TÉCNICO LISBOA

Cooperating Smart Cameras

Tiago Miguel de Oliveira Marques

Thesis to obtain Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Professor José António da Cruz Pinto Gaspar

Examination Committee

Chairperson: Professor João Fernando Cardoso Silva Sequeira
Supervisor: Professor José António da Cruz Pinto Gaspar
Member: Professor Jorge dos Santos Salvador Marques

November 2015

Resumo

A crescente necessidade de sistemas de vigilância em espaços públicos, bem como os avanços recentes em compressão de vídeo e comunicação em rede em sistemas embebidos tornaram as redes de vigilância comuns. Faltam, no entanto, metodologias para tratar automaticamente a quantidade massiva de dados capturados por cada rede de câmaras. De modo a detectar de modo eficiente múltiplos alvos, redes the camaras autonomas requerem algoritmos de detecção, seguimento, escalonamento e controlo adequados. Algoritmos de detecção e seguimento extraem features das imagens adquiridas que são usadas para actualizar o estado dos alvos visuais. Algoritmos de escalonamento atribuem alvos visuais a camaras. Metodologias de controlo geram referências precisas de orientação e zoom.

Nesta dissertação, elementos de Teoria de Informação são usados para resolver o problema de controlo e escalonamento. Cada alvo observável no ambiente corresponde a uma fonte de informação. Como tal, cada observação corresponde a um ganho de informação que fará reduzir a incerteza no estado do alvo. São apresentadas várias funções de observação para ajudar a evitar níveis extremos de zoom, suavizando o ganho de informação. Por último, é feita uma proposta de optimização expedita do ganho de informação para redes de camaras.

Palavras chave: Câmera pan tilt zoom, Teoria da Informação, Ganho de Informação, Função de observação

Abstract

The increase need of surveillance in public places and recent technological advances on embedded video compression and communications made camera networks ubiquitous.There are however missing methodologies for watching so much data captured by so many cameras having few staff. In order to effectively track multiple targets, autonomous active camera networks require adequate detection, tracking, scheduling and control methodologies. Detection and tracking algorithms extract features from captured imagery which are used to build state posterior of the visual targets. Scheduling algorithms assign visual targets to cameras. Control methodologies set precise orientation and zoom references of the cameras

In this thesis, Information Theory elements are used in order to solve the scheduling and control problem. Each observable target in the environment corresponds to a source of information for which an observation corresponds to a reduction of the uncertainty and, as such, a gain in the information. Observation functions are shown to help avoiding extreme zoom levels while keeping smooth information gains.

At last, an agile optimization procedure proposal is made, in order to maximize the information gain for camera networks.

Keywords: Pan and Tilt Zoom Camera, Information Theory, Information Gain, Observation Functions

Agradecimentos

Esta dissertação certamente não seria a mesma sem a contribuição especial de uma série de pessoas, às quais expresso o meu sincero agradecimento.

Ao Professor José Gaspar, por ter sido incansável ao longo do desenvolvimento desta dissertação na partilha de conhecimento e discussão de ideias

Ao Pedro Silva, pela amizade e pela partilha de conhecimento no início desta dissertação.

Ao Luka Lukic, pela amizade e pelo tempo despendido em troca de ideias e trabalho de revisão.

Por último, aos meus amigos, familiares e colegas de trabalho, por me incentivarem a continuar e alcançar os meus objetivos.

A todos um bem-haja e um grande obrigado

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The increasing need of surveillance in public places and recent technological advances on embedded video compression and communications made camera networks ubiquitous.There are however missing methodologies for watching so much data captured by so many cameras having few staff. In this thesis, Information Theory elements are used in order to solve the scheduling and control problem in smart camera networks.

1.1 Related Work

The first autonomous surveillance systems were composed of multiple static cameras, cooperating to solve complex tasks such as tracking moving objects. The need for considering overlapping field of views during camera deployment and wide field of views, resulting in low resolution images, led to the deployment of pan-tilt-zoom cameras in modern surveillance systems.

Some new architectures appeared, such as master-slave configurations, and cooperative smart networks [15]. In the master-slave configuration, static cameras are used for event detection to direct the PTZ camera to the target of interest, in contrast with more complex architecture, in which both static and PTZ camera streams are used for event analysis. This way, the global state of these systems is composed of both the individual state of each target and the camera state. In both architectures there is a need for target tracking methodologies and, in the more complex cooperative architectures, there is also a need for camera management methodologies, in order to compute the optimal configuration of the network.

There are many generic methodologies for target tracking. The classic approach, based on the Bayes filter, which can be found all across the literature ([1], [20], [19]), is characterized

at each iteration by the update of the state estimate based on the predicted state given by the motion model of the target and the observations given by the sensor. Another approach is taken in [10], [7], [3] and [2] where, contrary to the recursive approach, the trajectory is estimated in batches, making available both past and future observations for the estimation of the trajectory at a given time.

Regarding camera management, many different approaches are evaluated in the literature. Starzyk *et al* [21] propose a complete system for tracking multiple targets using cooperative cameras. The conflicts in behaviors are resolved using a central computer which combines the desired behaviors in a single behavior which reflects the best compromise between all.

The Multi-Armed Bandit is introduced in [23], [16] and [13] not directly as a camera management system, but as a decision methodology for allocating resources to projects, being them robots which can travel to certain locations in order to discover events or network packets which can be routed to various channels in order to maximize the throughput. This framework can be used to model the problem of camera management. Each camera is thought as a resource and each target as a project. The objective is to allocate cameras (resources) to targets (projects) in order to maximize some measure of reward over time.

Another approach based on the information theory framework is applied to the problem of camera management in [20] and [19], in which the camera parameters (both camera to target assignment and kinematic parameters of the camera) are chosen based on the mutual information gain between the state estimate and the estimate given an observation.

The work done in this thesis is based on previous work using the Information Gain as basis for solving the camera scheduling and control problem.

1.2 Problem Formulation

The difficulties in designing surveillance systems with pan-tilt-zoom cameras involve problems not only in video processing, as with systems consisting in static cameras, but also in camera management in order to capture the maximum information possible of all the targets present in the scene.

In video processing, a common problem is the change of the environment, such as illumination changes, target occlusions and situations of uncertainty in which two or more targets meet in the same location for a period of time, making it hard to distinguish between them. The problem is often solved using motion models to filter the target observations (Extended Kalman Filters, Particle Filter,...) or, when the motion model is too simple, such as the case of a random

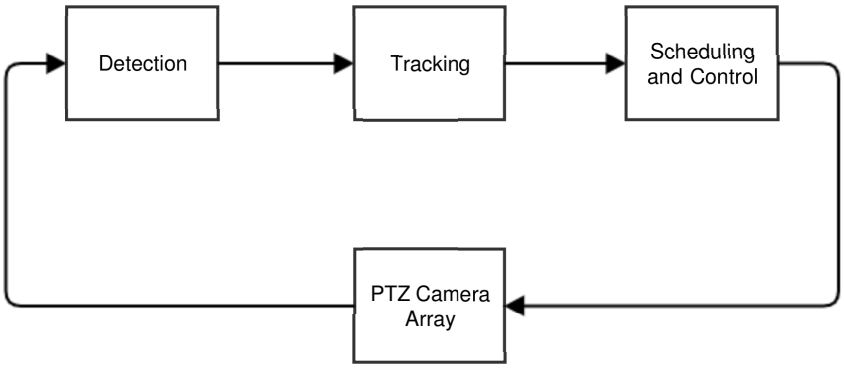


Figure 1.1: Fundamental tasks of a smart surveillance system

walk, using batch processing in the segments of the trajectory observed.

In the camera management domain, the problem is finding the optimal control policies for the system to be able to capture the maximum number of targets with the maximum detail possible. Using as degrees of freedom the pan-tilt-zoom parameters and the scheduling policy chosen for the camera, the challenge is in the tradeoff between the zoom level of each camera and the number of targets being tracked. In the camera management domain, the problem of finding the optimal control policies is solved by selecting targets for each camera based in observations available at the time of decision, let it be distance, variance or mutual information gain.

1.3 Thesis Structure

This document is organized as follows. Chapter 1 introduces the problem to approach in the thesis. In particular it presents a short discussion on the state of art in dynamic camera scheduling. Chapter 2 describes some related work in the fields of target tracking and camera scheduling. Chapter 3 describes the tracking methodologies used in this thesis. Chapter 4 describes the Information Theoretic approach for camera scheduling in which this work is based, along with the contribution on the analysis of the effect of different observation functions in the cost function. Chapter 5 provides an overview of the experiments executed as well as the results attained. Finally, chapter 6 summarizes the work performed and highlights the main achievements of this work, as well as future vectors of exploration which, due to the limited time of this thesis, were not explored.

1.4 Publications

The work developed in this thesis has been partially published in the paper [14] accepted for publication at the Second Iberian Robotics Conference, Robot'2015. This paper compares the effect of observation functions into the PTZ cameras scheduling and control problem based in an information theoretic approach.

Chapter 2

Video Surveillance Systems and Methodologies

The following chapter shows related work in both detection and tracking systems and camera scheduling systems. The chapter starts by describing typical surveillance systems architecture, from networks of passive wide field-of-view camera networks to cooperative smart cameras. Section 2.2 contains a description of two works in target tracking. Section 2.3 contains works in the camera scheduling field. Starting by describing a camera scheduling system which uses *ad-hoc* rules as performance measurements for the observation tasks, the chapter ends with the description of a decision framework with potential application in camera scheduling systems, the Multi-Armed Bandit (MAB).

2.1 Generic Surveillance System

The main goal of most surveillance systems is to detect and track moving targets of interest (pedestrians, vehicles, ...) in the environment with the largest resolution possible.

Traditional surveillance systems are composed of fixed wide field-of-view cameras which allow to cover large areas. However, due to the wide field of view, this systems fail to capture high resolution imagery of the targets of interest. Moreover, since the cameras are static, these systems tend to be dependent on camera deployment instructions.

With the use of PTZ cameras in surveillance systems, it became possible to adapt the system covered area by managing each camera field of view. This feature mitigates problems like occlusions and low target resolutions However, despite the qualities of the dynamic camera networks, they require the development of control methodologies in order to choose which pan,

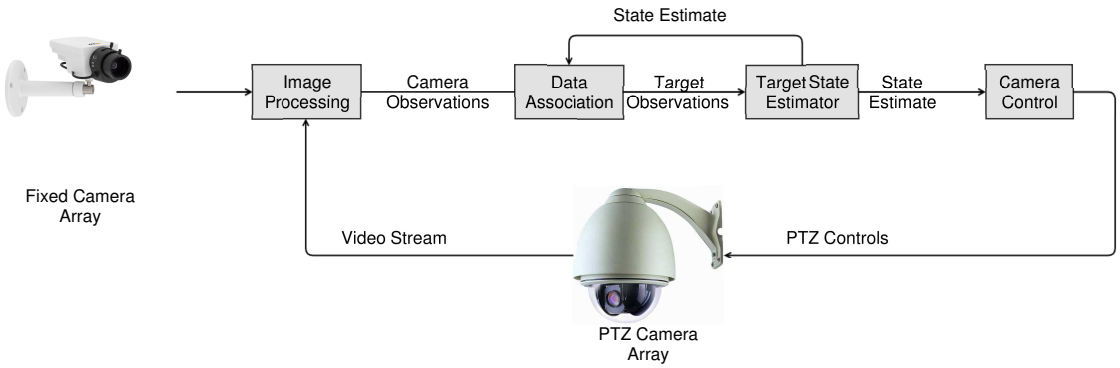


Figure 2.1: Generic surveillance system

tilt and zoom parameters to send to the camera. The problem becomes more difficult when we need to consider multiple cameras with possible overlapping fields of view. In particular we want to avoid behaviors such as having all cameras targeting the same target in detriment of the remaining ones or having all cameras in a wide field of view configuration, as it would happen in a static camera network.

Many control methodologies exist for incorporating PTZ cameras in surveillance networks [15]. One of the simplest approach is the master slave configuration, in which an array of static cameras are responsible for the image processing and event analysis. The resulting information is then used to control the PTZ cameras in order to acquire higher resolution imagery.

In more advanced networks both dynamic and static cameras perform image processing and event analysis. Figure 2.1 shows a typical surveillance system with centralized control. Each camera, both fixed and dynamic, feeds the system with image frames. Each class of camera can have different responsibilities. For instance, fixed cameras can be deployed pointing to possible entry points of the area under surveillance and dynamic cameras be responsible for keeping track of the existing targets.

Each observation is matched against the knowledge of the existing targets and used to update the state estimate. Finally, the combined knowledge of each target, state and uncertainty, is used to choose the controls to send to each camera.

2.2 Related Work in Detection and Tracking

When tracking objects in the environment with cameras (static or fixed) several challenges arise, such as changes in the illumination and the partial or total occlusion of the targets being tracked.

The following section describes some state of art algorithms in robust target tracking.

2.2.1 Probabilistic Occupancy Map

In [7] is proposed a multi-camera tracking algorithm using probabilistic occupancy maps. The goal is to track an unknown number of pedestrians from a sequence of synchronized video streams. The system is composed by two parts. The first part consists in an object detector which computes the occupancy map from binary images obtained using background subtraction. In the second part these probabilities are combined with the color and motion model of the objects.

Unlike the algorithms based on Bayesian filtering, that iteratively update the state estimates at each frame (and may therefore fail when not able to detect the object in several consecutive frames [7]), this system can handle those situations since it runs on batches of frames and, for each batch, takes into account the previous and the posterior frames.

Let each video stream be composed of T frames, each with C images, one for each camera. The ground plane is divided in G regularly spaced 2-D locations plus a virtual location H to model entrances and exits from the area. Let $\boldsymbol{L_t} = \{L_t^1, ..., L_t^N\}$ be the location of each of the individuals at time t[1], and $\boldsymbol{I_t} = \{I_t^1, ..., I_t^C\}$ the images from the current batch acquired at time t for each of the C cameras, then the goal is to maximize the likelihood of the individuals location given the batch of images acquired until time t

$$p(\boldsymbol{L_1}, ..., \boldsymbol{L_T} | I_1, ..., I_T). \tag{2.1}$$

In [7] the maximization of (2.1) is made by estimating each of the trajectories independently and sequentially, ordering by confidence level to reduce errors in position. This independence conditions the estimation of later trajectories by prohibiting two objects to occupy the same place at the same time.

For estimating a single trajectory from a batch of images, one wants to find the trajectory $(l_1^n, ..., l_T^n)$ of the pedestrian n which maximizes the likelihood $P(L_1^n = l_1^n, ..., L_T^n = l_t^n | I_1, ..., I_t)$. Let $\Phi_t(k)$ be the maximum likelihood of the individual location given the set of images, it is shown in [7] that it can be computed as

[1] Non-visible pedestrians are at the virtual position H

$$\Phi_t(k) = P(I_t|L_t^n = k) \max_{\tau} P(L_t^n = k|L_{t-1}^n = \tau)\Phi_{t-1}(\tau) \tag{2.2}$$

where $P(I_t|L_t^n = k)$ is the appearance model and $P(L_t^n = k|L_{t-1}^n = \tau)$ is the motion model.

For the motion model it is used an exponential distribution

$$P(L_t^n = k|L_{t-1}^n = \tau) = \begin{cases} \frac{1}{Z}e^{-\rho||k-\tau||} & if\ ||k-\tau|| \leq 0 \\ 0 & otherwise \end{cases} \tag{2.3}$$

over $||k - \tau||$, when the current position k is close to the last position τ. A threshold is used to force the probability to zero as the distance increases. In practical terms this means that no target moves at a higher speed than the one specified.

For the appearance model, let X_t^k be the true occupancy in position k at time t, T_t the colors of the pixels inside the blob belonging to a single pedestrian and B_t the binary image generated by background subtraction, then the appearance model can be modeled as

$$P(I_t|L_t^n = k) \ \propto \ P(L_t^n = k|X_t^k = 1, T_t)P(X_t^k = 1|B_t) \tag{2.4}$$

with $P(L_t^n = k|X_t^k = 1, T_t)$ denoting the color model for the pedestrian and $P(X_t^k = 1|B_t)$ the ground plane occupancy.

The color model

$$P(L_t^n = k|X_t^k = 1, T_t) = \frac{P(T_t|L_t^n = k)}{\sum_m P(T_t|L_t^m = k)}, \tag{2.5}$$

with

$$P(T_t|L_t^n = k) = \prod_{c \in C} \prod_{r \in T_i^c(k)} \mu_n^c(r) \tag{2.6}$$

and μ_n^c being the color distribution of the individual n as seen by camera c, is simply the normalized product of the color distributions of the individual n over all cameras regarding the pixels which belong to the pedestrian being tracked.

As for the ground plane occupancy, one wants to compute $P(X|B)$ from the data acquired by background subtraction $P(B|X)$. Assuming that the movement of each individual is independent from the remaining and the binary images B^k generated from each camera are independent, and $P(B|X)$ is defined as

$$P(B|X) = \prod_c P(B^c|X) = \frac{1}{Z} \prod_c e^{-\Psi(B^c, A^c)} \tag{2.7}$$

with Ψ a pseudo-distance operator between two binary images, A^c the image obtained by placing a rectangular shape where each individual is located in the image. The computation of $P(X|B)$ is made by minimizing the Kullback-Leibler divergence

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)} \tag{2.8}$$

between the approximation of $P(X|B)$ and the true distribution.

Finally, the trajectory is estimated using the Viterbi algorithm on the equation (2.2).

K-shortest Paths Optimization

A different approach is made in [3][2], where the problem to generate the trajectories is reformulated as a constrained flow optimization problem which falls in the linear programming domain. But, due to the problem dimension, it is demonstrated that the problem can be solved efficiently using the k-shortest paths algorithm. The only requirement for the algorithm to work is to have as input the occupancy map for each time t in the batch of images to process. This approach, however, completely ignores appearance models, despite that inclusion being possible [2].

As in the method above, the ground plane is discretized in K locations and the time interval in T instants. For any location k, $N(k)$ denotes the vicinity of the location k, that is, the positions to which an object in the location k can move in one time step.

The occupancy over time is modeled as a direct acyclic graph with KT vertices representing each valid position in time and space, and edges which represent valid object motions.

Each vertex i has a label which denotes its occupancy m_i^t at time t, and each edge has a label $f_{i,j}^t$ which denotes the number of objects to move from location i to location j at time $t+1$. The appearance or disappearance of objects is modeled by including two additional vertices v_{source} and v_{sink}, such that v_{source} is connected to any other vertex where there can be new objects and v_{sink} is connected to every vertex where there can disappear existing objects.

Let M_i^t be a random variable which stands for the true presence of an object at location i and time t, and $\rho_i^t = P(M_i^t = 1|I^t)$ an estimate of the marginal posterior probability of the presence of an object then, assuming conditional independence of M_i^t given I_t, the optimization problem can be formulated as

$$m^* = arg \max_{m \in \mathbb{F}} log \prod_{t,i} P(M_i^t = m_i^t | I^t) \tag{2.9}$$

$$= arg \max_{m \in \mathbb{F}} \sum_{t,i} \left(log \frac{\rho_i^t}{1 - \rho_i^t} \right) f_i^t \tag{2.10}$$

$$s.t.\ f_{i,j}^t \geq 0 \tag{2.11}$$

$$\sum_{j \in N(i)} f_{i,j}^t \leq 1 \tag{2.12}$$

$$\sum_{j \in N(i)} f_{i,j}^t - \sum_{k:i \in N(k)} f_{k,i}^{t-1} \leq 0 \tag{2.13}$$

$$\sum_{j \in N(v_{source})} f_{v_{source},j} - \sum_{k:v_{sink} \in N(k)} f_{k,v_{sink}} \leq 0 \tag{2.14}$$

with $\mathbb{F}$ being the set of all admissible maps. The equation (2.11) forces the flows to be positive, the equation (2.12) forces each vertex to have at most one object, (2.13) is a mass conservation constraint and finally (2.14) ensures that all flows created at v_{source} will eventually end up in v_{sink}.

To solve this problem as a Linear Problem, a relaxation is made and non integer values for $f_{i,j}^t$ are admitted. Due to *total unimodularity*, it can be proved that the relaxed LP formulation converges to the non-relaxed Integer Problem solution. Despite this property, the dimension of the state space makes the problem intractable for larger problems.

However, the problem can be efficiently solved by using the K-Shortest Paths algorithm with each edge $e_{i,j}^t$ having a cost of

$$c(e_{i,j}^t) = -log \left(\frac{\rho_i^t}{1 - \rho_i^t} \right). \tag{2.15}$$

2.2.2 Bayesian Network Based Tracking

A similar approach as the one described in section 2.2.1 is proposed in [11], in the way that the trajectories are also generated in batches. This approach is robust to occlusions of the target being tracked and indecisions which arise from targets joining and/or leaving groups. This is achieved by using a bayesian network to solve the problem of assigning disrupted segments of trajectory caused by time instances in which there is not a clear view of the target.

The system can be divided in two main components. The first component is responsible for detecting active regions in the image and generating trajectories when no uncertainty exists,

the second component is responsible for resolving the conflicts which the first component was unable to resolve, thus generating the complete trajectories for each target.

Trajectory Segment Estimation

The identification of the active regions in the first component is made using a background subtraction algorithm. For that intent it is used the Gaussian Mixture Model for background subtraction, which models each pixel as an independent random variable with a sum of Gaussians as probability distribution. The pixel is classified as background if the probability of pixel assuming the observed value is higher than a threshold, and foreground otherwise.

A succession of active regions belonging to the same target forms a segment of trajectory, identified by an unique label. When no occlusions or other difficulties occur, the match between active regions can be updated using simple algorithms [2] between frames and, for each trajectory generated, a label is associated to that segment.

Every segment ends whenever the target cannot be identified in the next frame. The match of active regions in consecutive frames is made using a multiple choice algorithm. Let $s_k(t)$ denote the value of the trajectory k at time t and $r_i(t)$ the active region i at time t, the algorithm makes the maximum likelihood estimation of

$$s_u^{t-1} = \arg\max_{s_k^{t-1}} p(r_i^t | s_k^{t-1}) \tag{2.16}$$

and

$$r_v^t = \arg\max_{r_j^t} p(s_u^{t-1} | r_j^t). \tag{2.17}$$

The active region is said to belong to the trajectory segment if both the region and the segment mutually choose each other, that is $r_i^t = r_v^t$. For the computation of the probability distributions in (2.16) and (2.17), metrics such as the relation between the bounding boxes of the active region at time $t-1$ and at time t and the velocity in the image plane of that target are used.

Labeling

Segments of trajectory belonging to the same trajectory (that is, generated by the same target) have different labels. The goal is to find the sets of segment labels which belong to the same trajectory and assigning them the same trajectory label.

[2] More complex algorithms such as Kalman or particle filters could be used at this stage.

This is achieved by generating a bayesian network in which each node corresponds to a trajectory segment computed in the preceding step and each edge corresponds to probable connections between trajectories. The edges are built assuming time coherency, that is, two segments of trajectory which are observed at the same time cannot belong to the same trajectory, and spatial coherency, so that no target can jump from one location to another above a certain threshold. For those which can belong to the same trajectory, an edge with an associated conditional probability is created using metrics such as the dominant colors in the active regions and the distance between trajectories.

The set of admissible trajectory labels for each segment is created by propagating in the network the labels of the parent nodes. The goal is to find the most likely labeling set.

Instead of depending on the local convergence of the classic recursive tracking algorithms upon the occurrence of events which block the view from the target, a whole new segment of trajectory is generated when the target is lost. The link between segments of trajectory is made *a posteriori* using a bayesian network, this way having access not only to the past information (available for recursive algorithms), but also the probable future information of the trajectory.

This freedom comes at the price of a delay in the computation of trajectories. The more the system waits before building the bayesian network, the more precise will be the matching between segments of trajectory (since there is more information available). Nevertheless, if the system waits for too long there can be a considerable delay between the generation of the image and the generation of the corresponding trajectory.

2.3 Related Work in Camera Scheduling

In a single camera surveillance system, we want to use a single camera to track multiple targets. In this case the task planning just needs to know the location of the targets to compute the pan-tilt-zoom camera parameters so that a subset of the existing targets are present in the camera field of view.

Likewise, in a multi camera surveillance system we want to capture in the field of view of each camera a subset of all the targets. However, the existence of multiple cameras raises the need to have a scheduling module to mediate each of the cameras tracking a specific target, preventing situations such as target negligence.

In the following sections several camera scheduling methodologies are described.

2.3.1 Metrics based Planning

Starzyk and Qureshi [21] proposed a layered architecture for an autonomous camera management system for pedestrian tracking, which is capable of capturing varying resolution imagery depending on the number of targets in the same scene. Each layer, named level 0, level 1 or level 2, is made of a collection of behaviors in such a way that higher level behaviors use the information from lower level behaviors for decision making.

Let h_j represent a pedestrian/target in the scene, the camera state is a combination of possible level 2 behaviors $A_i = \{idle, observing(h_j), evaluating(h_j)\}$. As the name says, while in the state $evaluating(h_j)$ the camera is evaluating the success probability of tracking the target h_j, in the state $observing(h_j)$ the camera is actively observing the target h_j. Each camera can have multiple level 2 behaviors as long as it doesn't represent an illogical global state (for instance, the same camera cannot be idle and observing a target at the same time).

Level 2 behaviors share the usage of level 1 behaviors (search, track, fixate, zoom and reset) among each other. This way it arises the need for a behavior arbitration module in order to solve conflicts between level 2 behaviors. Each level 2 behavior sends to the behavior arbitration module its preferred region of interest to evaluate/track. Then, based on the utility of each task, the behavior arbitration module computes the new region of interest as the union of all the level 2 regions of interest that meet the task requirements and has maximum utility. This new region of interest is sent to the level 1 behaviors layer to estimate the camera parameters.

Finding the optimal state sequence for the network is a combinatorial problem. It is solved in [17] by employing a greedy best-first search in the state graph over a shallow time horizon. Each state utility is computed based on the success probability of each level 2 behavior.

Let r denote a task from the set of all the available tasks, both active and potential, for the camera and $P(r)$ the success probability for the task r , the new region of interest is obtained by computing the set of tasks which maximize the joint probability of success, assuming probabilistic independence

$$\begin{aligned} R^* = \arg\max_R \prod_{r \in R} P(r) \\ subject\ to\ arg\ \min_{r \in R} Resolution(r) > threshold \end{aligned} . \tag{2.18}$$

The computation of the success probability, for the case of pedestrian tracking, is addressed by Terzopoulos and Qureshi [17] and [12], where a set of relevance metrics, for each tracking task and for each camera, are used in order to compute the best camera for the tracking task.

The metrics are the following

- **Camera-Pedestrian Distance** r_d: Higher relevance for targets closer to an *optimal* distance d' to the camera
$$r_d = exp\left(-\frac{d-d'}{2\sigma_d}\right) \tag{2.19}$$
with d' the optimal camera-pedestrian distance, computed experimentally;

- **Frontal View Direction** r_γ: Higher relevance for targets facing the camera
$$r_\gamma = exp\left(-\frac{\gamma^2}{2\sigma^2}\right) \tag{2.20}$$
with γ the angle between the fixation vector of the camera and the velocity vector of the pedestrian;

- **PTZ limits** $r_{\alpha\beta\theta}$: Higher relevance for targets far from the turn and zoom limits of the camera
$$r_{\alpha\beta\theta} = exp\left(-\frac{\theta-\theta'}{2\sigma_\theta} - \frac{\alpha-\alpha'}{2\sigma_\alpha} - \frac{\beta-\beta'}{2\sigma_\beta}\right) \tag{2.21}$$
with α and β the pan and tilt angles and θ the zoom required to capture the pedestrian;

- **Observational Range** r_o: Set to null relevance all the tasks outside the observational range of the camera. Equals to 1 if the pan and tilt angles are within the valid values and the distance is less than a maximum distance d_{max}, otherwise;

- **Handoff Success Probability** r_h: Higher relevance for cameras in the vicinity of the camera currently observing the target. This factor is only considered during handoff stages
$$r_h = exp\left(-\frac{\epsilon^2}{2\sigma_\epsilon}\right) \tag{2.22}$$
with ϵ the angle between the fixation vector of the camera in evaluation and the currently observing camera.

The computation of the success probability from the relevancies can be made by choosing a function $f(r_1, ..., r_n)$, with r_k being the k^{th} metric used. Since all the relevancies are, by nature, normalized, a common approach is to simply use the product of all relevancies. Another approach is to use a temporal decay factor to discount old captures of a target, prompting the objective function to recapture the target [12]. The success probability is then augmented as

$$p_{ij}(t) = p(S|c_{ij}, t) = q_{ij}\tau(t - t_{ij}, \sigma_\tau, \Delta t_s, \Delta t_{cut}) \tag{2.23}$$

with t_{ij} the time at which the capture c_{ij} with the camera i of the target j was taken, and τ a decaying weight function of the time elapsed since the first capture

$$\tau(t - t_{ij}, \sigma_\tau, \Delta t_s, \Delta t_{cut}) = \begin{cases} 1, \ t - t_{ij} < \Delta t_s \\ e^{-\frac{t - t_{ij} - \Delta t_s}{\sigma_\tau}}, \ \Delta t_s \leq t - t_{ij} < \Delta t_{cut} \\ 0, \ otherwise \end{cases} \tag{2.24}$$

with Δt_s being the time in which the captures are fully rated and Δt_{cut} the time in which after that the captures do not contribute to the success probability.

Notice that these metrics are relevant for pedestrian tracking, where one wants to capture, for instance, the face for facial recognition. Other applications may required small modifications. For instance, if the goal is to track vehicles it is more useful to prefer targets with the frontal view orthogonal to the camera view, so that it is possible to better estimate the movement of the vehicle.

2.3.2 Multi-Armed Bandit

The multi-armed bandit problems (MAB) are a class of sequential resource allocation problems concerning the allocation of a set of resources to alternate and competing targets across time. When a control action is taken to an arm at time t, the arm transits to a random state x_t and yields a reward depending on the final state. When the MAB action-state sequence is Markovian, then the MAB problem degenerates in a special case of a Markov Decision Process [8].

This formulation is quite similar to the camera scheduling problem. Each arm represents a target being tracked and each camera represents a bandit. In the camera scheduling problem, one wants to estimate which targets to track/arms to play in order to maximize some reward measure, for instance, the overall knowledge of the state of all targets.

Classic MAB Problem

In the classic MAB problem, each arm/process/bandit is described by a pair of random variables $\{X(n), R(X(n))\}$, where $X(n)$ denotes the state of the process after being operated n-times and $R(X(n))$ denotes the reward obtained by operating the machine by the n-th time. At each time step, one controller/processor chooses one from the k machines to operate and receive

the reward for operating that machine. Only one machine can be operated at each time and non-operated machines do not generate reward neither their states evolve.

Let $U(t) = (U_1(t), ..., U_k(t))$ be the control action taken by the controller at time t, $Z_i(t) = (X_i(0), ..., X_i(N_i(t)))$ the sequence of states of the machine i, $W_i(n)$ a random process with known probability distribution and $N_i(t)$ the number of times the machine i has been operated at time t, then each process evolves according to

$$X_i(t+1) = \begin{cases} f_{N_i(t)}(Z_i(t), W_i(N_i(t)),\ if\ U_i(t) = 1 \\ X_i(N_i(t)),\ if\ U_i(t) = 0 \end{cases} \tag{2.25}$$

with

$$R_i(X(N_i(t)), U_i(t)) = \begin{cases} R_i(N_i(t)) + 1,\ if\ U_i(t) = 1 \\ R_i(N_i(t)),\ if\ U_i(t) = 0 \end{cases}. \tag{2.26}$$

The MAB problem then consists in determining the scheduling policy γ that maximizes the functional

$$J^{\gamma} = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \sum_{i=1}^{k} R_i(X_i(N_i(t)), U_i(t)) | Z(0)\right]. \tag{2.27}$$

In the classic MAB problem, only one machine can be operated at each time and non-operated machines contribute with no reward, neither their states evolve. These features imply that an optimal solution can be achieved by forward induction, since decisions not made in the present can be made in the future with the same sequence of rewards.

The optimal policy can be made by first computing the Gittins index v_{X_i} for each machine then operate the machine with highest Gittins index until the accumulated reward for playing that arm achieves the value of v_{X_i}

$$v_{X_i}(x) = \max_{\tau > 0} \frac{\mathbb{E}\left[\sum_{t=0}^{\tau-1} \beta^t R_i(X_i(t)) | x_i(0)\right]}{\mathbb{E}\left[\sum_{t=0}^{\tau-1} \beta^t | x_i(0)\right]}. \tag{2.28}$$

The limitations of the classic MAB problem have led to the emergence of other, more general, MAB variations. However, some of these variations do not retain the properties of classic MAB and thus do not have an index type solution.

Super Process

In the classic MAB problem, each machine accepts binary control input ($U_i(t) \in \{0, 1\}$). In the super process variant, the control input may accept multiple values $U_i(t) \in \mathbb{U}_i := \{0, 1, ..., M_i\}$,

where $U_i(t) = 0$ corresponds to a freezing control.

With this change, the state evolution and the reward granted by each machine also depends on the control law, thus an index-type solution has not been discovered for some of this type of problems [9].

Arm-acquiring bandit

The arm-acquiring bandit problem is a variation of MAB in which there can be new machines at time $t \neq 0$. Both the state evolution and the reward obtained by operating a machine remain the same as in the classic MAB. In that way, one wants to determine the policy which maximizes the following functional with time dependent $K(t)$ number of machines

$$J^{\gamma} = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \sum_{i=1}^{K(t)} R_i(X_i(N_i(t)), U_i(t)) | Z(0)\right]. \tag{2.29}$$

An optimal policy can be achieved using the Gittins index. The reasoning behind this is the independence between the decision at time t and the arrival of new machines, supposing the arrival process as a sequence of independent identically distributed random variables [9].

MAB problem with switching penalties

The MAB problem with switching penalties is a variation from the classic MAB problem in which every time the controller switches from one machine to another, a switching penalty is incurred (such as a delay). This new feature makes an index-type solution no longer optimal.

MAB problem with multiple plays

In MAB problem with multiple plays, one has k independent processes and one controller with m processors available, with $m < k$. At each time the controller can allocate one processor to exactly one process and no process can be operated by more than one processor. In general, operating the m machines with higher Gittins index does not result in an optimal policy for the MAB with multiple plays.

Restless bandits

The Restless bandits problem is a generalization of the classic Multi-armed bandit, in which a player chooses from one out of N arms to activate at each time and receives a reward based on the state of the activated arm. Only the played arms change its state according to a Markovian

rule, and the objective is to maximize a reward by choosing which arm to active at each time. Assuming the Markov property, the state evolves as

$$X_i(t+1) = f_{i,t}(X_i(0), ..., X_i(t), U_i(t), W_i(t)). \tag{2.30}$$

Any machine not operated at time t cannot be operated later with the same reward, since the state of the machine doesn't remain frozen, and so neither the reward. This holds that, in general, an index-type solution does not result in an optimal scheduling policy. However, by relaxing the cost functional of the classical MAB formulation, Whittle [23] created a method for computing an index type solution for the restless bandits variation.

Whittle Index

In the general case, the Gittins index is the optimal solution of any MAB problem variation as long as the decisions which are not made at time t are still possible to be made at a future time, with the same reward for playing the arm [9]. This does not hold for the restless bandits, since the state of a not played arm can change and thus, all decisions made are irrevocable.

Whittle generalized the Gittins index to the Restless MAB problem by allowing multiple arms to be played simultaneously and allowing passive arms to change state and offer rewards [9]. This is achieved by considering a Lagrangian relaxation of the problem by allowing, in average, K arms to be played per turn, leading to the decoupling of the optimization problem into several optimization problems, one for each arm, giving the problem an index type solution.

This principle has been used to model problems in areas such as dynamic multichannel access [13] and UAV dynamic routing [16]. In the first, one wants to choose from one of N different channels to send a packet of data, without prior knowledge of the state of the channel. In the second the formulation is used in robot task planning to choose one of N places to visit. A reward is obtained when the vehicle visits a location in a specific place.

Let $S_i(t)$ be the state of the arm i at time t. The state of the arm is not directly observable, unless the arm is played. That way it arises the need to define a belief vector/information state denoted by $\Omega(t) := [w_1(t), ..., w_N(t)]$, where $w_i(t)$ is the conditional probability of $S_i(t) = 1$ given all past observations.

The belief is updated according to the Markov Model which describes each arm, with

$$w_i(t+1) = f(U_i(t), S_i(t)). \tag{2.31}$$

A policy is a function π which maps a belief vector $\Omega(t)$ to a corresponding action $U(t)$

at time t, so that it maximizes one of two performance measures, the discounted reward or the infinite horizon average reward.

An index policy assigns a measure of how attractive is to play an arm and chooses the arm with highest index, evaluating each arm independently. For instance, a myopic policy is such that at time t it chooses the arm with highest reward in the present time, independent of the future arm states.

For the case of the Whittle index, it measures how attractive is an arm according to the *subsidy of passivity* [23].

Let $P(m) = \{\omega : u_m^*(\omega) = 0\}$ be the passive set under the subsidy m, then an arm is indexable if the passive set monotonically increases from the null set to all the state space, as m increases from $-\infty$ to $+\infty$. That is, a bandit is indexable if the set of states from which is optimal to take the passive action (no action) increases with the subsidy of passivity [16]. With this condition, the Whittle index $W(\omega)$ of the state ω is the minimum subsidy m which makes both the passive and active actions equally rewarding. Although this condition may seem to hold in every situation, in some cases this may not be true, making the problem not indexable [23].

Chapter 3

Tracking

In this chapter it is described some methodologies for object tracking. In section 3.1 the the CAMSHIFT algorithm is presented as a methodology for tracking objects of interest in video streams. Finally, in section 3.2 an introduction to bayesian filtering is made, with special attention to the Extended Kalman (EKF) filtering, a special case of bayesian filtering. Some insights are also made on the application of EKFs in tracking using PTZ cameras in surveillance systems.

3.1 Continuously Adaptive Mean Shift

The Continuously Adaptive Mean Shift (CAMSHIFT) algorithm, originally developed for face tracking in vision based user interfaces, is based on the mean-shift algorithm, which consists on a non-parametric technique for finding modes in probability distributions [4]. However, unlike the Mean-shift algorithm, the modified algorithm dynamically changes its color probability distributions.

In image processing tasks, the Mean-shift algorithm is used to find regions in images with similar color histograms. In tracking tasks, one has to store a target color model and that color model is used to track the same target in different frames.

The algorithm starts by building a color model for the object to track, which assigns a probability value to each pixel value. A typical approach is to use the hue channel from the HSV space, in order to build models robust to lighting changes [4].

Then, for each image frame to be processed, the image is back projected with the pre-computed color model. The resulting image is a 2-dimensional probability distribution image encoding, at each pixel, the probability of a given pixel to belong to the original target.

The next step is to assign a window of fixed size I_k inside the image and compute the mean value

$$x_c^{k+1} = \frac{M_{10}}{M_{00}}, \tag{3.1}$$

$$y_c^{k+1} = \frac{M_{01}}{M_{00}} \tag{3.2}$$

of the normalized probability distribution inside the window I_k, with

$$M_{00} = \sum_x \sum_y I_k(x, y) \tag{3.3}$$

being the zeroth moment of the probability distribution $I_k(x, y)$ and

$$M_{10} = \sum_x \sum_y x I_k(x, y) \tag{3.4}$$

and

$$M_{01} = \sum_x \sum_y y I(x, y) \tag{3.5}$$

its moments of first order.

The new window I_{k+1} is centered around the new center (x_c^{k+1}, y_c^{k+1}). The process repeats until a convergence condition is met (such as number of iterations or the distance or the algorithm innovation $\|(x_c^{k+1}, y_c^{k+1}) - (x_c^k, y_c^k)\|$ is lower than a threshold). The CAMSHIFT algorithm adds a window resize step, in which the window size is step to a function of the M_{00}.

This results in a methodology robust to both lighting changes (by the use of a brightness invariant color model) and scale invariant, since the window size changes over time.

3.2 Bayesian Filtering

Given a set of observations, it is required a methodology to fuse information from the different sources. Bayesian filtering is the standard way of fusing sensor data and the knowledge of the system in order to estimate its state. Essentially it is composed of two separate steps. In the first step, the prediction step, the state belief x_t is predicted based on the last state estimate and the motion model $p(x_t|x_{t-1}, u_t)$ of the system and the control command u_t. In tracking applications, the control signal is not available and so it is assumed to be a stochastic signal.

The predicted state is then updated based on the observations z_t obtained by the sensors and

the likelihood of those observations given the sensor model $p(z_t|x_t)$, corresponding this to the update step

$$\begin{array}{c} p(x_t|z_{1:t-1}, u_{1:t}) = \int p(x_t|u_t, x_{t-1}) p(x_{t-1}|z_{1:t-1}, u_{1:t-1}) dx_{t-1} \\ p(x_t|z_{1:t}, u_{1:t}) = \eta p(z_t|x_t) p(x_t|z_{1:t-1}, u_{1:t}) \end{array}. \tag{3.6}$$

In the case of linear systems with Gaussian sensor and model noises the equation is solved in closed form, being its solution the equations of the Kalman filter,

$$Prediction\ Step: \tag{3.7}$$

$$x_{k|k-1} = f(x_{k-1|k-1}|u_k) \tag{3.8}$$

$$P_{k|k-1} = F_{k-1} P_{k-1|k-1} F_{k-1}^T + Q_{k-1} \tag{3.9}$$

$$Update\ Step: \tag{3.10}$$

$$y_k = z_k - h(x_{k|k-1}) \tag{3.11}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{3.12}$$

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{3.13}$$

$$x_{k|k} = x_{k|k-1} + K_k y_k \tag{3.14}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{3.15}$$

with $f(x_{k-1|k-1}|u_k)$ and $h(x_{k|k-1})$ being the motion and observation model equations, and F_{k-1} and H_k the linearized jacobian matrix of the motion and observation model matrices. In the prediction step the motion model is evaluated, leading to a rise in the uncertainty. In the update step, the new observation is incorporated and the state estimate and covariance are updated. This leads to a decrease in the state uncertainty.

The Kalman Filter is an optimal estimator under the restriction of linear motion and observation model. In this case, both the motion and observation model jacobians are computed in closed form and are constant for all possible values of the state x. The Extended Kalman filter constitutes a generalization of the Kalman filter and it is an approximation of the optimal linear counterpart for non-linear systems.

The non-linearity of the motion and observation models requires that, at each prediction and update step, the motion and observation models to be linearized anew.

In tracking scenarios with multiple cameras, all the targets state is described with respect to a ground plane referential. This common referential eases the process of fusion of observations from multiple cameras, using the appropriate sensor model. When using cameras for tracking

tasks, the most simple observation model is the pinhole camera model

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K_{3\times3}[R_{3\times3} \mid t_{3\times1}] \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = P_{3\times4} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \tag{3.16}$$

in projective coordinates, with P being the projective matrix, K being the intrinsic parameters matrix, R the rotation matrix from the world reference frame to the camera frame and t the camera position in the world reference frame. In PTZ cameras, the rotation matrix R can be computed by taking into account the current pan-tilt angles and the rotation from the world reference frame to the camera frame when the camera is at rest, that is, null pan and tilt values. The intrinsic parameters matrix K depends on the focal distance and thus depends on the zoom level. So, this matrix can be estimated by calibrating the camera at discrete zoom levels and interpolating the values in between. In cartesian coordinates it takes the form

$$\begin{pmatrix} u' \\ v' \\ w \end{pmatrix} = P_{3\times4} \begin{pmatrix} x' \\ y' \\ z' \\ h \end{pmatrix} \Rightarrow \begin{cases} u = \frac{f_1(x,y,z)}{f_3(x,y,z)} = \frac{p_{11}(\frac{x}{h})+p_{12}(\frac{y}{h})+p_{13}(\frac{z}{h})+p_{14}h}{p_{31}(\frac{x}{h})+p_{32}(\frac{y}{h})+p_{33}(\frac{z}{h})+p_{14}h} \\ \\ v = \frac{f_2(x,y,z)}{f_3(x,y,z)} = \frac{p_{21}(\frac{x}{h})+p_{22}(\frac{y}{h})+p_{23}(\frac{z}{h})+p_{24}h}{p_{31}(\frac{x}{h})+p_{32}(\frac{y}{h})+p_{33}(\frac{z}{h})+p_{14}h}, \end{cases} \tag{3.17}$$

where $f_k(x, y, z)$ is the internal product between the k^{th} row of the projective matrix and the real world position in projective coordinates and thus p_{ij} is the value of the projective matrix in the i^{th} row and j^{th} column.

By taking into account the structure of the sensor model, the Jacobian $H(x, y, z) \in \mathbb{R}^{2\times3}$ can be easily computed. Let x_i be the i^{th} state variable of the set $\{x, y, z\}$, and g_i the i^{th} observation variable from the set $\{u, v\}$, then the partial derivative in the position (i, j) of the matrix is given by

$$\frac{\partial g_i}{\partial x_j}(x, y, z) = \frac{\frac{\partial f_i}{\partial x_j}(x, y, z)f_3(x, y, z) - \frac{\partial f_3}{\partial x_j}(x, y, z)f_i(x, y, z)}{f_3(x, y, z)^2}. \tag{3.18}$$

This formulation enables the use of non-linear motion and observation models. However, error is still modelled as Gaussian noise. When the sensor and system noise are known to be non-Gaussian, non-parametric methods such as particle filtering can be used. The idea is to represent the posterior probability distribution $p(x_t|z_t)$ by a set of random particles and weights. As the number of particles increases, the estimate approximates to the optimal bayesian estimate [1].

Assuming the Markov property, the generic particle filter is made of the following steps. For each of the M particles, draw a new particle from the state transition distribution $p(x_t|x_{t-1}, u_t)$. This particle will represent the predict of the current state given the last state. The inclusion of observations is made by computing a weight w_t^i for each particle, corresponding to the likelihood of the prediction given the sensor model, $p(z_t^i|x_t^i)$. This weight is often called the *Importance Factor* [22]. Finally, each pair (x_t^i, w_t^i) is added to a temporary set $\overline{X}$.

The last step is called *Importance Resampling* and consists of drawing M particles from the set $\overline{X}$ computed in the last step according to the distribution w_t, so that the probability of drawing a particle x_t^i is its corresponding weight.

In chapter 5 are detailed tracking experiments and results based in the CAMSHIFT algorithm combined with a particle filter. In particular, the robustness of the algorithm to occlusions is evaluated in a single-camera single-target scenario.

Chapter 4

Information Theoretic Approach for PTZ Cameras Scheduling

In this chapter it is introduced the theoretic framework in which this thesis is based, previously explored in various works [20] [6] [18]. Essentially the camera scheduling is tackled as an information gathering problem. For each camera it is assigned a PTZ configuration in order to maximize the mutual information gain between the current state distribution and the expected state distribution after making a observation.

This framework allows to carry two types of tasks: scene exploration and target following. In the first, the goal is to explore the environment under surveillance and detect new potential targets. It corresponds to an evolution to methods previously described in which static cameras are deployed for that end.

In the later, the goal is to maintain inside the cameras field of view previously detected targets.

4.1 Scheduling Cameras Scene Exploration

In the scene exploration task, the goal is to find new targets for tracking tasks in the surrounding environment, using a set of PTZ cameras. This is achieved by optimizing the information gain between the existence e of a target in a set of locations an its detection d by some active camera, over the PTZ working space.

Let the $e \in \mathcal{E}$ be zero or one depending if the target exists or not and $d \in \mathcal{D}$ be zero or one depending if the target is detected or not, then the Information Gain $I(e; d)$ between the

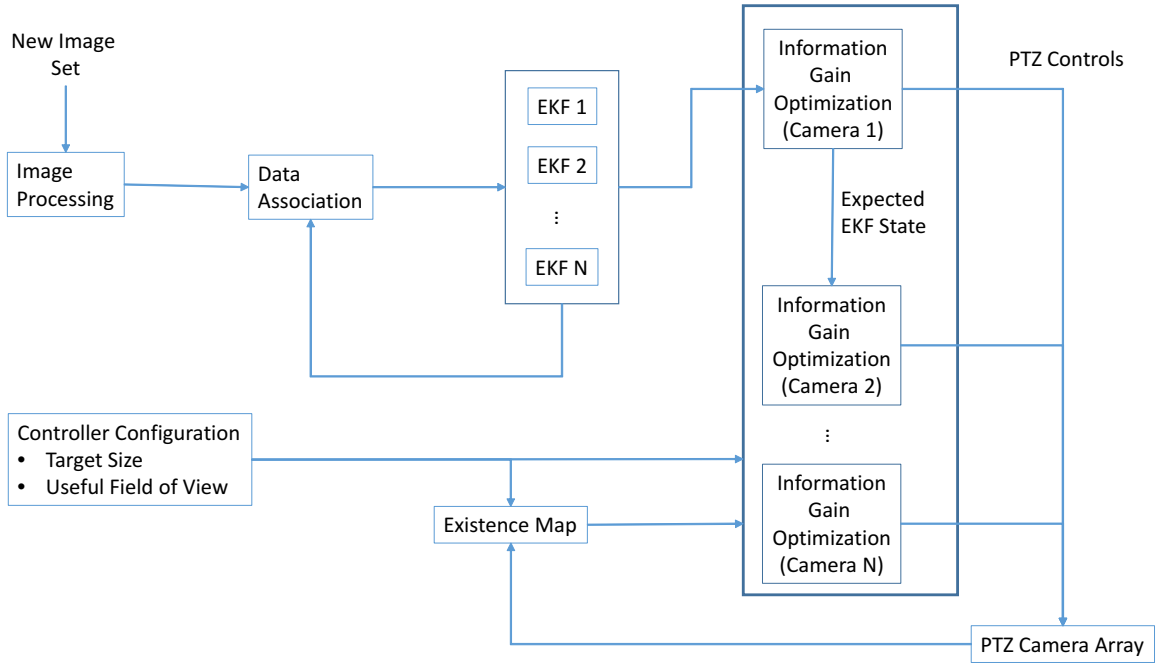


Figure 4.1: System architecture

existence of a target and its detection for a single location is given by

$$I(e;d) = H(e) - H(e|d) = H(d) - H(d|e) =$$
$$= -\sum_{d_i \in \mathcal{D}} p(d) log\ p(d) + \sum_{e_i \in \mathcal{E}} p(e_i) \sum_{d_j \in \mathcal{D}} p(d_j|e_i) log\ p(d_j|e_i),$$
$$with\ p(d) = p(d|e=0) + p(d|e=1), \quad (4.1)$$

with $H(x)$ being the Shannon Entropy of the discrete probability distribution x, defined as [5]

$$H(x) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x), \quad (4.2)$$

with $\mathcal{X}$ the domain of the random variable $p(x)$.

Thus, the information gain is dependent on three priors. The existence model $p(e)$ models the probability of existence of a target in a certain location. The last two $p(d|e=0)$ and $p(d|e=1)$ correspond to the detector performance, that is, the probability of detecting a target given its existence (true positive) or not (false positive).

These models encode the dependency between the information gain and the pan-tilt-zoom controls. High uncertainty in the existence of a target in a given location will result in a high

information gain when pointing the camera towards that location. As for the detector performance, note that both low and high zoom configurations result in acquisition of images in which it is difficult to find target features in the image, making the detector performance dependent on the zoom level. This makes the information gain a suitable metric for performance evaluation in exploration tasks.

4.1.1 Existence model

For any given location, the existence $e \in \{0, 1\}$ of a target is modeled by a birth-and-death process

$$p(e(t - t_0)) = (\alpha - 0.5)e^{-\lambda(t-t_0)} + 0.5 \tag{4.3}$$

with equal birth and death rates, with t_0 being the last time that location was observed by a camera. The uncertainty of the existence of a target in that location is given by the Shannon Entropy of the discrete distribution $p(e)$. After a location is observed, the probability of existence is reset to its initial value α. As time passes, $p(e)$ asymptotically evolves to the value of 0.5. This makes the entropy $H(e)$ increase with time, asymptotically approaching the maximum of 1 bit.

In other words, the existence model encodes the growing uncertainty of the existence of a target in a particular location, given that it was not observed for t time. In the limit, there is a complete lack of knowledge of the state of that particular location so that it is equally probable that there is (or there is not) a potential target in the location and, as such, the entropy grows to its maximum.

Figure 4.2 shows the effect of different initial probabilities in the entropy distribution. Note that for models in which the initial existence probability is close to 0.5 (very uncertain models), the entropy will always be close to 1.

4.1.2 Detector performance

If the birth-and-death process is responsible for the pointing direction of the cameras (which locations are going to be observed at any given time), the detector performance is responsible for the selection of the appropriate zoom level for the detection task.

Intuitively, the detection of a target in a given location is difficult for both high and low zoom levels, since in the former the projection of a possible target into the image plane would be larger than the acquired image and in the later its projection would be too small.

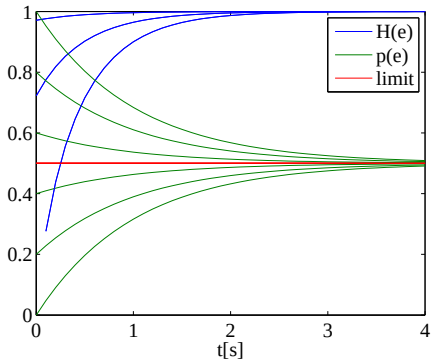


Figure 4.2: Birth-and-death process and respective entropy evolution for various values of α. The entropy scale is in bits

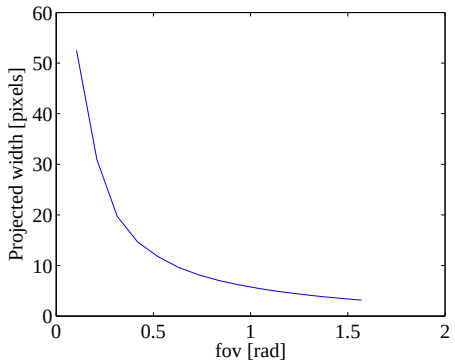


Figure 4.3: Evolution of the projected width of the bounding box in the image plane with the field of view (larger field of view results in lesser zoom)

In practice, to compute this prior a possible target can be modeled as a parallelepiped on top of the ground plane, with fixed width and height. The goal is to estimate the width of the solid projected into the image plane and use this distance as a measure for performance. This can be done by projecting the four edges in the base of the solid into the image plane and compute the largest distance between themselves. This distance is used as measure of performance. The figure 4.3 shows the evolution of the bounding box width with the increase of field of view using the described method. By carefully choosing the range of widths in which a target can be detected, one can build a prior $p(d|e = 1)$, for instance, a Gaussian distribution around an optimal point.

The same stream of thought can be applied to build models for false detection $p(d|e = 0)$.

4.1.3 Optimization

The search for the maximum value in equation (4.1) can be done by exhaustive search. For each PTZ configuration, the set of regions inside the field of view are evaluated and its contribution to the information gain computed. The global information gain is the sum of the information gain of all the visible regions.

The figure 4.4 shows the movement of a camera under an exploration task. For experiment purposes, the ground plane was discretized in equally space squares 1 meter wide. The pan range of the camera was discretized in 11 values between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$, the tilt range in 6 values between $-\frac{\pi}{2}$ and 0 and the field of view between $\frac{\pi}{30}$ and $\frac{\pi}{2}$.

It was assumed a true positive probability $p(d|e=1)$ of 0.7 when the potential target is 70 to 100 pixels wide in the image plane. The false positive probability was assumed to be zero.

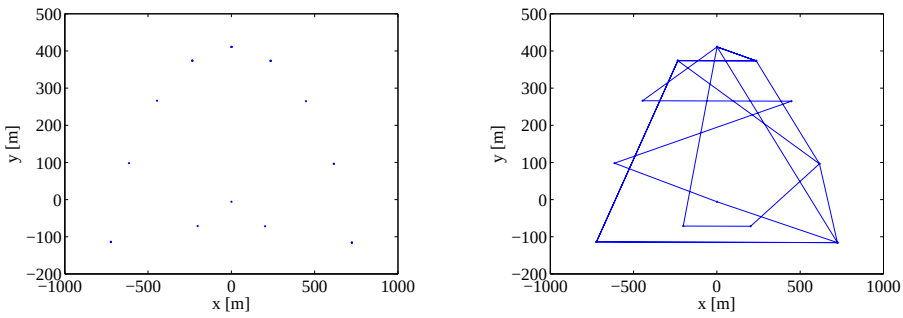


Figure 4.4: Camera Trajectory. Each point corresponds to the back-projected center of the image in the ground plane. At the left (a) it is only shown the back-projected points resulting from the optimization procedure. At right (b) it is shown the camera trajectory.

Figure 4.4 (a) shows the positions, in ground floor coordinates, to which the camera was pointing. Figure 4.4 (b) shows the transitions between configurations over time.

The results show that, under the assumption that no other kind of tasks are to be performed by the camera (for instance, tracking tasks concurrently with exploration tasks), the optimal pan-tilt-zoom configuration is a small subset of the search space. Thus, under the mentioned assumptions, the camera controls for exploration can be computed offline and the information gain evaluated for a smaller sample of the possible configurations.

4.2 Scheduling Cameras for Tracking tasks

As with the scene exploration task previously described, the scheduling methodology for tracking is also modeled as an information gathering problem.

The optimal pan-tilt-zoom configuration is the one which maximizes the information gain $I(x; o)$ between the state estimate and the observation. In other words, it is the one that leads to a greater increase in the certainty of the state estimate.

4.2.1 Single Camera Scenario

In a single camera scenario, a single camera is responsible for tracking and maintaining the trajectories of various targets which move freely across the environment.

Each target is tracked using an Extended Kalman Filter (EKF) with the motion model depending on the kind of target being tracked (e.g. pedestrian or vehicle) and the pinhole camera model as the observation model. The decision on which pan-tilt-zoom command to send to each camera is done by maximizing the mutual information gain $I(x_t;\ o_t)$ for all targets, with x_t being the target state estimate at time t and o_t the observation made.

Using the definitions of mutual information gain $I(x;\ o)$ and conditional entropy $H(x|o)$ [5], the cost function can be expanded

$$a^* = \arg\max_a I(x;\ o) = \arg\max_a H_a(x) - H_a(x|o) \tag{4.4}$$

$$= \arg\max_a H_a(x) + \int p(o) \int p(x|o) \log p(x|o) \partial x \partial o \tag{4.5}$$

$$\begin{aligned} = \arg\max_a H_a(x) &+ \int_v p(o) \int p(x|o) \log p(x|o) \partial x \partial o \\ &+ \int_{\bar{v}} p(o) \int p(x|o) \log p(x|o) \partial x \partial o, \end{aligned} \tag{4.6}$$

where distribution $p(o)$ denotes the probability of making an observation of the target. Its domain of integration can be divided into two subdomains: v, which denotes all the camera configurations in which the target is visible and $\bar{v}$, which represents all the configurations in which the target is not visible.

Without further assumptions, this problem is hard to solve. However, recalling the EKF structure, predict and update steps, one obtains the following properties: (i) The probability distribution $p(x|o)$ corresponds to the state distribution after the update step. In other words,

$p(x|o) \sim \mathcal{N}(x_{k|k}, P_{k|k})$. (ii) All state variables are gaussian distributed. This means all differential entropies can be computed in closed form. (iii) When there is no observation of the target, the update step is skipped. This means that the state is independent from the observation and $p(x|o) = p(x) \sim \mathcal{N}(x_{k|k-1}, P_{k|k-1})$. Applying these properties into equation (4.6), the optimal control action can be evaluated using the simplified optimization problem

$$a^* = \arg\max_a \left(H_a(x) + \omega(a)H(x^+) + (1 - \omega(a))H(x^-) \right), \tag{4.7}$$

where

$$\omega(a) = \int_v p(o)do \tag{4.8}$$

represents the observation function, which depends on the action a.

Note that all the state variables are Gaussian distributed and, by definition, the differential entropy for Gaussian distributed variables depends only on the variable covariance matrix. By observing the EKF equations, it can be seen that the observation o_k is only used in the innovation equation and in the state update equation. This enables the computation of P^+ without having an observation. Recalling that the differential entropy for a multivariate gaussian distribution is given by

$$H(x) = \frac{k}{2}(1 + \log(2\pi)) + \frac{1}{2}log(|\Sigma|) \tag{4.9}$$

with k the random variable dimension and Σ the distribution covariance, and using the covariance update equation in the EKF, the cost functional can be simplified to

$$a^* = \arg\min_a \omega(a)(\log|P^+| - \log|P^-|) \tag{4.10}$$

$$= \arg\min_a \omega(a) \log(I - K_k H_k). \tag{4.11}$$

Note that the final cost function has three essential contributions. First, the Kalman gain K_k encodes the uncertainty of the target position. Larger Kalman gain means that the future observation will have a higher weight in the update the of state estimate and, as such, will result in a larger information gain. Second, the observation model jacobian H_k encodes the sensor geometry. As it is shown in section 5.3, for the pinhole camera model it corresponds to higher information gain at higher resolutions and when capturing the target in the edge of the image. Finally, the observation function $w(a)$ encodes the probability of observing a target. As it is shown in section 5.3, this prevents, for instance, excessive zoom into the target.

The choice of the optimal configuration is made by optimizing the sum of the mutual infor-

mation gains $I(x;\ o)$ for all targets. Despite the elegance of this cost function, in the general case its evaluation is intractable because it requires the exhaustive search over the configuration space of the camera. For each pan-tilt-zoom configuration of the camera, the observation model must be linearized anew and an EKF update must be performed to obtain the new Jacobian matrix H_k and K_k. In the section 5.3, it is shown a comparison between different observation functions and its structure in the application with PTZ sensor models is used to ease the optimization procedure.

4.2.2 Observation function

The observation function $\omega(a)$, equation (4.11), is a central component of the target tracking methodology as it effects on the convexity of the information gain.

The observation function can be just a flag indicating whether a point representing the target location, in world coordinates, is visible or not in the image:

$$\omega(a) = \begin{cases} 1, & \mathcal{P}(X_{gnd};\ a) \in E_{img} \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

where X_{gnd} is a point in the ground plane representing the target position, $\mathcal{P}(\cdot;\ a)$ is the projection operator on a set of points, which is defined by action a, i.e. in (3.17) the projection matrix $P_{3\times4}$ is modified by action a and E_{img} is the ellipse in the image plane concentric with the image rectangle. However, this observation function does not perform a smooth regularization of the cost function, making difficult to design iterative optimization algorithms.

By changing the model of the target from a single point in ground plane into a surface in the ground plane, one obtains a smoother optimization function. Modeling the shape of the target as a rectangle, taking into account the visible part normalized by the area of the imaged shape not truncated by the field of view of the camera, the observation function takes the form

$$\omega(a) = \frac{A(\mathcal{P}(R_{gnd};\ a)\ \cap\ E_{img})}{A(\mathcal{P}(R_{gnd};\ a))} \tag{4.13}$$

where $A(\dot{})$ indicates area of a convex hull of points and R_{gnd} is a rectangle in the ground plane. Alternatively, modeling the target as an ellipse in the ground plane, the function becomes

$$\omega(a) = \frac{A(\mathcal{P}(E_{gnd};\ a)\ \cap\ E_{img})}{A(\mathcal{P}(E_{gnd};\ a))} \tag{4.14}$$

where E_{gnd} is an ellipsis in the ground plane.

4.2.3 Multiple Camera Scenario

In a multiple camera scenario, multiple cameras can have observations of the same target. The incorporation of these observations is done using an sequential Kalman Filter for each target. In this variation, a single prediction step is made per iteration. In case of existing observations, each observation is applied to the filter sequentially (with the proper observation model for each camera). Each observation contributes to the covariance matrix with a factor of $(I - K_c H_c)$, where K_c is the Kalman gain from the observation of camera c and H_c is the Jacobian of the observation model for camera c.

The mutual information gain of a target for multiple observations

$$I(x;\ o_1, ..., o_C) \propto \sum_{c \in C} log|I - \omega(a) K_c H_c|, \tag{4.15}$$

is then obtained by combining equation (4.10) with the new covariance matrix update

$$P^+ = \left(\prod_{c \in C} (I - K_c H_c) \right) P^- \ . \tag{4.16}$$

Chapter 5

Experimental Results

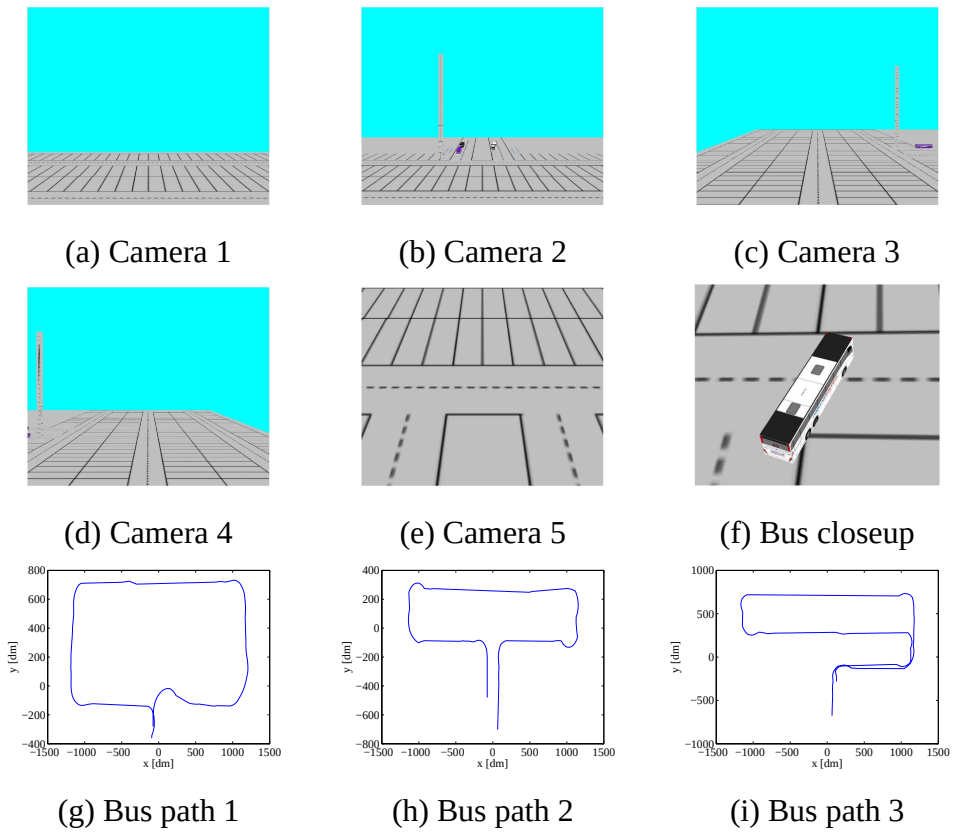
This chapter describes the experiments conducted in this work. Starting from a brief description of the simulation environment in section 5.1, in section 5.2 some tracking experiments conducted to evaluate the CAMSHIFT algorithm performance are presented.

In section 5.3 the effects of the observation function in the information gain cost functional previously described are shown, in order to give to the reader some intuition about the described entities.

Finally, section 5.4 presents some camera control experiments are performed using three different strategies for camera control. The effects of the observation functions are shown and strategies using different observation functions are compared against each other.

5.1 Simulator Setup

The experiments described in this section are based in a virtual reality environment simulating a parking lot (see Fig. 5.1(a-f)). Four buses, with the dimension of 2x10x2 m, cross the scene according to predefined trajectories shown in Fig. 5.1(g-i). These trajectories are generated using the car model. Tracking is performed using five pan-tilt-zoom cameras located at fixed positions on the sides of the parking lot and in the entrance, as described in figure 5.2.

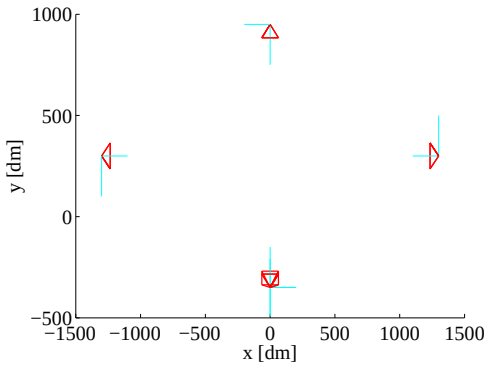


(a) Camera 1 (b) Camera 2 (c) Camera 3

(d) Camera 4 (e) Camera 5 (f) Bus closeup

(g) Bus path 1 (h) Bus path 2 (i) Bus path 3

Figure 5.1: Field of view of the cameras at rest orientation and typical bus trajectories.



Camera Number	Position/dm
Camera 1	(0, -350, 250)
Camera 2	(0, 950, 250)
Camera 3	(-1300, 300 250)
Camera 4	(1300, 300, 250)
Camera 5	(0, -350, 250)

Figure 5.2: Simulator Cameras' Position. At left, a spacial representation of the cameras is shown. The red triangles represent the cameras field of view angle. The blue axis the camera represent the camera orientation.

5.2 Tracking: Robustness to Occlusions

The objective of this experiment is to evaluate the robustness to occlusions of a tracking system composed by a CAMSHIFT module providing observations to a particle filter. This experiment is built on top of the VRML based virtual reality simulator in which a bus in a parking lot is followed by a single pan-tilt-zoom camera. An object was placed in the middle of the parking lot in order to evaluate the performance of the tracker when occlusions occur.

The results of this experiment are depicted in figure 5.3. Initially the observations have little error and the particle filter seems to work. However, when the bus starts to be occluded, the cam shift keeps trapped in the back of the bus. These observations are consistent with the motion model, since it appears that its linear velocity is approaching zero and the bus is stopping.

However, when the bus crosses the pillar and is again fully observable by the camera, the cam shift converges again to the true position of the bus. This is however inconsistent with the motion model, since it would require the bus to have a huge acceleration for going from rest to the observed position.

This makes the particle filter estimates to have a huge error when the bus crosses the obstacle when using the observations of a single camera.

Despite the tracking error upon the existence of an obstacle, the camera still maintains the bus in the field of view, since the zoom level is low.

This experiment shows how much the motion model and the tracked object geometry is being ignored, since the observations are only based on the similarity of the color histograms between frames. Before the target occlusion, the observations are unreliable but consistent with the motion model, since it appears the bus is breaking.

A more robust approach would take into account the geometry of the object to correct the observations in situations of sudden change (such as when the object is occluded).

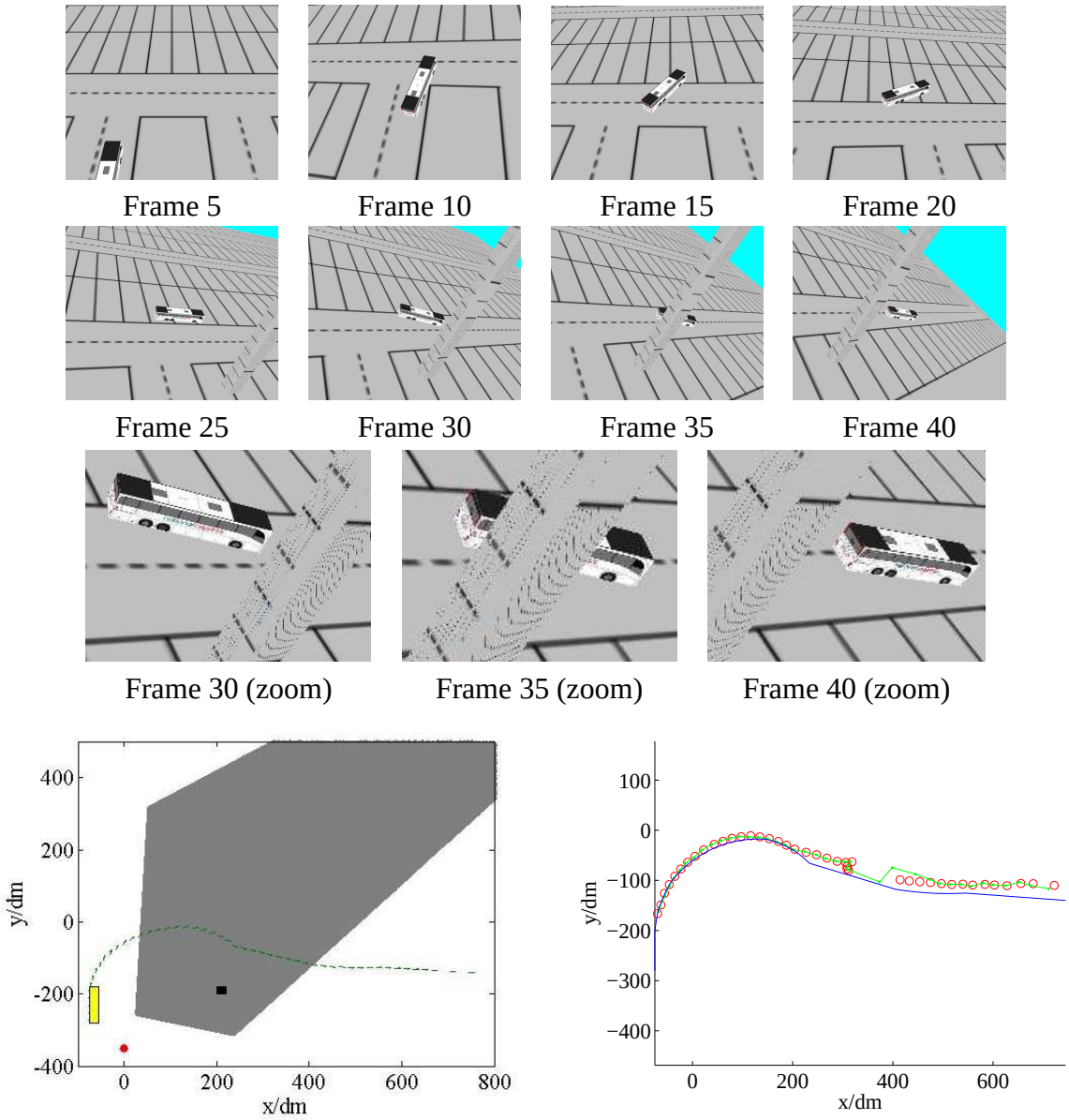


Figure 5.3: Tracking target partially occluded. In the top the sample frames obtained with the simulator. In the bottom estimated trajectories. At bottom left it is represented the ground truth of the bus, along with the camera position (red), the bus in its initial position (yellow), the obstacle (black square) and the camera field of view (without the obstacle). At bottom right it is represented the estimated trajectory (green) along with the ground truth (gray) and the cam-shift observations (red).

5.3 Scheduling: Comparison of Observations Functions

In order to evaluate the geometry of the cost function using different observation functions, the cost function was evaluated from a set of values from the pan-tilt-zoom space for a camera. This was achieved by placing a target in a fixed location in front of the camera and making an observation. The camera model and its Jacobian were computed for the new pan-tilt-zoom configuration and, along with the new observation, used to update an existing EKF. The resulting Kalman gain and the linearized observation model were obtained and used in the cost function (4.11).

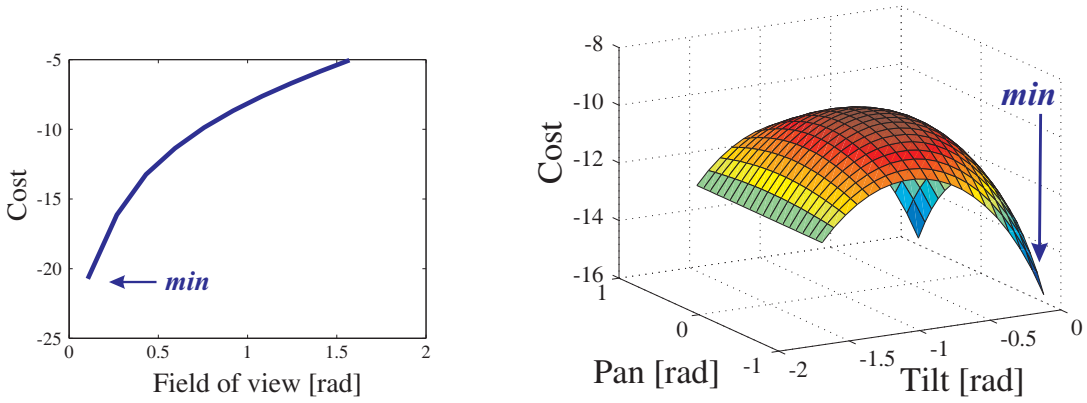
Figure 5.4(a) show the cost function evolution with both the zoom level and the pan-tilt values using the observation function (4.12). Figure 5.4(b) show the same evolution as in the previous case, but now using the observation function (4.14). The term $w(a)$ defined as in (4.11) changes the cost minimum from extreme values of zoom, pan and/or tilt to within-range, central, values.

The second set of figures was generated in a similar way from the former ones, however sampling the whole camera parameters space. Figure 5.5 shows slices of the cost function in the camera parameter space (pan-tilt-zoom) using different observation functions as described in Section 4.2.2. Each row represents a fixed zoom configuration, with lower rows representing configurations with higher field of view (less zoom).

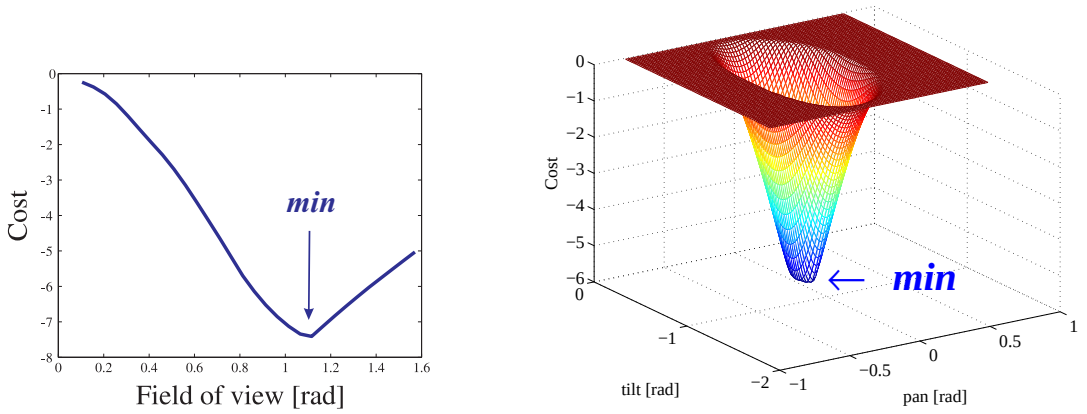
In column (a) of Fig. 5.5, is used $w(a)$ defined in (4.11). In column (b) is considered the same $w(a)$ however, in this experiment, there are two targets in the scene in different locations and with different state estimate covariances. In columns (c) and (d) the term $w(a)$ is the one defined by (4.13) and (4.14), respectively.

Figure 5.5(a) shows that the cost function is generally lower in configurations which make the target appear in the image plane with higher resolution, that is, when the camera is at its highest zoom or when the camera has the target of interest in the corner of the image. This result comes from the influence of the observation model (in particular, its Jacobian) used in the EKF in the cost function. While this is a good result in the sense resolution is maximized, uncontrolled zoom onto the target is not the desired behavior, since a minimum movement can make the target disappear from the camera field of view. Term $\omega(a)$, as defined in (4.13) or (4.14), acts as a regularizing factor, managing the optimal zoom level to observe the target at constant zoom level and preventing observations just using the "‘corner of the eye"’.

In section 5.4 both observation functions are used under the same simulated conditions and its performance is evaluated.

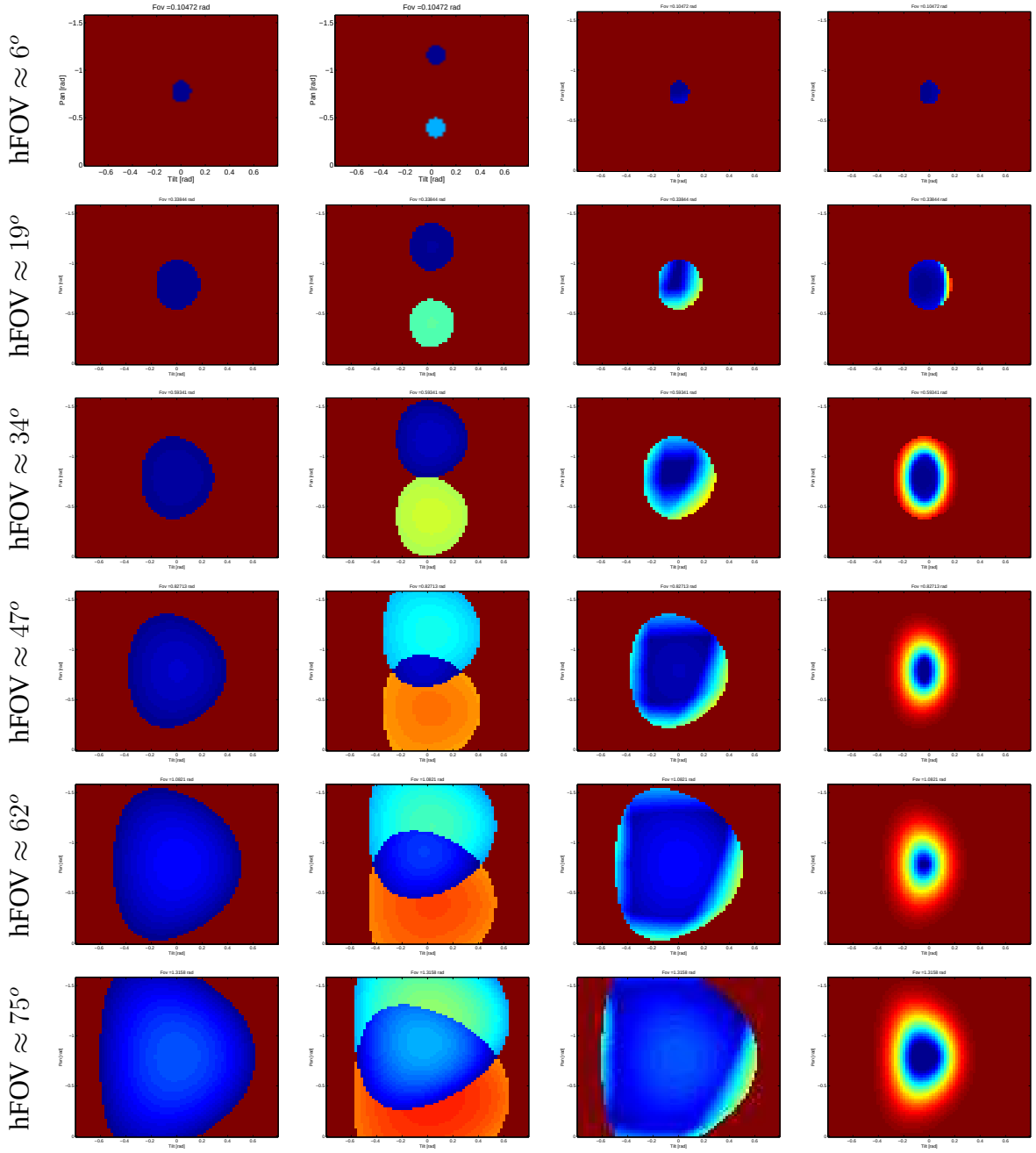


(a) Observation function indicating target in or out of the field of view.



(b) Observation function accounting imaged target size.

Figure 5.4: Effect of the observation function into the cost function. In case (a) the observation function $\omega(a)$ is defined by (4.12) and therefore just indicates whether the target in or out the field of view. In case (b) $\omega(a)$ is defined by (4.14) and thus takes in account the imaged size of the target and the field of view. Two plots for each one of the cases, cost vs zoom (left) and cost vs pan and tilt (right).



(a) one target $\omega(a)$ as (4.12) (b) two targets $\omega(a)$ as (4.12) (c) one target $\omega(a)$ as (4.13) (d) one target $\omega(a)$ as (4.14)

Figure 5.5: Cost function slices at fixed field-of-view (zoom). Each row represents a different zoom level (field of view, top row 6[deg] ≈ 0.1 [rad], bottom row 75[deg] ≈ 1.3 [rad]). Colder colors represent lower cost functions. Configurations in which the target was not visible were assigned high cost. In cases (a,c,d) one camera observes one target. In case (b) one camera observes two targets. The observation function $\omega(a)$ is defined by (4.12) in cases (a) and (b), and is defined by (4.13) or (4.14) in cases (c) or (d), respectively.

5.4 Scheduling: Camera Control Experiments

This section describes a series of experiments on camera control in the simulated environment.

In this simulation, each bus enters in the environment at the time instants $t = 1$, $t = 10$, $t = 30$ and $t = 60$, and their detection is made 10 seconds after their initial appearance. There is no image processing or data association steps involved. All the bus positions are read from a previously loaded file. Then, at runtime, it is computed which cameras have which buses inside its field of view, contributing each camera with an observation to the EKFs. The controller is then fed with the resulting EKFs state and outputs the corresponding pan-tilt-zoom commands for each of the cameras.

Three strategies were evaluated. In the first two the information gain is maximized using two observation functions described in previous sections. On made by modeling the bus as a rectangle in the ground plane (see equation 4.13) and other by modeling the bus as an ellipse in the ground plane (see equation 4.12). The final strategy is a greedy procedure in which each camera tracks the target with maximum uncertainty.

As for the first two, the optimization procedure is made sequentially over the PTZ space, starting from the camera 1 to the camera 5. After a prediction step for each EKF, each camera maximizes the information gain for the next observation, considering the decisions of the previous cameras. That is, camera 2 will maximize its information gain given the decision made by camera 1 and so forth.

The cost function is non-convex and as such the local optimum may not correspond to the global optimum. However, some techniques are used to ease the optimization procedure.

The optimization starting points are all contained in the minimum admissible zoom level plane.This is to enable the optimization procedure to evolve into convex curves between two targets, as described in previous sections. As for the initial pan and tilt values, they correspond to configurations which make the camera point to a bus or the middle ground between them, in groups of two, three, ... n buses.

It can be argued that the number of starting points has quadratic complexity with respect to the number of targets to track. However, for environments with few targets to track it can be considered acceptable. Using the minimum admissible zoom as starting point for the optimization, one can observe a finite number of local convex curves (see figure 5.5).

In the last strategy, it was defined that each camera would sequentially choose to follow the target with maximum entropy. Using this strategy, as a rule it was forced that no camera can observe two targets at the same time, to avoid having all cameras following the same bus. But, due to zoom selection, more than one bus can be inside the field of view. In these cases, the

Number of Targets	Number of Starting Points
1	1
2	3
3	7
n	$1 + n(n-1)$

Table 5.1: Optimization starting points complexity

camera will still, although incidentally, contribute with more than one observation.

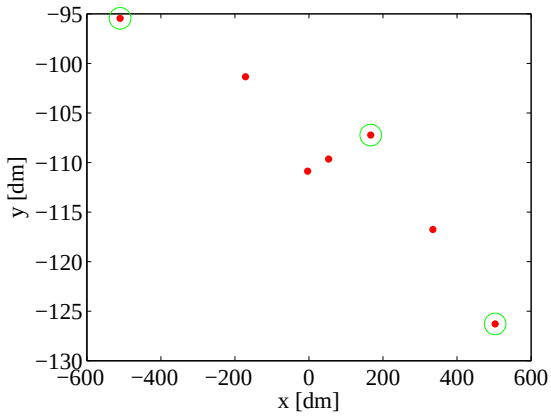
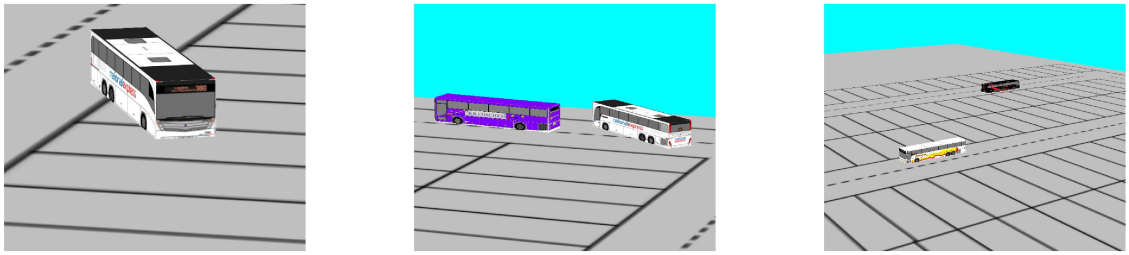


Figure 5.6: Optimization starting points in the ground plane. The points marked with a green circle correspond to bus positions.

5.4.1 Results

Figure 5.7 shows samples of camera frames acquired during the simulation, corresponding to typical behaviors of the controller. Figure 5.7 (a) shows the ability of the system to automatically adjust the zoom level into a single target. This is due to the effect of the observation function in the cost function, preventing excessive zoom.

Figures 5.7 (c) show a scenario in which the optimal decision is to observe multiple buses. As in the previous sample, the zoom level is also adjusted in order to keep both buses inside the camera field of view. Moreover, these results also show that the decision of which buses to cluster inside the same field of view is independent from the distance of the buses to the camera. Unlike other scheduling approaches, which take into account the distance of the targets to the camera in order to perform clusters of targets inside the field of view, this approach automatically takes into account the camera perspective effect. While on figure 5.7 (b) the



(a) Auto-PTZ in case of one target

(b) Auto-PTZ in case of two nearby targets

(c) Auto-PTZ in case of two far away targets

Figure 5.7: PTZ imaging showing automatic clustering of multiple buses in the same field of view. Frame (a) shows automatic zoom selection when focusing a single target, while in (b) both buses are close together. In (c) both buses are positioned far away, clustering is due to the projective effect of the camera.

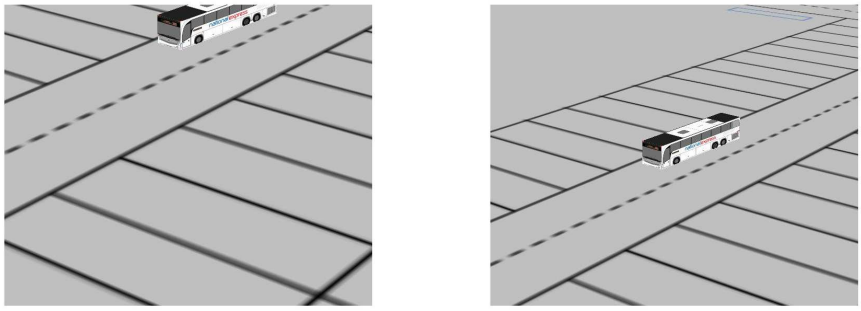
buses are at the same distance to the camera are they are close together, in figure 5.7 (c) both buses are far from each other and from the camera.

Figure 5.9 shows the evolution of the uncertainty of each target using the described strategy. In figures 5.9 (b) and (c) the information gain was maximized using different observation functions. While on 5.9 (c) it was used an ellipse projected into the ground plane (as described in equation 4.14), in 5.9 (b) it was used a rectangle projected into the ground plane (as described in equation 4.13). Figure 5.9 (a) uses the same input data but, as control strategy, it was defined that each camera would sequentially choose to follow the target with maximum entropy.

Each peak corresponds to a change in the number of observations. For the first target, after the initial uncertainty peak (due to the EKF initialization), the uncertainty rapidly converges into steady state, since all the cameras are directed into the new target. After the detection of a second bus ($t = 11$), and in posterior detections of new buses ($t = 20$, $t = 30$ and $t = 70$) there is an increase of the uncertainty base line.

First thing to notice in all the strategies is the intermittent behavior of the cameras. Sequence of peaks mean that one or more cameras are continuously switching between targets. Despite undesired, this is to be expected since there is not a regularization term in the cost function in order to impose a finite speed limit for each camera.

The second point to address is the comparison between the strategies (a) and (b), using different observation functions. The first phenomenon to observe when using as observation function as in case (b) is not directly observable in figure 5.9. Figure 5.8 shows a sample of the image frames acquired by the camera using both strategies As it is shown, modeling the targets as rectangles in the ground plane (as described in equation 4.13) and using the proposed

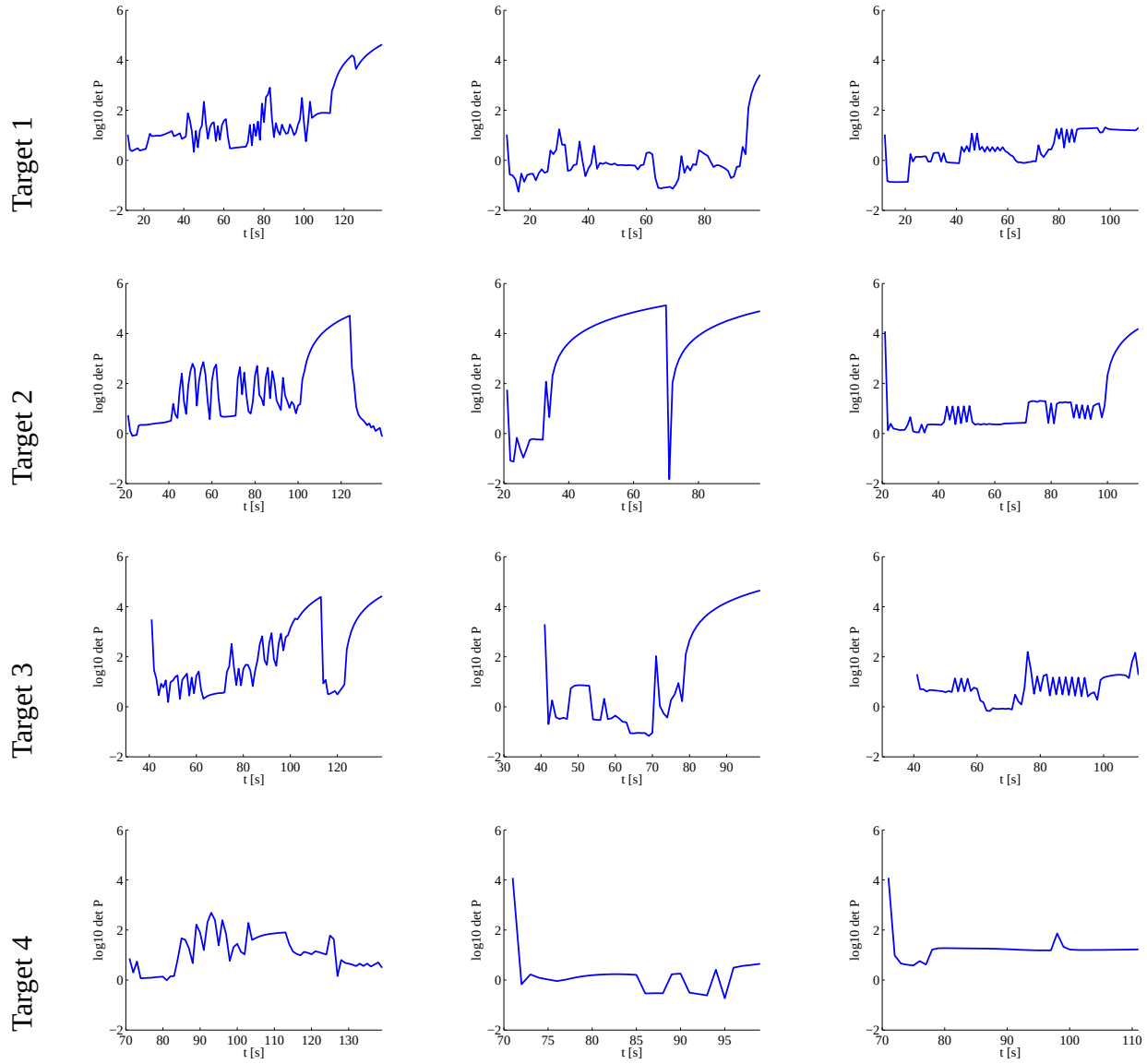


(a) $\omega(a)$ as in 4.13 (b) $\omega(a)$ as in 4.14

Figure 5.8: Frame detail using two different observation functions at the same time instant. On the left (a), the optimization procedure gets trapped in a local minimum which corresponds to observe the bus in the corner of the image. The same doesn't happen in (b).

optimization procedure, the optimal decision is to observe the bus from the corners of the image. This shows that the observation function 4.13 is not effective enough to smooth the cost function. Although a formal proof could not be made, this behavior was not observed when modeling the targets as ellipses in the ground plane, as described in equation 4.14 (see figure 5.8 (b)).

The final point to address is the rapid increase in the target uncertainty. This can be caused by several phenomena. Bad calibration of the EKF process noise and/or inadequate motion model can lead the camera to lose track of the moving target. Also, the current implementation is only composed of a tracking task (recall that, in chapter 4, the information theoretic framework was shown as being composed by an exploration and a tracking task). By combining both tasks in the implementation and adding a trigger for lost targets (for instance, based on a maximum uncertainty threshold), the cameras would automatically search the environment and the target could be reacquired. However, as at the time of writing of this thesis, there was no work done combining the tracking and exploration tasks using a non-exhaustive search method.



(a) greedy method (b) $\omega(a)$ as (4.13) (c) $\omega(a)$ as (4.14)

Figure 5.9: Uncertainty of targets location using three different strategies for camera control. Each row corresponds to a different target. In (a) each camera sequentially chooses the target with largest entropy. In (b) and (c) the information gain is maximized with different observation functions. In the former, it is used an ellipse in the ground plane to model the targets (as described in equation 4.14), while in the later a rectangle is used (as described 4.13).

Chapter 6

Conclusion and Future Work

In this thesis the Information theoretic approach for active camera scheduling proposed in [19] was explored, with special incidence on tracking tasks.

The main technical constraint when applying the information gain as objective function was the undesired behaviors which result from the usage of the pinhole observation model. These behaviors include observation of targets from the corner of the image and uncontrolled zoom onto the target to observe.

In order to overcome those problems, different observation functions were studied and their effect on the cost function was evaluated. In particular, it was discovered that modeling the targets as ellipses projected into the ground plane mitigates the unwanted behaviors.

Some desired behaviors emerged from this method. When a single target is present inside the range of the active camera, the optimal command is to zoom on the target, according to the cost function. The regularization term controls the optimal zoom on the target, independently of the target location. In a multi-target scenario, the overlapping cost functions can make the optimal command to keep both targets inside the cameras field of view.

As future work, topics such as the application of the proposed algorithms in real systems need to be tackled, starting from the creation of a datasets likely to represent PTZ cameras.

Bibliography

[1] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, Tim Clapp, and Sanjeev Arulampalam. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *Signal Processing, IEEE Trans. on*, 50(2):174–188, 2002.

[2] Horesh Ben Shitrit, Jerome Berclaz, Francois Fleuret, and Pascal Fua. Tracking multiple people under global appearance constraints. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 137–144. IEEE, 2011.

[3] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, 33(9):1806–1819, 2011.

[4] Gary R Bradski. Computer vision face tracking for use in a perceptual user interface. 1998.

[5] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[6] Joachim Denzler, Matthias Zobel, and Heinrich Niemann. Information theoretic focal length selection for real-time active 3d object tracking. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 400–407. IEEE, 2003.

[7] Francois Fleuret, Jerome Berclaz, Richard Lengagne, and Pascal Fua. Multicamera people tracking with a probabilistic occupancy map. *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, 30(2):267–282, 2008.

[8] Alfred O Hero and Douglas Cochran. Sensor management: Past, present, and future. *Sensors Journal, IEEE*, 11(12):3064–3075, 2011.

[9] Alfred Olivier Hero, David Castañón, Doug Cochran, and Keith Kastella. *Foundations and applications of sensor management*. Springer, 2007.

[10] Pedro Mendes Jorge, Arnaldo J Abrantes, and Jorge S Marques. Automatic Tracking of Multiple Pedestrians with Group Formation and Occlusions. In *VIIP*, pages 613–618, 2001.

[11] Pedro Miguel Torres Mendes Jorge. *Seguimento de Pessoas e Grupos em Sinais de Vídeo com Redes Bayesianas*. PhD in electrical and computers engineering, Universidade Técnica de Lisboa, 2008.

[12] Nils Krahnstoever, Ting Yu, Ser-Nam Lim, Kedar Patwardhan, Peter Tu, et al. Collaborative real-time control of active cameras in large scale surveillance systems. In *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications-M2SFA2 2008*, 2008.

[13] Keqin Liu and Qing Zhao. Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access. *IEEE Trans. on Information Theory*, 56(11):5547–5567, 2010.

[14] Tiago Marques, Luka Lukic, and José Gaspar. Observation functions in an information theoretic approach for scheduling pan-tilt-zoom cameras in multi-target tracking applications. In *Second Iberian Robotics Conference (ROBOT'2015)*, 2015.

[15] Christian Micheloni, Bernhard Rinner, and Gian Luca Foresti. Video analysis in pan-tilt-zoom camera networks. *Signal Processing Magazine, IEEE*, 27(5):78–90, 2010.

[16] Jerome Le Ny, Munther Dahleh, and Eric Feron. Multi-uav dynamic routing with partial observations using restless bandit allocation indices. In *American Control Conf.*, pages 4220–4225. IEEE, 2008.

[17] Faisal Z Qureshi and Demetri Terzopoulos. Planning ahead for ptz camera assignment and handoff. In *Third ACM/IEEE Int. Conf. on Distributed Smart Cameras (ICDSC)*, pages 1–8. IEEE, 2009.

[18] Eric Sommerlade. *Active Visual Scene Exploration*. PhD thesis, Brasenose College, University of Oxford, 2011.

[19] Eric Sommerlade and Ian Reid. Probabilistic surveillance with multiple active cameras. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 440–445. IEEE, 2010.

[20] Eric Sommerlade, Ian Reid, et al. Cooperative surveillance of multiple targets using mutual information. In *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications-M2SFA2 2008*, 2008.

[21] Wiktor Starzyk and Faisal Z. Qureshi. Multi-tasking smart cameras for intelligent video surveillance systems. *8th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, pages 154–159, August 2011.

[22] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[23] Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of applied probability*, pages 287–298, 1988.