^{\bar{j}} \int_{t_{i,j}}^{t_{i,j+1}} x_{i,j}(t)(\theta_{i,j}^l - \theta_{i,j}^r) + d_i(\theta_{i,j}^r - t)dt, \tag{5.5}$$

where $\theta_{i,j}^l$ and $\theta_{i,j}^r$ are the θ-functions for the interval $[t_j, t_{j+1}]$. Without loss of generality we set $l_i = 0$ and $r_i = d_i$. By using Equation (5.5) we obtain for the calculation of $\mathrm{ADT}_i(1a)$

$$\mathrm{ADT}_i(1a) = \sum_{j=1}^{5} \int_{t_{i,j}}^{t_{i,j+1}} x_{i,j}(t)(\theta_{i,j}^l - \theta_{i,j}^r) + d_i(\theta_{i,j}^r - t)dt \tag{5.6}$$

where

$$x_{i,1}(t) = t, \qquad x_{i,2}(t) = -t + h_1, \qquad x_{i,3}(t) = t - h_1,$$
$$x_{i,4}(t) = -t + h_1 + h_2, \quad x_{i,5}(t) = t - h_1 - h_2,$$

and

$$\theta^l_{i,1} = \theta^l_{i,2} = h_1,$$
$$\theta^l_{i,3} = \theta^l_{i,4} = h_1 + h_2,$$

and

$$\theta^r_{i,j} = d_i + h_1 + h_2 \quad \text{for } j = 1, \ldots, 4.$$

Evaluating Equation (5.6) with this parameters leads to

$$\begin{aligned} \mathrm{ADT}_i(1a) = h_1 h_2 (d_i - \frac{1}{4} h_1) + \\ \frac{1}{2} d_i (\frac{1}{2} h_1^2 + h_1 d_i + \frac{1}{2} h_2^2 + h_2 d_i + \theta_{i,5^l} d_i). \end{aligned} \tag{5.7}$$

Evaluating Equation (5.5) for case (1b) leads to the performance

$$\mathrm{ADT}_i(1b) = \frac{1}{2} d_i (h_1 h_2 + \frac{1}{2} h_1^2 + d_i h_1 + \frac{1}{2} h_2^2 + h_2 d_i + h_3 d_i). \tag{5.8}$$

Calculating the difference of the Equations (5.7) and (5.8) yields

$$\mathrm{ADT}_i(1a) - \mathrm{ADT}_i(1b) = \frac{1}{2} h_1 h_2 (d_i - \frac{1}{2} h_1).$$

This case is described by $t_{j+2} - t_j \leq 2d_i$, which is equivalent to $h_1 + h_2 \leq 2d_i$. Hence, $h_1 \leq 2d_i$ and therefore $d_i - \frac{1}{2} h_1 \geq 0$, which leads to $\mathrm{ADT}_i(1a) - \mathrm{ADT}_i(1b) \geq 0$. This means, case (1a), the case with 3 consecutive catch-points has, regardless of h_1 and h_2, always a greater average detection time than case (1b).

Consider now the case $t_{j+2} - t_j > 2d_i$, which is illustrated in Fig. 5.3 (2a-2b). For these two case we also calculate the average detection time with performance Equation (5.5). With the variables used in Fig. 5.3 (2a-2b), this leads to

$$\begin{aligned} \mathrm{ADT}_i(2a) = -\frac{1}{4} \epsilon^2 (c - \epsilon) + \\ d_i (\frac{1}{2} h_3 d_i + c d_i + \frac{1}{4} c^2 - \epsilon d_i + \epsilon^2 + \frac{1}{2} c \epsilon + \frac{1}{2} d_i^2) \end{aligned} \tag{5.9}$$

and

$$\mathrm{ADT}_i(2b) = \frac{1}{2}d_i(h_3 d_i + \frac{1}{2}c^2 + d_i^2). \tag{5.10}$$

Calculating the difference of Equations (5.9) and (5.10) leads to

$$\mathrm{ADT}_i(2a) - \mathrm{ADT}_i(2b) = c(d_i^2 - \frac{1}{4}\epsilon^2) + \epsilon^2(d_i - \frac{1}{4}\epsilon) + d_i\epsilon(\frac{1}{2}c - d_i).$$

Since $2d_i < c < 4d_i$, $0 < \epsilon < 4d_i - c$, and hence $\epsilon < 2d_i$ the three terms in brackets, $(d_i^2 - \frac{1}{4}\epsilon^2)$, $(d_i - \frac{1}{4}\epsilon)$, and $(\frac{1}{2}c - d_i)$ are all positive, and therefore $\mathrm{ADT}_i(2a) - \mathrm{ADT}_i(2b) > 0$. This means, case (2b), the case with just two consecutive catch-points has regardless of c and ϵ a lower average detection time.

Since for both cases, $t_{j+2} - t_j \leq 2d_i$ and $t_{j+2} - t_j > 2d_i$, the case with just two consecutive catch-points has the lower average detection time, we can state that an optimal cluster performance can never be obtained with three consecutive catch-points. □

From Lemma 5.2.1 and 5.1.3 we know the necessary sequence of catch-points and the trajectory between these catch-points for an optimal cluster performance. With this insight of the trajectory shape, it is now possible to create an parameterized model of the lower bound performance. This model is illustrated in Fig. 5.4 and an explanation follows.

The trajectory of the lower bound model could start at every position of the cluster, because the trajectory is a periodic trajectory, which must visit every point of the cluster. In our model it starts at t_0 at the left boundary. The next catch-point after t_0 could be either at the right or left boundary, here t_1 is set to the left boundary in distance h_1. The trajectory between this two, and any other two, catch-points, is known from Lemma 5.1.3. According to Lemma 5.2.1 we cannot have three consecutive catch-points on one boundary, hence the third catch-point must be on the opposite boundary, which leads to t_2 at the right boundary. With setting $h_1 = 0$ this setup also includes the case that the second catch-point is at the right boundary, therefore it represents the most general case. Catch-point number three, t_2, must be after time d_i from t_1. It cannot be later, because in that case it is known from Lemma 5.1.3 that the trajectory first stays at the left boundary. Staying at the left boundary can

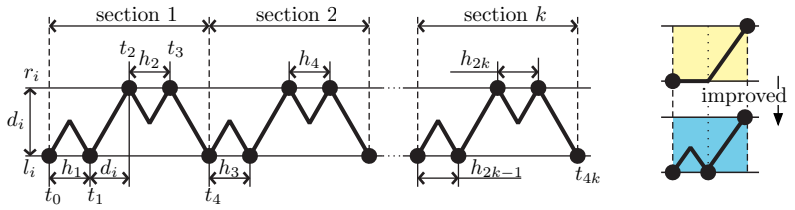


Figure 5.4: Parameterized lower bound model. Fig a) shows the parameterized model for a single cluster. Fig b) explains why the catch-points on opposite boundaries must be in distance d_i.

always be improved, as shown in Fig. 5.4 b), which leads to three consecutive catch-points on one side and according to Lemma 5.2.1 this is not optimal. Therefore for an optimal solution t_2 must be exactly after time d_i. From t_2 onwards, the same reasoning holds as from t_0 onwards, which leads to t_3 on the right boundary and t_4 at the left boundary. We call the movement between t_0 and t_4 section 1, and this movement then can be performed k times. This leads to a movement with $k \leq \lfloor R_i \rfloor$ sections, where $R_i = d_{\max}/d_i$ is the ratio of the longest cluster length to the i-th cluster length. The parameter k is bounded, because in each section the camera needs to sweep back and forth between the boundaries of the cluster, and for a fixed period length $T = 2d_{\max}$ at most $\lfloor R_i \rfloor$ back and forth movements can be performed. The average detection time of this parameterized model can be calculated, which leads to the lower bound performance stated in the following Lemma.

Lemma 5.2.2 (Lower bound)**.** *For a set of n cameras with clusters lengths $d_1, \ldots, d_n$, the optimal average detection time for a smart intruder satisfies the lower bound:*

$$\mathrm{ADT}^* \geq \frac{\sum_{i=1}^{n} d_i^2}{L},$$

where $L = \sum_{i=1}^{n} d_i$.

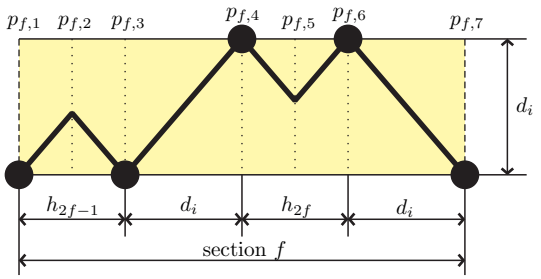


Figure 5.5: Parameterized model for a section f. The model shows the catch-points (black dots), whose position is described by parameters, and the trajectory between the catch-points.

Proof. Consider a cameras trajectory $X(t)$, and let the parameters $p = (k, h_1 \dots h_{2k})$ describe the trajectory $x_i(t)$ as in Fig. 5.4. Let ADT_i be the average detection time of cluster i for the trajectory $X(t)$ (see the proof of Lemma 5.1.3). The proof consists then of 3 steps. First, the performance of the lower bound model $\mathrm{ADT}_i(k, h_1 \dots h_{2k})$ is calculated. Second, $\mathrm{ADT}_i(k, h_1 \dots h_{2k})$ is minimized subject to $k, h_1 \dots h_{2k}$, which leads to an discontinuous function. And in the last step this discontinuous function is lower bounded by an continuous function.

1. Calculation of $\mathrm{ADT}_i(k, h_1 \dots h_{2k})$

   To calculate the cost subject the variables $k, h_1 \dots h_{2k}$ the modified performance Equation (5.1) is evaluated. This leads to the calculation of

   $$\begin{aligned}\mathrm{ADT}_i(x_i(t)) = \int_{t_0}^{t_{4k}} &(x_i(t) - l_i)(\theta_i^l(t) - t) \\ &+(r_i - x_i(t))(\theta_i^r(t) - t)dt.\end{aligned}$$

   This function is evaluated by splitting it up into the sum of the average detection times of the sections $\mathrm{ADT}_{i,f}$, $f = 1, \dots, k$,

that is

$$\mathrm{ADT}_i(x_i(t)) = \sum_{f=1}^{k} \mathrm{ADT}_{i,f}, \tag{5.11}$$

where

$$\begin{aligned}\mathrm{ADT}_{i,f} = \int_{t_{4(f-1)}}^{t_{4f}} &(x_{i,f}(t) - l_i)(\theta^l_{i,f}(t) - t) + \\ &(r_i - x_{i,f}(t))(\theta^r_{i,f}(t) - t)dt,\end{aligned}$$

and $x_{i,f}$, $\theta^l_{i,f}(t)$, and $\theta^r_{i,f}(t)$ describes the variables within section f. The integral of each section can be further slitted up into six intervals, such that in each interval the function $x_i(t)$ can be expressed as a smooth function. These parts are illustrated in Fig. 5.5, and this leads to

$$\begin{aligned}\mathrm{ADT}_{i,f} = \sum_{j=1}^{6} \int_{p_{f,j}}^{p_{f,j+1}} &(x_{i,f,j}(t) - l_i)(\theta^l_{i,f,j}(t) - t) + \\ &(r_i - x_{i,f,j}(t))(\theta^r_{i,f,j}(t) - t)dt,\end{aligned} \tag{5.12}$$

where $x_{i,f,j}(t)$ describes the trajectory, $\theta^l_{i,f,j}(t)$ the next left catch-point, and $\theta^r_{i,f,j}(t)$ the next right catch-point within the time interval $[p_{f,j}, p_{f,j+1}]$. All variables of Equation (5.12) can be expressed subject to $d_i, f, k, h_1, \ldots, h_{2k}$, which enables the calculation of $\mathrm{ADT}_{i,f}(k, h_1 \ldots h_{2k})$, and then by using Equation (5.11) the calculation of $\mathrm{ADT}_i(k, h_1 \ldots h_{2k})$. The specification of these variables is done in the following.

First the seven points $p_{f,1}, \ldots, p_{f,7}$, which characterize the intervals of the smooth trajectories, are described. These points

are

$$p_{f,1} = \sum_{j=1}^{2f-2} h_j + 2d_i(f-1), \qquad p_{f,2} = p_{f,1} + \frac{1}{2}h_{2f-1},$$
$$p_{f,3} = p_{f,2} + \frac{1}{2}h_{2f-1}, \qquad p_{f,4} = p_{f,3} + d_i,$$
$$p_{f,5} = p_{f,4} + \frac{1}{2}h_{2f}, \qquad p_{f,6} = p_{f,5} + \frac{1}{2}h_{2f},$$
$$p_{f,7} = p_{f,6} + d_i.$$

Notice, the point $p_{f,1}$ is the starting point of section f. That means its time is given by the summation of the duration of all previous sections g, $g = 1, \ldots, f-1$. The duration of each previous section g can be expressed as $2d_i + h_{2g-1} + h_{2g}$. Next, with the points $p_{f,1}, \ldots, p_{f,7}$, the trajectories $x_{i,f,j}(t)$, $j = 1, \ldots, 6$ can be formulated as

$$x_{i,f,1} = t - \sum_{j=1}^{2k-2} h_j - 2d_i(f-2),$$
$$x_{i,f,2} = -t + \sum_{j=1}^{2k-2} h_j + h_{2f-1} + 2d_i(f-1),$$
$$x_{i,f,3} = t - \sum_{j=1}^{2k-2} h_j - h_{2f-1} - 2d_i(f-1),$$
$$x_{i,f,4} = -t + \sum_{j=1}^{2k-2} h_j + h_{2f-1} + 2d_i f,$$
$$x_{i,f,5} = t - \sum_{j=1}^{2k-2} h_j - h_{2f-1} - h_{2f} - 2d_i(f-1),$$
$$x_{i,f,6} = -t + \sum_{j=1}^{2k-2} h_j + h_{2f-1} + h_{2f} + 2d_i f.$$

Finally, the left and right catch-points of the intervals, $\theta^l_{i,f,j}$

and $\theta^r_{i,f,j}$, need to be defined, these are

$$\theta^l_{i,f,j} = p_{f,3} \quad \text{for } j = 1, 2,$$
$$\theta^l_{i,f,j} = p_{f,7} \quad \text{for } j = 3, \ldots, 6,$$

and

$$\theta^r_{i,f,j} = p_{f,4} \quad \text{for } j = 1, 2, 3$$
$$\theta^r_{i,f,j} = p_{f,6} \quad \text{for } j = 4, 5$$
$$\theta^r_{i,f,6} = p_{f,7} + d_i + h_{2f+1}.$$

Evaluating Equation (5.12) with these variables leads to

$$\mathrm{ADT}_{i,f} = \frac{1}{4} d_i (h^2_{2f-1} + h^2_{2f})$$
$$+ d_i^2 (\frac{3}{2} h_{2f} + h_{2f-1} + \frac{1}{2} h_{2f+1}) + 2 d_i^3.$$

The summation of $\mathrm{ADT}_{i,f}$ over all f then yields

$$\mathrm{ADT}_i = \sum_{f=1}^{k} \mathrm{ADT}_{i,f} = \frac{1}{4} d_i \sum_{f=1}^{k} (h^2_{2f-1} + h^2_{2f})$$
$$+ d_i^2 \sum_{f=1}^{k} (\frac{3}{2} h_{2f} + h_{2f-1} + \frac{1}{2} h_{2f+1}) + 2k d_i^3.$$

This equation can be simplified by combining the summands of the sums, that leads to

$$\mathrm{ADT} = \frac{1}{4} d_i \sum_{j=1}^{2k} h_j^2 + d_i^2 \sum_{j=1}^{2k} (\frac{3}{2} h_j) + \frac{1}{2} h_{2k+1} - \frac{1}{2} h_1 + 2k d_i^3.$$

The parameter h_{2k+1} is not defined, but it stands for the first parameter which follows the end of the period T. After the end of the period the same period starts over, hence $h_{2k+1} = h_1$. Inserting this yields

$$\mathrm{ADT} = \frac{1}{4} d_i \sum_{j=1}^{2k} h_j^2 + \frac{3}{2} d_i^2 \sum_{j=1}^{2k} h_j + 2k d_i^3. \tag{5.13}$$

With the use of the ratio R_i,

$$R_i = \frac{d_{\max}}{d_i}, \tag{5.14}$$

Equation (5.13) transforms to

$$\mathrm{ADT} = \frac{1}{4}\frac{d_{\max}}{R_i}\sum_{j=1}^{2k} h_j^2 + \frac{3}{2}(\frac{d_{\max}}{R_i})^2 \sum_{j=1}^{2k} h_j + 2k(\frac{d_{\max}}{R_i})^3.$$

The period length of the trajectory is a priori fixed, that means

$$T = 2d_{\max} = 2kd_i + \sum_{j=1}^{2k} h_j.$$

With R_i from Equation (5.14), $\sum_{j=1}^{2k} h_j$ can be written as

$$\sum_{j=1}^{2k} h_j = 2d_{\max} - 2kd_i = 2d_{\max}(1 - \frac{k}{R_i}).$$

Inserting that, leads to the final function of the lower bound model

$$\mathrm{ADT}_i(k, h_1 \dots h_{2k}) = \frac{d_{\max}}{4R_i}\sum_{j=1}^{2k} h_j^2 + d_{\max}^3(\frac{3R_i - k}{R_i^3}). \tag{5.15}$$

To obtain the minimum of this function, the optimal parameters $k, h_1, \dots, h_{2k}$, which minimize the function, must be calculated. This calculation is shown in the step two.

2. Minimizing $\mathrm{ADT}_i(k, h_1 \dots h_{2k})$

   Suppose that the integer k is fixed. Then, a minimizer of ADT_i can be computed by solving the optimization problem

$$\begin{aligned} \min_{h_1,\dots,h_{2k}} \quad & \frac{d_{\max}}{4R_i}\sum_{j=1}^{2k} h_j^2 \\ \text{subject to} \quad & \sum_{j=1}^{2k} h_j = 2d_{\max}(1 - \frac{k}{R_i}), \end{aligned} \tag{5.16}$$

where the constraint arises from the fixed period length $T = 2d_{\max} = 2kd_i + \sum_{i=1}^{2k} h_j$. This optimization problem is solved by using the method of Lagrange multipliers. The method of Lagrange multipliers is an optimization tool to calculate the minima of a function, subject to equality constraints. A detailed explanation of the method can be found in chapter 8. To use this method, first the Lagrange function needs to be defined. For the optimization problem 5.16, that is

$$L(x, \lambda) = \frac{d_{\max}}{4R_i} \sum_{j=1}^{2k} h_j^2 + \lambda(\sum_{j=1}^{2k} h_j - 2d_{\max}(1 - \frac{k}{R_i})).$$

The partial derivatives of this Lagrange function are

$$\frac{\partial L}{\partial h_j} = \frac{d_{\max}}{2R_i} h_j + \lambda = 0, \tag{5.17}$$

$$\frac{\partial L}{\partial \lambda} = \sum_{j=1}^{2k} h_j - 2d_{\max}(1 - \frac{k}{R_i}) = 0. \tag{5.18}$$

Solving Equation (5.17) for h_j yields

$$h_j = -\lambda \frac{2R_i}{d_{\max}}. \tag{5.19}$$

Inserting the result of Equation (5.19) into (5.18) leads to

$$\begin{aligned} 0 &= \sum_{j=1}^{2k} -\lambda \frac{2R_i}{d_{\max}} - 2d_{\max}(1 - \frac{k}{R_i}) \\ &= -2k\lambda \frac{2R_i}{d_{\max}} - 2d_{\max}(1 - \frac{k}{R_i}), \end{aligned}$$

and finally,

$$\lambda = \frac{d_{\max}^2}{2kR_i}(\frac{k}{R_i} - 1).$$

With this expression of λ and Equation (5.19), h_j can be calculated as

$$h_j = \frac{R_i - k}{R_i k} d_{\max} \quad j = 1, \ldots, 2k. \tag{5.20}$$

Inserting the result for h_j from (5.20) into the parameterized performance function (5.15) leads to

$$\mathrm{ADT}_i(k) = \frac{d_{\max}}{4R_i} \sum_{j=1}^{2k} (\frac{R_i - k}{R_i k} d_{\max})^2 + d_{\max}^3 (\frac{3R_i - k}{R_i^3}). \quad (5.21)$$

Observe now that $k \in \mathbb{N}$ and $1 \leq k \leq \lfloor R_i \rfloor$, and that the terms $\frac{R_i - k}{R_i k}$ and $\frac{3R_i - k}{R_i^3}$ are monotonically decreasing with k. Then, the optimal parameter k equals $\lfloor R_i \rfloor$. Inserting $k = \lfloor R_i \rfloor$ into Equation (5.21) and simple manipulation yields

$$\min \mathrm{ADT}_i = d_{\max}^3 (\frac{1}{2R_i \lfloor R_i \rfloor} + \frac{2}{R_i^2} - \frac{\lfloor R_i \rfloor}{2R_i^3}).$$

Notice, this function is discontinuous because of the term $\lfloor R_i \rfloor$. In order to proof a performance bound for the equal-waiting trajectory, we lower bound this discontinuous function with a continuous function in step three.

3. Calculating a continuous lower bound

   In this last step, we want to show, that $\frac{2}{R_i^2} d_{\max}^3$ is a lower bound for the average detection time of a cluster, i.e.

$$\min \mathrm{ADT}_i = d_{\max}^3 \left(\frac{1}{2R_i \lfloor R_i \rfloor} + \frac{2}{R_i^2} - \frac{\lfloor R_i \rfloor}{2R_i^3} \right) \geq \frac{2}{R_i^2} d_{\max}^3.$$

   Simple manipulation yields

$$\frac{1}{2R_i \lfloor R_i \rfloor} \geq \frac{\lfloor R_i \rfloor}{2R_i^3},$$

   and finally

$$R_i^2 \geq \lfloor R_i \rfloor^2,$$

   which is true, because $R_i \geq \lfloor R_i \rfloor$.

Since the lower bound performance of the domain, that means all clusters, is greater or equal to the sum of the clusters lower bound

performance, it follows

$$\mathrm{ADT}^* \geq \frac{1}{TL} \sum_{i=1}^{n} \min \mathrm{ADT}_i$$

where $T = 2d_{\max}$ and $L = \sum_{i=1}^{n} d_i$. Moreover, we showed

$$\min \mathrm{ADT}_i \geq \frac{2}{R_i^2} d_{\max}^3,$$

which can be modified with $R_i = d_{\max}/d_i$ to

$$\min \mathrm{ADT}_i \geq 2 d_{\max} d_i^2.$$

Combining these results leads to,

$$\begin{aligned} \mathrm{ADT}^* &\geq \frac{1}{TL} \sum_{i=1}^{n} \min \mathrm{ADT}_i \\ &\geq \frac{1}{TL} \sum_{i=1}^{n} 2 d_{\max} d_i^2 = \frac{\sum_{i=1}^{n} d_i^2}{L}. \end{aligned}$$

Hence, the statement is true. □

In the next lemma we characterize the performance of the equal-waiting trajectory.

Lemma 5.2.3 (Equal-waiting trajectory performance)**.** *For a set of n cameras with clusters lengths $d_1, \ldots, d_n$, let $X(t)$ be the equal-waiting trajectory defined in Trajectory 1. Then*

$$\mathrm{ADT}(X(t)) = \frac{1}{2} d_{\max} + \frac{1}{2} \frac{\sum_{i=1}^{n} d_i^2}{L}, \tag{5.22}$$

where $d_{\max} = \max\{d_1, \ldots, d_n\}$ and $L = \sum_{i=1}^{n} d_i$.

Proof. For the proof we show that the performance of one cluster with the equal-waiting trajectory can be calculated as

$$\mathrm{ADT}_i(x_i(t)) = (d_{\max} + d_i) d_i d_{\max}.$$

The summation of the single cluster performances and normalizing the performance, that is

$$\mathrm{ADT}(X(t)) = \frac{1}{TL}\sum_{i=1}^{n}\mathrm{ADT}_i(x_i(t)),$$

where $T = 2d_{\max}$, leads with some simple modification to the statement. In the following the calculation of the cluster performance is stated. From Lemma 5.1.1 follows

$$\mathrm{ADT}_i(x_i(t)) = \int_0^T (x_i(t) - l_i)(\theta_i^l(t) - t) + (r_i - x_i(t))(\theta_i^r(t) - t)dt.$$

Without loss of generality we set $l_i = 0$ and $r_i = d_i$. Moreover we split the time integral up into 4 intervals, such that in each interval $x_i(t)$ can be expressed as a smooth function, that leads to

$$\begin{aligned}&\mathrm{ADT}_i(x_i(t)) = \\ &\sum_{j=1}^{4}\int_{p_{i,j}}^{p_{i,j+1}} x_{i,j}(t)(\theta_{i,j}^l(t) - \theta_{i,j}^r(t)) + d_i(\theta_{i,j}^r(t) - t)dt.\end{aligned} \tag{5.23}$$

The values for the variables $p_{i,j}$ for $j = 1, \ldots, 5$ and $x_{i,j}$, $\theta_{i,j}^l(t)$, and $\theta_{i,j}^r(t)$ for $j = 1, \ldots, 4$ follow from Fig. 5.6 and they are

$$\begin{aligned}&p_{i,1}(t) = 0, && p_{i,2}(t) = d_{\max} - d_i, && p_{i,3}(t) = d_{\max},\\ &p_{i,4}(t) = 2d_{\max} - d_i, && p_{i,5}(t) = 2d_{\max},\end{aligned}$$

and

$$\begin{aligned}&x_{i,1}(t) = 0, && x_{i,2}(t) = t - (d_{\max} - d_i),\\ &x_{i,3}(t) = d_i, && x_{i,4}(t) = -t + 2d_{\max},\end{aligned}$$

and

$$\theta_{i,j}^l(t) = 2d_{\max}, \quad \text{for } j = 1, \ldots, 4,$$

and

$$\begin{aligned}&\theta_{i,j}^l(t) = d_{\max}, && \text{for } j = 1, 2,\\ &\theta_{i,j}^l(t) = 3d_{\max}, && \text{for } j = 3, 4.\end{aligned}$$

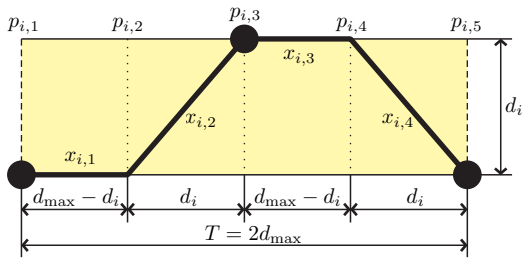


Figure 5.6: Performance calculation of the equal-waiting trajectory for a cluster d_i. The bold black line illustrates the trajectory and the black dots the catch-points.

Inserting these values into Equation (5.23) leads to

$$\mathrm{ADT}_i(x_i(t)) = (d_{\max} + d_i)d_i d_{\max}.$$

The statement follows. □

We are now ready to prove the result about the constant factor approximation of the equal-waiting trajectory. This result, combined with Lemma 5.2.2 and Lemma 5.2.3, concludes the proof of Theorem 3.0.1.

Lemma 5.2.4 (Equal-waiting trajectory approximation)**.** *For a set of n cameras with clusters lengths $d_1, \ldots, d_n$, let $X(t)$ be the equal-waiting trajectory defined in Trajectory 1. Then,*

$$\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*} \leq \min\left\{\frac{1}{2} + \frac{d_{\max}}{2d_{\min}}, \frac{3+\sqrt{n}}{4}\right\},$$

where $d_{\min} = \min\{d_1, \ldots, d_n\}$ and $d_{\max} = \max\{d_1, \ldots, d_n\}$.

Proof. We show the two bounds in two separate calculations. First we show the bound $1/2 + d_{\max}/2d_{\min}$ by manipulating and estimating the term $\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*}$. The other bound, $(3+\sqrt{n})/4$, is then shown by using the method of Lagrange multipliers, cf. Chapter 8. For both calculations we need the ratio $\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*}$. Using Lemma 5.2.2, Lemma

5.2.3, and the Ratio $R_i = d_{\max}/d_i$ we obtain

$$\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*} = \frac{\sum_{i=1}^n R_i^{-1} + R_i^{-2}}{2\sum_{i=1}^n R_i^{-2}}. \tag{5.24}$$

1. Bound $1/2 + d_{\max}/2d_{\min}$

   This bound is obtained by estimating the performance ratio. Modification of the term $1/2+d_{\max}/2d_{\min}$ with $R_{\max} = d_{\max}/d_{\min}$ yields

   $$\begin{aligned}\frac{1}{2} + \frac{d_{\max}}{2d_{\min}} &= \frac{\frac{d_{\max}}{R_{\max}} + d_{\max}}{2\frac{d_{\max}}{R_{\max}}} = \frac{1+R_{\max}}{2} \\ &= \frac{R_{\max}^{-2} + R_{\max}^{-1}}{2R_{\max}^{-2}}.\end{aligned} \tag{5.25}$$

   By using the performance ratio (5.24) and the stated bound (5.25) we show

   $$\frac{\sum_{i=1}^n R_i^{-1} + R_i^{-2}}{2\sum_{i=1}^n R_i^{-2}} \leq \frac{R_{\max}^{-2} + R_{\max}^{-1}}{2R_{\max}^{-2}},$$

   which means that the performance ratio is upper bounded by the term (5.25). Notice that

   $$\begin{aligned}&\frac{\sum_{i=1}^n R_i^{-1} + R_i^{-2}}{2\sum_{i=1}^n R_i^{-2}} - \frac{R_{\max}^{-2} + R_{\max}^{-1}}{2R_{\max}^{-2}} \\ &= \frac{R_{\max}^{-2}\sum_{i=1}^n \left(R_i^{-1} + R_i^{-2}\right) - (R_{\max}^{-2} + R_{\max}^{-1})\sum_{i=1}^n R_i^{-2}}{2R_{\max}^{-2}\sum_{i=1}^n R_i^{-2}} \\ &= \frac{R_{\max}^{-2}\sum_{i=1}^n R_i^{-1} - R_{\max}^{-1}\sum_{i=1}^n R_i^{-2}}{2R_{\max}^{-2}\sum_{i=1}^n R_i^{-2}} \\ &= \frac{R_{\max}^{-1}\sum_{i=1}^n \left((R_{\max}^{-1} - R_i^{-1})R_i^{-1}\right)}{2R_{\max}^{-2}\sum_{i=1}^n R_i^{-2}} \leq 0,\end{aligned}$$

   since $R_{\max} \geq R_i$ for all i. Then, the first part of the statement follows.

2. Bound $(3+\sqrt{n})/4$

This bound follows from using the method of Lagrange multipliers (cf. Chapter 8) for the performance ratio (5.24). We show that for $F = (3 + \sqrt{n})/4$

$$\frac{\mathrm{ADT}(X(t))}{\mathrm{ADT}^*} = \frac{\sum_{i=1}^{n} R_i^{-1} + R_i^{-2}}{2\sum_{i=1}^{n} R_i^{-2}} \leq F$$

is satisfied. Without loss of generality we can say that camera 1 has the largest cluster, which yields $R_1 = 1$ and therefore

$$\frac{\sum_{i=1}^{n} R_i^{-1} + R_i^{-2}}{2\sum_{i=1}^{n} R_i^{-2}} = \frac{2 + \sum_{i=2}^{n} R_i^{-1} + R_i^{-2}}{2 + 2\sum_{i=2}^{n} R_i^{-2}} \leq F. \qquad (5.26)$$

This can be formulated as the optimization problem

$$\begin{aligned} &\text{minizie} && F \\ &\text{subject to} && 2 + \sum_{i=2}^{n}(R_i^{-1} + R_i^{-2}) - F(2 + 2\sum_{i=2}^{n} R_i^{-2}) \leq 0 \\ & && F \geq 1. \end{aligned}$$

Two comments about the constraints of the optimization problem follow. The first constraint,

$$2 + \sum_{i=2}^{n}(R_i^{-1} + R_i^{-2}) - F(2 + 2\sum_{i=2}^{n} R_i^{-2}) \leq 0,$$

follows from Equation (5.26), and means that the performance ratio $\mathrm{ADT}(X(t))/\mathrm{ADT}^*$ must be smaller than a certain factor F for all n and R_i. The second constraint means, that the factor F must be greater than 1, because otherwise the equal-waiting trajectory would have a lower performance than the lower bound.

This optimization problem can be solved with the method of Lagrange multipliers. The Lagrange function for this problem is

$$L = F + \lambda(2 + \sum_{i=2}^{n}(R_i^{-1} + R_i^{-2}) - F(2 + 2\sum_{i=2}^{n} R_i^{-2})),$$

with the partial derivatives

$$\frac{\partial L}{\partial R_i} = \lambda(-R_i^{-2} - 2R_i^{-3} + 4FR_i^{-3}) = 0, \tag{5.27}$$

$$\frac{\partial L}{\partial \lambda} = 2 - 2F + \sum_{i=2}^{n} R_i^{-1} + (1-2F)\sum_{i=2}^{n} R_i^{-2} = 0 \tag{5.28}$$

Solving Equation (5.27) for R_i leads to

$$R_i = 4F - 2.$$

Inserting this result into (5.28) and simple manipulation yields,

$$\begin{aligned} 0 &= 2 - 2F + \sum_{i=2}^{n}(4F-2)^{-1} + (1-2F)\sum_{i=2}^{n}(4F-2)^{-2} \\ &= 2 - 2F + (n-1)(4F-2)^{-1} + (1-2F)(n-1)(4F-2)^{-2} \\ &= (2-2F)(4F-2)^2 + (n-1)(4F-2) + (1-2F)(n-1) \\ &= -32F^3 + 64F^2 + F(2n-42) + 9 - n. \end{aligned} \tag{5.29}$$

By solving Equation (5.29) subject to F, we obtain the three solutions

$$F_1 = \frac{1}{2},$$

$$F_2 = \frac{3+\sqrt{n}}{4},$$

$$F_3 = \frac{3-\sqrt{n}}{4}.$$

Since $n \geq 1$, $F_1 \leq 1$ and $F_3 \leq 1$. This leads to the unique solution

$$F = \frac{3+\sqrt{n}}{4}.$$

This result concludes the statement. □

With the proof Lemma 5.2.2, 5.2.3 and 5.2.4 the lower bound performance, the equal-waiting trajectory performance and the constant factor approximation of the eqaul-waiting trajectory are proofed. This concludes the proof of Theorem 3.0.1.

Chapter 6

Partial homogenous field of view

In this chapter we study a special case. For this special case the lengths of the clusters are restricted by $d_{\max}/d_{\min} < 2$, but the number of cameras can be arbitrary. This restriction means, that every camera can just move once back and forth between the boundaries within one period. Since the cluster lengths are not arbitrary, we call this special case *partial homogenous field of view*. For the partial homogenous field of view case we state a parameterized model and cost function, which can be solved with numerical optimization tools for arbitrarily n. The model and the cost function can be developed, because of the cluster length restriction all possible movements of the cameras are known, and therefore the trajectory can be expressed with a known number of parameters. On the other hand, for arbitrary cluster lengths we do not know how often the cameras will sweep back and forth between the cluster extremes within a period, and therefore we cannot create such a model. In the following subsection we first explain the model for the cost function and in the second subsection we state and derive the parameterized cost function.

6.1 Parameterized model

The model for the partial homogenous field of view for 4 cameras is illustrated in Fig. 6.1. Since the period length is given by $T = 2d_{\max}$ the movement of the camera with $d_{\max}$, in our model camera 1, is given by moving back and forth with maximum speed between the boundaries. The movement of camera 2 during one period can be described with 4 catch-points, because it is known from Lemma 5.2.1 that 3 consecutive catch-points on one boundary are never optimal . Starting from a catch-point at the left boundary at time $t_{1,2}$ this is then two right catch-points at time $t_{2,3}$ and $t_{2,5}$ and then a left catch-point at time $t_{2,7}$. The trajectory between the catch-points is known from Lemma 5.1.3. The movement of the camera 3 and all following cameras can then be described by 5 catch-points, as it is illustrated in the model. The trajectory of each camera can then be parameterized, with parameters that describe the distance between the catch-points. The movement of camera 1 is fixed and therefore no parameter is needed. For each of the other cameras 3 parameters are needed for a complete description of the trajectory. Observe that the distance between time $t_{3,7}$ and $t_{3,9}$ is the same as the distance between $t_{2,3}$ and $t_{2,5}$ and therefore camera 4 needs only 3 parameters for a complete description of the trajectory.

Notice that the model in Fig 6.1 assumes that the camera with the largest cluster is at an end of the open path. If the camera with $d_i = d_{\max}$ would not be at one of the extremes of the path, the model can still be used to calculate the performance, because the movement of the camera with $d_i = d_{\max}$ is a priori fix and the movements of the cameras left and right of this camera are independent of each other. Therefore, the calculation can be done independently for the right and left side.

In the following subsection we derive the parameterized cost function for this model.

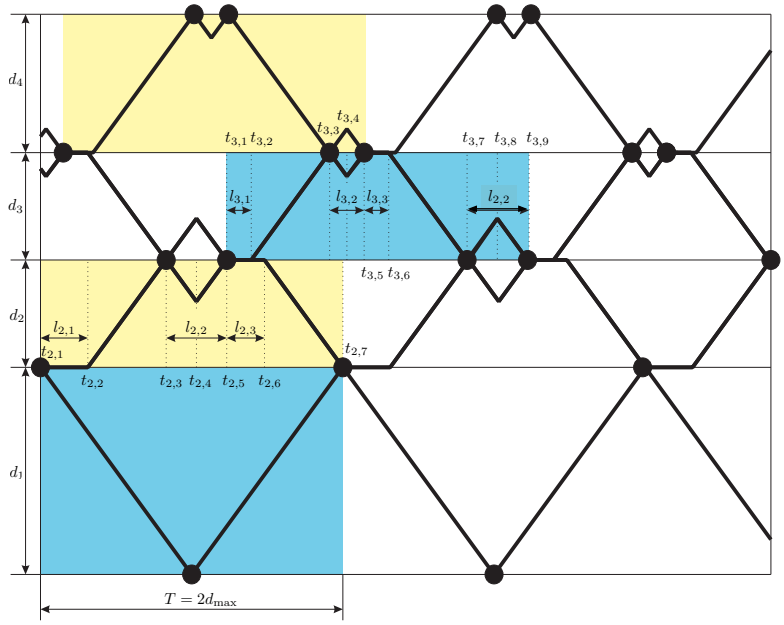


Figure 6.1: Partial homogenous field of view model. The model of 4 cameras with a partial homogenous field of view, that means $d_{\max}/d_{\min} < 2$, is illustrated. The color boxes illustrate a period for each cluster, and the black dots the catch-points of the cameras.

6.2 Cost function

The following Lemma states the performance function for the partial homogenous field of view case.

Lemma 6.2.1 (Partial homogenous field of view performance)**.** *Let* $p = \{l_{2,1}, l_{2,2}, l_{2,3}, l_{3,1}, \ldots, l_{n,1}, l_{n,2}, l_{n,3}\}$ *be the parameterized description of the team trajectory X(t) as in Fig. 6.1 for n cameras with cluster lengths* $d_1, \ldots, d_n$*, where* $d_1 = \max_i d_i$ *and* $d_1/d_i < 2$ *for all*

i. Then the average detection can be calculated as

$$\mathrm{ADT}(p) = \frac{1}{LT}\Bigg(2d_1^3 + \sum_{i=2}^{n}\Bigg[2d_i^3 + 3d_i^2(d_1 - d_i) - \frac{1}{4}(l_{i,1}l_{i-1,2}^2 + l_{i,3}l_{i,2}^2) + d_i\left(\frac{1}{2}(l_{i,1}^2 + l_{i,3}^2) + \frac{1}{4}(l_{i,2}^2 + l_{i-1,2}^2) + l_{i,1}l_{i-1,2} + l_{i,2}l_{i,3}\right)\Bigg]\Bigg), \tag{6.1}$$

where $L = \sum_{i=1}^{n} d_i$, $T = 2\max\{d_1, \ldots, d_n\}$, *and* $l_{1,2} = 0$.

Proof. From Lemma 5.1.1 Equation (5.1) it is known that the performance of all clusters equals the sum of the single clusters, i.e

$$\begin{aligned}\mathrm{ADT}(p) &= \frac{1}{LT}(\sum_{i=1}^{n}\mathrm{ADT}_i(p)) \\ &= \frac{1}{LT}(\mathrm{ADT}_1(p) + \mathrm{ADT}_2(p) + \sum_{i=3}^{n}\mathrm{ADT}_i(p)).\end{aligned} \tag{6.2}$$

We show now that

$$\mathrm{ADT}_1(p) = 2d_1^3, \tag{6.3}$$

$$\mathrm{ADT}_2(p) = 2d_2^3 + 3d_2^2\,(d_1 - d_2) - \frac{1}{4}l_{2,3}l_{2,2}^2 + d_2\left(\frac{1}{2}(l_{2,1}^2 + l_{2,3}^2) + \frac{1}{4}l_{2,2}^2 + l_{2,2}l_{2,3}\right), \tag{6.4}$$

$$\mathrm{ADT}_i(p) = 2d_i^3 + 3d_i^2\,(d_1 - d_i) - \frac{1}{4}(l_{i,1}l_{i-1,2}^2 + l_{i,3}l_{i,2}^2) + d_i\left(\frac{1}{2}(l_{i,1}^2 + l_{i,3}^2) + \frac{1}{4}(l_{i,2}^2 + l_{i-1,2}^2) + l_{i,1}l_{i-1,2} + l_{i,2}l_{i,3}\right), \tag{6.5}$$

for $i = 3, \ldots, n$, which leads with Equation (6.2) to the statement. The cluster performances of Equation (6.3), (6.4), and (6.5) can be calculated with the performance Equation (5.5) from the proof of Lemma 5.2.1, which is

$$\mathrm{ADT}_i(p) = \sum_{j=1}^{\bar{j}}\int_{t_{i,j}}^{t_{i,j+1}} x_{i,j}(t)(\theta_{i,j}^l - \theta_{i,j}^r) + d_i(\theta_{i,j}^r - t)dt.$$

The clusters $i = 3, \ldots, n$ have the most general trajectory. The trajectory of the other two clusters, cluster 1 and 2 can be expressed as a special case of this parameterized trajectory. For cluster 1 that is setting $l_{1,1} = l_{1,2} = l_{1,3} = l_{0,2} = 0$, and for cluster 2 it is setting $l_{1,2} = 0$. Therefore, we calculate the performance of a cluster $i = 3, \ldots, n$ and then derive the performances of cluster 1 and 2.

To evaluae Equation (6.2) for a cluster $i = 3, \ldots, n$, the cluster is split up into $\bar{j} = 8$ intervals, and it is evaluate with the following variables:

$$
\begin{aligned}
&t_{i,1} = 0, && t_{i,2} = t_{i,1} + l_{i,1}, && t_{i,3} = t_{i,2} + d_i,\\
&t_{i,4} = t_{i,3} + \frac{1}{2} l_{i,2}, && t_{i,5} = t_{i,4} + \frac{1}{2} l_{i,2}, && t_{i,6} = t_{i,5} + l_{i,3},\\
&t_{i,7} = t_{i,6} + d_i, && t_{i,8} = t_{i,7} + \frac{1}{2} l_{i-1,2}, && t_{i,9} = t_{i,8} + \frac{1}{2} l_{i-1,2},
\end{aligned}
$$

and

$$
\begin{aligned}
&x_{i,1} = 0, && x_{i,2} = t - l_{i,1},\\
&x_{i,3} = -t + l_{i,1} + 2d_i, && x_{i,4} = t - l_{i,1} - l_{i,2},\\
&x_{i,5} = d_i,\\
&x_{i,6} = -t + l_{i,1} + l_{i,2} + l_{i,3} + 2d_i,\\
&x_{i,7} = t - l_{i,1} - l_{i,2} - l_{i,3} - 2d_i,\\
&x_{i,8} = -t + l_{i,1} + l_{i,2} + l_{i,3} + l_{i,4} + 2d_i,
\end{aligned}
$$

and

$$
\begin{aligned}
\theta^l_{2,j} &= t_{i,7}, && \text{for } j = 1, \ldots, 6,\\
\theta^l_{2,j} &= t_{i,9}, && \text{for } j = 7, 8,
\end{aligned}
$$

and

$$
\begin{aligned}
\theta^l_{2,j} &= t_{i,3}, && \text{for } j = 1, 2,\\
\theta^l_{2,j} &= t_{i,5}, && \text{for } j = 3, 4,\\
\theta^l_{2,j} &= t_{i,9} + t_{i,3}, && \text{for } j = 5, \ldots, 8.
\end{aligned}
$$

Evaluating Equation (5.5) with these parameters yields

$$\begin{aligned}\mathrm{ADT}_i(p) = {} & 2d_i^3 + 3d_i^2\,(d_1 - d_i) - \frac{1}{4}(l_{i,1}l_{i-1,2}^2 + l_{i,3}l_{i,2}^2) \\ & + d_i\left(\frac{1}{2}(l_{i,1}^2 + l_{i,3}^2) + \frac{1}{4}(l_{i,2}^2 + l_{i-1,2}^2) + l_{i,1}l_{i-1,2} + l_{i,2}l_{i,3}\right),\end{aligned}$$

which is equivalent to Equation (6.5). To obtain the performance of cluster $i = 2$ we set $l_{1,2} = 0$. This parameter is zero, because the period of the trajectory ends at time $t_{i,7}$ and the movement to time $t_{i,8}$ and $t_{i,9}$, which is described by the parameter $l_{i-1,2}$, does not exist. Inserting $i = 2$ and $l_{1,2} = 0$ in Equation (6.5) leads to

$$\begin{aligned}\mathrm{ADT}_2(p) = {} & 2d_2^3 + 3d_2^2\,(d_1 - d_2) - \frac{1}{4}l_{2,1}l_{1,2}^2 \\ & + d_2\left(\frac{1}{2}(l_{2,1}^2 + l_{2,3}^2) + \frac{1}{4}l_{2,2}^2 + l_{2,2}l_{2,3}\right),\end{aligned}$$

which is equivalent to (6.4). The trajectory of cluster $i = 1$ can be described with the parameters $l_{1,1} = l_{1,2} = l_{1,3} = l_{0,2} = 0$ (cf. Fig. 6.1). Inserting these values into Equation (6.5) yields

$$\mathrm{ADT}_1(p) = 2d_1^3,$$

which is equivalent to Equation (6.3). Hence, the statement is true. □

By minimizing Equation (6.1) from Lemma 6.2.1 subject to the parameters $p = \{l_{2,1}, l_{2,2}, l_{2,3}, l_{3,1}, \ldots, l_{n,1}, l_{n,2}, l_{n,3}\}$ it is possible to obtain the trajectory with the optimal average detection time. Because of the terms $-1/4l_{i,1}l_{i-1,2}^2$ and $d_i l_{i-1,2}^2$ the performance of two neighboring clusters are coupled. This means, the clusters cannot be optimized on their own. Since we could not find a closed form solution for the case $n \geq 4$, we cannot provide a general analytical solution for this optimization problem. However, this optimization problem should be solvable with numerical methods, but since we did not investigate this, further research is needed in that direction. Moreover, since the equal-waiting trajectory for this special case is within factor 3/2 of the optimum, the precise solution is not very interesting.

Chapter 7

Conclusion

7.1 Summary

In this work we address the problem of surveilling an one dimensional open path by means of a team of autonomous cameras against smart and moving intruders. The smart intruders have full knowledge about the cameras movement and they plan their movement such that they are undetected as long as possible. As performance function for this problem we define and adopt two criteria, the worst-case detection time and the average detection time criterium. The worst-case detection time measures how long an intruder can hide in the worst case, and the average detection time criterium measures how long the average intruder remains undetected.

For the general case, that is an arbitrary number of cameras and arbitrary cluster lengths, we propose the equal-waiting trajectory as a solution. This trajectory has an optimal worst-case detection time and is within a constant factor of the optimum regarding the average detection time criterium. For the computation of the equal-waiting trajectory only local information and the longest cluster length is necessary. Additionally, we design a distributed feedback algorithm, which steers the cameras towards the equal-waiting trajectory. This algorithm is very easy to implement and only neighboring cameras need to communicate if they reach the extreme of their observed cluster. In a simulation we show, that the trajectory solution of the pro-

posed algorithm converges to the equal-waiting trajectory and that the algorithm is robust against cameras failures and motion uncertainties.

For a special case with length restrictions for the observed clusters, we present a parameterized trajectory and cost function that exactly describes the performance of the average detection time. The minimization of this cost function with respect to the parameters, leads to the optimal trajectory. However, we cannot propose a closed form for this optimization problem, but it can be solved numerically.

7.2 Outlook

A variety of unaddressed problems about this topic remains. First, we assumed that all cameras have the same speed. It would be interesting to investigate the case that the cameras have different speed. Second, it would be interesting to work on the optimal cluster partitioning of the cameras. In this work we assumed that the clusters of the cameras are non overlapping, therefore every camera has to observe the whole cluster. If the clusters are overlapping, there must be an optimal partitioning of these clusters, with respect to the average detection time criteria. Third, the synchronization problem of cameras would be very interesting for planar cases. In the first step it could be extended to tree graphs, which could be interpreted as corridors of a building, and in a further step to general graphs.

Chapter 8

Appendix

8.1 Method of Lagrange multipliers

The method of Lagrange multipliers is a mathematical optimization tool, which yields a necessary condition for a maxima or minima of a differentiable function subject to equality constraints [10]. Such an optimization problem can be written as

$$\begin{aligned} &\text{minizie} && f(x) \\ &\text{subject to} && g(x) = c, \end{aligned}$$

where $x \in \mathbb{R}^n$, $f(x)$ and $g(x)$ are differentiable functions $\mathbb{R}^n \to \mathbb{R}$ and c is a constant.

The idea of the Lagrange method is that at an extrema, the counter lines of $f(x)$ and $g(x)$ must be tangential. Since the gradient of a function is perpendicular to the contour lines, the functions $f(x)$ and $g(x)$ are parallel if and only if the gradients of $f(x)$ and $g(x)$ are parallel. This yields the necessary condition for an extrema

$$\nabla f(x) = -\lambda \nabla g(x),$$

where $\nabla f(x) = \text{grad}(f(x))$, and λ is a scalar, which is needed because the gradients may be parallel but still have different lengths. To combine these two conditions, $g(x) = c$ and the gradients of $f(x)$ and $g(x)$ are parallel, into one equation, we introduce the Lagrange

function

$$L(x, \lambda) = f(x) - \lambda(g(x) - c)$$

and solve for

$$\nabla_{x,\lambda} L(x, \lambda) = 0.$$

Notice, the partial derivate of L with respect to λ, that is $\nabla_\lambda L(x, \lambda) = 0$, implies $g(x) = c$, and therefore it is ensured that the constraint is always satisfied.

Bibliography

[1] J. Clark and R. Fierro, "Mobile robotic sensors for perimeter detection and tracking," *ISA Transactions*, vol. 46, no. 1, pp. 3–13, 2007.

[2] D. B. Kingston, R. W. Beard, and R. S. Holt, "Decentralized perimeter surveillance using a team of UAVs," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1394–1404, 2008.

[3] S. Susca, S. Martínez, and F. Bullo, "Monitoring environmental boundaries with a robotic sensor network," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 2, pp. 288–296, 2008.

[4] Y. Elmaliach, A. Shiloni, and G. A. Kaminka, "A realistic model of frequency-based multi-robot polyline patrolling," in *International Conference on Autonomous Agents*, Estoril, Portugal, May 2008, pp. 63–70.

[5] A. Machado, G. Ramalho, J. D. Zucker, and A. Drogoul, "Multi-agent patrolling: An empirical analysis of alternative architectures," in *Multi-Agent-Based Simulation II*, ser. Lecture Notes in Computer Science. Springer, 2003, pp. 155–170.

[6] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology*, Beijing, China, Sept. 2004, pp. 302–308.

[7] F. Pasqualetti, A. Franchi, and F. Bullo, "On cooperative patrolling: Optimal trajectories, complexity analysis and approximation algorithms," *IEEE Transactions on Robotics*, Jan. 2011, submitted.

[8] M. Baseggio, A. Cenedese, P. Merlo, M. Pozzi, and L. Schenato, "Distributed perimeter patrolling and tracking for camera networks," in *IEEE Conf. on Decision and Control*, Atlanta, GA, USA, 2010, pp. 2093–2098.

[9] R. Carli, A. Cenedese, and L. Schenato, "Distributed partitioning strategies for perimeter patrolling," in *American Control Conference*, San Francisco, CA, USA, 2011, pp. 4026–4031.

[10] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.