This is to certify that the
dissertation entitled

PLANNING AND CONTROL OF MOBILE SURVEILLANCE
NETWORKS

presented by

AMIT YOGESH GORADIA

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in ELECTRICAL AND
COMPUTER ENGINEERING

Major Professor's Signature

9/20/06

Date

MSU is an Affirmative Action/Equal Opportunity Institution

PLANNING AND CONTROL OF MOBILE SURVEILLANCE NETWORKS

BY

Amit Goradia

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering

2006

P

move

sensor

The t

ing re

monit

and ex

tracke

worked

scale n

stable

The

capabil

based

an ade

simult

dorff t

targets

the sur

as the

using f

represe

ABSTRACT

Planning and Control of Mobile Surveillance Networks

By

Amit Goradia

Pervasive surveillance implies the continuous tracking of multiple targets as they move about the monitored region so that they do not leave the field of view of the sensors observing them and maintain discernable resolution for feature identification. The tasks to be performed by a surveillance system are expressed as the following requirements:(1) Automatically track the identified targets over the region being monitored; (2) Coordinate tasking of multiple sensors to maintain target visibility and execute other requested tasks;(3) Provide concise feedback and video data of a tracked target to multiple operators. Due to the inherent complexity of large networked systems and the diversity of the sensors used, a major challenge for such large scale networked systems is to design an efficient modeling and analysis tool and devise stable control algorithms for accomplishing the surveillance task.

The ability to track multiple targets simultaneously using sensors with motion capability is an important aspect of surveillance systems. Current feature-point-based visual surveillance and tracking techniques generally employed do not provide an adequate framework to express the surveillance task of tracking multiple targets simultaneously using a single sensor. This dissertation presents the method of Hausdorff tracking, which can express the surveillance task succinctly and track multiple targets simultaneously using a single sensor. Hausdorff tracking can readily express the surveillance tasks of maintaining the visibility and adequate resolution of a target as the minimization of an error (shape function) and accomplish the tracking task using feedback directly from the acquired image. Mutational equations are used to represent the motion of the target sets with respect to the motion of the camera.

Us

mo

to

trac

lanc

task

tiona

of th

disse

sive s

for m

switch

ios m

appro

Th

Hence

needs

based

over a

can b

camer

movin

design

have b

the pr

Using the mutational dynamics model, a feedback map to control the sensor motion module for accomplishing the surveillance task can be developed. Further, in order to take advantage of the redundancy in the motion control, an optimal Hausdorff tracking framework is presented.

Despite the limited sensing capability and range of the individual sensors, a surveillance network can track targets over a large region by transferring the target-tracking task from one group of sensors to another based on the motion of the target. Mutational analysis and shape-based control fail to capture the discrete switching nature of the surveillance task of tracking the target using multiple switched sensors. This dissertation presents a mutational hybrid automata (MHA) model for such pervasive surveillance networks that retains the advantages of using mutational equations for modeling the active surveillance task while also being able to model the discrete switching between various sensors. Analysis of example pervasive surveillance scenarios modeled using the MHA model and experimental results that verify the proposed approach are presented.

The active sensors needed to track the target keep changing due to target motion. Hence the concise video feedback provided to the operator to assist decision-making needs to be switched to the sensors currently involved in the tracking task. A task-based metric representing the number of cameras required to track a moving target over a monitored region is presented. This metric (in conjunction with other metrics) can be used to optimally place cameras in a monitored environment. An optimized camera selection algorithm for selecting the minimum number of cameras to track a moving target over a large area is also presented. A surveillance test-bed has been designed based on these requirements and the algorithms and subsystems described have been implemented on it. Results of the experimental implementation validate the proposed approaches.

To my parents and Prerna...

hav

duri

com

have

sugg

I

and I

I am

orato

Dr. Y

Uche

and b

ton H

of pa

M

disser

contin

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Ning Xi, without whom this would not have been possible. I would like to thank Dr. Matt Mutka for providing acute insights during our numerous discussion sessions. I would like to thank all my other guidance committee members, Dr. Hassan Khalil, Dr. Erik Goodman and Dr. Gowei Wei, who have devoted many hours on the committee meetings. Their insightful comments and suggestions have enhanced the technical soundness of this dissertation.

I would also like to express my deepest gratitude to the Dean Dr. Satish Udpa and Dr. Subir Biswas for their support and advise through the course of my career. I am grateful to my friends and colleagues from the Robotics and Automation Laboratory. The discussions with Zhiwei Cen, Dr. Yu Sun, Mike(Quan) Shi, Yang Liu, Dr. Yantao Shen, Long Pan, Dr. Jindong Tan, Dr. Imad Elhajj, Dr. Guangyong Li, Uchechukwu Wejinya, and Jiangbo Zhang have substantially contributed to my work and broadened my knowledge. I would like to express my great appreciation to Clayton Haffner, Michael Huntwork, Mat Prokos and Craig Pomeroy for the endless hours of painstaking work they have put in at the Robotics and Automation Laboratory.

My thanks go to my family, especially my fiancee Prerna and my parents. This dissertation would not have been possible without their years of encouragement and continuous support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

C

In

1.1

A se

comm

netw

consi

advan

creasi

Infras

rapidl

N

reaso

senso

infra

be ins

opera

Locom

and s

Chapter 1

Introduction

1.1 Background

A sensor network is a collection of individual nodes with each node having sensing, communication, computation and motion capabilities. Distributed wireless sensor networks have recently emerged as an important research area. Sensor networks consist of a variety of different types of sensors in distributed areas. Technological advances in wireless networking and distributed robotics have been leading to increasing research on distributed sensing applications using wireless sensor networks. Infrastructure-less surveillance and monitoring are important applications of such rapidly deployable sensor networks.

Networked surveillance systems provide an extended perception and distributed reasoning capability in monitored environments through the use of multiple networked sensors. Various types of sensors with varying sensing modalities such as cameras, infrared detector arrays, laser rangefinders, omnidirectional acoustic sensors etc., can be instantly deployed in hostile environments, inaccessible terrain and disaster relief operations to obtain vital reconnaissance information about the area being surveyed. Locomotion and active sensing on individual nodes can greatly increase the range and sensing capability of the network. Such instantly deployable, wireless, mobile,

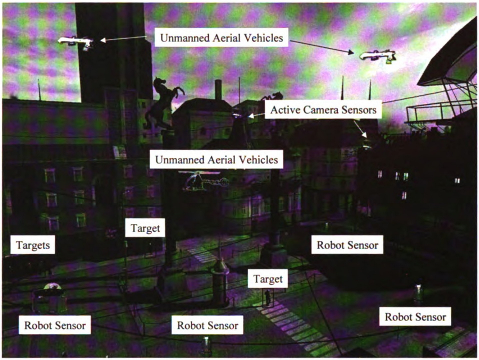


Figure 1.1: Pervasive Surveillance Scenario

distributed sensor networks networks find myriad civilian and military applications, such as battle field surveillance, environment monitoring, scene reconstruction, motion detection and tracking, remote sensing and global awareness.

There is a growing interest in surveillance applications due to the growing availability of sensors and processors at a reasonable cost. There is also a growing need from the public for improved safety and security in large urban environments and improved usage of resources of public infrastructure. This, in conjunction with the increasing maturity of algorithms and techniques, is making possible the application of this technology in various application sectors such as security, transportation, and the automotive industry. In particular, the problem of remote surveillance of unattended environments has received growing attention in recent years, especially in the context of:

- Safety in transport applications [1], [2], such as monitoring of railway stations

S

be c

How

coop

tasks

const

due t

Pe

gets a

impor

a targ

condu

a part

region

all reg

[4], [5], underground stations [3, 4], airports [5, 6] and airplane routes [7–9], motorways [10], [11], urban and city roads [12, 13], maritime environments [14, 15];

- Safety or quality control in industrial applications, such as monitoring of nuclear plants [16] or industrial processing cycles [1, 2];
- Improved security for people's lives, such as monitoring of indoor or outdoor environments like banks [29], supermarkets [3], car parking areas, waiting rooms [17], buildings [18, 19], etc., remote monitoring of the status of a patient [20], remote surveillance of human activity [21, 22]; d) military applications for surveillance of strategic infrastructure [23, 24], enemy movements in the battlefield [25, 26] and air surveillance [27, 28].

Surveillance networks are composed of numerous independent nodes which can be controlled individually. Hence they have high fault tolerance to node failure. However, this redundancy and fault tolerance comes at a price. One needs to design cooperation and coordination schemes in order for the individual nodes to perform tasks together. Further, these cooperation and coordination mechanisms should not consume too much power and other resources, as they would render the advantages due to redundancy useless.

Pervasive surveillance can be defined as the active monitoring of recognized targets as they move through a large monitored area using a network of sensors. An important characteristic of pervasive surveillance systems is their capability to track a target over a large area using multiple sensors. Figure 1.1 depicts a scenario for conducting reconnaissance on a city square. Each individual sensor can view only a part of the monitored area, but collectively the network can monitor the entire region. There are a number of reasons that optimal coverage may not be available for all regions of a large area being monitored, such as: a lack of prior knowledge of the

e
a
a
n
o
se
m
ro
th
in
th
sy
iss

lan
netw
lang
spar
T
cise
mult
and r
There
ator o
stream

environment, a paucity of locations for placement of sensors in the environment, or a shortage of sensors to be deployed. Motion capability of the sensors and the use of active sensing can greatly enhance the coverage quality of the network. For example, monitoring of areas with sparse or no coverage or acquiring higher resolution coverage of the targets can be accomplished by changing the active field of view of the fixed sensors or deploying mobile sensors. Further, the area monitored using a limited number of sensors can be increased by using active (mobile) sensors such as UAV's, robot-mounted cameras or cameras mounted on pan-tilt heads etc. The mobility of these sensors can be leveraged to increase the area monitored by each sensor, which in turn increases the total area monitored. However, the mobility combined with the scalability requirements renders modeling and analysis of pervasive surveillance systems using active sensors a challenging task and brings to light many research issues.

Compared with traditional surveillance networks, an IP-network-based surveillance system is easy to deploy, maintain, and expand. The ubiquitous nature of IP networks will dramatically reduce the cost to set up a traditionally expensive surveillance network. We envision that the possible usage of surveillance networks could span the venues of industry, military and daily lives.

The surveillance network must provide, to the human operator, a timely and concise view of the relevant activities within the environment being monitored. Providing multiple video feedback streams often causes loss of attention span of the operator and makes it hard to keep track of the various activities over the various cameras. Therefore only video streams from relevant sensors should be presented to the operator on a per activity basis. This involves automatically switching the active video stream presented to the operator.

1

T

re

the

shou

safe

sens

track

A

sors

amon

in gen

mains

sensor

involve

of targ

1.2 Challenges and Problems in Surveillance Networks

The tasks to be performed by a surveillance system can be expressed as the following requirements:

1. Deploy the sensors in order to monitor the region under surveillance;
2. Identify multiple moving targets based on predefined models;
3. Automatically track the identified targets over the region being monitored;
4. Coordinate tasking of multiple sensors to maintain target visibility and execute other requested tasks;
5. Provide concise feedback and video data of a tracked target to multiple operators.

The sensors should be deployed in the monitored region in order to maximize the likelihood of identifying and tracking the targets. In adversarial scenarios, care should be taken in order to conceal the location of the sensors in order to ensure their safety and the integrity of the task. A major challenge in the deployment phase is sensor placement in order to maximize the covered area while also ensuring continuous tracking of a target when it traverses the coverage regions of multiple sensors.

An important aspect of sensor deployment is calibration. Calibration of the sensors enables multiple sensors to coordinate with each other and share information amongst themselves. This cooperation amongst sensors leads to better sensing and is in general a desirable property. Calibration of the sensors can be viewed in many domains (feature spaces). The most literal one is calibrating the location of the deployed sensors with respect to each other and a global frame. Other types of calibration can involve calibrating the color spaces of various cameras, which will facilitate the sharing of target models amongst the sensors.

de
is
vi
ta
is
res
de
ord
is t
ord

at le
track
must
and
agen
very
motio

Th
the s
of th
to th
feedb
prede

V
the va

In order to perform automated surveillance there are two major subtasks: target detection and target tracking. A target perception and video understanding module is responsible for detecting and classifying the various targets in the active field of view (FOV) of the sensor and performing temporal consolidation of the detected targets over multiple frames of detection. Moving target detection and classification is known to be a difficult research problem and has been the focus of many recent research efforts [29]. The next challenge would be to classify and associate the various detected image blobs to discernable targets and maintain their temporal tracks in order to pervasively track them. Once the targets have been identified, the challenge is to recognize the relationships of the target to the environment and other targets in order to attribute aims and motives to the tracked targets.

The target tracking problem involves continuously keeping the target in view of at least one sensor. When the target is escaping the viewing area of the current tracking sensor, the current sensor needs to be moved (relocated) or the tracking task must be transferred to another sensor which can currently view the target. Modeling and design of such cooperative tracking using multiple sensors is a challenge. Active agents further make the modeling and design of such pervasive surveillance systems very difficult. A modeling and design framework for coordinating and controlling the motion of the active cameras to pervasively track targets is necessary.

The various methods through which the the human operator communicates with the surveillance network form an integral part of the usability, usefulness and efficacy of the surveillance network. These methods include passing commands and queries to the network indicating the intention of the surveillance task and receiving realtime feedback and results of the queries in progress and alarms for violations of various predefined conditions.

Video feedback is provided to the operator in order to apprise the operator of the various tracking and monitoring tasks being currently executed by the sensors.

He

ov

sh

be

fe

th

eva

the

int

and

surv

dela

rand

31].

1.3

Netw

muni

ature

Depl

capal

deplo

chara

dedica

[34 36

However, due to the large number of sensors involved, the operator can be easily overwhelmed by the amount of video information. Hence, only relevant video feeds should be provided to the operator and, based on the task, the video feed should be switched from sensor to sensor. There are various choices for the switched video feedback implementation and they have their advantages and disadvantages based on the scenario they are being applied to. A performance evaluation metric is needed for evaluating the efficacy of a particular scheme when being applied to certain scenarios.

Direct human intervention and control of the various cameras can be used for the logging and tracking of targets which are not predetermined and programmed into the network. Hence the human operator needs to move the active cameras and receive real-time video feedback to manually track targets. However for such surveillance systems, direct manual tele-control of the active active camera over the delayed IP network suffers from stability and synchronization problems due to the random, unbounded and asymmetric communication time delay in the network [30, 31]. This can result in reduced performance and even loss of tracking.

1.3 Literature Review

Networked surveillance systems have received much attention from the research community due to their many pervasive applications [32], [33]. Previous work from literature is reviewed here.

Deployment and Calibration: Based on their mobility and rapid-deployment capability, the operation of MSN's can be divided into two separate phases, namely deployment and surveillance. Infrastructure-less rapid deployment is an important characteristic of mobile surveillance networks and many research efforts have been dedicated to optimally deploy sensor nodes for increasing their total sensing area [34–36]. A global kinematic representation of a network of connected sensors $\mathcal{R} =$

$\{R$

sug

me

Ta

me

and

The

targ

solid

detec

appro

tion

rangi

next

discer

them.

and B

target

Targe

keepi

specif

[39] [3

involv

the ta

succin

blurri

$\{R_1, R_2, ..., R_n\}$ using only localization information between one-hop neighbors is suggested in [34]. This relationship between the various nodes is key to sharing meaningful information between them.

Target Identification and Video Understanding: Beyond the initial deployment phase, in the surveillance phase there are two major subtasks: target detection and target tracking. Figure 5.8 depicts the general architecture of a sensor node. The target perception module is responsible for detecting and classifying the various targets in the active field of view (FOV) of the sensor and performing temporal consolidation of the detected targets over multiple frames of detection. Moving target detection and classification is known to be a difficult research problem [29]. Many approaches such as active background subtraction [37] [38] and temporal differentiation have been suggested for detecting and classifying various types of moving targets ranging from single humans and human groups to vehicles and wildlife [39] [38]. The next problem would be to classify and associate the various detected image blobs to discernable targets and maintain their temporal tracks in order to pervasively track them. Various approaches such as extended Kalman filtering, pheromone routing and Bayesian belief nets have been suggested for maintaining the track of the various targets [40].

Target Tracking with Active Camera: A general surveillance task involves keeping moving targets in the active sensing region of the sensors with a certain prespecified resolution. Research approaches to this problem found in recent literature [39] [37], generally use visual servo control [41] or gaze control [42], which mainly involve feature-point-based tracking and fail to describe the basic task of maintaining the target in the sensor's active field of view with certain resolution, effectively and succinctly. These approaches result in excessive camera motion that may result in blurring and other undesired effects, which in turn have detrimental effects on the su

mu

Se

tia

mo

the

fron

itse

in a

requ

In or

of th

to an

the o

opera

both

Hy

of ma

switch

in a p

model

suffers

Netwo

individ

ple nod

nisms

surveillance task. Further, using these approaches it is not possible to describe a multiple target tracking task with randomly moving targets using a single sensor.

Sensor Cooperation and Coordination: Multiple groups of nodes are sequentially involved in order to pervasively track the target as it traverses through the monitored area which is much larger than the range of an individual sensor. Hence there needs to be a handoff mechanism which essentially "switches" the tracking task from one group of sensors to another based on the movement of the target. Tracking itself can be described as a switching phenomenon since the act of tracking a target in a given monitoring area which is greater than the viewing area of any given sensor requires there to be some form of hand-off, or switch, between a number of sensors. In order to assure that the target visibility will be maintained even when it goes out of the capable field of view of one sensor, the system must switch the tracking task to another sensor that has to "pick up" the target before it leaves the field of view of the original sensor. This discrete switching phenomenon along with the continuous operation of the individual nodes can be modeled as a hybrid system consisting of both continuous and discrete transitions.

Hybrid automata models for describing switching systems have been the focus of many recent research efforts [43], [44]. They can effectively capture the discrete switching nature of the surveillance task. However, their continuous part is modeled in a point-based framework using differential equations. This model fails to effectively model the continuous operation of the individual nodes in the surveillance task and suffers from the drawbacks mentioned previously.

Network topology The efficient and timely exchange of information between the individual sensing nodes is imperative for cooperation and coordination of multiple nodes in order to perform pervasive target tracking. Ad-hoc, multi-hop mechanisms are best suited for such mobile infrastructure-less wireless networked systems.

N

at

al

si

re

si

ne

to

Op

sur

by

ove

hib

pro

the

ple

is n

and

jects

Force

GUI

that

ates

comp

Al

used

Networking and routing mechanisms for distributed wireless sensor networks have attracted a lot of attention in recent literature [45] [46] [47]. These approaches generally assume scalar (low volume) data transfer between the individuals nodes and the sink node (which is just a consumer of the information.) However, surveillance tasks require fast and timely transfer of vector (large volume) data, e.g. video data, to the sink and also within the nodes themselves. This scenario may create an unbalanced networking load on the nodes along the optimal relay path, which can be detrimental to the longevity of the network.

Operator Interface Design: Video feedback is an essential component of the surveillance system. A single human operator cannot effectively monitor a large area by looking at dozens of monitors showing raw video output. That amount of sensory over-load virtually guarantees that information will be ignored and requires a prohibitive amount of transmission bandwidth. In [39] an approach is presented which provides an interactive, graphical user interface (GUI) showing a synthetic view of the environment, upon which the system displays dynamic agents representing people and vehicles. This approach has the benefit that visualization of scene events is no longer tied to the original resolution and viewpoint of a single video sensor and the operator can therefore infer proper spatial relationships between multiple objects and scene features. Another program called the the Modular Semi-Automated Forces (ModSAF) program provides a 2-D graphical interface similar to the VSAM GUI [48], with the ability to insert computer-generated human and vehicle avatars that provide simulated opponents for training [49]. The ModStealth program generates an immersive, realistic 3-D visualization of texture-mapped scene geometry and computer-generated avatars [50].

Although automatic image analysis and video understanding tools [39] can be used to facilitate identification of targets and activation of alarms or logs for certain

s
t
n
pı
Tl
tic
net
dev

curı
sens
mov
unne
tenti
over
to th
switc

L
mobi
targe
in lit
tainir
to us
cann
targe
as ser
for pe

surveillance tasks, the operator needs the video feedback to make decisions about the tracking task which may not have been pre-programmed or to independently task the network based on the current feedback received from the sensors. The video feedback provided to the operator can be transmitted over either analog or digital channels [51]. The use of analog modulating and transmission techniques for surveillance applications has been reported in [52]. Receiving digital video feedback over an IP-based network from the networked camera sensors has also received much attention with the development of the various video compression and communication protocols [53–57].

Since multiple cameras are deployed to track the identified targets, multiple, concurrent feedback video streams maybe required for monitoring the target. These sensors initiating these streams will be changing from time to time as the target moves out of range of the current sensors tracking it. However, providing multiple unnecessary (unrelated to the task) video feedback streams often causes loss of attention span of the operator and makes it hard to keep track of the various activities over the cameras. Hence only video streams from relevant sensors should be presented to the operator on a per activity basis. This is done through automatic or manual switching of the camera streams that are presented to the operator

Literature Review Summary: From the above review on recent literature on mobile surveillance networks, the focus of the major works reviewed revolves around target identification and video understanding. However, most of the approaches cited in literature lack a generalized framework for modeling the surveillance task of maintaining visibility of multiple targets with adequate resolution. These approaches tend to use visual servoing which involves only feature point based target tracking and cannot adequately describe multi-target tracking using a single camera sensor. Multi target tracking is addressed in a very ad-hoc fashion in literature using methods such as sensor slaving and gaze control. A generalized modeling and analysis framework for pervasive target tracking using multiple distributed sensors is not available in

c

c

a

b

b

ov

ar

ap

1.4

The

usin

sens

of th

effort

scena

Th

model

infra-

toring

the co

tracki

of a s

is in t

Hausd

the se

For

current literature. Although switched surveillance implementations using multiple distributed sensors have received much attention in recent literature, a comparative analysis of video feedback schemes for various visual surveillance scenarios has not been reported. Further, optimized sensor deployment based on minimizing the number of sensor switches in order to pervasively track a target using multiple sensors over a large area has not been explored. This dissertation tries to address these issues and develop a formal modeling, analysis and design framework for mobile surveillance applications.

1.4 Objectives of this Research

The main objective of this research is to implement a pervasive surveillance network using active camera sensors connected to each other using an IP-based network. The sensors have motion capability and are required to continuously maintain visibility of the identified targets in the surveillance region. The objectives of this research effort are to provide a modeling and design framework for such pervasive surveillance scenarios.

This dissertation proposes a mutational-analysis-based topological framework for modeling and design of mobile surveillance networks which find applications in myriad infrastructure-less, rapidly deployable pervasive multi-target surveillance and monitoring scenarios. Using the concept of mutations of domains and shape analysis, the conditions for accomplishing various surveillance tasks such as continuous target tracking while maintaining appropriate image resolution can be derived. The design of a surveillance task using image-based Hausdorff tracking, used when the target is in the active field of view of the sensor is presented. The method of cooperative Hausdorff tracking, used to track targets which are outside the active field of view of the sensor using the observations from other sensors, is presented.

For pervasive surveillance scenario which requires a hand-off or switching mechanism between a number of sensors, this dissertation proposes the Mutational Hybrid Automata (MHA) model, that combines the discrete switching structure of hybrid automata with the domain-based evolution of mutational systems in order to overcome their individual limitations. It further presents example surveillance tasks modeled using the proposed MHA model and provides an analysis and proof of stability of the example systems.

A switched video feedback system is an important component of the pervasive surveillance network. This dissertation compares different design alternatives for the implementation of the switched video feedback system, such as MJPEG [57] H.261 [53] and H.263 [54] transported over RTP transport protocol [56]. Their performance under certain scenarios/tasks are measured and the advantages and disadvantages of different alternatives are analyzed. Various performance metrics are proposed to compare the suitability of these schemes based on a given scenario/task. Further an optimal sensor selection algorithm for selecting minimum number of sensors to track a moving target is presented.

1.5 Major Contributions of this Dissertation

The major contributions of this dissertation can be classified into various categories. This section provides a brief statement about the major contributions of this dissertation.

Theoretical Contribution

- A new domain based modeling framework called Hausdorff tracking for tracking multiple targets using continuous camera motion is developed.

- A mutational hybrid automata model is developed for modeling multi-sensor pervasive target tracking tasks. Using this model, autonomous as well as controlled sensor switching scenarios can be modeled.

- Stability analysis of the proposed Hausdorff tracking scheme using the shape Lyapunov theorem has been presented.

- Target tracking stability of the pervasive surveillance task modeled using the mutational hybrid automata models is analyzed for example systems using a two-step procedure.

Technological Contributions

- A switched video feedback system that enables streaming of relevant video streams to a human operator for multiple clients has been implemented using a H.264 video encoding scheme. The operator clients developed allow the automated monitoring as well as active control of the networked sensors.

- A task/scenario based performance metric is proposed to analyze the suitability of application of a particular video transport scheme for that task. Various schemes such as MJPEG, H.261, H.263 and H.264 are compared for different scenarios using the proposed metric.

Design Contributions

- An optimality framework for exploiting the redundancy in task accomplishment using Hausdorff tracking is developed. Physical programming has been used to provide a physical meaning to the combinations of the various cost functions.

- The video switching algorithm is modeled as a discrete-time finite-state dynamic system and an assessment metric is proposed to analyze the locations and configurations of the cameras for switched video feedback in order to minimize the number of switches while maximizing the resolution sustained by the tracked target at the tracking camera.

1.

Th
for
elin
on
mod

of a
Hau
is de
deve

C
sens
exam
auto
coop
is als

C
lance

Contributions for Experimental Work

- A visual surveillance test-bed has been implemented for testing and validating the efficacy of the various approaches proposed.

- A mobile robot and articulated robot arm have been integrated as a part of the motion platform for the visual sensors. These can be used in the future for further extension and experiments of this research.

1.6 Organization of this Dissertation

The scope of this dissertation is to investigate a novel modeling and design framework for mobile surveillance networks. The work is divided into three parts - namely modeling, analysis and design. First, in Chapter 2, we present the necessary background on the mathematical tools which will be required to develop and analyze the formal modeling and design framework proposed.

In Chapter 3, the new modeling framework using mutational equations for motion of a single sensor is presented. Using this mutational equation model, a shape-based Hausdorff tracking control scheme to track multiple targets with viewing constraints is developed. Chapter 3 further presents the Mutational Hybrid Automata model developed for modeling multi-sensor cooperative target tracking scenarios.

Chapter 4 presents the shape Lyapunov theorem for analyzing the stability (in the sense of Lyapunov) of the Hausdorff tracking method proposed. It further presents examples of multi camera surveillance systems modeled using the mutational hybrid automata model. A procedure for analyzing the stability of the example multi-sensor cooperative tracking systems modeled using the mutational hybrid automata model is also presented.

Chapter 5 presents the work done toward design of the switched video surveillance system. It presents a brief review on grouping architectures and presents a sensor allocation algorithm which can be used to allocate sensors to targets. A design framework for optimal tracking of multiple targets by exploiting the redundancy of task accomplishment is presented. Further algorithms for sensor placement in order to minimize sensor switching while tracking moving targets and algorithms for optimized target tracking using switched sensors are also presented.

Experimental validations of the proposed algorithms are presented in Chapter 6. The experimental results for Hausdorff tracking, multi-sensor cooperation using the mutational hybrid automata model and multi-sensor target allocation are presented. Further results of optimal Hausdorff tracking and multi-target tracking experiments are also presented. Chapter 6 also provides a performance analysis of the various video transmission schemes and proposes a task- and scenario- based metric for evaluating the various video compression and transmission schemes for switched video surveillance tasks. Further, simulation results for optimized target tracking using switched sensors is also presented.

Conclusions and future directions are presented in Chapter 7.

C

M

Th
the
equ
Fur
mod
vari

2.1

Let
and
be cl
subs
close
used

A

•

Chapter 2

Mathematical Background

This chapter presents the mathematical preliminaries that will be used in the rest of the dissertation. A definition of a dynamic systems is presented. Then mutational equations, which are used to represent domain based dynamic systems are introduced. Further hybrid dynamic systems are introduced and the Mutational Hybrid Automata model is presented. Shape analysis and shape functions are introduced to provide various measures on sets (shapes).

2.1 Introduction

Let X be a collection of variables; then $\mathcal{X}$ denotes the valuations of these variables and x will be used to refer to both the variable and its valuation (the reference should be clear from the context). 2^X denotes the power space of X i.e., the family of all subsets of X and $\mathcal{K}(X)$ denotes the space of all nonempty compact subsets of a given closed set $W \subset \mathbb{R}^n$. We will use $\mathcal{M}$ to denote a general metric space and $\mathcal{X}$ will be used to denote a finite-dimensional vector space.

A controlled dynamical system is a system $\Sigma = [\mathcal{X}, \Gamma, \mathcal{U}, \phi]$ where,

- $\mathcal{X}$ is an arbitrary topological space called the state space of Σ;

th

Ξ

if l

dir

ma

pro

con

enti

call

wh

l

are

elem

W ⊂

inclu

a ge

syste

as a

mode

- Γ is the time set, which is a transition semigroup with identity;
- $\mathcal{U}$ is a nonempty set called the control-value space of Σ;and
- The map $\phi : \mathcal{X} \times \Gamma \times \mathcal{U} \mapsto \mathcal{X}$ is a continuous function satisfying the identity and semigroup properties [58].

The dynamical system can also be denoted by the system $\Sigma = [\mathcal{X}, \Gamma, \mathcal{U}, f]$ where the transition function f is the generator of the extended transition function ϕ [58].

A discrete-time dynamic system is a dynamic system Σ for which $\Gamma = \mathbb{Z}$ where, $\mathbb{Z}$ is the set of integers. The discrete-time dynamic system, Σ is finite dimensional if both $\mathcal{X}$ and $\mathcal{U}$ are finite dimensional and the dimension of the system Σ is the dimension of $\mathcal{X}$. A system Σ is complete if every input is admissible for every state.

A hybrid dynamic system is an indexed collection of dynamic systems with some map defined for jumping among them. Various models for hybrid systems have been proposed in the literature [43] [44]. Generally, these approaches assume that the continuous dynamics part of the hybrid systems are defined by the solutions of differential equations ($\dot{x} = f(x(t))$) where $x(t) \in \mathcal{X} \subset \mathbb{R}^n$ and the function $f : \mathcal{X} \mapsto \mathbb{R}^n$ is called a vector field on $\mathbb{R}^n$. The resulting dynamical system is given by $\phi(x_0, t) = x(t)$ where $x(t)$ is a solution of the differential equation starting at x_0 at $t = 0$.

For the case of the pervasive surveillance problem, the variables to be tracked are not elements of a vector space but geometrical domains themselves which are elements of the space $\mathcal{K}(W)$ of all non-empty compact subsets of a given closed set $W \subset \mathbb{R}^n$ with distance $\boldsymbol{d}$. Hence we need to enhance the hybrid systems model(s) to include the cases where the continuous state is not an element of a vector space but a geometric domain $K \in \mathcal{K}(W)$. In this case we describe the continuous dynamical system as the solution of a mutational equation $\mathring{K} \ni \varphi(K(t))$ [59] which is defined as a tube. Using this formalism we propose a Mutational Hybrid Automata (MHA) model which can adequately describe a pervasive surveillance scenario.

u

as

Def

of a

on t

i

poin

2.2

This

tions

Haus

We also define the projection of a point x which is an element of a vector space V on a set A whose elements are also the elements of the vector space V as:

Definition 2.1.1 *(Projection of a point x on a set A: $\Pi_A(x)$) The projection $\Pi_A(x)$ of a point x on set A is defined as the set of points in A having the distance $d_A(x)$ from point x and is written as:*

$$\Pi_A(x) = \{z \in A | \|z - x\| = d_A(x)\} \tag{2.1}$$

where, $d_A(x)$ is the distance of set A from point x defined as:

$$d_A(x) = \inf_{y \in A} \|x - y\| \tag{2.2}$$

Extending this concept further, we now define the projection of a set A on set B as:

Definition 2.1.2 *(Projection of a set B on a set A: $\Pi_A(B)$) The projection $\Pi_A(B)$ of a set B on set A is defined as the union of the projections of all the points in B on the set A as:*

$$\Pi_A(B) = \bigcup_{x \in B} \Pi_A(x) \tag{2.3}$$

where, $Pi_A(x)$ is the projection of a point x on a set A which is the set of all the points in A having the distance of $d_A(x)$ from point x.

2.2 Mutational Equations

This section provides a basic introduction to shape analysis using mutational equations [60] and relevant concepts useful for developing the theoretical foundation for Hausdorff tracking. The sensor coverage area and the target are readily represented

as sets (domains) in $E \subset \mathbb{R}^n$. In order to study the motion of these domains we need to define a differential calculus on the space of all compact, non-empty subsets $\mathcal{K}(E)$ of the closed set E. The mathematical framework of mutational analysis allows us to extend the concept of differential equations to the metric space $\mathcal{K}(E)$.

For defining mutational equations, we supply the space $\mathcal{K}(\mathbb{R}^n)$ with a distance $\boldsymbol{d}$, for example the Hausdorff distance between domains $K_1, K_2 \in \mathcal{R}^n$ defined by $\boldsymbol{d}(K_1, K_2) = \sup_{q \in \mathbb{R}^n} \|d_{K_1}(q) - d_{K_2}(q)\|$, where, $d_K(q) = \inf_{p \in K} \|q - p\|$ represents the distance between the point q and domain K.

2.2.1 Evolution of a Tube

Tubes are domains evolving with time and can be defined as a map $K(\cdot) : \mathbb{R}^+ \mapsto \mathcal{K}(E)$. The deformation (motion) of the coverage and the target sets can be represented using tubes:

$$K(\cdot) : \begin{cases} \mathbb{R}^+ \mapsto \mathcal{K}(E) \\ t \mapsto K(t). \end{cases} \tag{2.4}$$

The evolution of a tube can be described using the notion of a time derivative of the tube as the perturbation of a set. Associate with any Lipschitz map $\varphi : E \mapsto E$, a map called the transition $\vartheta_\varphi(h, q) := q(h)$, which denotes the value at time h of the solution of the differential equation:

$$\begin{cases} \dot{q} = \varphi(q) \\ q(0) = q_0. \end{cases} \tag{2.5}$$

Extend this concept of a transition to the space $\mathcal{K}(E)$ by introducing the reachable set from set K at time h of φ as

$$\vartheta_\varphi(h, K) := \{\vartheta_\varphi(h, q_0)\}_{q_0 \in K} \tag{2.6}$$

The curve $h \mapsto \vartheta_\varphi(h, K)$ plays the role of the half lines $h \mapsto x + hv$ for defining differential quotients in vector spaces. Using the concept of the reachable set the time derivative of a tube can be defined as a mutation:

Definition 2.2.1 *(Mutation) Let $E \subset \mathbb{R}^n$ and $\varphi : E \mapsto E$ be a Lipschitz map ($\varphi \in Lip(E, \mathbb{R}^n)$). If for $t \in \mathbb{R}^+$, the tube $K : \mathbb{R}^+ \mapsto \mathcal{K}(E)$ satisfies:*

$$\lim_{h \to 0^+} \frac{\boldsymbol{d}(K(t+h), \vartheta_\varphi(h, K(t)))}{h} = 0, \tag{2.7}$$

then, φ is a mutation of K at time t and is denoted as:

$$\overset{\circ}{K} \ni \varphi \tag{2.8}$$

It should be noted that φ is not a unique representation of the mutational equation, which justifies the use of the notation ($\ni$) [60].

Consider a function $\varphi : \mathbb{R} \times \mathcal{K}(E) \mapsto Lip(E, \mathbb{R}^n)$ as a map associating a Lipschitz map φ to a pair (t, K). Using this map, we can define the mutation of tube $K(t)$ as:

$$\overset{\circ}{K}(t) \ \ni \ \varphi(t, K(t)), \ \forall t \geq 0 \tag{2.9}$$

2.2.2 Controlled Mutational Equations

A controlled mutational equation can be written as:

$$\overset{\circ}{K}(t) \ \ni \ \varphi(t, K(t), u(t)), \ \forall t \geq 0 \tag{2.10}$$

U

c

2.

Hy
dis
sys
ere
equ
the
can
hyb
and
wor
equ
syst
the

$$u(t) \in U, \tag{2.11}$$

where, $\varphi : \mathbb{R}^+ \times \mathcal{K}(E) \times U \mapsto Lip(E, \mathbb{R}^n)$ is a continuous map associating a Lipschitz map with (t, K, u) and $u(t) \in U$ is the control input.

A feedback law can be defined as a map $\mathcal{U} : \mathcal{K}(E) \mapsto U$ associating a control u with a domain $K(t)$ as:

$$u(t) = \mathcal{U}(K(t)) \tag{2.12}$$

Using a controlled mutational equation, we can model the motion of the target and coverage sets due to the motion input u to the camera/robot.

2.3 Hybrid Dynamic Systems

Hybrid systems generally refer to dynamical systems which consist of a mixture of discrete and continuous components. Such systems describe the time evolution of systems with both continuous and discrete parts. Hybrid dynamical systems considered in the literature generally have continuous dynamics modeled using differential equations which depend on some discrete phenomena like switching and jumping in the continuous and discrete state of the system respectively. These hybrid phenomena can be modeled using hybrid automata proposed by Lygeros and Sastry [44] [61]. A hybrid automaton describes the interaction and time evolution of both the discrete and continuous parts of a hybrid dynamical system. The hybrid automata framework models the continuous part of a hybrid dynamical system as using differential equations. Edge maps and reset maps are used to capture the discrete nature of the system while guard conditions and domains are used to describe the interaction of the discrete and continuous components.

A hybrid automaton (HA) is a collection

$$H = [Q, X, \Sigma, \text{Init}, D, E, G, R] \tag{2.13}$$

with the constituent parts as follows.

- Q is the set of index states or discrete states.
- X is the set of continuous variables.
- $\Sigma = \{\Sigma_q\}_{q \in Q}$ is a collection of controlled dynamical systems, where each $\Sigma_q = [\mathcal{X}_q, \Gamma_q, f_q, \mathcal{U}_q]$ is a controlled dynamical system. $\mathcal{X}_q$ are the valuations of the continuous state variables X_q, f_q are the continuous dynamics and $\mathcal{U}_q$ is the set of continuous controls.
- $Init \subset \mathcal{Q} \times \mathcal{X}$ is a set of initial states.
- $D : \mathcal{Q} \mapsto 2^{\mathcal{X}}$ is a domain. Each discrete state has a domain of possible continuous states associated with it. The domain is the set of valuations of the continuous states where the particular associated discrete state is valid.
- $E \subset \mathcal{Q} \times \mathcal{Q}$ is a set of possible edge transitions between the discrete states.
- $G : E \mapsto 2^{X}$ is a guard condition. Each edge in E is mapped to a manifold in $\mathcal{X}$ which triggers a mode switch when the continuous state hits this manifold.
- $R : E \times \mathcal{X} \mapsto 2^{\mathcal{X}}$ is a reset map which causes the continuous state to be reset due to the mode switch in the discrete state.

The state of the above hybrid dynamical system can be denoted as $S = \cup_{q \in \mathcal{Q}} \mathcal{X}_q \times \mathcal{Q}$. The above model can be used to model the discrete phenomena mentioned earlier. For more details about this model refer to [44].

2.4

Mut

tems

Usin

tions

whe

U

Hyb

with

-

-

-

-

-

2.4 Mutational Hybrid Dynamic Systems

Mutational equations can be incorporated into the definition of general dynamical systems as they satisfy the various nontrivialitry, semigroup and identity properties [58]. Using the concept of transitions of domains introduced earlier and mutational equations, we can write a general dynamic system as:

$$\Sigma = [\mathcal{K}(W), \Gamma, \vartheta_{\varphi}] \quad \text{or} \tag{2.14}$$

$$\Sigma = [\mathcal{K}(W), \Gamma, \varphi] \tag{2.15}$$

where ϑ_{φ} is the transition function and φ is the generator of that transition function.

Using the above definition of a dynamic system, we can now define a Mutational Hybrid Automaton (MHA) as:

$$H_m = [Q, M, \Sigma, \text{Init}, D, E, G, R] \tag{2.16}$$

with the constituent parts as follows.

- Q is the set of index states or discrete states.
- M is a collection of the continuous variables $K_i \in \mathcal{K}(W)$
- $\Sigma = \{\Sigma_q\}_{q \in Q}$ is a collection of controlled dynamic systems, where each $\Sigma_q = [\mathcal{K}(W)_q, \Gamma_q, \varphi_q, \mathcal{U}_q]$ is a controlled dynamic system. $\mathcal{K}(W)_q$ are the continuous state spaces, φ_q can be construed as the continuous dynamics and $\mathcal{U}_q$ is the set of continuous controls.
- Init $\subseteq \mathcal{M} \times \mathcal{Q}$ is a set of initial states.
- $D : \mathcal{Q} \mapsto 2^{\mathcal{M}}$ is a domain which associates a domain of operation with every $q \in \mathcal{Q}$.

t

e

v

f

2

S

o

E

fu

re

2.

Sh

su

- $E \subseteq \mathcal{Q} \times \mathcal{Q}$ is a set of edges along which discrete transition takes place.
- $G : E \mapsto 2^{\mathcal{M}}$ is a collection of guard conditions which enable discrete transitions.
- $R : E \times \mathcal{M} \mapsto 2^{\mathcal{M}}$ is a reset map which resets the continuous state when a discrete transition is enabled.

Thus, $S = \bigcup_{q \in Q} \mathcal{M} \times \{q\}$ is the hybrid state space of H_m.

A MHA can be pictured as an automaton as in Figure 4.2. There, each node is a constituent dynamical system with the index state as the name of the node.

This definition is a fairly broad and powerful modeling framework which subsumes continuous differential equations, mutational equations, discrete time difference equations, finite automata as well as switched controlled systems and non-smooth differential/mutational equations. A more intuitive explanation of the various components will be provided via an example application to pervasive surveillance networks in the following section.

2.5 Shape Analysis

Shape analysis deals with problems where the variables are not vectors of parameters or functions, but the shapes of geometric domains or sets K contained in a subset $E \subset R^n$. The measure of deformation of a set K can be expressed using shape functions as: $J : \mathcal{K}(E) \mapsto \mathbb{R}$. The directional derivative of the shape function represents the change in the shape function $J(K)$ due to the deformation of the shape K.

2.5.1 Shape Functions

Shape functions are set-defined maps defined from $\mathcal{K}(E) \mapsto \mathbb{R}$ and provide a "measure" of the deformation of $K \in \mathcal{K}(E)$. For example we can use a shape function r

s

(

T

T

Th

$t =$

to check the distance of the set K to another set $\hat{K}$, or whether the reference set $\hat{K}$ is contained within the current set K. Shape acceptability and optimality can be studied using shape functions J as optimization cost functions as is shown in [62]. Various shape functions can be readily developed which can adequately describe the surveillance task of set coverage or a measure of the size of a set, etc.

2.5.2 Directional Derivative of Shape Functions

The directional derivative of the shape function [62] represents the change in the shape function $J(K)$ due to the deformation of the shape K. It can be construed as the analog of the directional derivative in vector spaces and provides us with a measure of the change in the task criterion due to motion of the coverage or target set.

Consider a function $J : H \mapsto \mathbb{R}$ where H is a real Hilbert space such as $\mathbb{R}^n$. The Gateaux (directional) derivative of J at the point q in the direction of v is defined as:

$$DJ(q)(v) = \lim_{t \to 0} \frac{J(q + tv) - J(q)}{t} \tag{2.17}$$

This can also be looked at in a different way. Suppose $q(t)$ is defined as:

$$\dot{q}(t) = v, \qquad q(0) = q \tag{2.18}$$

This means that if $j(t) = J(q(t))$ then,

$$\begin{aligned} \frac{dj(t)}{dt} = \dot{j}(t) &= \lim_{t \to 0} \frac{J(q(t)) - J(q(0))}{t} && (2.19) \\ &= \lim_{t \to 0} \frac{J(q + tv) - J(q)}{t} = DJ(q)(v) && (2.20) \end{aligned}$$

Thus, the Gateaux (directional) derivative involves the point q and the direction v at $t = 0$.

sha

tul

as:

wh

forn

can

2.6

This

tati

hyb

nan

tinu

use

Au

surv

acce

Extending this same concept when q is not an element of a Hilbert space, but a shape (domain) K and v is replaced by φ, which is the direction of mutation of the tube K. Thus the Eulerian semi-derivative (Gateaux directional derivative) is defined as:

$$\mathring{J}(K)(\varphi) = \lim_{t \to 0} \frac{J(\vartheta_\varphi(t, K)) - J(K)}{t} \tag{2.21}$$

where, $\vartheta_\varphi(t, K)$ is the t reachable tube of K under the mutation φ.

From [60] and [63], the directional derivative of the shape function having the form of

$$J = \int_K f(q)\, dq \tag{2.22}$$

can be written as:

$$\mathring{J}(K)(\varphi) = \int_K \mathrm{div}(f(q)\varphi(q))\, dq \tag{2.23}$$

2.6 Chapter Summary

This chapter presents a discussion on various dynamic systems and introduces mutational equations as a tool to model the dynamics of shapes. It also presented the hybrid automata model used to model a combination of discrete and continuous dynamic systems. Hybrid automata models use differential equations to model the continuous system dynamics. This chapter further extends the hybrid automata model to use mutational equations to model the continuous dynamics. The Mutational Hybrid Automata (MHA) model proposed allows us to model large scale networked pervasive surveillance systems. This chapter also introduces shape analysis used to measure the acceptability of shapes using shape functions.

C

M

N

Thi

lan

tra

nific

the

mul

(MI

of m

3.1

In a

whi

of a

one

jum

Chapter 3

Modeling Mobile Surveillance Networks

This chapter presents the modeling of a tracking task using a multiple camera surveillance network. which is a mutational analysis based technique christened "Hausdorff tracking" (in honor of the German mathematician Felix Hausdorff who made significant contributions to set theory, descriptive set theory, measure theory, function theory, and functional analysis), is introduced in order to track multiple targets simultaneously using a single camera sensor. Further the Mutational Hybrid Automata (MHA) model is introduced for tracking a target as it moves across the viewing region of multiple cameras.

3.1 Introduction - Example Surveillance Scenario

In a multiple camera surveillance scenario with active (moving) cameras, the target, which moves in a continuous fashion, can be tracked using either continuous motion of a camera sensor or by discontinuous jumps of the "current" tracking sensor from one camera to the next or a combination of continuous motion and a discontinuous jump. Consider the example surveillance scenario form Figure 3.1(a) which depicts the

T

in F

tha

the

its a

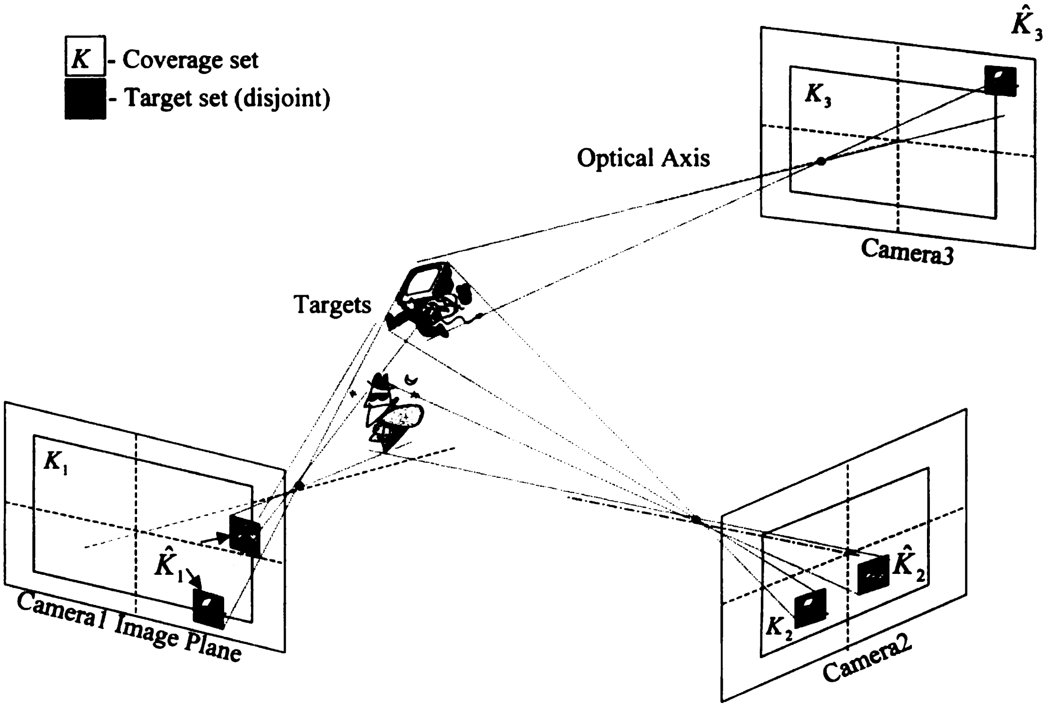
be t

cont

task

T

sor a



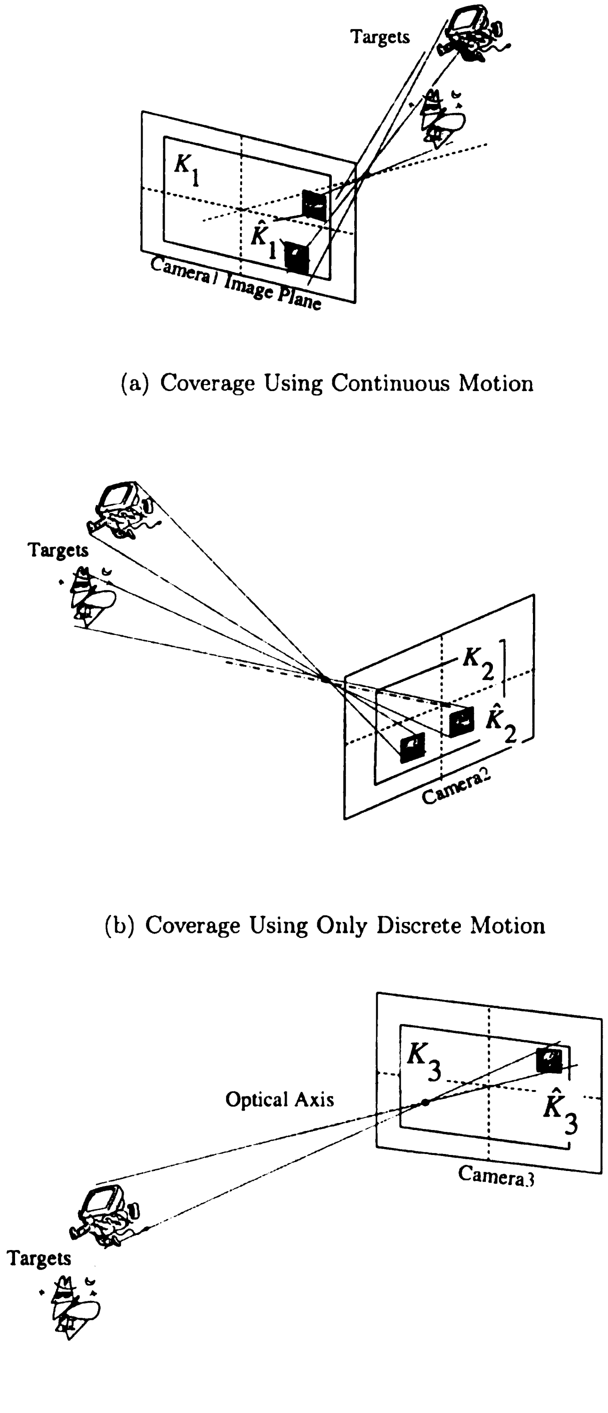
(a) Three-Camera Pervasive Surveillance Scenario

Figure 3.1: 3 Camera Pervasive Surveillance Scenario

the image planes of three camera sensors and a couple targets.

The targets can be tracked using the continuous motion of Camera 1 as is shown in Figure 3.2(a). We see that Camera 1 is moved in a continuous fashion to ensure that the targets are within the active field of view of the sensor. In Figure 3.2(b), the tracking task is transferred to Camera 2 which can already view the targets in its active field of view. Finally, as shown in Figure 3.2(c), the tracking task can be transferred to camera 3 which will first need to acquire the targets by executing continuous motion and then a discrete switch will be executed to transfer the tracking task to camera 3.

This chapter first presents the modeling of continuous motion of the camera sensor and target (continuous dynamics) using mutational equations. These dynamics



(a) Coverage Using Continuous Motion

(b) Coverage Using Only Discrete Motion

(c) Coverage Using Hybrid Motion

Figure 3.2: Modeling Pervasive Surveillance Networks

e

e

a

v

d

c

e

n

m

eq

models can be used for developing a servo mechanism to control the continuous camera motion. The continuous input to the camera for motion can be controlled using an error feedback from either the image space analysis or the world space analysis. Image space analysis implies that the error feedback for moving the camera is derived from analysis of the target location in the image plane of the camera, while world space analysis implies that the error input for moving the camera is derived from the world space coordinates of the target (acquired from other sensors that can observe the target) and the current world space location of the camera coverage set. Hence, we develop models for image space servoing and world space servoing separately.

The mutational equation model for the target motion in the image space is derived using the perspective projection model and the image jacobian [64]. The motion of the camera coverage set in the world space model is derived separately for 2-D and 3-D scenarios. In the 2-D scenario, the camera coverage set is taken to be a pie cut out from a circle, while in the 3-D scenario, the camera coverage is modeled as a 3-D pie (cone bounded by a sphere) cut out from a sphere. The motion of these coverage sets due to the motion of the camera is modeled as a controlled mutational equation. Further, we relax the implicit assumption of holonomic motion constraints and extend the continuous motion modeling using mutational equations for a sensor with non-holonomic constraints.

The Hausdorff tracking method for image space and world space models is introduced. This method uses shape functions to derive feedback errors which are used to control the motion input to the camera. Image space Hausdorff tracking and cooperative Hausdorff tracking are presented as two examples of the Hausdorff tracking method in the image space coordinates and world space coordinates, respectively.

This chapter also introduces the Mutational Hybrid Automata (MHA) model for modeling the hybrid camera motion scenario shown in Figure 3.2(c). Using mutational equations, the mutational hybrid automata model can describe the continuous motion of the cameras and target and the discrete switching behaviors when a target leaves or enters the field of view of a camera.

Using Hausdorff tracking method developed in this section, there are numerous choices of the input applied to the camera which can accomplish the task. This is because the robot system has a certain amount of redundancy built in for the surveillance task. An example of this redundancy in task execution can be evinced by noting that in order to bring a target, which is currently located at one of the sides of the camera FOV, into the cameras FOV, the camera can execute an angular motion by panning or a lateral motion by moving behind. Both these motions individually or a combination of them can be used to accomplish the surveillance task. Hence we need to select an appropriate input to the camera for task accomplishment.

We can select various other tasks that the camera can perform in order to utilize the redundancy offered by the camera motion. For example the energy consumed due to the motion of the sensor should also be minimized while maintaining a safe distance from sensed obstacles (in the case of sensors mounted on mobile robots). These combined tasks can be expressed as a multi-objective optimization problem which tries to minimize the resolution inadequacies and energy consumption subject to the constraint of maintaining the visibility of the target.

Various other tasks can also be considered for the multi-objective optimization problem and the complete problem can be represented as an optimal control framework for Hausdorff tracking. Once the task criteria are expressed in a multi-objective constrained optimization framework, various methods such weighted sum method ,physical programming, non-linear programming, search space methods, etc. can be used to find the solution to the optimization problem.

w

n

ai

3.

Th

usi

flo

inp

ima

to t

wher

A

spect

3.2 Controlled Mutational Equation Model for Camera Sensor and Target

A controlled mutational equation can be written as:

$$\overset{\circ}{K}(t) \ni \varphi(t, K(t), u(t)), \ \forall t \geq 0 \tag{3.1}$$

$$u(t) \in U, \tag{3.2}$$

where, $\varphi : \mathbb{R}^+ \times \mathcal{K}(E) \times U \mapsto Lip(E, \mathbb{R}^n)$ is a continuous map associating a Lipschitz map with (t, K, u) and $u(t) \in U$ is the control input.

Using a controlled mutational equation, we can model the motion of the target and coverage sets due to the motion input u to the camera/robot.

3.2.1 Perspective Projection Model

The deformation of the target set w.r.t. the motion of the camera can be represented using a mutational equation. The mutational equation can be derived using the optic flow modeling as shown in [41]. Let u be a velocity screw which represents the velocity input to the camera and let q be a vector of image space feature parameters. The image jacobian J_v is a linear transformation that relates the camera velocity screw u to the tangent space $\dot{q}$ of the image space feature parameter vector q as:

$$\dot{q} = J_v(c)u \tag{3.3}$$

where, c represents the camera pose such that $\dot{c} = u$.

Assuming that the projective geometry of the camera is modeled by the perspective projection model, a point $P = [x, y, z]^T$, whose coordinates are expressed w

co

m

m

by

whe

spa

$u =$

the

T

3.2.

We

in th

with respect to the camera coordinate frame, will project onto the image plane with coordinates $q = [q_x, q_y]^T$ as:

$$\begin{bmatrix} q_x \\ q_y \end{bmatrix} = \frac{\lambda}{z} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.4}$$

where λ is the focal length of the camera lens [41]. Using the perspective projection model of the camera, the velocity of a point in the image frame with respective to the motion of the camera frame [41] can be expressed. This is called the image jacobian by [41] and is expressed as:

$$\begin{bmatrix} \dot{q}_x \\ \dot{q}_y \end{bmatrix} = \varphi_c(q) = B_c(q) \begin{bmatrix} u \\ \dot{\lambda} \end{bmatrix} = B_c(q) u_c \tag{3.5}$$

$$B_c(q) = \begin{bmatrix} -\frac{\lambda}{z} & 0 & \frac{q_x}{z} & \frac{q_x q_y}{\lambda} & -\frac{(\lambda^2 + q_x^2)}{\lambda} & q_y & \frac{q_x}{\lambda} \\ 0 & -\frac{\lambda}{z} & \frac{q_y}{z} & \frac{\lambda^2 + q_y^2}{\lambda} & -\frac{q_x q_y}{\lambda} & -q_x & \frac{q_y}{\lambda} \end{bmatrix}$$

where, $q = (q_x, q_y)^T$ are the image space coordinates of the point P whose task space coordinates are $(x, y, z)^T$. λ is the focal length of the lens being used and $u = [v_x,\ v_y,\ v_z,\ \omega_x,\ \omega_y,\ \omega_z]^T$ is the velocity screw of the camera motion and $\dot{\lambda}$ is the rate of change of the focal length.

The mutational equation of the target set can now be written as:

$$\begin{aligned} \dot{q} = \varphi(q) &= \varphi_c(q) + \varphi_t(q) \\ \overset{\circ}{\hat{K}} &\ni \varphi(\hat{K}) \end{aligned} \tag{3.6}$$

3.2.2 World Space Model

We can derive the mutational equation for the sensor coverage set and target set in the world space. Here we consider two cases for a 2-D model and 3-D model se

di

2

Co

wh

poi

for

whe

are t

moti

where

muta

separately and present the mutational model for the 2-D and 3-D cases based on the dimensionality and geometry.

2 Dimensional World Space Model

Consider a point $q \in \mathbb{R}^2$ defined as:

$$\begin{bmatrix} q_x \\ q_z \end{bmatrix} = \begin{bmatrix} x + r\cos(\rho) \\ z + r\sin(\rho) \end{bmatrix} \tag{3.7}$$

where $[x, z, \theta]^T$ are the coordinates of the robot and r, ρ are the polar coordinates of point q with respect to the robot coordinates. Using equation 4.15, the coverage set for the focusing constraint can be written as:

$$K = \{q \mid r \in (D_{min}, D_{max}), \rho \in (\alpha_{min}, \alpha_{max})\} \tag{3.8}$$

where D_{min}, D_{max} are the minimum and maximum focus distances and $\alpha_{min}, \alpha_{max}$ are the minimum and maximum angles of the view lines as shown in Figure 3.5. The motion of each point in the coverage set can be written as:

$$\begin{bmatrix} \dot{q}_x \\ \dot{q}_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & -r\sin(\rho) \\ 0 & 1 & r\cos(\rho) \end{bmatrix} u$$

$$\dot{q} = \varphi(q) = B\ u \tag{3.9}$$

where $u = [\dot{x}, \dot{z}, \dot{\theta}]^T$ is the velocity input to the camera. Using equation 4.17, the mutational equations for the motion of the coverage set can be derived.

3 Dimensional World Space Model

The coverage set can be specified as the set of points which satisfies the viewing constraints of the sensor and can be defined as the active FOV of the camera after incorporating the depth of field constraints such as the maximum and minimum focus distances. Consider a point $q \in \mathbb{R}^3$ in the sensor coordinate frame defined as:

$$\begin{bmatrix} q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} r\cos(\theta)\sin(\phi) \\ r\cos(\phi) \\ r\sin(\theta)\sin(\phi) \end{bmatrix} \tag{3.10}$$

where (r, θ, ϕ) are the polar coordinates of the point q in the sensor frame. Using the above equation, the coverage set can be defined as :

$$\begin{aligned} K = \{q \mid r \in (D_{min}, D_{max}), \theta \in (\alpha_{min}, \alpha_{max}), \\ \phi \in (\beta_{max}, \beta_{min})\} \end{aligned} \tag{3.11}$$

where, (D_{min}, D_{max}) represent the depth of field and $(\alpha_{min}, \alpha_{max}), (\beta_{max}, \beta_{min})$ are the view angles for the sensor. The global coordinates of the target set $\hat{K}$ can be transformed to the local sensor coordinates using the localization information available.

The mutational equation for the motion of the coverage set can be written as:

$$\begin{aligned} \begin{bmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{q}_z \end{bmatrix} &= r \begin{bmatrix} \cos(\theta)\cos(\phi) & -\sin(\theta)\sin(\phi) \\ \sin(\phi) & 0 \\ \sin(\theta)\cos(\phi) & \cos(\theta)\sin(\phi) \end{bmatrix} u \\ \dot{q} &= \varphi(q) = B\, u \end{aligned} \tag{3.12}$$

where $u = [\omega_x, \omega_y]^T$ is the velocity input to the camera. The objective is to cover the

I

(1

of

a

th

an

(3.

2.

said

a ca

in th

set of targets $\hat{K}$ with the coverage set K such that $K \subset \hat{K}$.

3.3 Incorporating Non-holonomic Sensor Constraints

In section 3.2 and 3.2.2 the mutational model relating the motion of the sensor u to the motion of the target and coverage sets was introduced. However, the implicit assumption made was that the individual components of the vector u are independent of each other i.e., the system has a holonomic structure. Consider the example of a car moving in a horizontal plane. Select the reference point to be the midpoint of the two rear wheels and the reference direction to be the main axis of the car moving forward. This car's configuration can be defined as a vector (state) $(x, y, \theta) \in \mathbb{R}^3$. However, these parameters and their derivatives are related by the following relation:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta \tag{3.13}$$

where $\dot{x}$ and $\dot{y}$ are the derivatives of x and y respectively. At any configuration (x, y, θ) along a trajectory parameterized by time, the triplet $(\dot{x},, \dot{y}, \dot{\theta})$ is the velocity of the robot (car). The set of all vectors $(\dot{x},, \dot{y}, \dot{\theta})$ passing through the point (x, y, θ) is a vector space. If the robot was not constrained by the constraint in equation (3.13), this vector space would be the tangent space of the configuration space at (x, y, θ) and would have a dimension 3. However, due to the kinematic constraint in equation (3.13), the velocity vectors only span a subspace of the tangent space of dimension 2. This is called a nonholonomic constraint and the system dynamics of the car are said to be nonholonomic.

Various sensor motion platforms have a nonholonomic structure with examples like a car which can be steered left and right only when the car has a velocity component in the forward direction. Hence it is required to model such nonholonomic platforms for an accurate description of system dynamics. Another example of a nonholonomic platform would be a sensor mounted on a unicycle-type robotic motion module with a differential drive, which has been quite popular recently since they require only two motors for forward motion and steering.

Further, due to the large amount of energy consumed by the motion module, multiple sensors with varying sensing modalities may be mounted on a single motion module. An example is the mobile manipulator depicted in Figure 6.3, which has a PTZ camera mounted on the mobile base and a bullet camera mounted on the robot arm.

This section presents an example of a unicycle-type robot with nonholomic motion constraints which can be modeled as:

$$\begin{aligned} \dot{x} &= v\cos\theta_i \\ \dot{y} &= v\sin\theta_i \\ \dot{\theta} &= \omega \end{aligned} \tag{3.14}$$

where, v and ω are the linear and angular velocity control inputs of the robot. The relationship between u and $u_h = (v,\omega)^T$ can be written as:

$$u = B_h u_h = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.15}$$

Thus, the controlled mutational equation for camera coverage model in equation 3.6 can be written as:

$$\varphi(q) = B(q)u = B(q)B_h u_h = B_o u_h \tag{3.16}$$

3.4 Hausdorff Tracking

The task of a surveillance camera is to continuously keep a moving target or multiple moving targets in its active field of view. The surveillance task of maintaining multiple targets in the active field of view (FOV) of the sensor with an adequate resolution can be readily expressed in a set-based framework where the variables taken into consideration are not vectors of parameters but shapes (domains) themselves. However, due to lack of vectorial structure on the space, classical differential calculus cannot be used to describe the evolution of such domains. However, mutational equations [60] can be used to describe the dynamics (change in shape) of the sensor FOV and target domains and further be used to calculate feedback mechanisms to accomplish the surveillance task.

The surveillance task can be expressed, using shape functions [62], as the minimization of a Hausdorff distance-based metric or the size of the target, etc. The shape function essentially represents the error between the desired and actual shapes and reducing it to zero will accomplish the task. This section presents the method of Hausdorff tracking using mutational equations for performing the surveillance task.

Shape or a geometric domain can be defined as the set $K \in \mathcal{K}(E)$, $E \subset \mathbb{R}^n$ where $\mathcal{K}(E)$ represents the space of all nonempty, compact subsets of E. The target and the camera coverage can be readily expressed as shapes. Mutational equations can then be used to express the change (deformation) in the coverage and target sets based on the motion of the sensor. Shape analysis [62] can be used to address problems involving geometric domains or shapes. Shape functions, which are set-defined maps from $J(K) : \mathcal{K}(E) \mapsto \mathbb{R}$, can be used to provide a "measure" of acceptability and optimality of the shape K. For example we can use a shape function to see if a reference set $\hat{K}$ is contained within a current set K. In order to accomplish the task defined using shape functions, we need to derive a feedback map $\mathcal{U} : \mathcal{K}(E) \mapsto U$, where $u = \mathcal{U}(K(t))$ is the input to the sensor, which will reduce the shape function

c

(

t

t

t

re

in

us

sp

de

dy

sh

ca

$\int_K$

is

is

po

co

is t

3.2

to zero. The convergence of the shape function can be analyzed using the shape Lyapunov theorem [65]. The convergence to zero of the task function would imply task accomplishment.

The task requirements for Hausdorff tracking can be expressed as a positive semi-definite shape function which provides a measure of acceptability of a particular shape (set). For the example shown in Figure 3.2(a), the measure of acceptability of the target image (sets) is whether they are within a certain region of the image space and the measure can be expressed as a shape function $J(\hat{K}) \in \mathcal{R}_{+}$. The task of Hausdorff tracking can be defined as finding an input in order to move the camera in order to reduce the shape function to zero. Hence we need to express the evolution (change) in the shape of the target w.r.t. the movement of the camera. This is accomplished using mutational equations, which are the analog of differential equations for shape spaces. Using the mutational equation, we should be able to calculate the directional derivative of the shape function w.r.t. the motion of the camera, and using an inverse dynamics procedure, calculate a motion input to the camera in order to reduce the shape function (error) to zero.

For the example in Figure 3.1(a), the task requirements for Hausdorff tracking can be mathematically expressed using a positive definite shape function $J(\hat{K}) = \int_{\hat{K}} d_K^2(p)\, dp$ where K is the coverage set for the camera and $d_K(p) = \inf_{q \in K} \|q - p\|$, is the point to set distance from point p to set K. Note that the shape function is zero only when set $\hat{K}$ is completely covered by set K, otherwise it is a non-zero positive value. Initially in Figure 3.1(a) the target set $\hat{K}$ is not covered by the camera coverage set K and hence the shape function $J(\hat{K} > 0$. The Hausdorff tracking task is to reduce the shape function to zero by moving the camera as shown in Figure 3.2(a) i.e., minimize the shape function.

b

i

te

th

pr

fra

me

sha

ba

the

Ta

Figu

on t

Gene

defin

the n

Shap

The s

can be

3.4.1 Image-Based Hausdorff Tracking

When the target is located in the active FOV of the sensor, the method of image-based Hausdorff tracking can be used to accomplish the surveillance task. The target is detected using image processing algorithms and represented as a collection of pixels (blob) on the image plane. The sensor coverage set is represented as a rectangle centered at the image center and one of the tasks is to maintain the target blob within the sensor coverage set. Surveillance tasks such as set coverage and maintaining appropriate image resolution can be readily expressed in a setbased mutational analysis framework using a shape function as the minimization of a Hausdorff distance-based metric or the size of the target, etc. The task can be accomplished by reducing the shape function to zero. The task of visual servoing can also be expressed in a set-based framework [66]. The shape function essentially represents the error between the desired and actual shapes.

Target and Coverage Sets for Image-Based Hausdorff Tracking

Figure 3.3 depicts the coverage and target sets. The coverage set K is the set of points on the image plane (X, Y), within which we require the targets to be maintained. Generally it is taken as a rectangle centered in the image plane. The target $\hat{K}$ is defined as the union of the individual image plane blobs (sets of pixels) formed by the multiple (n) targets as: $\hat{K} = \bigcup_{i=1}^{n} \hat{K}_i$.

Shape Functions for Image-Based Hausdorff Tracking

The surveillance task of maintaining visibility of the target(s) with a certain resolution can be represented using a shape function as:

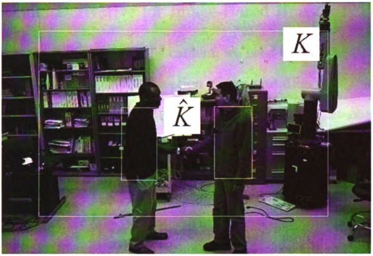


Figure 3.3: Target and coverage set for image-based Hausdorff tracking.

$$J(\hat{K}) = J_{FOV}(\hat{K}) + J_{Amin}(\hat{K}) + J_{Amax}(\hat{K}) \tag{3.17}$$
$$J_{FOV}(\hat{K}) = \int_{\hat{K}} d_K^2(p)\ dq$$
$$J_{Amin}(\hat{K}) = \max(\int_{\hat{K}}\ dq - AREA\_MIN,\ 0)$$
$$J_{Amax}(\hat{K}) = \min(AREA\_MAX - \int_{\hat{K}}\ dq,\ 0)$$

where q is a point on the image set $\hat{K}$ and $AREA\_MAX$ and $AREA\_MIN$ denote the maximum and minimum admissible areas of the target set K for maintaining adequate resolution.

Note that the shape function $J(\hat{K})$ is zero only when set $\hat{K}$ is completely covered by set K and when the area of set $\hat{K}$ is within the limits $(AREA\_MIN, AREA\_MAX)$. Otherwise it is a non-zero positive value.

Shape Directional Derivative for Image-Based Hausdorff Tracking

In order to derive the feedback u to the camera, we need to find the shape directional derivative $\mathring{J}(\hat{K})(\varphi)$ of $J(\hat{K})$ in the direction of the mutation $\varphi(\hat{K})$ using the shape function and the mutational equation. Assuming a relatively flat object, i.e., the z coordinate of all the points on the target are approximately the same, we can derive an expression for $\mathring{J}(\hat{K})(\varphi)$ by substituting equations 6.1 and 3.6 into 2.23 as:

$$\mathring{J}(\hat{K})(\varphi) = \int_{\hat{K}} \nabla f(q)\varphi(q) + f(q)\mathrm{div}\varphi(q)\; dq \tag{3.18}$$

where, $f(q)$ can be construed as the aggregated integrand from equation 6.1. For the shape function of the type

$$J(\hat{K}) = \int_{\hat{K}} d_K^2(q)\; dq \tag{3.19}$$

Substituting Equations (3.19) and (3.6) into Equation (3.21) we get

$$\begin{aligned} \mathring{J}(\hat{K})(\varphi(\hat{K},u)) &= \int_{\hat{K}} (2(q - \Pi_K(q)) \cdot B(q)u \\ &\quad + d_K^2(q)\mathrm{div}(B(q)u))\; dq \\ &= \int_{\hat{K}} (2(q - \Pi_K(q)) \cdot B(q) \\ &\quad + d_K^2(q)\mathrm{div}B(q))\; dq \cdot u \end{aligned} \tag{3.20}$$

The above equation can be written as:

$$\mathring{J}(\hat{K})(\varphi) = 2\int_{\hat{K}} \inf_{q \in \Pi_K(p)} \langle (q-p), \varphi(p) \rangle\; dp + 2\int_{\hat{K}} \mathrm{div}(\varphi(p))\; dp \tag{3.21}$$

Here it must be noted that for the existence of the shape directional derivative, the integrand $\alpha \in C^1(\mathbb{R}^n, \mathbb{R})$. J_{FOV} in Equation (6.1) is continuous. However, the elements J_{Amin} and J_{Amax} in Equation (6.1) do not satisfy this criteria. Hence we modify the shape function in order to satisfy the continuity criteria using smooth approximations of the min and max functions as:

$$J_{Amin} = \frac{\sqrt{(\int_{\hat{K}} dp - A_{min})^2 + \epsilon^2} + (\int_{\hat{K}} dp - A_{min})}{2} \tag{3.22}$$

$$J_{Amax} = -\frac{\sqrt{(A_{max} - \int_{\hat{K}} dp)^2 + \epsilon^2} - (A_{max} - \int_{\hat{K}} dp)}{2} \tag{3.23}$$

where, ϵ is a very small positive number. This approximation physically manifests as adding a very small area (of size ϵ) to the target set size and hence this approximation will hardly affect the validity of the shape function to represent the task of maintaining adequate resolution.

Feedback Map u for Image-Based Hausdorff Tracking

The problem now is to find a feedback map u_c such that the shape function J is reduced to zero. Equation 3.21 can be approximated as:

$$\mathring{J}(\hat{K})(\varphi) \leqslant \frac{1}{z_t} C_1(K)u_1 + C_2(K)u_2 + C_3(K)u_c \tag{3.24}$$

where,$u_c = [u_1^T u_2^T]^T$, $u_1 = [v_x, v_y, v_z, \dot{\lambda}]^T$ and $u_2 = [\omega_x, \omega_y, \omega_z]^T$. z_t is an estimated minimum bound on the target z position. The relation $z_t > z$ will guarantee the inequality in 3.24.

The matrices $C_1(K), C_2(K)$ and $C_3(K)$ can be defined as:

$$C_1(K) = 2\int_{\hat{K}} \inf_{q \in \Pi_K(p)} B_1(q)^T (q - p) dp, \tag{3.25}$$

$$C_2(K) = 2 \int_{\hat{K}} \inf_{q \in \Pi_K(p)} B_2(q)^T (q - p) dp, \tag{3.26}$$

$$C_3(K) = 2 \int_{\hat{K}} B(q) dp. \tag{3.27}$$

where, the matrix $B(q)$ is defined as: $B(q) = [B_1(q), B_2(q)]$ and its elements are defined as:

$$B_1(q) = \frac{1}{z_t} \begin{bmatrix} -\lambda & 0 & q_x \\ 0 & -\lambda & q_y \end{bmatrix} \tag{3.28}$$

$$B_2(q) = \begin{bmatrix} \frac{q_x q_y}{\lambda} & -\frac{(\lambda^2 + q_x^2)}{\lambda} & q_y & \frac{q_x}{\lambda} \\ \frac{\lambda^2 + q_y^2}{\lambda} & -\frac{q_x q_y}{\lambda} & -q_x & \frac{q_y}{\lambda} \end{bmatrix} \tag{3.29}$$

After combining the relevant terms from equation 3.24, it can now be rewritten as:

$$\mathring{J}(\hat{K})(\varphi) \leqslant C(K) u_c \tag{3.30}$$

A controller can now be proposed based on equation 3.30 using the notion of a generalized pseudoinverse of a matrix. The pseudoinverse $C^{\#}(K)$ of the matrix $C(K)$ must satisfy:

$$u_c = \alpha C^{\#}(K) J(\hat{K}) \tag{3.31}$$

where α is a gain value. It should be noted that the estimate z_t of the target distance only affects the gain of the control and not its validity. Further it is important to note that the gain distribution between the various redundant control channels depends on the selection of the null space vector when calculating the generalized

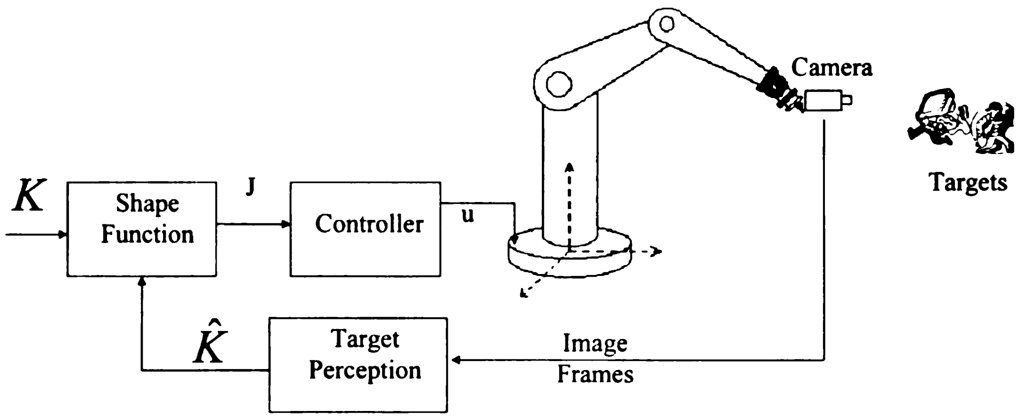


Figure 3.4: Block diagram for image-based Hausdorff tracking.

pseudoinverse $C^{\#}(K)$ of matrix $C(K)$.

Figure 3.4 depicts the block diagram representation of the Hausdorff tracking controller. Once the targets to be tracked are identified by the target perception module, the Hausdorff tracking method can be used to track the multiple identified targets simultaneously irrespective of the relative motion of the targets with respect to each other.

3.4.2 Cooperative Hausdorff Tracking

Cooperative Hausdorff tracking is used when a sensor node cannot sense the target in its current configuration and needs to move in order to bring the target into its active sensing region. The individual sensor nodes maintain information regarding the observations of their neighboring nodes and broadcast (within their locality) their own observations. Based on the combined observations, each node develops a list of targets being actively tracked and the status of its peer nodes. Once the target is detected in the active FOV, the sensor switches to image-based Hausdorff tracking to perform the surveillance task.

For establishing cooperation of multiple sensors, which may not be able to observe the same object simultaneously, there is a need transform the image blobs and measurements into a common coordinate system referenced by all the sensors and task the sensors according to the movement of the targets expressed in the common frame of reference. Further, this common representation will be helpful in combining the object detection hypothesis from multiple camera sensors for sensor fusion. It can also be used for designing efficient routing mechanisms and algorithms for communication.

Target and Coverage Sets for Cooperative Hausdorff Tracking

The coverage set can be construed as the set of points that satisfy the viewing constraints of the camera. It can be defined as the active field of view of the camera after incorporating the depth of field constraints such as the maximum and minimum focus distances. Section 3.2.2 provides the coverage set model for the 2D and 3D cases considered and Figures 3.5 and 3.6 depict the 2D and 3D coverage sets, respectively.

For the world space model, the target is a set of points in 2D or 3D based on the scenario. For ease of representation, the target set is represented as a bounding box of the target. This simplification tends to simplify the structures and computation required for tracking the target. Also, it should be noted that if the bounding box of the target is covered, the target is also covered.

Shape Function for Cooperative Hausdorff Tracking

The objective is to cover the set of targets $\hat{K}$ with the coverage set K such that $K \subset \hat{K}$. This condition can be written using a shape function as:

$$J(K) = \int_{\hat{K}} d_K^2(p)\, dp \tag{3.32}$$

Note that when the target set is completely covered by the camera coverage set, the value of the shape function is zero. Otherwise the shape function takes a positive non-zero value. Hence the coverage task can be attained by reducing the value of the

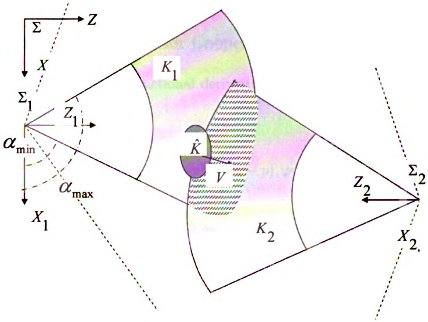


Figure 3.5: Target and coverage set 2D for cooperative Hausdorff tracking.

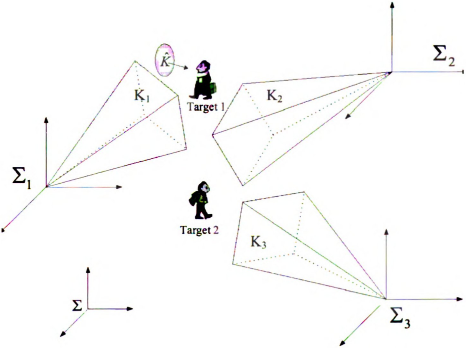


Figure 3.6: Target and coverage set for 3D cooperative Hausdorff tracking.

shape function in equation (3.32) to zero.

Shape Directional Derivative for Cooperative Hausdorff Tracking

Using the equation 3.21, the directional derivative of $J(K)$ represented by $\mathring{J}(K)(\varphi)$ can be written as:

$$\mathring{J}(K)(\varphi) = 2 \int_{\hat{K}} \inf_{q \in \Pi_K(p)} \langle (q-p), \varphi(q) \rangle dp \tag{3.33}$$

The above equation can be rewritten as:

$$\mathring{J}(K)(\varphi) = 2\langle \int_{\hat{K}} \inf_{q \in \Pi_K(p)} B(q)^T (q-p) dp, u \rangle \tag{3.34}$$

where u is the input to the camera and $B(q)$ is defined in equation 4.17.

Feedback Map u for Cooperative Hausdorff Tracking

In order to find a u, let us denote:

$$C(K) = \int_{\hat{K}} \inf_{q \in \Pi_K(p)} B(q)^T (q-p) dp. \tag{3.35}$$

Let $C(K)^{\#}$ be the pseudo inverse of $C(K)$.

A choice of u can be taken as:

$$u = -\frac{1}{2} \alpha C(K)^{\#} J(K), \tag{3.36}$$

where α is the gain of the control. The gain distribution of the task among the various control channels of u is controlled by assigning the null space vector of the matrix $C(K)$.

3.5 Mutational Hybrid Automata Model for Pervasive Surveillance Networks

Consider a network of N cameras which comprise the pervasive surveillance system. With each camera cam_n we associate a coverage set $K_n \in \mathcal{K}(W_n)$ which signifies the region cam_n can view actively and W_n represents the total capable FOV of cam_n. A target to be tracked is represented by a target set $\hat{K} \subset \mathcal{K}(W), W = \bigcup_{n \in N} W_n$. An example of the graphical MHA model for two cameras tracking a target is shown in Figure 4.2

Groups of cameras are sequentially involved in tracking a target as it traverses the monitored region. Within each group, the behavior of each camera is governed by target visibility from a particular camera. This combination of continuous tracking and switching behavior can be modeled using the Mutational Hybrid Automata (MHA) model presented in section 2.4. This section discusses the various parts of the MHA model for the pervasive surveillance network H_{PSN} and further elaborates on the design of stable controllers for a pervasive surveillance system modeled using the MHA model.

3.5.1 Mutational Hybrid Automata Model

It is relatively straight forward to write the MHA model for a pervasive surveillance system as:

$$H_{PSN} = (Q, M, \Sigma, \text{Init}, D, E, G, R) \tag{3.37}$$

where Q represents the discrete variables, M represents the continuous variables, $\Sigma = \{\Sigma_q\}_{q \in Q}$ is a collection of the continuous mutational dynamics, $\text{Init} \subseteq \mathcal{Q} \times \mathcal{M}$ is the set of initial values of the variables in Q and M. D is a domain which associates a domain of operation of the continuous states for every discrete state $q \in Q$, E is an edge set along which discrete transition takes place, G is a set of guard conditions which enable the discrete transitions and R is a reset map which resets the continuous states when a discrete transition is enabled.

Discrete Variables

In the above multi-camera surveillance scenario the target can be tracked by a single camera or a group of cameras at any given time. Further, if the target is in the FOV of a particular camera, it can visually track the target, *i.e.*, perform image-based tracking, while if the target is not visible, the camera can cooperatively track the target based on the location of another camera which can view the target. Hence one discrete (binary) variable b_n is associated with each camera to indicate whether the target is visible to that camera, such that $b_n = 1$ if target is visible in cam$_n$ and $b_n = 0$ otherwise. Another discrete variable c_n indicates whether a camera (cam$_n$) is involved in the particular tracking scenario. Thus the discrete variables collection can be defined as: $Q = \{\{b_n, c_n\}_{n \in N}, \}$.

Edge Set

The edge set E defines which particular transitions are possible. Here we describe general rules to develop E. A handoff between cameras can be initiated when the target is visible to both cameras. An example edge set for a two-camera scenario is developed in section 4.3.

Domains

The domains of operation D_q for each mode can be developed based on location of the target set $\hat{K}$, which will decide which cameras are capable of viewing the target, and based on the continuity of the target motion, will decide which cameras can acquire the target as it moves through the monitored region W.

Continuous Variables

The collection of continuous variables M consists of the individual camera coverage sets K_n and the target set $\hat{K}$; *i.e.,* $M = \{\{K_n\}_{n \in N}, \hat{K}\}$. Let $K \in \mathcal{M}$ represent an element of the valuations of M.

Assumption: We can safely make the assumption that the target set $\hat{K}$ is convex. This assumption does not have significant effect on the model as all non-convex targets can be modeled as their convex cover. This assumption will be invoked to aid the proof of stability later in this section. We further make the assumption that the coverage sets K_n are also convex. This assumption can be substantiated by the fact that the active fields of view of most cameras are convex.

Continuous Dynamics

The continuous dynamics represent the change (deformation) of the sets in M which can be modeled using mutational equations as: $\mathring{K}_n \ni \varphi_n(K_n, u_n)$ where $u_n \in \mathcal{U}$ is the control input to cam$_n$. Based on the mode of operation of cam$_n$, the controller model can be changed, giving rise to different continuous dynamics for each camera in different modes of operation q. The motion of the target can also be represented as a mutational equation as: $\mathring{\hat{K}} \ni \varphi_t(\hat{K})$ which can be estimated from the observations of the target.

Guard Condition

A guard condition or switching condition G associated with each edge $e \in E$ is designed based on the location of the target set. Various events can be recognized and processed by the multiple camera surveillance network such as the visibility of a target in a particular target or the global location of the target or camera coverage set reaching some predefined value.

For example, when the target moves into the visible region of cam$_n$ then the b_n associated with it changes, which effects the transition along an edge e to a mode where $b_n = 1$.

Further, based on the current location of the target (sensed and communicated from other cameras), the variable c_n associated with camera cam$_n$ will change in order to involve the camera in the tracking task.

Reset Map

Since we are dealing with continuous positions of the targets and camera coverage sets, there are no jumps affected in the continuous variables. Hence the reset map R can be defined as $R : \{(p, q), K\} \mapsto K$, $K \in \mathcal{M}$. However, in case of multiple groups of cameras being involved in the surveillance scenario, the coverage sets, which are defined as the union of the individual coverage sets, will jump from one group to another based on the change in the discrete variable c. For example, when switching the camera group from state q to p, the reset map is given as: $R(p, q, K_i) \mapsto K_j$ where $K_i = \cup_{l \in p} K_l$, $K_j = \cup_{l \in q} K_l$.

Init Set

The target can be recognized only if it is in the view of at least one camera. Hence the set of initial conditions is restricted to the set of discrete modes $Init : \{q, M :$

$\wedge_{n \in N} b_n = 1\}$ where b_n is the target visibility condition for cam_n.

3.5.2 Controller Design for Individual Cameras

Events can be recognized by the controller which can be used to generate the control inputs provided to the various cameras and trigger mode switches. Such events include the recognition of the target by a particular camera and the location of the coverage set of cam_n hitting certain boundaries within its domain of operation W_n. Also, within each mode q we are provided with the number of cameras actively involved in the surveillance c and the cameras which can view the target b_n.

Camera Active, Target Visible($c_n = 1, b_n = 1$)

When the target is visible in the image plane of cam_n ($b_n = 1$), the method of image-based Hausdorff tracking [67] can be used to track the target. In this controller, the control input to the controller is derived based on the shape function ${}^iJ_n({}^{in}K_n, {}^{in}\hat{K}) : \mathcal{K}(W_n) \times \mathcal{K}(W) \mapsto \mathbb{R}_+$ where the superscript ^{in} represents the projection of a set on the image plane ${}^iW_n$ of cam_n. The control input u_n can be calculated as is detailed in [68].

Camera Active, Target Not Visible ($c_n = 1, b_n = 0$)

When the target is not visible to cam_n but is visible to cam_m ($b_m = 1$), we can use the method of cooperative Hausdorff tracking proposed in [67]. If the target $\hat{K} \not\subset W_n$ then cam_n should track the projection $\Pi_{W_n}(\hat{K})$ of the target on the domain W_n. This strategy will ensure that as soon as the target moves into the region W_n it will be encompassed in the visible region of cam_n.

However, in a real tracking scenario the projection $\Pi_{W_n}(\hat{K})$ may not be easily available. But taking into account the fact that $b_m = 1$, which implies that $\hat{K} \subset K_m$, we can deduce that $\Pi_{W_n}(\hat{K}) \subset \Pi_{W_n}(K_m)$. Thus, tracking $\Pi_{W_n}(K_m)$ will also ensure tracking the set $\Pi_{W_n}(\hat{K})$ (proof of this proposition not presented due to space limitations but it can be done in a very straightforward manner).

We can derive a control input u_n using cooperative Hausdorff tracking technique in order to track the set $\Pi_{W_n}(K_m)$ by minimizing the shape function ${}^cJ_n(K_n, \Pi_{W_n}(K_m)) : \mathcal{K}(W_n) \times \mathcal{K}(W_n) \mapsto \mathbb{R}_+$.

Camera Inactive$(c_n = 0)$

When the distance of the target set to the coverage set of cam$_n$ is large i.e., the target is vary far away from cam_n, the variable c_n is set to zero and cam$_n$ enters into inactive mode. This mode prevents all the cameras from continuously servoing based on small motions of a target which is rather far away and is used to conserve energy of the camera. In this mode, the input to the camera, cam$_n$ is set to 0 and the camera does not move. The target dynamics are still represented using the mutational equation of the target set.

3.6 Chapter Summary

In this chapter, we have introduced the use of controlled mutational equations to model the motion of the sensor coverage and target sets. Using the mutational equation model, we have introduced the method of Hausdorff tracking which can be used to control the motion of the sensors (cameras) to maintain target visibility for moving targets. Specific examples of the Hausdorff tracking in image space and world space have also been presented.

Cooperation among various sensors is an important characteristic for pervasive surveillance. We have introduced the mutational hybrid automata (MHA) model for modeling motion of multiple sensors tracking a target. The implementation of this model for a pervasive surveillance scenario can be done in a distributed fashion as the various events occurring, such as target visibility, etc., can be detected and processed in a distributed fashion.

Chapter 4

Analysis of Surveillance Networks

This chapter presents the stability analysis of the proposed model and controllers. The shape Lyapunov theorem is presented in section 4.1, and can be used to analyze the asymptotic behavior of tubes. The stability analysis of the proposed Hausdorff tracking method using the shape Lyapunov theorem is presented next, followed by the analysis of an example two-camera system modeled using the mutational hybrid automata model proposed earlier.

4.1 Lyapunov Theorem for Shape Space

The asymptotic behavior of the measure $J(K(t))$ of the deformation of the set K can be studied using the shape Lyapunov theorem. The deformation is described as the reachable tube $K(t)$ and is the solution to the mutational equation in equation 3.1. We usually look for assumptions that guarantee the convergence to 0 of $J(K(t))$.

Definition 4.1.1 *(Shape Lyapunov Function) Consider a subset E of $\mathbb{R}^n$ and a mutational map φ defined on the set E as shown in section 1, a shape functional $J : \mathcal{K}(E) \mapsto \mathbb{R}^+$ and a continuous map $f : \mathbb{R} \mapsto \mathbb{R}$. The functional J is a f-Lyapunov* *fu*

w

W

J

as

T

se

m

d

if.

U

U

I

function for the mutational map φ if, for any $K \in Dom(J)$ we have,

$$J(\vartheta_\varphi(t,K)) \leqslant w(t), \forall t \leqslant 0, \quad J(K) = w(0) \tag{4.1}$$

where, $w(\cdot)$ is a solution of $w' = -f(w)$.

When the solution $w(\cdot)$ converges to 0 as $t \longrightarrow \infty$, the shape Lyapunov functional $J(\vartheta_\varphi(t,K))$ also converges to 0.

Using shape derivatives, the above definition of Lyapunov functions can be stated as the shape Lyapunov theorem.

Theorem 4.1.1 *Consider a subset E of $\mathbb{R}^n$ and a mutational map φ defined on the set E as shown in section 1, a shape functional $J : \mathcal{K}(E) \mapsto \mathbb{R}^+$ and a continuous map $f : \mathbb{R} \mapsto \mathbb{R}$. Let the Eulerian semi-derivative of J in the direction φ exist and be defined as $\mathring{J}(K)(\varphi)$. The functional J is an f-Lyapunov function for φ if and only if, for any $K \in Dom(J)$, we have*

$$\mathring{J}(K)(\varphi) + f(J(K)) \leqslant 0. \tag{4.2}$$

Proof. (from [65])Assume that for any positive t, we have $J(\vartheta_\varphi(t,K)) \leqslant w(t)$. Using $J(K(0)) = w(0)$, we can write

$$\frac{1}{t}(J(\vartheta_\varphi(t,K)) - J(K)) \leqslant \frac{1}{t}(w(t) - w(0)) \tag{4.3}$$

Using the definition of directional shape derivative this yields,

$$\mathring{J}(K)(\varphi) \leqslant w'(0) \tag{4.4}$$

Then, observing that $w'(0) = -f(w(0)) = -f(J(K))$ we can conclude the result.

Conversely, denote $y(t) = J(\vartheta_\varphi(t, K))$. We can now write

$$\begin{aligned} y(t) - y(0) &= \int_0^t \frac{d}{dt} J(\vartheta_\varphi(s, K))ds \\ &= \int_0^t \mathring{J}(\vartheta_\varphi(s, K))(\varphi)ds \end{aligned} \tag{4.5}$$

By the hypothesis, we obtain

$$y(t) - y(0) \leqslant -\int_0^t f(y(s))ds \tag{4.6}$$

Now, we can write

$$y'_d(t) \leqslant -f(y(t)) \tag{4.7}$$

Now consider a solution $w(\cdot)$ of the differential equation

$$\dot{w}(t) = -f(w(t)), \qquad w(0) = J(K) \tag{4.8}$$

Using the Gronwall Lemma, for any positive t, we obtain

$$\begin{cases} y(t) = J(\vartheta_\varphi(t, K)) \leq w(0) \\ x(0) = V(K) = w(0). \end{cases} \tag{4.9}$$

This proves the theorem. □

Using the shape Lyapunov theorem we can derive assumptions on the input u, to move the camera/robot, in order to accomplish the surveillance task.

4.2 Stability Analysis of Hausdorff Tracking

In this section we analyze the stability of the Hausdorff tracking method using the shape Lyapunov theorem discussed in the previous section.

Stability in the sense of Lyapunov for the proposed Hausdorff tracking method would imply the asymptotic convergence to zero of the shape function $J(K)$. This asymptotic convergence to zero of $J(K)$ will imply that the task criterion for the system is accomplished.

For both the image-based and cooperative Hausdorff tracking methods proposed, the shape directional derivative has been reduced to the following form:

$$\mathring{J}(K)(\varphi) \leqslant C(K) \cdot u \tag{4.10}$$

We can now propose the following theorem for stability of the Hausdorff tracking method.

Theorem 4.2.1 *For a automated tracking system modeled using the mutational equation $\mathring{K} \ni \varphi(K)$ where K is the target set and $\varphi(q) = B(q)u$ with the plant input vector u. The shape (error) function is defined as $J(K)$ with the shape directional derivative in the direction of the mutation ($\mathring{J}(K)(\varphi)$) defined by equation 4.10. The system is stable, i.e., the shape function asymptotically converges to zero for the input $u = -\alpha C^{\#}(K)J(K)$, $\alpha > 0$.*

Proof. In order to prove the stability of this system, we have to show that it satisfies the shape Lyapunov theorem from section 4.1. That would imply that the shape function $J(K)$ asymptotically converges to zero. The shape Lyapunov theorem can be written as:

$$\mathring{J}(K)(\varphi) + \alpha J(K) \leqslant 0 \tag{4.11}$$

Substituting the values of $\mathring{J}(K)(\varphi)$ into the above equation we get

$$C(K) \cdot u + \alpha J(K) \leqslant 0. \tag{4.12}$$

Substituting for u we get

$$-\alpha C(K) \cdot C^{\#}(K) \cdot J(K) + \alpha J(K) \leqslant 0 \tag{4.13}$$

$$\alpha J(K) \leqslant \alpha J(K) \tag{4.14}$$

which concludes the proof. □

4.3 Analysis of Example Surveillance System using Mutational Hybrid Automata Model

This section presents the stability analysis for a two-camera example system modeled using the mutational hybrid automata (MHA) model. The mutational hybrid model for a two-camera example system along with the various constituent parts is presented, followed by a stability analysis of the example system.

The two-camera model presented is extended by providing an inactive state for each camera. This implies that the cameras need not be tracking the target continuously and can shut down (go into an inactive state) or be assigned to other tasks when a particular target is not close by. Owing to this fact, the proposed model will be more energy efficient.

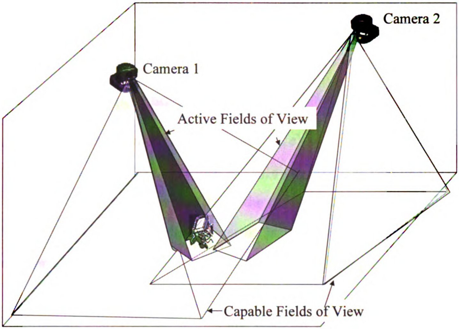


Figure 4.1: Example pervasive surveillance scenario with 2 cameras

4.3.1 Two-Camera Example for Pervasive Surveillance with Continuous Servoing Cameras

Consider the example as shown in Figure 4.1, which has two cameras with intersecting domains. With each camera we can associate a coverage set that represents the current FOV of the camera as represented in section 3.2. The total capable FOV of the camera cam_n can be represented as a closed set W_n. The closed set W can be modeled as the union of the total capable FOV of the two cameras ($W = W_1 \cup W_2$). The target (modeled as a domain in W, $\hat{K} \in \mathcal{K}(W)$) can move around in the set W with bounded velocity and acceleration. The problem is to find a hybrid controller (continuous control as well as switching conditions) which can pervasively track (cover using the active FOV) the target as it moves about randomly (maybe adversarially) in the domain W.

Consider the MHA model for the pervasive surveillance network given in equation

3.37. The constituent parts of the model can now be defined with respect to the two-camera pervasive surveillance system in consideration.

Hybrid State Space (Q and M)

The set of discrete states is modeled as $Q = \{b_1, b_2, c\} = \{q\}$ with its set of valuations $\mathcal{Q} = \{(0,0,2),(0,1,2),(1,0,2),(1,1,2)\} = \{q_0, q_1, q_2, q_3\}$. In this example we will assume that $c = 2$ which implies that both the cameras are involved in the tracking task. This assumption will be relaxed later on in the discussion. Each mode $q \in \Pi$, uses a different controller for the two cameras hence effecting distinct dynamics in them.

The continuous states for all modes are modeled as the target set $\hat{K}$ and the current coverage sets K_1 and K_2 of cameras cam_1 and cam_2, respectively. The coverage sets for the cameras can be defined as the sets of points which satisfy the viewing constraints of the camera including field of view and depth of field constraints such as the maximum and minimum focus distances. Consider a point $q \in \mathbb{R}^3$ in the coordinate frame O_n of cam_n defined as:

$$\begin{bmatrix} k_x \\ k_y \\ k_z \end{bmatrix} = \begin{bmatrix} r\cos(\theta)\sin(\phi) \\ r\cos(\phi) \\ r\sin(\theta)\sin(\phi) \end{bmatrix} \tag{4.15}$$

where (r, θ, ϕ) are the polar coordinates of the point k in the camera frame O_n. Using the above equation, the coverage set can be defined as :

$$\begin{aligned} K_n = \{k \mid r \in (D_{min}, D_{max}), \theta \in (\alpha_{min}, \alpha_{max}), \\ \phi \in (\beta_{max}, \beta_{min})\} \end{aligned} \tag{4.16}$$

where, (D_{min}, D_{max}) represent the depth of field and $(\alpha_{min}, \alpha_{max})$, $(\beta_{max}, \beta_{min})$ are the view angles for the sensor. The collection of continuous variables can be written as $M = \{\hat{K}, K_1, K_2\}$ with the set of valuations of M defined as $\mathcal{M} = \mathcal{K}(W) \times \mathcal{K}(W_1) \times \mathcal{K}(W_2)$.

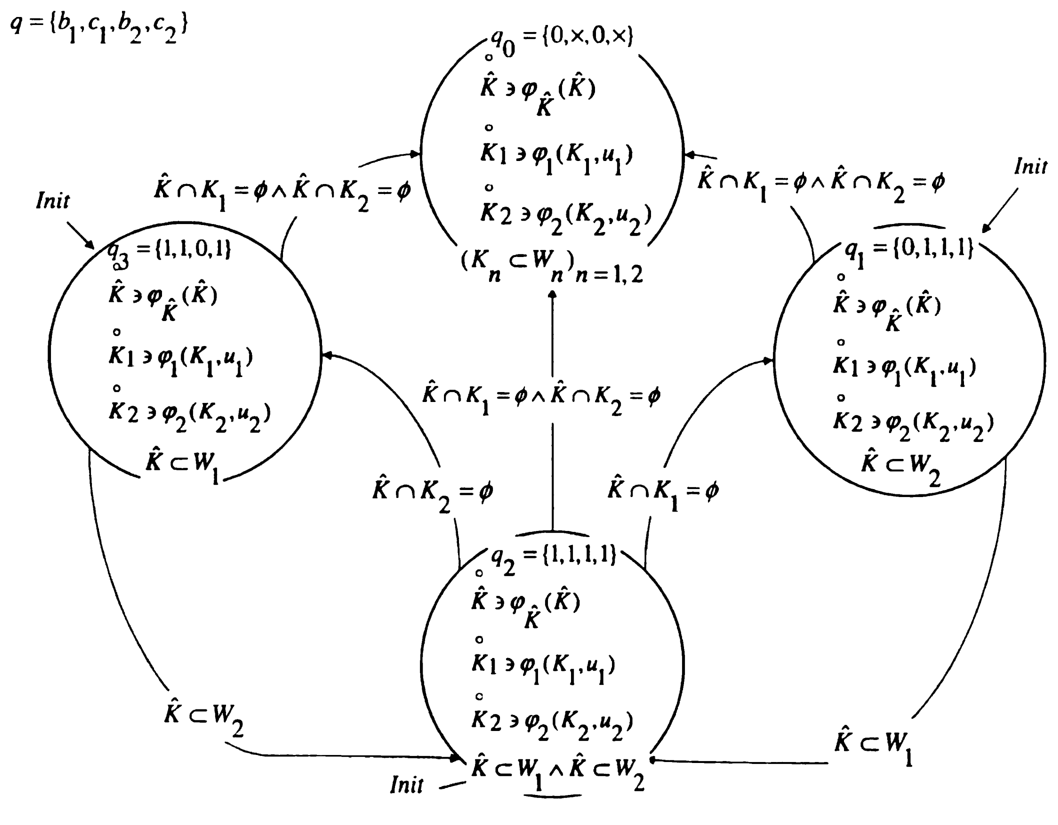

$q = \{b_1, c_1, b_2, c_2\}$
$q_0 = \{0, \times, 0, \times\}$
$\hat{K} \ni \varphi_{\hat{K}}(\hat{K})$
$K1 \ni \varphi_1(K_1, u_1)$
$K2 \ni \varphi_2(K_2, u_2)$
$(K_n \subset W_n)_{n=1,2}$
$q_1 = \{0, 1, 1, 1\}$
$\hat{K} \subset W_2$
$q_2 = \{1, 1, 1, 1\}$
$\hat{K} \subset W_1 \wedge \hat{K} \subset W_2$
$q_3 = \{1, 1, 0, 1\}$
$\hat{K} \subset W_1$
Init
Init
Init
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi$
$\hat{K} \subset W_2$
$\hat{K} \subset W_1$

Figure 4.2: Graphical MHA model for the two-camera scenario

Assumption: We can safely make that assumption that the target set $\hat{K}$ is convex. This assumption does not have significant effect on the model as all non-convex targets can be modeled as their convex cover. This assumption will be invoked to aid the proof of stability later in this section.

Domain (D) and Init Set (Init)

The domain $D(q)$ associated with every discrete mode q can be modeled as the union of the capable FOV W_n of the cameras involved in that particular mode such that the coverage sets of the individual cameras lie in their respective capable FOV's ($K_n \subset W_n$). We can further restrict the definition of the domain by using the condition that the target is a subset of the union of the capable FOV's of the cameras which can see the target ($\hat{K} \subset \cup_{n \in N, b_n = 1} W_n$). For this example the domains for each mode can be defined as:

$$\begin{aligned} D(q_0) &= \{K \in \mathcal{M} | (K_n \subset W_n)_{n \in \{1,2\}}\} \\ D(q_1) &= \{K \in \mathcal{M} | (K_n \subset W_n)_{n \in \{1,2\}} \wedge \hat{K} \subset W_2\} \\ D(q_2) &= \{K \in \mathcal{M} | (K_n \subset W_n)_{n \in \{1,2\}} \wedge \hat{K} \subset W_1\} \\ D(q_3) &= \{K \in \mathcal{M} | (K_n \subset W_n)_{n \in \{1,2\}} \wedge \hat{K} \subset \cup_{n \in \{1,2\}} W_n\} \end{aligned}$$

In order to track a target, it must be recognized by a camera for which it must be in the active FOV of a camera. The above condition dictates that the discrete state q must satisfy $\vee_{n \in N} b_n = 1$. If this condition is satisfied, then the target is visible in at least one camera. Hence we can define the Init set as $Init = \{q, K | q \in \{q_1, q_2, q_3\}\}$

Edge Set (E) and Reset Map (R)

The possible discrete transitions denoted by the edge set E are:

$$E = \{(q1, q0), (q1, q3)(q2, q0)(q2, q3), (q3, q1), (q3, q2), (q_3, q_0)\}$$

Note that under the assumptions of no occlusions in the camera views, continuity of target motion, finite size of the target set and non-empty overlap area between the two cameras, the transitions (q_1, q_2) and (q_2, q_1) do not occur. Further once the target is lost from both the active fields of view, the target is considered lost and cannot be recovered again, which implies that q_0 is an invariant mode and does not let any trajectories escape. Hence in our model we do not take into account the transitions $(q_0, q_1), (q_0, q_2), (q_0, q_3)$. However, in a real surveillance scenario, strategies based on searching for the target around its last known location can be developed and implemented in order to escape the failure state q_0.

The reset relation does not bring about any jumps in the continuous states. Hence

$$R(\cdot, \cdot, K) = K$$

Continuous Dynamics: Mutational Equations (Σ)

The target motion is modeled as a mutation $\overset{\circ}{\hat{K}} \ni \varphi_t(\hat{K})$ which is unknown but can be predicted as a piecewise linear trajectory using the image sequences of the target (the derivation of which is out of the scope of this paper). The 3D model for motion of the coverage sets K_1 and K_2 can be derived using the procedure outlined in section 3.2. Mutational equations are used to model the motion of the sets K_n.

Using equation 4.15, the mutational equation for the motion of the coverage set

K_n can be written as:

$$\dot{k} \;=\; \varphi_n(k) = B\; u_n \qquad \forall\;\; k \in K_n \tag{4.17}$$

where $u_n = [\omega_x, \omega_y]^T$ is the velocity input to cam$_n$. The above equation can be succinctly written as:

$$\mathring{K}_1 \ni \varphi_1(K_1, u_1), \;\; \mathring{K}_2 \ni \varphi_2(K_2, u_2)$$

where u_1 and u_2 are the inputs to the cameras cam$_1$ and cam$_2$, respectively, and are specified by a feedback map $\mathcal{U} : \mathcal{K}(W_n) \mapsto U$. The feedback maps (control inputs) u_i are selected based on the mode of operation of the network (q).

Target Visible($b_n = 1$) If the target is visible to cam$_n$ *i.e.,* $b_n = 1$, the control input u_n is calculated directly from the sensed image using image-based Hausdorff tracking proposed in [67]. The shape function (error) can be calculated as:

$${}^iJ_n = \int_{{}^{in}\hat{K}} d^2_{{}^{in}K_n}(p)\; dp \tag{4.18}$$

where ${}^{in}K$ and ${}^{in}\hat{K}$ are the projections of the coverage set of cam$_n$ and the target on the image plane ${}^iW_n$ of cam$_n$.

Target Not Visible ($b_n = 0$) When the target is not visible to cam$_n$ we use the method of cooperative Hausdorff tracking [67] to track a projection of the target $\Pi_{W_n}\hat{K}$ on the domain W_n, which is the capable field of view of cam$_n$. It should be noted that W_n is assumed to be convex.

However, the exact 3D location of the target set $\hat{K}$ is not known during the surveillance scenario. Here we point out that the target is visible to cam$_m$, *i.e.,* $b_m =$ 1. Hence, using proposition 1, we can say that tracking the projection $\Pi_{W_n}(K_m)$ of the coverage set K_m on the capable field of view W_n of cam$_n$ will suffice for tracking the projection of the target set $\Pi_{W_n}(\hat{K})$, since $\hat{K} \subset K_m$. Now we can derive the shape function (error) ${}^cJ_n$ which will be used as an error input to the controller for cam$_n$ for deriving the control input u_n. The shape function can be written as:

$$ {}^cJ_n = \int_{\Pi_{W_n}(K_m)} d^2_{({}^{in}K_n)}(p)\, dp \tag{4.19}$$

Continuous Control u_n Given the shape functions (errors) for the various cameras in different modes and the mutation equations for the coverage sets, the control input u_n can be calculated using the directional derivative of the shape function $J(K)$ in the direction of the mutation φ_K denoted as $\overset{\circ}{J}(K)$. The shape Lyapunov theorem can now be used to study the asymptotic behavior of the measure $J(K)$ of the set K and further derive assumptions on the input u_n which guarantee the convergence of $J(K)$ to zero.

Hence by selecting a control input u_n which ensures that $\overset{\circ}{J}(K) \leq -\alpha J$ we can prove that the continuous dynamics of each mode $\{q_1, q_2, q_3\}$ are exponentially stable. The detailed procedure is derived in [67].

Guard Conditions: Switching Control (G)

The guard condition can be construed as the switching conditions for the hybrid controller. These conditions have a profound impact on the stability of the system and should be carefully chosen. For our example system, transitions are enabled based on the recognition of certain events such as the recognition of the target by cam$_n$ in its field of view ($b_n = 1$). Assuming no occlusions and deterministic target recognition, the change in b_n can be equated to the target coming into or falling out of view of the coverage set K_n i.e, $b_n = 1 \Leftrightarrow \hat{K} \subset K_n$ and $b_n = 0 \Leftrightarrow \hat{K} \not\subset K_n$. Assuming that parts of the target can be recognized when they come into view of the camera, we can write the switching conditions as:

$$
\begin{aligned}
G(q_1, q_3) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 \neq \emptyset\} \\
G(q_2, q_3) &= \{K \in \mathcal{M} : \hat{K} \cap K_2 \neq \emptyset\} \\
G(q_1, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_2, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_3, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_3, q_1) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset\} \\
G(q_3, q_2) &= \{K \in \mathcal{M} : \hat{K} \cap K_2 = \emptyset\} \qquad (4.20)
\end{aligned}
$$

A hysteresis is inherently associated with the above switching conditions which will prevent infinite switching in a finite time period (Zeno execution).

4.3.2 Stability of the Two-Camera Example System

The question that arises at this point is "*How can we ensure that the tracked target does not leave the active FOV of at least one sensor at any time and the network behaves in a desired manner?*" This corresponds to the systems and control theoretic notion of "stability" - i.e., we need to guarantee that the hybrid flows within the network do occur as designed, states converge, the system does not "blow up", etc.

We propose a two-step process for the stability analysis of the pervasive surveillance scenario in section 6.7.3 modeled using the MHA model. First we require that each permissible mode of the model be internally stable. Internal stability implies that the continuous flow within a particular mode of the hybrid system converges and the hybrid state (Q, M) remains bounded away from the forbidden regions while the system operates in any particular mode.

The next step will be to show that the goal set of the mutational hybrid system will be invariant. Hence any trajectory that reaches in the goal set of the mutational hybrid system will always remain in the goal set. Now if we restrict the *Init* set to the goal set, the system trajectory will never leave the goal set and hence the system will be stable.

The goal set will be comprised of the various stable models of the MHA model. In order to prove the invariance of the goal set, we would be required to prove that every hybrid trajectory which exits a particular mode in the goal set, will only traverse to another mode in the goal set with the continuous variables initialized to permissible states in the new mode.

This procedure will ensure that the trajectory will never leave the goal set. If the goal set is taken as the set of modes for which the target is visible to at least one sensor, we can prove the pervasive tracking of the target.

Internal Stability for the Two-Camera Example System

Given an equilibrium point for each mode of operation of the MHA, we can now define the notation of internal stability [69] for each operational mode of the MHA. Internal stability implies that

Definition 4.3.1 *(Internally stability): A mode q of a mutational hybrid automaton is said to be internally stable if the local deliberative logic and hybrid control primitives have been designed such that:*

- *continuous evolution within each mode is stable in the sense of Lyapunov relative to an equilibrium point defined for that mode.*
- *the mechanism of mode transition is such that there is some dwell time in the mode q_i switched to (i.e., infinite switching does not occur)*

We can now put forward the following lemma for the stability in the sense of Lyapunov for the operating modes $\{q_1, q_2, q_3\}$ of the system.

Lemma 4.3.1 *Consider the continuous closed loop dynamics of the example two-camera system in the operating modes $\{q_1, q_2, q_3\}$ and the guard conditions 4.20. The operating modes $\{q_1, q_2, q_3\}$ are internally stable.*

Proof. For each state in $\{q_1, q_2, q_3\}$ we can define a candidate Lyapunov function V_i as:

$$q_1 : V_1 =^c J_1 +^i J_2$$
$$q_2 : V_2 =^i J_1 +^c J_2$$
$$q_3 : V_3 =^i J_1 +^i J_2$$

where, ${}^iJ_n \geq 0$ and ${}^cJ_n \geq 0$ are defined as in equation 4.18 and 4.19 respectively. Hence V_i $\{i = 1, 2, 3\}$ are candidate Lyapunov functions, since $V_i \geq 0$, $V_i = 0$ implies that the target is covered. Based on the procedure for calculating the motion input u_n to the cameras outlined in 4.3.1, we can show that ${}^iJ_n$ and ${}^cJ_n$ exponentially tend to zero, which implies that V_i also exponentially tends to zero. Hence we can say that the continuous evolution in the modes $\{q_1, q_2, q_3\}$ converges exponentially to the equilibrium state.

We further note that the switching conditions G defined in equation 4.20 have a certain hysteresis associated with them due to the finite size of the non-empty target set $\hat{K}$ and the non-empty overlap region $W_1 \cap W_2$. Hence we can prove that Zeno executions are prevented, which completes the proof. □

Hybrid System Stability for the Two-Camera Example System without Controlled Switches

Based on Definition 4.3.1 of internal stability and lemma 4.3.1, it can be deduced that each state (except q_0) is internally stable. We can now define stability for the pervasive surveillance scenario as:

Definition 4.3.2 *(Stability of a pervasive surveillance system) A pervasive surveillance system is said to have the stability property if a target set ($\hat{K}$) will be continuously covered in the monitored region if it enters the coverage set (K_i) of one of the sensors comprising the network.*

Now, if we can prove that the set of modes $\{q_1, q_2, q_3\}$ is invariant, then any hybrid execution (τ, q, K) starting in these modes will always remain in these exponentially stable modes, which implies that the target will be pervasively tracked. Based on these conditions we can put forward the following theorem for stability of a pervasive surveillance network.

Theorem 4.3.1 *Consider the graphical MHA model of a pervasive surveillance system in Figure 4.2. The set of operating modes $\{q_1, q_2, q_3\}$ is an invariant set.*

Proof. Consider a hybrid trajectory $\xi_H = (\tau, q, K)$ of the system with its initial conditions within the set $\{q, K | q \in \{q_1, q_2, q_3\}\}$. The modes $\{q_1, q_2, q_3\}$ are exponentially stable, which implies that the energy of the system in that mode can be reduced to a very small value in any finite amount of time t. For mode q_1 this implies that when the target $\hat{K} \subset K_2$ hits the boundary of the domain W_1, its projection $\Pi_{W_1}(\hat{K}) \subset \Pi_{W_1}(K_2)$ is already covered by K_1 as $\Pi_{W_1}(K_2) \subset K_1$. When $\hat{K}$ is inside the boundary of W_2, $\Pi_{W_2}(\hat{K}) \subseteq \hat{K} \subset K_1$. Hence the target is within the coverage region of K_1 and will be recognized by K_1 which will generate the event to set $b_1 = 1$, which in turn will switch the mode to q_3 (assuming $W_1 \cap W_2 \neq \emptyset$). Thus as soon as the target moves out of W_2 and into W_1 it will be covered by K_1. A similar argument can be presented for the mode q_2 and the target hitting the boundary of W_2.

Now in mode q_3, the target is located in the intersection of the domains W_1 and W_2 ($\hat{K} \subset W_1 \cap W_2$) and is being covered by both K_1 and K_2 such that $\hat{K} \subset K_1$ and $\hat{K} \subset K_2$ due to the exponential stability of mode q_3. Now the target can only move into the regions $W_1 \setminus (W_1 \cap W_2)$ or $W_2 \setminus (W_1 \cap W_2)$ and in both regions either K_1 or

K_2 can cover the target, respectively. This implies that the system can switch modes to q_1 or q_2, both of which possess the exponential stability (in the sense of Lyapunov) property.

We have shown that all trajectories originating in q_1 and q_2 can switch only to state q_3 and all trajectories originating in q_3 can switch only to either q_1 or q_2. This implies that the set of modes $\{q_1, q_2, q_3\}$ is invariant, which proves the theorem. □

Based on the above theorem, and the fact that all the modes in the invariant set $\{q_1, q_2, q_3\}$ are exponentially stable (in the sense of Lyapunov), we can conclude that the hybrid system is stable i.e., the target set is continuously covered once it enters the invariant set.

4.3.3 Two-Camera Example for Pervasive Surveillance with Controlled Switches

For the two-camera pervasive surveillance example considered in the previous section 4.3, the discrete variables were comprised of $Q = (b_1, b_2, c) = \{q\}$ where b_i is a binary value which denotes whether the target is visible to sensor Cam_i and c is a discrete variable which indicates which cameras are involved in the tracking scenario.

This representation of the discrete state c is not very intuitive. Hence for a truly distributed implementation, we propose to modify the discrete variable c and separate it into its constituent parts. Each camera will be associated with its own c_i which will assist in a distributed implementation.

For the example system presented earlier, only the modes in which the discrete variable c was set to 2 were considered. This implies that for all possible locations of the target all the cameras will be servoing the target. This scenario will be wasteful for energy consumption and does not take into account the fact that the sensors need not track the target if it is relatively far away from the sensor's domain.

Since the cameras need not be tracking the target continuously and can shut down when the target is not close by, the MHA model with controlled switching will be more energy efficient. Figure 4.3 shows the MHA model of the two camera system with controlled a controlled switch which is used render a camera into an inactive state.

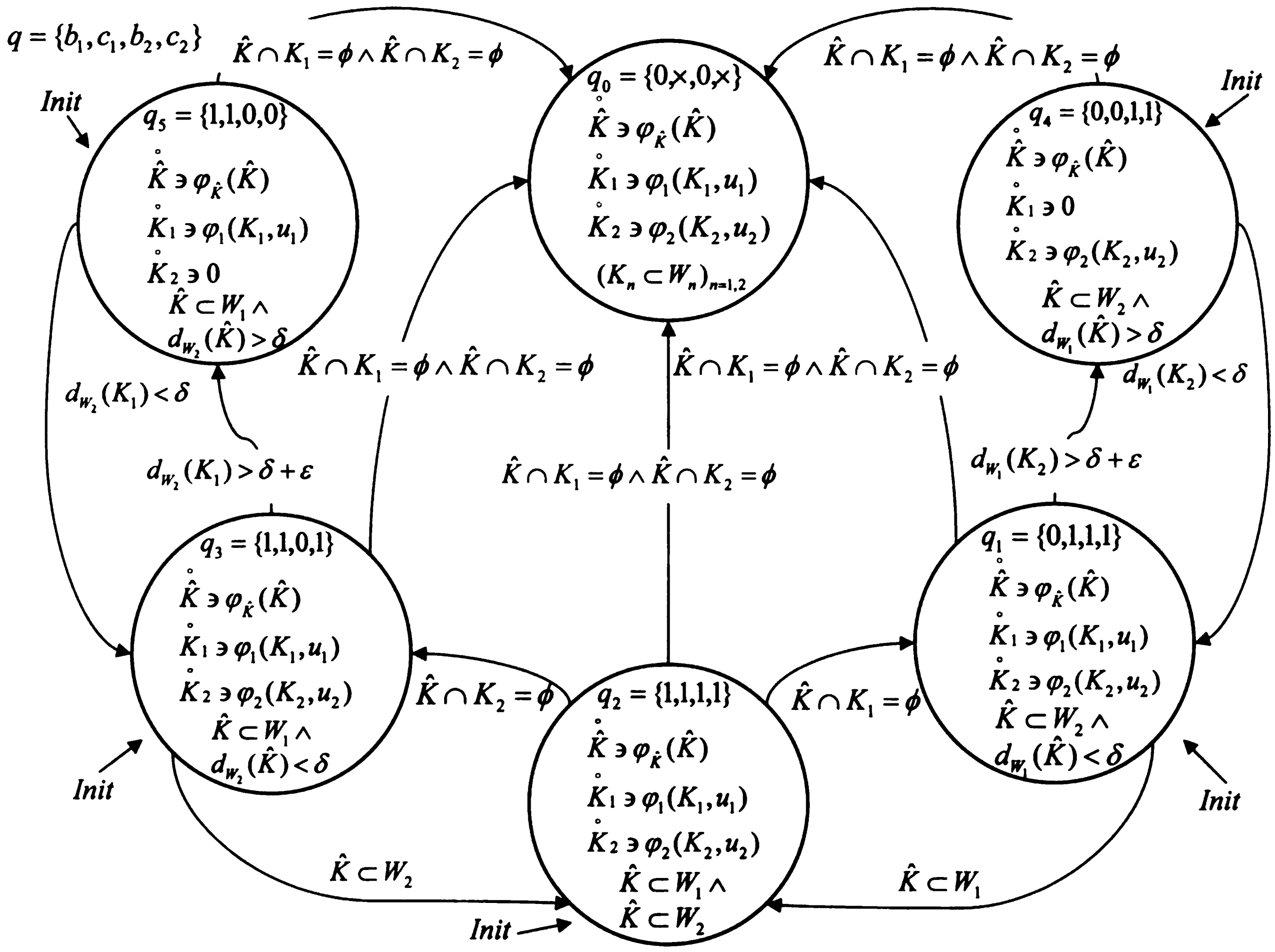

$q = \{b_1, c_1, b_2, c_2\}$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
Init
Init
$q_5 = \{1,1,0,0\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni \varphi_1(K_1, u_1)$
$\overset{\circ}{K_2} \ni 0$
$\hat{K} \subset W_1 \wedge$
$d_{W_2}(\hat{K}) > \delta$
$q_0 = \{0{\times},0{\times}\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni \varphi_1(K_1, u_1)$
$\overset{\circ}{K_2} \ni \varphi_2(K_2, u_2)$
$(K_n \subset W_n)_{n=1,2}$
$q_4 = \{0,0,1,1\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni 0$
$\overset{\circ}{K_2} \ni \varphi_2(K_2, u_2)$
$\hat{K} \subset W_2 \wedge$
$d_{W_1}(\hat{K}) > \delta$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$d_{W_2}(K_1) < \delta$
$d_{W_1}(K_2) < \delta$
$d_{W_2}(K_1) > \delta + \varepsilon$
$\hat{K} \cap K_1 = \phi \wedge \hat{K} \cap K_2 = \phi$
$d_{W_1}(K_2) > \delta + \varepsilon$
$q_3 = \{1,1,0,1\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni \varphi_1(K_1, u_1)$
$\overset{\circ}{K_2} \ni \varphi_2(K_2, u_2)$
$\hat{K} \subset W_1 \wedge$
$d_{W_2}(\hat{K}) < \delta$
$q_1 = \{0,1,1,1\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni \varphi_1(K_1, u_1)$
$\overset{\circ}{K_2} \ni \varphi_2(K_2, u_2)$
$\hat{K} \subset W_2 \wedge$
$d_{W_1}(\hat{K}) < \delta$
$\hat{K} \cap K_2 = \phi$
$\hat{K} \cap K_1 = \phi$
$q_2 = \{1,1,1,1\}$
$\overset{\circ}{\hat{K}} \ni \varphi_{\hat{K}}(\hat{K})$
$\overset{\circ}{K_1} \ni \varphi_1(K_1, u_1)$
$\overset{\circ}{K_2} \ni \varphi_2(K_2, u_2)$
$\hat{K} \subset W_1 \wedge$
$\hat{K} \subset W_2$
Init
Init
$\hat{K} \subset W_2$
$\hat{K} \subset W_1$
Init

Figure 4.3: Graphical MHA model for the two-camera scenario with controlled switches

Discrete State Space (Q)

The discrete states are modeled as $Q = \{c_1, b_1, c_2, b_2\} = \{q\}$ where b_n denotes the visibility of the target from camera cam$_n$ and c_n denotes if cam$_n$ is active or inactive for that state.

Domain (D) and Init Set (Init)

For the MHA model depicted in Figure 4.3 the domains of operation of the various modes are modified as:

$$\begin{aligned}
D(q_0) &= \{K \in \mathcal{M} | (K_n \subset W_n)_{n \in \{1,2\}}\} \\
D(q_1) &= \{K \in \mathcal{M} | \hat{K} \subset W_2 \wedge d_{W_1}(\hat{K}) < \delta\} \\
D(q_2) &= \{K \in \mathcal{M} | \hat{K} \subset \cup_{n \in \{1,2\}} W_n\} \\
D(q_3) &= \{K \in \mathcal{M} | \hat{K} \subset W_1 \wedge d_{W_2}(\hat{K}) < \delta\} \\
D(q_4) &= \{K \in \mathcal{M} | \hat{K} \subset W_1 \wedge d_{W_1}(\hat{K}) > \delta\} \\
D(q_5) &= \{K \in \mathcal{M} | \hat{K} \subset W_2 \wedge d_{W_2}(\hat{K}) > \delta\}
\end{aligned}$$

where, δ is a positive scalar distance whose value is selected by the system designer based on the controller design and stability requirements of the model.

The Init set can be defined as $Init = \{q, K | \vee_{i=1toN} b_i = 1\}\}$. This implies that the Init set encompasses all the states where the target is visible to at least one camera.

Edge Set (E)

The possible discrete transitions denoted by the edge set E are:

$$\begin{aligned} E = \{&(q1, q0), (q1, q2)(q2, q0)(q2, q3), (q2, q1), (q3, q2), (q_3, q_0), \\ &(q_4, q_0), (q_5, q_0), (q_4, q_1), (q_1, q_4), (q_5, q_3), (q_3, q_5)\} \end{aligned} \tag{4.21}$$

Continuous Dynamics: Mutational Equations (Σ)

The continuous dynamics in states q_1, q_2 and q_3 are the same as in the previous example presented in section 6.7.3. However for the sates q_4 and q_5, the continuous dynamics are modified such that the camera that can view the target will perform image based Hausdorff tracking while the other camera will have zero input. Hence there will be a third case such that $c_n = 0$.

Camera Inactive, ($c_n = 0$) When the camera is inactive, the input the the system is set to zero i.e., $u_n = 0$.

Guard Conditions: Switching Control (G)

The various events recognized by the distributed sensors are:

- Target visible to camera cam_n
- Location of coverage set of all neighboring cameras.

Using the location of the neighboring cameras, the current camera can estimate the approximate location of the target as shown in 4.3.1. The guard conditions for the various transitions can be written as:

$$\begin{aligned}
G(q_1, q_2) &= \{K \in \mathcal{M} : \hat{K} \subset W_1\} \\
G(q_3, q_2) &= \{K \in \mathcal{M} : \hat{K} \subset W_2\} \\
G(q_1, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_3, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_2, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_4, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_5, q_0) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset \wedge \hat{K} \cap K_2 = \emptyset\} \\
G(q_2, q_1) &= \{K \in \mathcal{M} : \hat{K} \cap K_1 = \emptyset\} \\
G(q_2, q_3) &= \{K \in \mathcal{M} : \hat{K} \cap K_2 = \emptyset\} \\
G(q_3, q_5) &= \{K \in \mathcal{M} : d_{W_2}(K_1) > \delta + \epsilon\} \\
G(q_5, q_3) &= \{K \in \mathcal{M} : d_{W_2}(K_1) > \delta\} \\
G(q_1, q_4) &= \{K \in \mathcal{M} : d_{W_1}(K_2) > \delta + \epsilon\} \\
G(q_4, q_1) &= \{K \in \mathcal{M} : d_{W_1}(K_2) > \delta\}
\end{aligned}$$

where, $\epsilon > 0$ is an arbitrarily small positive integer. A hysteresis is inherently associated with the switching conditions for every mode which will prevent infinite switching in a finite time period (zeno execution).

4.3.4 Stability Analysis of the Two Camera Example with Controlled Switches

We can apply a two step procedure similar to the one presented in 4.3.2 in order to analyze the stability of the mutational hybrid automata model presented in Figure 4.3. First we prove that each mode in the set $q = \{q_1, q_2, q_3, q_4, q_5\}$ is internally stable as defined in Definition 4.3.1. Next we show that the goal set of the mutational hybrid system is invariant which would imply that a trajectory which reaches the goal set will remain in the goal set. Further, by restricting the *Init* set to the goal set, we can show that all trajectories of the mutational hybrid system initialized in the *Init* set will remain in the goal set hence completing the proof.

Internal Stability

Lemma 4.3.2 *Consider the continuous closed loop dynamics of the example two-camera system represented in Figure 4.3 in the operating modes* $\{q_1, q_2, q_3, q_4, q_5\}$ *and the guard conditions 4.22. The operating modes* $\{q_1, q_2, q_3, q_4, q_5\}$ *are internally stable.*

Proof. For each state in $\{q_1, q_2, q_3, q_4, q_5\}$ we can define a candidate Lyapunov function V_i as:

$$\begin{aligned}
q_1 : V_1 &= {}^cJ_1 +^i J_2 \\
q_2 : V_2 &= {}^iJ_1 +^c J_2 \\
q_3 : V_3 &= {}^iJ_1 +^i J_2 \\
q_4 : V_4 &= {}^iJ_2 \\
q_5 : V_5 &= {}^iJ_1
\end{aligned}$$

where, ${}^iJ_n \geq 0$ and ${}^cJ_n \geq 0$ are defined as in equation 4.18 and 4.19 respectively. Hence V_i $\{i = 1, 2, 3, 4, 5\}$ are candidate Lyapunov functions, since $V_i \geq 0,\ V_i = 0$ implies that the target is covered. Based on the procedure for calculating the motion input u_n to the cameras outlined in 4.3.1, we can show that ${}^iJ_n$ and ${}^cJ_n$ exponentially tend to zero, which implies that V_i also exponentially tends to zero. Hence we can say that the continuous evolution in the modes $\{q_1, q_2, q_3, q_4, q_5\}$ converges exponentially to the equilibrium state.

We further note that the switching conditions G defined in equation 4.22 have a certain hysteresis associated with them due to the finite size of the non-empty target set $\hat{K}$ and the non-empty overlap region $W_1 \cap W_2$. We can also assume that the size of the camera coverage set for any camera is larger than the size of the target which implies that that $d_{W_1}(K_2) \leqslant d_{W_1}(\hat{K})$ when the target set is covered by the coverage set K_2 of camera cam$_2$. We can write a similar condition for the target being covered by camera 1.Hence we can prove that Zeno executions are prevented, which completes the proof. □

Hybrid System Stability

In order to prove that the target will always be visible to at least one camera at all times, i.e., stability of the hybrid system, we need to prove that the set of modes $q = \{q_1, q_2, q_3, q_4, q_5\}$ is invariant. This will imply that once the target enters this set of invariant modes, i.e., the target is visible to at least one camera, the hybrid trajectory will never leave the invariant set.

Theorem 4.3.2 *Consider the graphical MHA model of a pervasive surveillance system in Figure 4.3. The set of operating modes $\{q_1, q_2, q_3, q_4, q_5\}$ is an invariant set.*

Proof. Consider a hybrid trajectory $\xi_H = (\tau, q, K)$ of the system with its initial conditions within the set $\{q, K | q \in \{q_1, q_2, q_3, q_4, q_5\}\}$. The modes $\{q_1, q_2, q_3, q_4, q_5\}$ are exponentially stable, which implies that the energy of the system in that mode can be reduced to a very small value in any finite amount of time t.

In the mode q_4, $\hat{K} \subset K_2$. Further as the target moves closer to the domain W_1, $d_{W_1}(K_2) \to \delta$ and as soon as the $d_{W_1}(K_2) < \delta$, there is a mode switch to mode q_1.

For mode q_1 this implies that when the target $\hat{K} \subset K_2$ hits the boundary of the domain W_1, its projection $\Pi_{W_1}(\hat{K}) \subset \Pi_{W_1}(K_2)$ is already covered by K_1 as $\Pi_{W_1}(K_2) \subset K_1$. When $\hat{K}$ is inside the boundary of W_2, $\Pi_{W_2}(\hat{K}) \subseteq \hat{K} \subset K_1$. Hence the target is within the coverage region of K_1 and will be recognized by K_1 which will generate the event to set $b_1 = 1$, which in turn will switch the mode to q_2 (assuming $W_1 \cap W_2 \neq \emptyset$). Thus as soon as the target moves out of W_2 and into W_1 it will be covered by K_1.

Now within the mode q_1, consider the target moving away from the domain W_1. This will imply that the distance $d_{W_1}(\hat{K})$ is increasing which would in turn imply that $d_{W_1}(K_2)$ is increasing. When $d_{W_1}(K_2) > \delta + \epsilon$, the system switches to mode q_4.

At the instant when there is a mode switch from mode q_4 to q_1, $d_{W_1}(\hat{K}) > \delta$. Assuming that the maximum velocity of the target centroid is v_{max} it will take the target $t_1 = \delta / v_{max}$ time to reach the boundary of domain W_1. From the continuous dynamics system, the error ${}^cJ_2$ can be reduced to zero in a finite amount of time $t_2 \leqslant 5\alpha$ where α is the gain of the Hausdorff tracking control input. Now, if $t_2 < t_1$ i.e., $\delta > 5\alpha v_{max}$, we can show that the Lyapunov function V_1 in mode q_1 will be reduced to zero before the target set intersects with the boundary of domain W_1. Hence we have shown that for all mode switches out of mode q_1, the Lyapunov function of q_1 is reduced to zero.

A similar argument can be presented for the mode q_3 and the target hitting the boundary of W_2 or moving away from the domain W_2 such that $d_{W_2}(K_1) > \delta + \epsilon$.

Now in mode q_2, the target is located in the intersection of the domains W_1 and W_2 ($\hat{K} \subset W_1 \cap W_2$) and is being covered by both K_1 and K_2 such that $\hat{K} \subset K_1$ and $\hat{K} \subset K_2$ due to the exponential stability of mode q_2. Now the target can only move into the regions $W_1 \setminus (W_1 \cap W_2)$ or $W_2 \setminus (W_1 \cap W_2)$ and in both regions either K_1 or K_2 can cover the target, respectively. This implies that the system can switch modes to q_1 or q_3, both of which possess the exponential stability (in the sense of Lyapunov) property.

We have shown that all trajectories originating in $q_1(q_3)$ can switch only to state $q_2(q_2)$ or $q_4(q_5)$, all trajectories orignating in state $q_4(q_5)$ can switch only to state

$q_1(q_3)$ and all trajectories originating in q_2 can switch only to either q_1 or q_3. This implies that the set of modes $\{q_1, q_2, q_3, q_4, q_5\}$ is invariant, which proves the theorem. □

4.4 Chapter Summary

This chapter presents the analysis of the Hausdorff tracking method and the mutational hybrid automata model presented in Chapter 3. The Lyapunov theorem for shape spaces is presented and it is used to prove the stability (in the sense of Lyapunov) for the Hausdorff tracking method presented. The stability analysis of an example two-camera pervasive surveillance system is also presented.

Chapter 5

Design of Surveillance Networks

Collaboration among multiple sensors can generally lead to better sensing performance and higher fault tolerance to individual node failure. Calibration of the sensors aids collaboration by establishing a common frame of reference for the various sensors. Calibration needs to be handled at many levels including location calibration of the sensors, color map calibration, calibration of the target recognition modules etc.

Coordinating the tracking of multiple targets using multiple sensors can be addressed in two ways:

1. Use a central monitoring station to integrate the track information from each node [39].

2. Allow sensor nodes to operate autonomously and exchange track information with one another in a geographical vicinity and provide a handover mechanism when the target transitions from one sensor field of view to another [37].

The first approach simplifies track management and communications but is not scalable to large networks due to the limited communication and processing capability of the central monitoring station. Also, the system is not fault tolerant to the failure of the central monitoring station. The second approach is highly scalable and fault tolerant to the failure of nodes, but involves significant communication overhead for target track management and disambiguation.

5.1 Wide Area Camera Calibration

Calibration of the sensors in a wide area environment is a difficult task without physical access to the environment that the sensors are placed in. Normal techniques of system calibration such as using precise planar grids [70] are not possible in a previously unknown and inaccessible environment. Only the intrinsic properties of the sensors can be calculated and accounted for prior to deployment. Therefore, a robust calibration method that allows for a wide dispersion of sensors to be calibrated must be used.

5.1.1 Unitless Calibration

An unsynchronized network of cameras can be calibrated by tracking an identifiable point in motion. This provides a unitless orientation of all sensors that have sensed the point. The sensor network must meet the requirement of viewing area overlap for each camera with at least one other camera. Also, all sensors must be interconnected via overlapping viewing areas of intermediary sensors. Without this interconnectedness the network would be segmented into groups that could not be related spatially. This method is ideal for our proposed environment for the following reasons: 1) with a drop-in-place environment there is no assurance that sufficient identifiable markers will be available for automatic calibration of the type proposed by [71] [72] 2.) A Kalman filtering technique can be used to reduce the error between pairs of sensors and the propagation of error across the network 3.) The world coordinate system can be defined with respect to any camera in the network including the mobile sensors.

Initial Calibration

It is desirable to have taken the intrinsic characteristics of the cameras into account prior to deployment to reduce the number of actions necessary to calibrate the system. These intrinsic properties are represented in matrix form as

$$\mathbf{A} = \begin{bmatrix} \alpha_u & c & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

and includes the focal length (α_u, α_v), image center (u_0, v_0), and skewness (c) of the image [73]. Focal length is the distance from the lens to the projection plane. The image center is the center of the projected image on the projection plane. This allows a more accurate pose to be generated. Finally, the skewness is the ratio of the pixel width to pixel height and is used for correction of a flawed lens or to normalize wide view images. We will assume that these characteristics are already well known for each sensor in the network.

The extrinsic parameters of the cameras can be found using a point-in-motion scheme. By identifying a detectable point in motion it is necessary, for every pair of cameras viewing the point, to calculate both the cameras' transformation matrices. A camera's transformation matrix is composed of the rotation matrix and the translation vector for the camera with respect to another. This matrix is represented as:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \tag{5.2}$$

The intrinsic and extrinsic matrices are then combined to provide a complete description of the mapping of a point in 2D image coordinates to 3D coordinates in the world frame.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.3}$$

One method of wide area calibration of extrinsic parameters is a point in motion calibration technique. This method, proposed by [74], allows for a single identifiable point to be passed throughout the observable environment. When all sensors have detected the point while another sensor has the point in view simultaneously, a rough estimate of the unitless calibration can be generated. This method is better suited for wide area calibration because of the relatively large distances between sensors. In order for a calibration object to be used to generate accurate results, resolution is very important. The most accurate calibration takes place with the object filling most of the screen. For a good calibration to be performed with a calibration object in a wide area, it would need to be quite large to ensure the pose was properly modeled. This is one of the strengths of the point-in-motion technique. The points can be distributed throughout the volume of the coverage set, creating a virtual calibration object.

The scalar λ is called the depth and accounts for perspective effects [75] and can have any value. This ambiguity is a result of the loss of the z coordinate in relation between the object in space and the object on the image plane. For any point (u_i,v_i) in the image plane i, a matrix $\mathbf{T}_i$ exists that satisfies

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \mathbf{A}\mathbf{T}_i \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5.4}$$

where λ_i is the depth at that point i.

The collected data is then processed assuming that times are synchronous for each pair of positions and recorded for the sensors. Processing includes generating an initial rough extrinsic calibration. Since no actual synchronization is used to gather data, there will be time domain discrepancies in every pair of 2D points used to find location and pose. This error in the developing map is reduced later using an extended Kalman filter [76].

5.1.2 Method for Assigning Units

In order to find the scale factor associated with the calibration matrix it is necessary to make some accurate real world measurements. With a drop-in-place system a number of methods can be used to find the scalar λ associated with the calibration matrices. In our proposed environment there can be no assurance that a known dimension will be found or that an object with a known dimension can be placed.

We have implemented a pre-calibrated dual head camera rig. This rig can be used to accurately assign units to the unitless calibration data that has already been generated. An accurate calibration such as that proposed by Zhang [72] will have been performed and the unit's characteristics will be considered well known so that minimal error is introduced into the scaling of the sensor map. All sensors in the network of calibrated pairs that now have a shared normalized transformation can have units assigned with only a single shared marker with one other sensor by the pre-calibrated dual sensor rig.

5.2 Grouping Architectures

One important issue we need to address for the surveillance network is group formation and management. There is extensive previous research done to address forming and managing ad hoc network groups based on the proximity of the sensor nodes. In most surveillance tasks, spatial proximity of sensor nodes is a vital characteristic of sensor networks. In order to take advantage of the proximity of the nodes to save energy or enable efficient sensing information processing, sensors are often organized into local collaborative groups.

The group management approaches differ from each other depending on how the target object is identified and the parameters are detected. In the work of Chen et al. [77], the target is detected based on the acoustic signal strength and a group management algorithm is proposed based on the Voronoi diagram. Liu et al. [78], proposed a distributed group management framework for target localization applications. In their framework, each node runs a likelihood ratio test (LRT) once the target is detected. Later the group formation and leader election process is initialized based on the result of LRT. In RoamHBA (Roaming Hub Based Architecture, [79]), a subnetwork is established among the sensor nodes that are observing the same target. Communication within this subnet is done through a multicast tree, called a roaming hub. The subnetwork is updated by adding or deleting nodes at the edge.

For the visual surveillance network, we propose to use the RoamHBA(Roaming Hub Based Architecture, [79]), based group management approach. One group will be set up for all the nodes in a particular locality. In most cases, it is necessary for the tracking group to elect a leader to coordinate the tracking and surveillance task. Previous research work addressed the leader election problem in a wireless sensor network

The dynamically elected leader is responsible for coordinating the tracking tasks of each node in the group. Periodically the group leader runs the coverage optimization algorithm. A timer is set up to invoke the process of coverage optimization. The value of the time is determined based on the capabilities of the sensor node and the characteristics of the targets being tracked. For example, if the targets move very fast, the timer should have a very short timeout, and at the same time, the requirement for the capabilities of the sensor nodes are increased. The result of the optimization algorithm, that is, the (multiple) target assignments, should be transmitted to the member nodes using the networking transport services provided by the sensor nodes. The actual transport services being used can vary based on the network infrastructure on which the sensor network is built up.

It is reasonable to assume that the sensor nodes are equipped with geographical-location-aware devices. Extensive research has been done about forwarding packets in a wireless network based on the geographical location of the nodes. Different from traditional wireless routing protocols, in location-based routing each node only needs to know its own position and the position of its one-hop neighbors. This makes location-based routing perform well in a dynamic environment such as a mobile surveillance network. More details about location-based routing can be found in [80].

If the sensor nodes are geographically distributed and connected to the Internet, common multicast protocols can be used for the group leader to disseminate the coverage information to the group members. The group management can be done using protocols like Internet Group Management Protocol (IGMP) and multicast routing protocols can be used to set up the routes. In more common cases, where the sensor nodes are connected using wireless networking technologies such as IEEE 802.11 series technologies, group management techniques presented at the beginning of the section can be used. The dissemination of the control information for sensor tasking can be done using the multicasting or broadcasting techniques in wireless networks. For access-point-based wireless networks, multicast among the group can be done with the aid of the access points. For wireless ad hoc networks, much research has been done to solve the multicasting problem. In the most straightforward case, a broadcast simulated multicast, which involves the nodes boradcasting within their locality, can serve the goal of a surveillance network.

5.3 Sensor Allocation for Multi-Sensor Cooperation

For successful completion of the surveillance task, it is important to allocate the sensors to the targets that are actively being tracked. Generally, each target should be tracked by the sensor that can view it at an optimal resolution for best visibility of that target. However, when the system is heavily loaded, that is, the target density in a particular region with a limited number of sensors is high, one sensor might need to track multiple targets. Even when the system is underloaded, multiple target tracking by individual sensors is highly desirable as viewpoint redundancy will lead to better estimates of the target coordinates and also present the viewer a clearer view of all the targets that are being tracked. In this section we propose a heuristic method to determine an optimal sensor allocation strategy.

At each system interval, the leader node computes a cost and visibility weight for the targets being tracked. Assume m sensors with cameras were setup in the system and there are n targets that need to be covered. One camera can cover from 0 to up to n targets, so for each camera, the number of possible coverage sets is $\sum_{k=0}^{n}\binom{n}{k} = 2^n$. In order to determine the optimal coverage set for each camera, we set up a weight matrix W for the coverage options. An element of the matrix W is defined as ($i \in [1,m]$ and $k \in [0, 2^n - 1]$):

$$w_{ik} = \sum_{s=1}^{C(L(k))} V_{i,L(k)} \cdot (R_{s,i,L(k)} \cdot w_v + P_s \cdot w_p) \tag{5.5}$$

The following parameters are explained as follows:

- $L(k)$ is the k^{th} combination of the set $[1, n]$.
- $V_{i,L(k)}$ is a binary value (either 0 or 1) that indicates the ability for the sensor node to cover all the targets in the set $L(k)$.
- $R_{s,i,L(k)}$ is the resolution of target s viewed by sensor node i when all the targets

in the set $L(k)$ are covered by sensor node i. The resolution is calculated as follows:

$$R_{s,i,L(k)} = \lambda \frac{x_s}{z_{s,i}} \tag{5.6}$$

where x_s is the width of the target s and $z_{s,i}$ is the distance between target s and sensor node i. $\lambda \in (\lambda_{min}, \lambda_{max})$ is the zoom factor (focal length) of the camera of sensor node i.

- P_s is the preassigned priority of target s in the tracking system.
- w_v and w_p are adjustable parameters to reflect the importance of resolution and priority in the coverage weight.

For each row of the matrix, a larger value will indicate a more desirable target assignment. However, it is required that all targets should be covered by the sensor nodes. Thus, an optimal solution of the problem can be defined as $\{(y, L_y)|y \in [1, n]$ and L_y is a combination of the set $[1, n]\}$ which satisfies that $\sum_{y=1}^{n} L_y$ is the largest among all the possible coverage set for each camera.

By brute force the complexity of the algorithm is $O(2^{mn})$. When n and m are large, heuristic algorithms are needed to find the optimal assignment. In reality, many heuristic approaches can be taken into account when solving the problem. For example, we can prove that under normal target density, the coverage matrix will be a sparse matrix, where most of the elements are zero. Further, in most cases, we do not need to find the optimal solution. Finding a sub-optimal solution would reduce the complexity of the algorithm dramatically. For example, after setting a threshold of the acceptable weight, we can take the first solution encountered as long as all the targets are covered.

5.4 Optimal Hausdorff Tracking

Given the task specified by equation 3.31, notice that $C(K, \hat{K})$ is not a square matrix, which indicates that the system is redundant. Further the condition from Equation (3.30) indicates that there are numerous choices of the input u which satisfy the stability requirements of the Hausdorff tracking system. This implies that the system has infinite solutions for a choice of input u which will accomplish the task. Therefore the designer can choose an algorithm in order to achieve an optimal solution for this system. An appropriate choice for u will minimize the resolution inadequacies and energy consumed.

A review on recent literature shows that a commonly used approach for the optimal redundancy resolution relies on the use of generalized or pseudoinverse transformation to degenerate the system, while optimizing the given task criterion constructed from multiple objective functions using a weighted sum method [81], [82].

Optimality can be quantified by defining supplementary task functions which the system needs to accomplish. These functions are called objective functions which need to be minimized in order to achieve an optimal solution. The optimization problem can be expressed as:

Find the target and coverage sets (K and $\hat{K}$ respectively) and input u which satisfy:

$$\begin{array}{ll} \min\limits_{K,\hat{K},u} : & \mathbf{D}(K,\hat{K},u) = \begin{bmatrix} D_1(K,\hat{K},u) & \dots & D_N(K,\hat{K},u) \end{bmatrix}^T \\ \text{Subject to:} & \begin{cases} J(K,\hat{K}) = \int_{\hat{K}} d_K^2(p)\, dp \\ \mathring{J}(K,\hat{K})(\varphi_1,\varphi_2) = C(K,\hat{K})u \leq -\alpha J(K,\hat{K}) \end{cases} \end{array} \tag{5.7}$$

where, $D_i(K, \hat{K}, u)$, $i = 1$ to N are the N objective functions, $D(K, \hat{K}, u) \in \mathbb{R}^N$ is a vector of these objective functions.

For the case of visual surveillance, the choice of objective functions can include (but is not restricted to):

1. Energy consumed, which can be represented by $\|u\|^2$.
2. Resolution constraints, which can be represented as $Area\_min \leqslant \int_{\hat{K}} dp \leqslant Area\_max$.

Various methods such as weighted sum approach and physical programming can be used to solve the above optimization problem. It is proposed to use physical programming for solving the constrained multi-objective optimization problem. The significant advantage of the physical programming approach is it capability of placing the design into a more flexible and natural framework.

5.4.1 Framework for Optimal Hausdorff Tracking Design

Surveillance systems for target tracking are designed to perform tracking tasks with different requirements under varying environmental conditions. Task requirements vary even during the progress of a single task. The task criterion will vary for different kinds of task requirements. Furthermore, a dynamically varying environment implies the task criterion should be generated on-line. Thus, the new challenge for optimal task distribution for the target tracking task is dynamic modification of the task criterion based on task requirements. A new dynamic multi-objective optimal task distribution algorithm for a target tracking system employing Hausdorff tracking method proposed in [67] is developed. The algorithm's architecture is shown in Figure 5.1.

The dynamic optimal task distribution algorithm consists of three modules: task analyzer, optimization scheduling, and optimal solution calculation. The task analyzer takes current condition of motion module, target state and working environment as inputs, analyzes requirements of the current task, and generates a collection of pa-

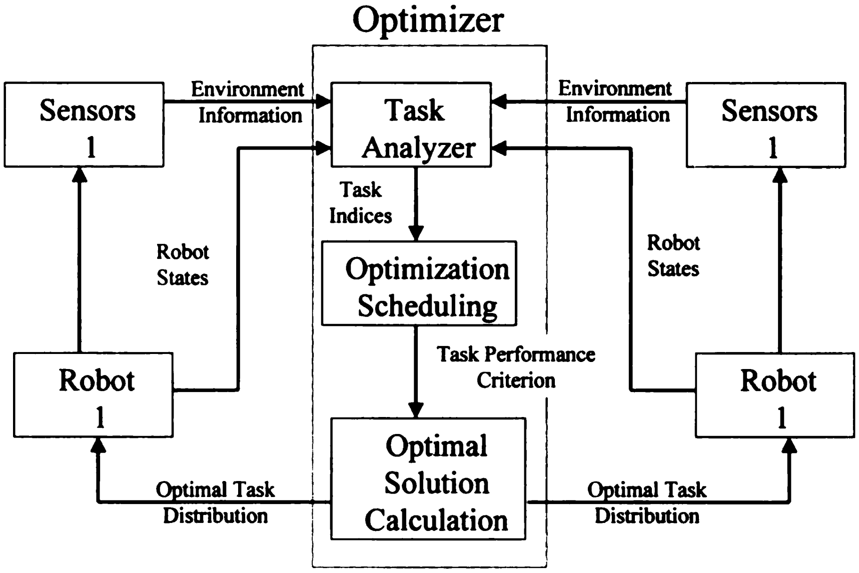


Figure 5.1: The architecture of the optimal task distribution scheme.

rameters called as Task Indices (TI) to represent task requirements. The task analyzer passes the task requirement information, TI, to the optimization scheduling module. Based on TI, the optimization scheduling module builds the task criterion function for the optimal task distribution. The optimal solution calculation algorithm then finds an optimal solution for the criterion.

The optimal task distribution system may be located anywhere on the network and can also be incorporated as a separate subsystem in one of the sensors. The current environment conditions and state of the target and various sensors are input to the optimizer over the network. The optimizer calculates the optimal solution as a task distribution amongst the various sensors and robots involved in the tracking task. This task distribution is represented as a vector of additional inputs to the various sensors and is communicated to the sensors over the network.

It should be noted that in the presence of random network delay, which will affect the timely communication of the task distribution vector, the various sensor nodes can still perform the tracking task assigned to them, albeit in a less optimized fashion. This is because the optimizer only assigns an optimal task distribution to the various sensors and not the complete tracking task.

The various parts of the optimal task distribution module are described in the following sections.

Task Analyzer

Different kinds of tasks have various requirements which depend on many factors such as the nature of the task, the targets being tracked, the location and state of the various sensors involved in the tracking task, the obstacles in the working environment *etc.*. A complex task may have varying requirements even during execution of the task. The task analyzer takes as input the various stated of the task including the location and states of the various sensors, targets and obstacles and provides a task index vector as an out which represents the requirements of the task in a concise form.

Task Indicators

For optimal performance, the task criterion function should be dynamically modified according to time-varying task requirements. An on-line task analyzer is proposed to analyze task requirements. Using the information from the sensors and the working environment as inputs, a collection of task indicators is generated online, which represent the actual conditions of the tracking task being performed. Using a fuzzy inference scheme, the task analyzer maps the task conditions represented by task indicators u, into TI, which represents task requirements. The task indicator vector

$\mathbf{u} \in R^M$ (set of all task indicators) is defined as:

$$\mathbf{u} = \{u_1, u_2, \cdots, u_M \mid \forall u_i \in U, \quad i = 1 : M\}^T. \tag{5.8}$$

where, M is the number of task indicators.

Task Indices The Task Indices (TI) vector is used to mathematically describe the task requirements. These indices are generated by a fuzzy mapping which takes the task indicators as input variables. Figure 5.2 shows the structure of the Task Analyzer. Various task requirements have their own physical and functional meanings according to which they can be categorized into classes. For example, various classes, such as maintaining the manipulability of the motion module, singularity avoidance, conserving energy, maintaining adequate target resolution, etc., can be used.

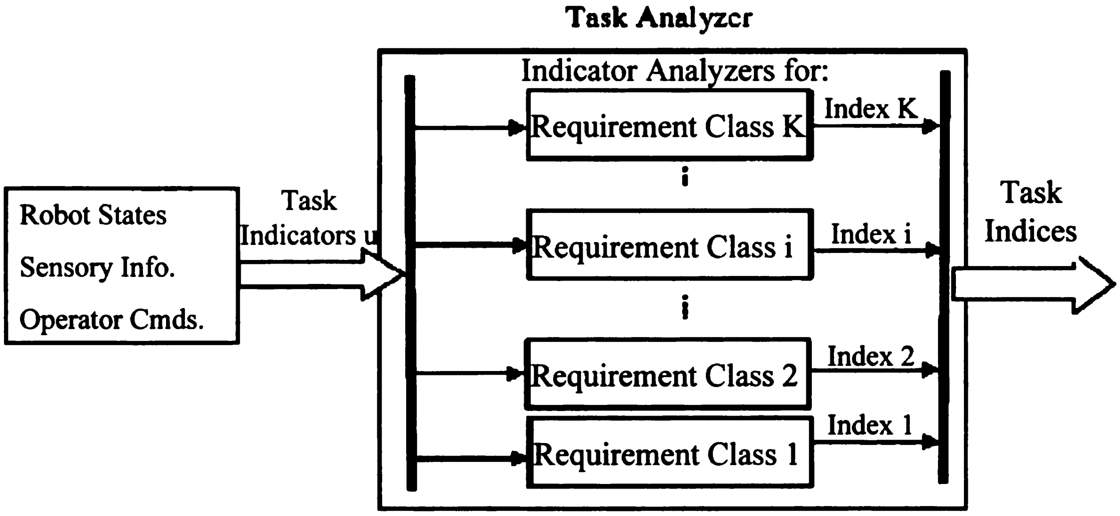


Figure 5.2: The structure of the task analyzer module.

For each class of task requirement, we generate one Task Index. For all task requirements, a vector of Task Indices is generated:

$$\begin{aligned}\mathbf{Index} &= \{Index_1, Index_2, \cdots, Index_K\}^T, \\ Index_i &= f_i(\mathbf{u}) = \frac{\sum\limits_{j=1}^{M} a_{i,j} g_{i,j}(u_j)}{\sum\limits_{j=1}^{M} a_{i,j}}.\end{aligned} \tag{5.9}$$

where $Index_i$ is the indicator analyzer for the i^{th} requirement class. $a_{i,j}$ is the relative weight of task indicator u_j in the task requirement class i, $g_{i,j}$ is the mapping function of the task indicator u_j for the task requirement class i. Equation (5.9) maps the task indicator value into the extent of task requirement. The relative weights are assigned based on the physical meaning and importance of a particular task indicator within a particular task requirement class.

Optimization Scheduling

The TI vector, generated in the task analyzer module, is the input to the optimization scheduling module. This module generates the task criterion based on task requirements represented in the TI using physical programming. The structure of the optimization scheduling module is shown in Figure 5.3. Taking vector **Index** as the input, a task criterion function is generated. This module consists of two parts: physical programming and PP (Physical Programming) analyzer.

Physical programming, which is a multi-objective optimization approach, maps the collection of objective functions into a utility function. Since the task requirements vary according to transformation of tasks and environment conditions, the mapping model for physical programming should change with the task requirements. Therefore, the PP analyzer is built to analyze the physical programming mapping model based

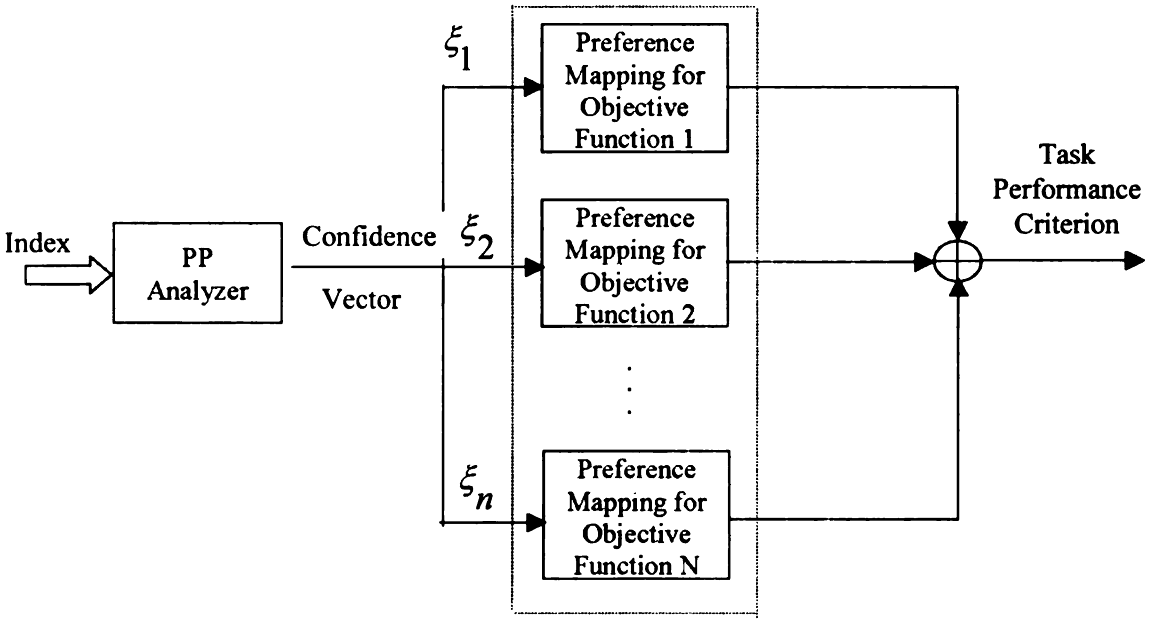

ξ_1
Preference Mapping for Objective Function 1
Task Performance Criterion
Index
PP Analyzer
Confidence
Vector
ξ_2
Preference Mapping for Objective Function 2
ξ_n
Preference Mapping for Objective Function N

Figure 5.3: The structure of the optimization scheduling module.

on the TI. A confidence vector $\Xi \in R^N$,

$$\Xi = \{\xi_1, \xi_2, \cdots, \xi_N\}^T \tag{5.10}$$

is generated to represent the degree of the system designer's satisfaction about the physical programming mapping model according to the current task's conditions. Finally, physical programming mapping is modified based on Ξ.

Physical Programming Optimal task distribution for Hausdorff tracking using multiple robots is a multi-objective optimization problem for optimizing a vector of objective functions, **D**, in equation (5.7). Usually, the system designer maps the collection of multiple objective functions C_i into a utility function $Z \in R$, which we call the task criterion, using a mapping $P : R^N \rightarrow R$ as:

$$Z = P(D_1, D_2, \cdots, D_N) \tag{5.11}$$

D_i is the objective function for the i^{th} task requirement. The optimal solution is achieved by choosing the system variables $[u, p]$ which calculates the optimal value for the criterion Z, where u is the input velocity of the various axes and p is the state of the sensors and targets.

Weighted Sum Method The weighted sum method is the most popular mapping method for multi-objective optimization problems. In this method, the task criterion is formulated as a weighted sum of the objective functions, where the system designer assigns a relative importance (weight) to each objective function. However, the task criterion obtained by this method lacks physical meaning. Misinterpretation of the theoretical and practical meaning of the weights can make the final solution unsatisfactory. Although there are many methods for choosing weights, *a priori* selection of weights does not necessarily guarantee the acceptability of the solution [81]. Furthermore, it is impossible to obtain a solution on the non-convex regions of the Pareto frontier [83].

Thus, physical programming [84], is used to generate the task criterion. Using this method, the system designers only need to specify a preference structure for each objective instead of assigning a weight which may be meaningless. This usually has more physical meaning and can better guarantee a satisfactory solution. Using physical programming, the system design can be put into a more flexible and natural framework. Another advantage of physical programming is that it can obtain the solutions in the non-convex regions of the Pareto frontier [83].

Preferences Mapping Using the physical programming approach, objective functions are mapped into a preference space. The preference is a parameter which represents the extent of the designer's satisfaction. The task criterion is set up as the aggregation of the preferences of all the objective functions. Thus, instead of being physically meaningless, the criterion becomes a satisfaction factor of the solution.

The designer's expression of the preferences with respect to each objective function can be categorized into four different classes: smaller-is-better (1S), larger-is-better (2S), value-is-better (3S), range-is-better (4S) as in Figure 5.4. According to Messac's research [85], the mapping is depicted as Figure 5.4. In this Figure, μ_i is the value of the ith objective function, P_i is preference, a real positive scalar, of the ith objective function. Each objective belongs to one of these classes. [83–85] provide a more deeper analysis of physical programming.

According to the degree of the desirability, the objective function values are categorized into several regions as seen in Figure 5.4. Taking the case of Class 1S as example, the preference ranges are:

Ideal range	$(\mu_i \leq v_{i1})$
Desirable range	$(v_{i1} \leq \mu_i \leq v_{i2})$
Tolerable range	$(v_{i2} \leq \mu_i \leq v_{i3})$
Undesirable range	$(v_{i3} \leq \mu_i \leq v_{i4})$
Highly Undesirable range	$(v_{i4} \leq \mu_i \leq v_{i5})$
Unacceptable range	$(v_{i5} \leq \mu_i)$

The parameters, v_{i1} to v_{i5}, are physically meaningful constants which express the designer's preference associated with the ith objective. For physical programming, higher the desirability of the objective, the lower the value of preference. Therefore, physical programming problems are always of the minimization type.

The class functions for preference mapping should have several important properties such as nonnegativity, continuity and convexity. Further, the preference value P at range intersections (e.g., Tolerable-Undesirable) is the same for all class types and objectives.

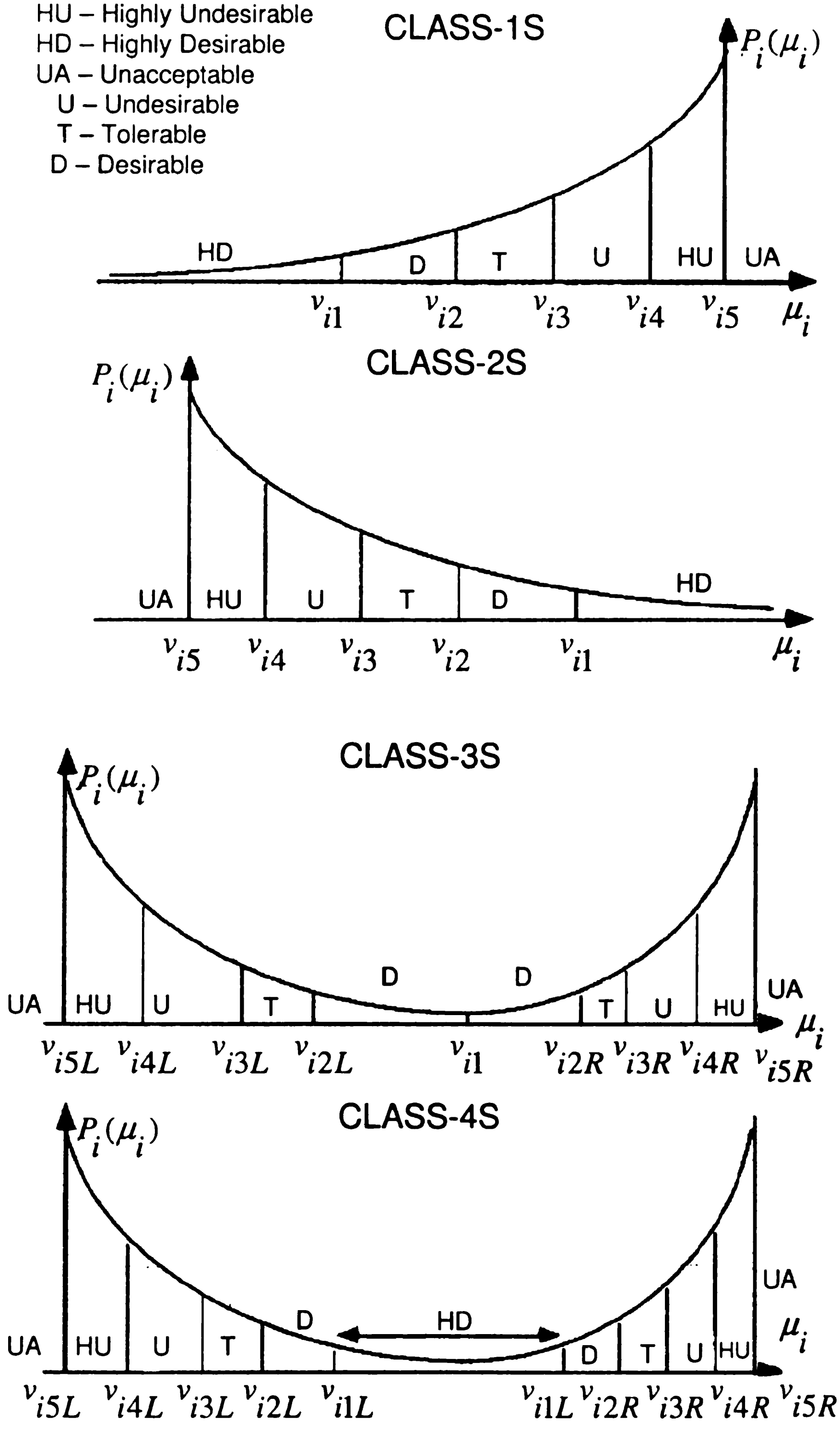


Figure 5.4: Class functions for physical programming.

Physical Programming Formulation Based on the preference mapping of the objectives, physical programming problems can be stated as:

$$\min_{\mathbf{x}} Z = \sum_{i=1}^{M} P_i \left(g_i(\mathbf{x}) \right)$$

subject to:

$$\begin{array}{ll} g_i(\mathbf{x}) \leq v_{i5} & \text{(for class 1S)} \\ g_i(\mathbf{x}) \geq v_{i5} & \text{(for class 2S)} \\ v_{i5L} \leq g_i(\mathbf{x}) \leq v_{i5R} & \text{(for class 3S and 4S)} \end{array} \tag{5.12}$$

where $\mathbf{x}$ is the system variables, M is the number of the objective functions, $g_i(\mathbf{x})$ is the ith objective function, P_i is the preference class function of g_i.

Physical Programming Analyzer (PP Analyzer) The preference structure (say, class type and v_1 to v_5) for each objective function must be specified to set up the global criterion function. However, due to changing task requirements, the initially specified preferences for the objectives may not represent the objective's desirability accurately at later time instants. Therefore, physical programming should be modified according to the task requirements.

A PP analyzer module is built to handle this problem. The PP analyzer takes the Task Dexterity Indices as inputs, and generates a vector $\Xi = [\xi_1, \cdots, \xi_M]^T$ to represent the degree of designers' confidence about the preference structure specified for the objective function. It passes this confidence vector to the physical programming module. The latter module will modify its result according to the confidence parameters. The analyzer is formulated as:

$$\xi_i = (a_i + k_i \times index_i) \quad i = 1, \cdots, M \tag{5.13}$$

where a_i is a constant which represents the system designer's confidence about the original preference mapping function, k_i is a constant. The criterion generated by physical programming will be modified as:

$$Z = \sum_{i=1}^{M} \xi_i \times P_i\left(g_i(x)\right) \tag{5.14}$$

Equations (5.13) and (5.14) show that when $index_i$ increases, the preference value of the i^{th} objective function will increase. Thus, when the task has high preference on the requirement class i, the objective function D_i should have a higher preference for further minimization. Thus, the task requirement condition governs the path of optimization.

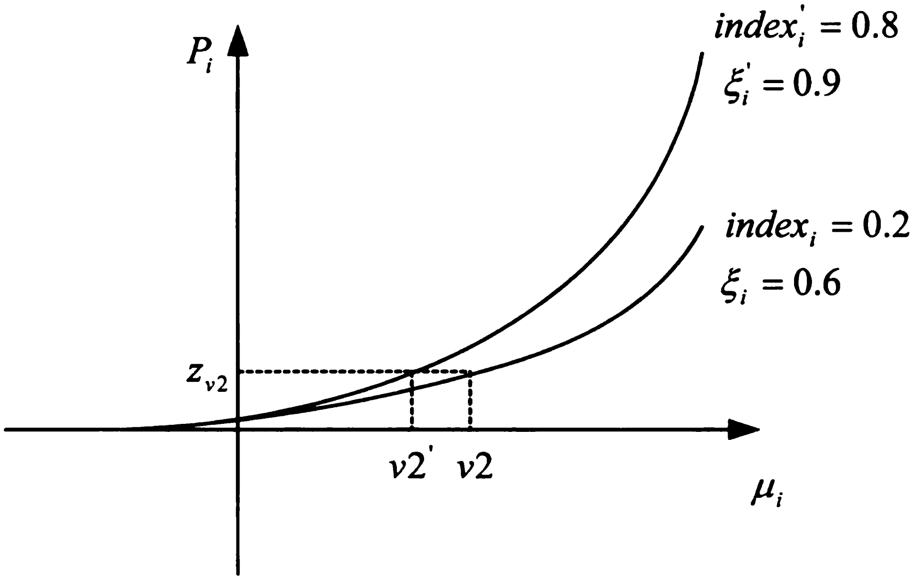


Figure 5.5: Modification of the preference mapping by confidence factor.

From another viewpoint, multiplying a factor with the preference mapping function can be treated as modifying the structure of the mapping. Taking the class 1S as an example (from Figure 5.5), if the preference value at the range intersection (say z_{v2} Desirable-Tolerable) is kept unchanged, the desirability ranges of the objective function change with the task index (confidence factor). In this Figure, $index' > index$

will induce $v2' < v2$ which makes the Tolerant range closer to the ideal range.

Therefore, combining equations (5.7), (5.12), and (5.13), and replacing g_i by D_i, and $\mathbf{x}$ by u, p, the optimal Hausdorff tracking problem is formulated as:

$$\min_{u,p} Z = \sum_{i=1}^{M} \xi_i \times P_i\left(D_i(u,p)\right) \tag{5.15}$$

subject to

$$\begin{aligned} & -\alpha J \geq Cu & & \\ & D_i(u,p) \leq v_{i5} & & \text{(for class 1S)} \\ & D_i(u,p) \geq v_{i5} & & \text{(for class 2S)} \\ & v_{i5L} \leq D_i(u,p) \leq v_{i5R} & & \text{(for class 3S and 4S)} \end{aligned} \tag{5.16}$$

where, α is a positive definite gain and J is the vector of shape functions described for a specific task. The system constraint equation $-\alpha J = Cu$ is derived from the Hausdorff tracking task controller derived in [67, 68].

Optimal Solution Calculation

The next step is to find the optimal solution for the optimization problem specified by equations (5.15) and (5.16. In this formulation the system constraint $-\alpha J = Cu$ takes the form of the linear equation. Therefore, we can use the generalized elimination method [86] to eliminate the system dependent variables: suppose that there exist matrices $A_{n\times m}$ and $B_{n\times(n-m)}$ such that $[A\ B]$ is non-singular. Usually, for the single target Hausdorff tracking task, $n = 6$ and $m = 1$. If matrices A and B satisfy the following equations:

$$\begin{aligned} CA &= I \\ CB &= 0, \end{aligned} \tag{5.17}$$

the system solution is formulated as:

$$u \leqslant -\alpha AJ + B\epsilon \tag{5.18}$$

where, ϵ denotes the Null space of the matrix C.

The reduced optimization problem is shown as:

$$\min_{\epsilon} Z = \sum_{i=1}^{M} \xi_i \times P_i\left(D_i(u,p)\right) \tag{5.19}$$

subject to

$$\begin{aligned} &D_i(u,p) \leq v_{i5} \text{ (for class 1S)} \\ &D_i(u,p) \geq v_{i5} \text{ (for class 2S)} \\ &v_{i5L} \leq D_i(u,p) \leq v_{i5R} \text{ (for class 3S and 4S)} \end{aligned} \tag{5.20}$$

$$\begin{aligned} \text{where,}\quad u &\leqslant -\alpha AJ + B\epsilon \\ p &= \bar{p} + u/f \end{aligned} \tag{5.21}$$

$\bar{p}$ is the initial value of the system variables p, f is the control frequency, ϵ is and arbitrary vector and A and B are as defined in Equation (5.17). Using an online pattern search algorithm [82], the system can easily obtain the optimal solution.

5.5 Switched Video Feedback

Video feedback is an essential component of the surveillance system. The task of capturing, transmitting and displaying the live video stream from the various sensors to the requesting clients is handled by the video subsystem. Since multiple cameras are deployed to track the identified targets, multiple, concurrent feedback video streams may be required for monitoring the target. These sensors initiating these streams will be changing from time to time as the target moves out of range of the current sensors tracking it. However, providing multiple unnecessary (unrelated to the task)

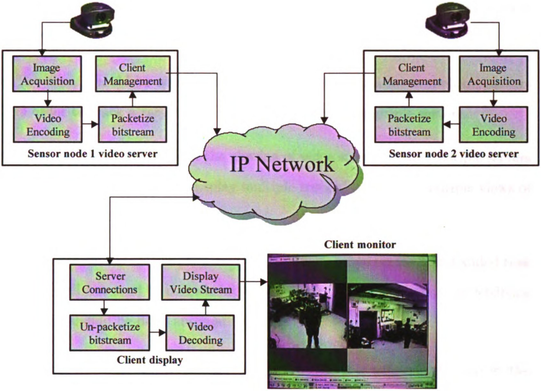


Figure 5.6: Switched video architecture implementation

video feedback streams often causes loss of attention span of the operator and makes it hard to keep track of the various activities over the cameras. Hence only video streams from relevant sensors should be presented to the operator on a per-activity basis. This is done through automatic or manual switching of the camera streams that are presented to the operator.

The system is designed to support video feedback to multiple clients over an IP network. The video service is designed to support various types of live video stream feedback from the sensors to the individual clients such as MJPEG and H.263. Various resolutions such as CIF, QCIF and 4CIF are supported for the live video streams.

A switched video client and server architecture have been implemented as shown in Figure 5.6.

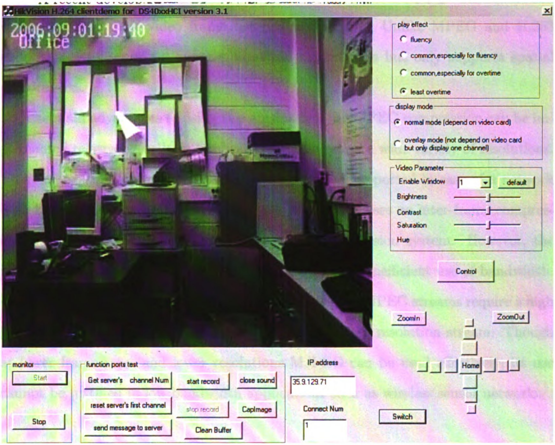
The various properties and tenets of the design of the switched video feedback system are

- The client should be able to switch the video stream between multiple servers in order to track a target as it ’handed-off’ between multiple servers.
- The client should also afford the viewing of video streams from multiple servers simultaneously in order to display multiple tracked targets or multiple views of the same target.
- The client should receive selected video feedback based on the commanded task which will not tend to overwhelm the operator. This selection of the feedback streams should be done automatically.

In order to transmit real-time video over a network, it is necessary to convert the output of the encoder into a packet sequence - i.e., packetize it. Traditional TCP and UDP services are not sufficient for real-time applications, and applications-oriented protocols such as the Real-time Transport Protocol RTP provide an alternate solution for managing the essential tradeoffs for quality and bandwidth. RTP provides end-to-end delivery services such as payload type identification, sequence numbering and time stamping.

The proposed architecture has been implemented on a PC running Linux OS. The Qt toolkit [87], which is a complete C++ application development framework, including a class library and tools for cross-platform development, has been used to develop the client and application. A Windows-only client has also been implemented using Microsoft Visual Studio 2005. The current implementation of the user interface is shown in Figure 5.7.

We have implemented the MJPEG, H.261, H.263 and H.264(hardware based) video compression schemes which are packetized using RTP/UDP transport protocol



play effect
○ fluency
○ common,especially for fluency
○ common,especially for overtime
● least overtime

display mode
● normal mode (depend on video card)
○ overlay mode (not depend on video card but only display one channel)

Video Parameter
Enable Window: 1
Brightness
Contrast
Saturation
Hue

IP address: 35.9.129.71
Connect Num: 1

Figure 5.7: H.624 Based Switched Video Streaming Client.

for transferring the surveillance video to the client. Characteristics and implementation details of the video subsystems implemented are summarized in the following discussion.

5.5.1 Characteristics and Implementation Details for MJPEG

MJPEG stands for Motion-JPEG and is a technique which simply performs a JPEG compression on each video frame before transmission. Unlike MPEG and H.263 codecs, MJPEG does not support temporal compression but only supports spatial compression.

The main advantages to using this approach is that JPEG compression can be implemented relatively easily in hardware and it supports a wide variety of resolutions. This implies that a wide variety of hardware can be supported in case of a network with heterogenous video capture hardware. Further, it uses no inter-frame compression which results in low latency in the video transmission system. However, the major disadvantage for using MJPEG technology is its inefficient use of bandwidth. Due to the lack of inter-frame (temporal) compression, MJPEG streams require a high bandwidth of the order of 2 Mbits/s for a 30 fps NTSC resolution stream. Though at lower frame rates and lower resolutions MJPEG can be used effectively, its use cannot be justified in low bandwidth applications such as wireless sensor networks.

5.5.2 Characteristics and Implementation Details for H.261

H.261 is a video coding standard published by the International Telecommunications Union (ITU) [53]. It supports CIF (352×288 pixels) and QCIF (176×144 pixels) resolutions. It supports both temporal and spatial compression for reducing the size of the encoded image. The coding algorithm is a hybrid of inter-picture prediction, transform coding, and motion compensation. The data rate of the coding algorithm was designed to be able to be set to between 40 Kbits/s and 2 Mbits/s. The compressed video stream is structured into a hierarchial bitstream consisting of four parts, namely: (1) Blocks which correspond to 8 × 8 pixels; (2) Macro blocks which correspond to 16 × 16 pixels of luminance and two 8 × 8 pixel chrominance components; (3) Group of blocks (GOB) which corresponds to 1/12 of CIF picture or 1/3 QCIF picture; (4) Picture layer which corresponds to one video frame.

The H.261 standard actually only specifies how to decode the video stream. Encoder designs are constrained only such that their output can be properly decoded by any decoder meeting the standard spec [53]. In our implementation, based on the Vic software, the H.261 bitstream is encoded such that macro blocks that have changed significantly from the previous frame are updated in the current frame. Also, each macro block is updated at least once every 32 picture frames using a block aging process.

5.5.3 Characteristics and Implementation Details for H.263

H.263 is video coding standard by ITU [54]. It was designed for data rates as low as 20 Kbits/s and is based on the ITU H.261 standard. It supports 5 resolutions (CIF, QCIF, sub-QCIF, 4CIF and 16CIF, where CIF is standard 352×288 pixels resolution). Like H.261, it uses both temporal and spatial compression and also provides for advanced coding options (not included in H.261) such as unrestricted motion vectors, advanced prediction and arithmetic coding (instead of variable length coding) for improvement of video quality at the expense of video codec complexity. It allows for fixed bit rate coding for transmission over a low bandwidth network as well as variable bit rate coding for preserving a constant image quality and frame rate for storage and transmission over high bandwidth networks.

Each picture frame can be coded as either an 'I'(intra), 'P'(predictive), 'B'(bidirectional) or 'PB' frame. 'I' frames are intra-coded reference frames which contain all the information needed to initialize the decoder and display the image. 'P' and 'PB' frames are inter-coded frames which require information from other frames at the decoder in order to decode them. In our implementation for the H.263 codec, which is based on the Vic software, a completely intra-coded 'I' frame is transmitted every 10 seconds.

5.5.4 Characteristics of H.264 / MPEG4-Part10

A recent development for video coding is H.264/MPEG-4 Part 10 [55], also named Advanced Video Coding (AVC), which is jointly developed by ITU and ISO. H.264/MPEG-4 supports video compression (coding) for realtime video delivery over low bandwidth networks. In contrast to the frame based coding in MJPEG, H.261 and H.263+ schemes, the coding in MPEG4 streams is object based where each scene is composed of objects, which are coded separately. This object based coding structure can be effectively used for video surveillance systems [88]. However, current software implementations of the coding scheme have been tested to have best-case capture and compression latency of the order of 500ms - 1s which precludes their use in a realtime video scenario.

We have integrated a commercially available hardware based H.264-baseline and H.264-enhanced encoder in our switched surveillance implementation which supports a frame resolution of up to 1080×768 pixels^2 at a frame rate.of up to 30 Frames/sec. The hardware encoder supports constant bit rate encoding as well as variable bit rate encoding. For a frame rate of 30 fps and an image resolution of 800×600 pixels^2 with variable bit rate encoding and a tracking scenario involving camera motion, the baseline encoder has a bit rate of approximate 500 Kbps. The interface developed is shown in Figure 5.7.

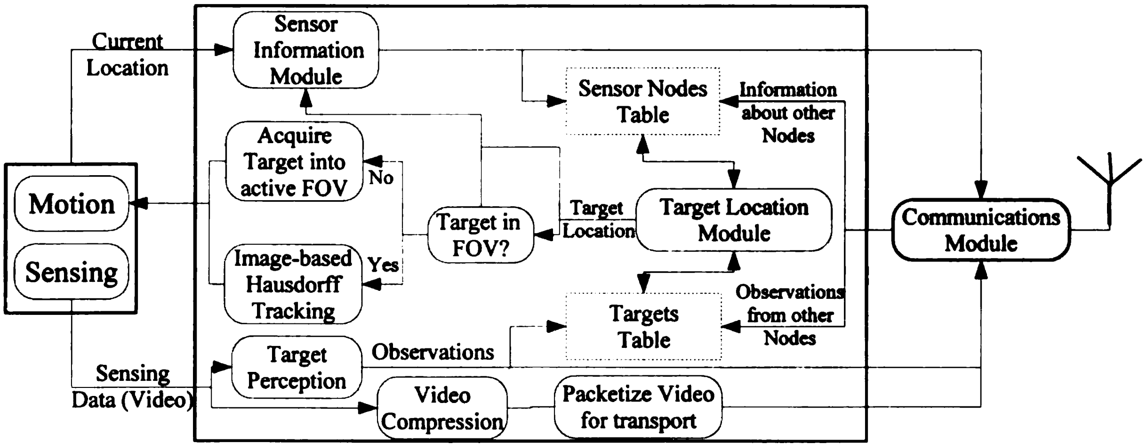


Figure 5.8: Architecture of Sensor Node.

5.6 Sensor Node Architecture

Figure 5.8 depicts the general architecture of a sensor node. The individual sensor nodes maintain information regarding the observations of their neighboring nodes and broadcast (within their locality) their own observations. Based on the combined observations, each node develops a list of targets being actively tracked and the status of its peer nodes and stores this information in the targets table and the sensor nodes table, respectively. In the targets table, the native as well as observed characteristics of the target objects, observed by the respective sensors, are stored. The targets table also stores information indicating the node that sensed these characteristics. Nodes also store peer information, such as location, active FOV and total capable FOV of the peer.

When the target to be tracked is outside the active FOV of the sensor (but in the capable FOV), the node can still ensure that the target is acquired in the active FOV using target location information available from other sensors in the targets table. This paper proposes the method of cooperative Hausdorff tracking for deriving assumptions on the input u to the robot/camera to bring the target into the active FOV of the sensor. Using the cooperative Hausdorff tracking method the target can be brought into the active FOV of the sensor, and assuming that the visual characteristics of the target can be recognized, the sensor will switch over to image-based Hausdorff tracking to maintain the target in the active FOV.

The sensed video information is broadcast to the requesting clients using the video subsystem, which consists of a video server that compresses and transmits the captured video using either MJPEG or H.263 bit stream mounted over the RPT/UDP/IP transport protocol.

5.7 Modeling Switched Video Feedback as a Dynamic System

The configuration of the cameras $\mathcal{C}$ can be defined as the collection of all the parameters of the cameras involved. That is:

$$\mathcal{C} = \{C_1, C_2, \ldots, C_N\} \tag{5.22}$$

where, $C_i \in \mathbb{R}^p$ is a vector containing all the intrinsic and extrinsic parameters of the camera. The words "layout" and "placement" will also be alternatively used with camera configuration but they will reflect the internal parameters of the camera, too.

Consider a monitored region as a domain $E \subset \mathbb{R}^n$ under surveillance. A finite number (N) of cameras is distributed in this region and involved in the surveillance task of maintaining visibility of a target as it moves within the monitored region. The monitored region E can be discretized on a finite-dimensional grid $\mathcal{G} = \{V_i | V_i \in \mathbb{R}^n, \text{i=1,..,g}\}$ which consists of a finite number, g, of vertices V_i. Note that grid $\mathcal{G}$ can be generated from the domain E using various approximate cell decomposition methods where the cells have a pre-defined shape and size in order to achieve a certain resolution.

Consider any continuous-time target trajectory in two dimensions $k_c(\tau) : \Gamma_c \mapsto \mathbb{R}^2$. For every $\tau \in \Gamma_c$, $k_c(\tau)$ provides as an output the location of the target. This continuous time trajectory can be approximated as a discrete-time finite-dimensional map on a grid as

$$k(t) : \Gamma \mapsto \mathcal{U}_{\mathcal{G}} \tag{5.23}$$

where, $\Gamma = \mathbb{Z}$ and $\mathcal{U}_{\mathcal{G}}$ represents the finite-dimensional space of the locations of the vertices V_i of the grid $\mathcal{G}$.

The dynamic sensor switching problem can be modeled as a discrete-time, finite-dimensional dynamic system. The state space of this system is the finite set of camera states defined as: $\mathcal{Q} = \{q_i | q_i = (i, C_i), i \in (1, ..., N), C_i \in \mathcal{C}\}$ where N is the total number of cameras and $\mathcal{C}$ is space of configurations of all cameras.

Using the above definitions we can now describe the model of a dynamic sensor switched system as a finite-dimensional discrete-time dynamic system $\mathcal{V}$:

$$\mathcal{V} = [\mathcal{Q}, \Gamma, \mathcal{U}, \phi]$$

$$\phi : \mathcal{Q} \times \Gamma \times \mathcal{U}_{\mathcal{G}} \times \mathcal{U}_1 \mapsto \mathcal{Q} \tag{5.24}$$

where $\mathcal{U} := \mathcal{U}_{\mathcal{G}} \times \mathcal{U}_1$. Here, $u \in \mathcal{U}_1$ is the vector of control inputs while $k(t) \in \mathcal{U}_{\mathcal{G}}$, $t \in \Gamma$, the sequence of grid locations in $\mathcal{G}$ that the target visits, are the reference inputs to the system. The reference inputs evolve according to a predefined function of time t and depend only on the target motion, which may not be known *a priori*. The control inputs are used to control the trajectory of the system to ensure the target is covered and can also be used for optimization of the various metrics being used.

The system $\mathcal{V}$ may not be complete, as all the transitions between various cameras may not be enabled at all times, based on the visibility constraints of the various states for the target locations provided.

5.7.1 *Examples of Video Feedback Algorithms - Best Resolution*

Consider a set of N cameras $\mathcal{Q} = \{(i, C_i) | i = 1, ..., N \;\; C_i \in \mathcal{C}\}$. The sensor selection strategy is dependent on the resolution sustained by the target at each of the cameras and the one with the best resolution will be selected to provide feedback to the human operator.

This sensor selection strategy takes into account only the best resolution sustained at all the cameras, which implies that the control input u is based solely on the camera configurations $\{C_i\}$ and the target locations $k(t)$. The strategy does not depend on the current camera tracking the target - i.e., sensor switching costs are not taken into account.

In order to implement this strategy in the dynamic systems model, we define a function $R_{res} : \mathcal{U}_G \times \Gamma \mapsto \mathcal{U}_1$ which maps the locations of the target at various time instances to the space of the control input variables u based on the resolution sustained by the target at each camera. Notice that R is not a function of the current state of the dynamic system - i.e., the current camera. This implies that the system does not have memory.

The above algorithm can be implemented as follows. Let q_{best} represent the camera that has the least distance to the target; i.e., under the assumption of homogenous camera sensors, q$_{best}$ can view the target with the best resolution. For a given target location $k(t)$ and time t, q_{best} can be written as:

$$q_{best}(t) = q_i | \min_{i=1toN} \text{dist}(C_i(t), k(t)), \text{visible}(q(t), k(t)) == 1 \qquad (5.25)$$

Using the definition of q_{best} the next sensor to switch to i.e., $q(t+1)$ can be written as: $q(t+1) = q_{best}(k(t+1))$.

5.7.2 *Example Video Feedback Algorithm - Persistent Camera*

In the best-resolution-based video feedback algorithm, the current camera tracking the target was not taken into account in the input to the dynamic system. Switching cameras in a surveillance network can lead to the disorientation of the human operator and is also generally associated with a switching time delay. Hence we should minimize the number of switches.

In order to implement this strategy, define a function $R_{per} : \mathcal{Q} \times \mathcal{U}_{\mathcal{G}} \times \Gamma \mapsto \mathcal{U}_1$ which maps the current tracking camera and the locations of the target at various time instances to the space of the control input variables u. This implies that the state of the system, $q(t)$, is used along with the resolution in order calculate the control input to the dynamic system.

The above algorithm can be implemented using the definition of q_{best} from the previous section. The sensor selected at the next time step $q(t+1)$ can be written as:

$$q(t+1) = \begin{cases} \min(\text{dist}(C_{best}(t+1), k(t+1) + \epsilon, \text{dist}(q(t), k(t+1))), \\ \qquad \text{if visible}(q(t), k(t+1)) = 1 \\ \qquad q_{best}(t+1) \\ \qquad \text{if visible}(q(t), k(t+1)) = 0 \end{cases} \tag{5.26}$$

5.7.3 *Assessment Metric for Video Switching Based on Camera Locations*

In sections 5.5 and 5.5, a switched video feedback system for realtime video feedback to the human operator to assist in the monitoring and decision making task has been described. However, the constant switching of the video stream can be quite disruptive for human cognizance of the remote environment and adversely affect decision making. Hence the system should try to reduce the number of video feed switches required for the same tracking task.

The video feedback switch is based on the capability of the tracking cameras to visually resolve the target and discern its features - i.e., the resolution of the target sustained by the cameras. This switching is directly influenced by the locations of the cameras in the scene. In the case of manually deployed camera configurations, the system designer must take into consideration the adverse effects of video switching and must decide the camera locations in order to minimize the switches while continuously tracking the target. In this section we propose a metric as a performance measure of the video feedback switching algorithm, based on the configuration of the cameras.

The switching metric $M \in \mathcal{R}_+$ maps the configuration space of the video algorithm [89] (which includes the previous feedback camera) and a scalar potential field over the configuration space of the video algorithm to a positive scalar. Given the configuration of the cameras $\mathcal{C}$, the video algorithm can be thought of as a mapping from the combined space of current feedback camera and target location to the feedback camera space. The scalar potential $\varphi(p) \in \mathcal{R}_+, p \in E$ provides a relative importance to the current target location and can be chosen to bias the importance of the target locations.

$$M = \int_F \nabla\mathcal{V}(f,\mathcal{C})\ \varphi(p)\ df \tag{5.27}$$

where, $f \in F \subset \mathcal{Q} \times E$ and $p \in E$. $\nabla\mathcal{V} \in [0,1]$ is the spatial derivative of the output of the video feedback algorithm and represents at which points in the video feedback configuration space F the output of the video feedback algorithm changes. It is akin to edge segmentation in image processing [90]. Integration of these points over the configuration space represents the number of switching surfaces present in the complete space E. A lower number of switching surfaces will imply that the feedback camera location does not switch a lot with the free motion of the target. The term $\varphi(p)$ is just a scalar potential which reflects the importance of the particular point $p \in E$ and can be used to bias the surveillance space E.

The switching metric M in conjunction with other metrics such as target resolution at various locations in E can be computed for a large number of randomly generated configurations and a sub-optimal solution can be derived for the placement of the cameras. Owing to the large configuration space of the deployment algorithm, various evolutionary computation and optimization schemes can be utilized in order to minimize the total placement metric.

5.8 Optimization for Target Tracking using Switched Sensors

Given the scenario having a large number of static cameras distributed in an environment with significant overlap of their viewing regions, the problem is to identify the minimum number of cameras required to track a given target trajectory.

We propose to use dynamic programming as an optimal control strategy in order to minimize the total number of cameras required to view the target with adequate resolution.

5.8.1 Optimal Control and Dynamic Programming

Consider a finite dimensional discrete-time dynamic system $\mathcal{V}$ and function $b : \mathcal{X} \times \Gamma \times \mathcal{U} \mapsto \mathbb{R}_+$ that takes on non-negative real values. We can now define a trajectory cost function as:

$$\mathcal{B}(\tau, \sigma, x, \omega) = \sum_{i=\sigma}^{\tau-1} b(i, \xi(i), \omega(i)) \tag{5.28}$$

where, ω is a sequence of inputs and $\xi = \varphi(x, \omega)$ is a sequence of states of the dynamical system given the initial state x and a sequence of inputs ω. The trajectory cost functions is read as "the cost of the trajectory produced by the control sequence $\omega \in \mathcal{U}^{[\sigma,\tau)}$ if we start at state x."

An optimization problem can be stated as: Given a discrete-time finite dimensional dynamic system $\mathcal{V}$, a trajectory cost function $\mathcal{B}$ and a pair of times $\sigma < \tau$, and an initial state $x(\sigma)$ find a control sequence w admissible for state $x(\sigma)$ which minimizes $\mathcal{B}$.

More generalized problems can be introduced which require the total cost to be a function of the final state or by adding constraints to the final state, etc.

In order to find a control input sequence ω that minimizes the trajectory cost function $\mathcal{B}$, we could list all possible control input sequences $\omega = u_\sigma, u_{\sigma+1}, ...u_\tau$ which are elements of $\mathcal{U}$ and compute the complete cost of all the trajectories generated by all $\omega \in \mathcal{U}^{[\sigma,\tau)}$. This could indeed entail a prohibitively large computation cost.

Alternately, we could use the dynamic programming method and inductively construct the Bellman function V and the optimal control input law K, backwards in time i.e, from τ towards σ as:

$$V : [\sigma,\tau] \times \mathcal{X} \mapsto \mathbb{R}_+ \tag{5.29}$$

$$K : [\sigma,\tau) \times \mathcal{X} \mapsto \mathcal{U} \tag{5.30}$$

The Bellman function $V(s,x)$ should satisfy for any $s \in [\sigma,\tau]$, and each $x \in \mathcal{X}$,

$$V(s,x) = \min_{\omega} \mathcal{B}(\tau,\sigma,x,\omega) \tag{5.31}$$

and the optimal control input satisfies the condition

$$\begin{aligned} \xi(j+1) = \phi(\xi(j), j, K(j,\xi(j))),\ j = s, s+1, ..., \tau \\ \xi(s) = x \end{aligned} \tag{5.32}$$

for each $s \in [\sigma, \tau)$ and each $x \in \mathcal{X}$.

The computation effort required to solve this problem will be significantly lower than tabulating all the control input sequences. However storage requirements for this procedure will be significantly large.

5.8.2 Dynamic Programming Algorithm Implementation

In order to minimize the switching cost and maintain adequate resolution of the target, we construct a graph based on the camera location and the visibility of the targets to the cameras, given the predicted motion of the target, Figure 5.9. The procedure for constructing the graph is shown in Algorithm 1. After the graph is constructed, we use the dynamic programming method, a modified version of the Dijkstra algorithm, to find the optimal switching strategy (Algorithm 2). In the graph generating stage, we enumerate all the cameras that can see the current grid point with an acceptable resolution. The resolution metric is marked over each camera as a weight on the vertex. However, in order to run the dynamic programming method, it is desirable to have all the weights only on the edges. Thus each camera is represented by two vertices in the graph, with an edge connecting them that has a weight equaling to the resolution metric. We also enumerate all the cameras that can see the grid points specified by the prediction vector. Between two prediction grid points we connect the cameras based on their switching metric. If the two cameras are the same, the switching metric is set to 0, otherwise a non-zero switching metric is set on the edge. In Figure 5.9, all non-zero metrics are set to 1 for the sake of simplicity.

5.9 Chapter Summary

This chapter presents the design of the various components of a surveillance network. For exploiting the redundancy in motion along the various axes of the sensor, an

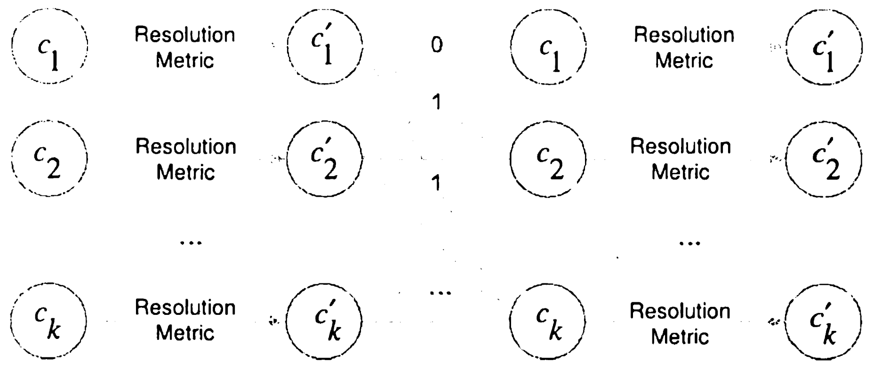


Figure 5.9: Generating Graph for the Camera Switching Strategy

Algorithm 1 $[G, W] = \text{GraphGen}(p_v)$

```
for all cam in CameraSet do
    if $p_v(1)$ is visible by cam then
        Add two nodes representing cam to $G$
        Connect the two nodes with the resolution metric of cam in $W$
    end if
end for
$p_{prev} = p_v(1)$
Delete $p_v(1)$ from $p_v$
for all $p$ in $p_v$ do
    for all cam in CameraSet do
        if $p$ is visible by cam then
            Add two nodes representing cam to $G$
            Connect the two nodes with the resolution metric of cam
            Connecting the nodes generated by $p$ and $p_{prev}$ and set the switching cost
            to $W$ if they are for different cameras.
        end if
    end for
end for
```

optimization framework is proposed. This framework allows the accomplishment of various other tasks under the constraints of the main tracking task, such as energy conservation and better sensor location for sensing.

The characteristics and implementation details of the various choices for video feedback are discussed and finally the design of a sensor node is presented. The architecture of the sensor node consists of a camera, a target perception module, a motion control module, a video feedback module and a communications module.

Algorithm 2 $p = \text{Dijkstra}(G, W, s, t)$

```
for all $v$ in $V(G)$ do
  $d(v) \leftarrow +\infty$ {Initialize the distance vector}
  $p(v) \leftarrow$ undefined {Initialize the previous node vector}
end for
$d(s) \leftarrow 0$ {The source has 0 distance to itself}
$C \leftarrow \emptyset$ {Initialize the checked node set}
$Q \leftarrow V(G)$ {Copy the vertex set into a working set}
while $Q \neq \emptyset$ do
  $u \leftarrow ExtractMin(Q)$ {Extract the vertex with minimum value in $d$}
  $C \leftarrow C \cup \{u\}$
  if $u = t$ then
    return
  end if
  for all edge $(u, v)$ do
    if $d(u) + W(u, v) < d(v)$ then
      $d(v) \leftarrow d(u) + W(u, v)$
      $p(v) \leftarrow u$
    end if
  end for
end while
if $!t \in C$ then
  $p \leftarrow \emptyset$
end if
```

Chapter 6

Experimental Implementation and Testing

Figure 6.1 shows the general architecture of the surveillance system implemented over an IP network. This chapter presents a discussion of the implementation issues such as switched video feedback to the human operator/client and a performance analysis of the experimental results of the proposed algorithms.

6.1 Experimental Test-bed Setup

A visual surveillance test-bed comprising of seven video cameras has been set up at the Robotics and Automation Laboratory at Michigan State University. The test-bed is used to demonstrate the integration of multiple active sensors with active target tracking algorithms to perform a coherent pervasive surveillance task of tracking multiple targets as they move across the monitored landscape. The cameras are attached to processing and communication computers and some are mounted on pan-tilt drives or robots for moving the cameras. The surveillance test-bed developed has the functionality of an end-to-end, multi-camera monitoring system that allows a single or multiple human operator(s) to monitor activities in the region of surveillance.

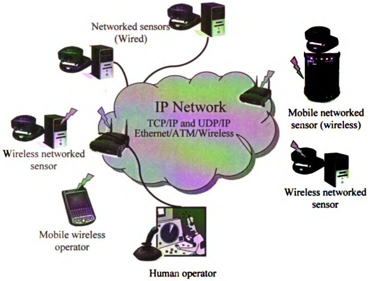


Figure 6.1: Surveillance network implementation over an IP network.

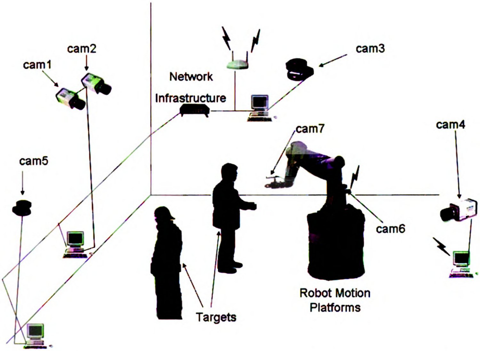


Figure 6.2: Surveillance test-bed designed along with the cameras and the motion modules for camera motion.

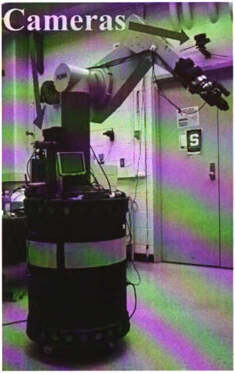


Figure 6.3: Sony EVI-D30 mounted on Robot Nomad XR-4000

Figure 6.2 shows the test-bed setup. This section describes the hardware and software architecture of the individual sensors used.

6.1.1 Hardware Setup

The sensor node setup consists of three Sony EVI-D30 active PTZ (pan-tilt-zoom) cameras, one dual head stereo vision camera consisting of two Hitachi KPI-D50 cameras mounted on a Digital Perception DP-250 pan-tilt drive and one fixed view KPI D-50 camera. One of the Sony EVI-D30 cameras was mounted on a Nomad XR 4000 mobile robot for extended mobility and sensing range. Figures 6.3 and 6.4 show the sensor and motion hardware for the implemented system.

The cameras were connected to Pentium 4 2.4 GHz computers which had PCI-based video capture hardware cards attached to them. The camera capture cards are

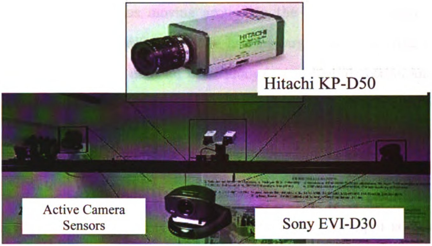


Figure 6.4: Active Cameras: Sony EVI-D30 and dual head Hitachi KP-D50

capable of transcoding, in real time, the input from the cameras to an H.264 stream (on a separate software interface) and transmiting that stream over the Internet to clients that request the video stream. The PTZ modules for the various cameras were controlled through serial port communications. The various computers were connected to each other using wired ethernet and wireless 802.11g connections over an IP network. The individual sensor nodes were provided with publicly addressable IP addresses and hence could be accessed from the Internet. The human interface clients were comprised of Pentium 4 laptop computers which could be connected to the surveillance network through a direct wired or wireless connection or through the Internet.

6.2 Software Setup

The sensor outputs are processed on the local computer for target detection and identification. The local computers used for sensing and processing are running a Linux Operating system. The targets to be tracked using the automated tracking system were multiple humans moving around in the monitored region. For ease of target detection and identification, the human targets were wearing solid color clothing and *CMVision* was used for color analysis and blob detection and merging [91]. The acquired 640×480-pixel images were quantized on a 128×96 grid for reduced computation load. The identified targets were represented using their bounding boxes which were quantized on the 128×96 grid. The coverage set is also quantized on the grid and the distances of the grid points to the target set are pre-computed and stored. The tracking task is carried out at 25 fps - i.e., a new velocity input is computed for the motion module inputs at the rate of 25 Hz.

At the same time, the raw camera feed is input to the H.264 transcoder card, which is provided as a separate software device (`/dev/dsp0`) on the computer. There are various implemented on Linux and Windows operating systems for receiving video stream and other pertinent information from the camera servers.

6.3 Experimental Results for Wide Area Camera Calibration

The individual camera sensors involved in the setup of the camera system need to be calibrated for their location and pose information in order to share information amongst each other. A wide area calibration procedure described in Section 5.1 has been used to calibrate the configuration of the cameras.

BlueCCal, developed for use at Czech Technical University by Tomas Svoboda [92], is an attempt at calibrating many cameras without the necessity of a large calibration object, or the necessary handwork of more classical methods. The system does, however, have necessary conditions for the calibration to yield sufficiently accurate results, which, among others, is a minimum of three cameras for the setup.

The only action required as a reference for calibration is the mapping of 1-point objects that is easily detectable in the images captured. For our setup, this involved capturing images in a dark setting with a plastic-capped laser pointer to act as the 1-point reference in all images. Therefore, the only necessary work was moving the laser pointer around within our viewable domain, and capturing images during the course of this motion. These images would later be used in the calibration software, to locate the relevant points necessary for calibration.

The software itself can be broken down into a number of simple steps: find all points in the images for each sensor, discard all mis-detected points by pair-wise RANSAC analysis, estimate projective depths, fill missing points to make the scaled measurement matrix complete, perform the rank 4 factorization to get projective shape and motion, and estimate parameters of non-linear distortion.

Results of the calibration aren't only just the necessary conversion matrices between each of the sensors, but also the distortion models of the system, the location of all captured points in each of the sensors, and a scaled model of the location of each of the sensors. This information proves vital in the immediate testing of the results of the calibration; i.e., if the scaled model displays some erratic location of a given sensor, it is likely the calibration is off w.r.t. that sensor. One can then look at the location of the points in that sensor and notice if that sensor got the necessary coverage of its viewing area to accurately calibrate w.r.t. the rest of the system. This allows for a more intuitive check of the accuracy of a given calibration.

An advantage of this calibration tool, besides those stated above, is that the system is an adaptive calibration, and among the images captured, the one-point object, - i.e. the laser pointer - does not have to be in all of the images for each sensor. The adaptive style of this network means that the more cameras involved, the better the calibration results. So this proves beneficial for heavily overlapped multiple sensor networks. The only necessary condition w.r.t. the number of sensors that can view a given point is that the resultant calibration requires a minimum number of points that can be viewed by at least three cameras at a time; this proves incredibly helpful

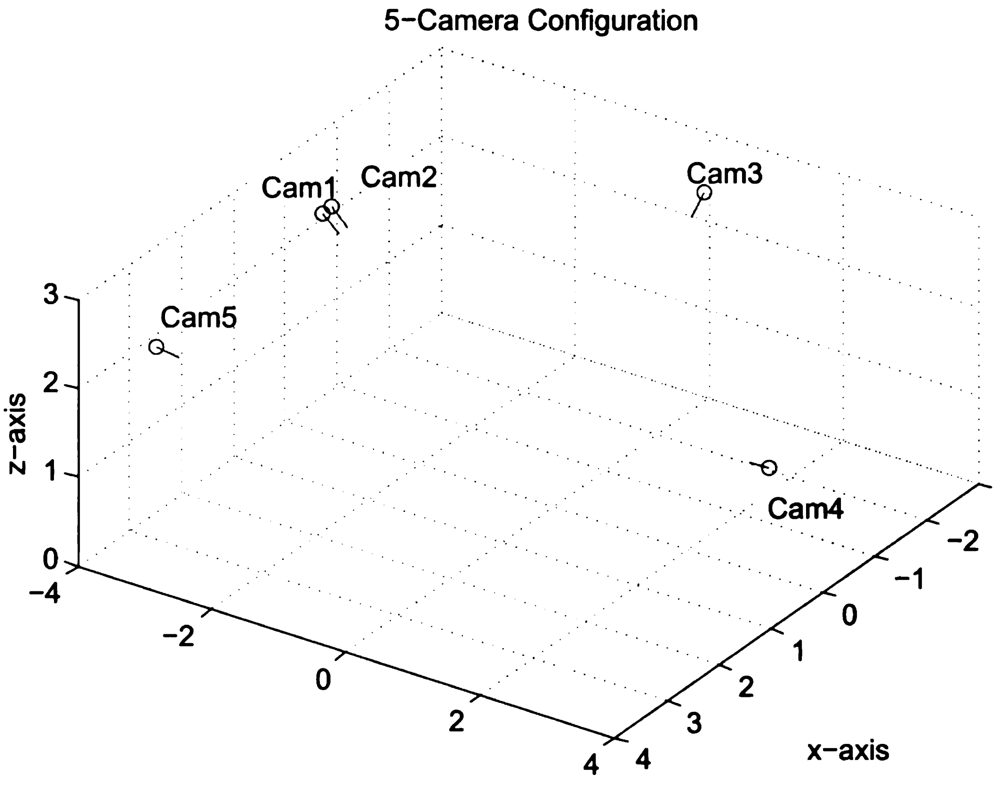


Figure 6.5: Camera Setup: Location and pointing direction of the camera setup.

when working with a disjoint network, where there exists no point that all of the cameras can see. By only requiring three cameras to see it, as long as the majority of the monitored region is covered by at least 3 cameras, then the system is able to calibrate with reasonable efficiency.

Figure 6.5 shows the results of the calibration carried out on 5 cameras. The cameras were located at different heights and were pointing to the center of the room.

The results of this experiment demonstrate the calibration of the surveillance testbed setup for conducting the remaining experiments outlined. From Figure 6.5 we can see that the camera setup was calibrated with an adequate degree of accuracy.

6.4 Experimental Implementation for Image-Based Hausdorff Tracking

The automated tracking task was defined as maintaining the visibility of multiple moving targets in a region with adequate resolution. The shape function used to track the targets is:

$$\begin{aligned} J(\hat{K}) &= \textstyle\sum_{i=1}^{N} J_{FOV}(\hat{K}_i) + J_{Amin}(\hat{K}_i) + J_{Amax}(\hat{K}_i) \\ J_{FOV}(\hat{K}_i) &= \textstyle\int_{\hat{K}_i} d_K^2(p)\ dq \\ J_{Amin}(\hat{K}_i) &= \max(\textstyle\int_{\hat{K}_i}\ dq - AREA\_MIN_i,\ 0) \\ J_{Amax}(\hat{K}_i) &= \min(AREA\_MAX_i - \textstyle\int_{\hat{K}_i}\ dq,\ 0) \end{aligned} \tag{6.1}$$

where, N is the number of targets, q is a point on the target set $\hat{K}$ and $AREA\_MAX_i$ and $AREA\_MIN_i$ denote the maximum and minimum admissible areas of the target set $\hat{K}_i$ for maintaining adequate resolution. Note that the shape function $J(\hat{K})$ is zero only when the set of targets $\bigcup_{i=1}^{N} \hat{K}_i$ is completely covered by the sensor coverage set K and when the area of each target set $\hat{K}$ is within the limits $(AREA\_MIN, AREA\_MAX)$ specified for that target. Otherwise $J(\hat{K})$ is a non-zero positive value.

Based on the value of the shape function $J(\hat{K})$, the velocity input vector u to the camera motion units (PTZ drives or robot) is calculated and applied at the rate of image acquisition - i.e., 25 frames per second (fps).

6.4.1 Tracking a Single Target

The surveillance task is to maintain the target in the active FOV while maintaining a discernable resolution of the target on the image plane. The task is described

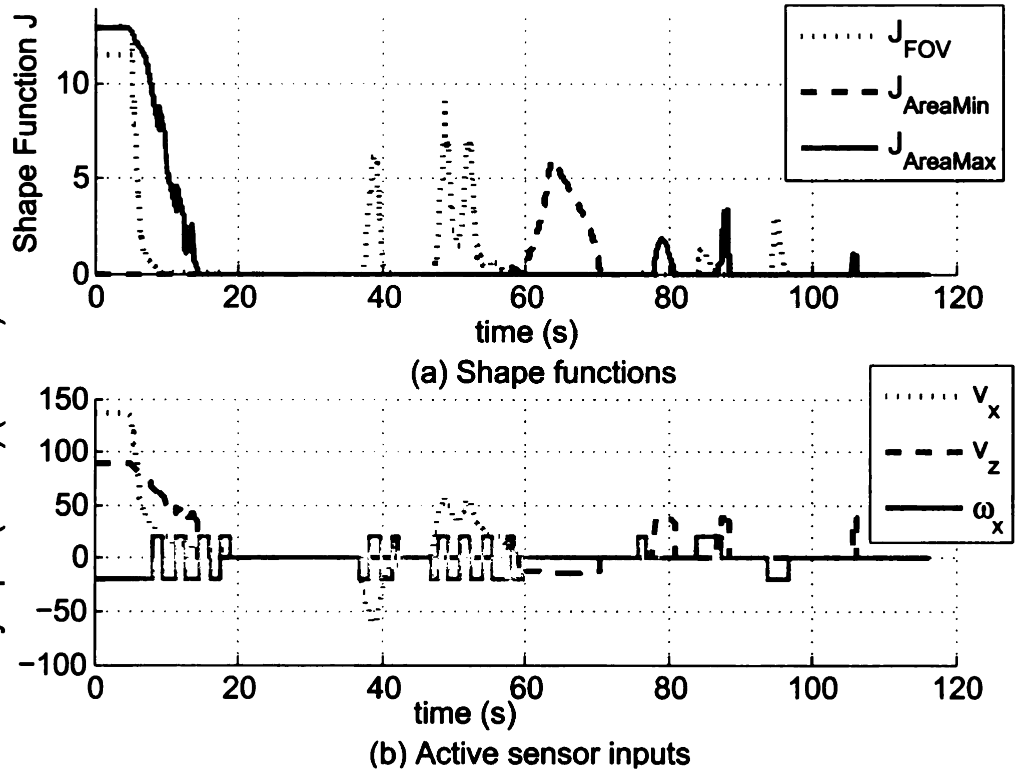
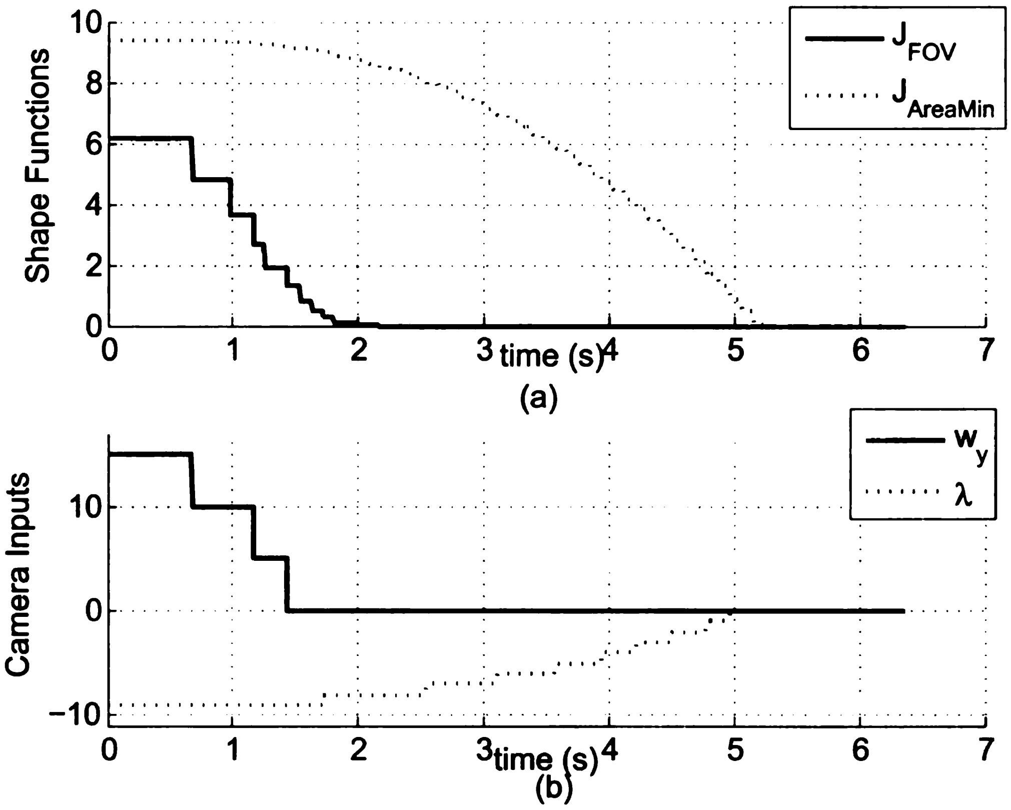


Figure 6.6: Image-based Hausdorff tracking using active camera

mathematically using equation 6.1. The target was a ball that was manually moved around in 3D space during the duration of the experiment. The image was taken as a regular grid of 128x96 pixels evenly spread over the 640x480 original image in order to reduce the computation load. The target set K was approximated as occupying a certain number of pixels on this grid. Assumptions on the input $u = [\dot{x}, \dot{z}, \omega_x, \omega_y, \dot{\lambda}]^T$ to the robot and camera system were derived using equation 3.31. The target was initially placed where the task critera were not satisfied and then the target moved around manually, which generated a disturbance input to the system and the system immediately tried to reduce the value of the shape function to zero.

Figures 6.6 and 6.7 depict the results of the image space Hausdorff tracking task using an active camera. The shape functions $J_{FOV}(\hat{K})$, $J_{AreaMin}(\hat{K})$ and $J_{AreaMax}$ and inputs $u = [\dot{x}, \dot{y}, \omega_x]^T$ are plotted in 6.6 (a) and (b), respectively. The figure



(a)

(b)

Figure 6.7: Image-based Hausdorff tracking using active camera: Maintaining adequate target resolution

shows that the system is stable and that the target is maintained in the camera field of view with a desired resolution despite the seemingly random motion of the target.

These experiments demonstrate Hausdorff tracking using multiple shape functions. We see that the shape functions are are non-zero when the target moves but they are quickly reduced to zero by the motion of the camera. The motion of the target is taken as a disturbance input for the tracking control system shown in Figure 3.4 and it can be seen that the Hausdorff tracking controller can reject disturbance input from motion of the target.

6.4.2 Tracking Multiple Targets Simultaneously

The surveillance task is to maintain the multiple targets in the active FOV of the sensor. The targets were two humans moving around and interacting with each other.

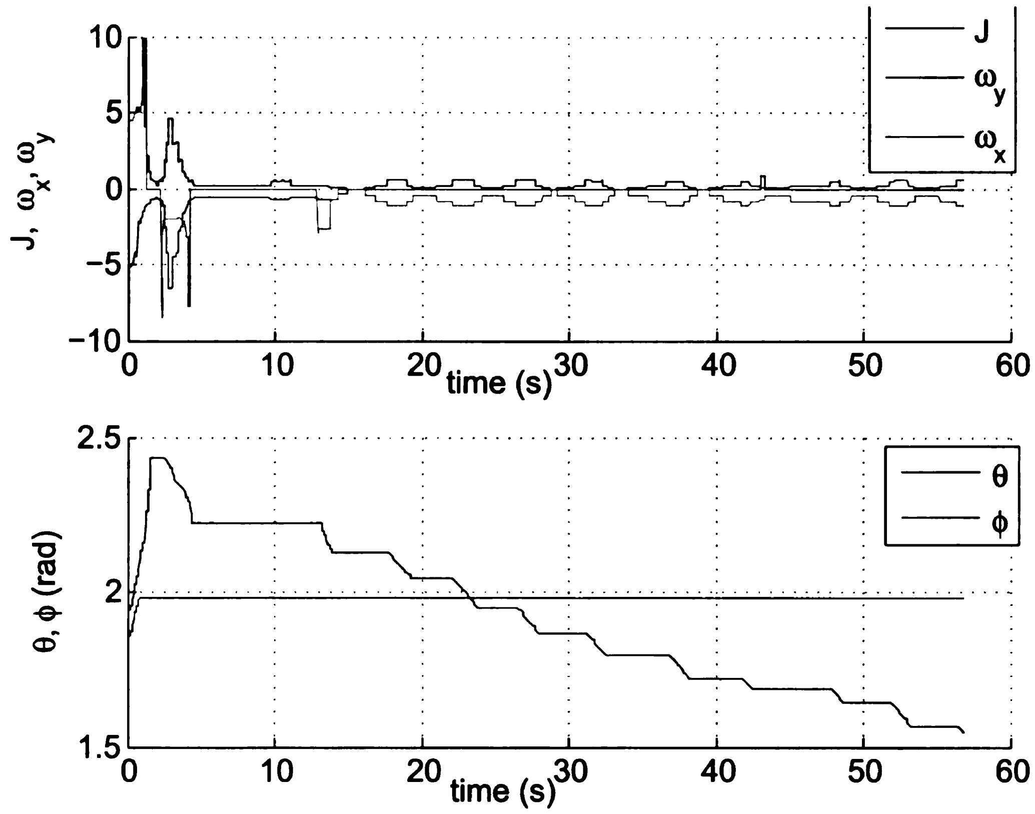


Figure 6.8: Tracking multiple targets using Image-based Hausdorff tracking

The target set K was approximated as occupying a certain number of pixels (in multiple disjoint locations) on this grid. Assumptions on the input $u = [\omega_x, \omega_y]^T$ to the camera system were derived using equation 3.31. The targets were initially located such that they were out of the field of view of camera. The targets were detected by the monitor and the sensor was tasked to cover the two targets simultaneously. At, $t = 0$, the targets are just in the active FOV of the sensor and the task criterion is not satisfied. The camera then moves to reduce the shape function J to zero so the targets are now covered. The targets then randomly move around the room and the camera tries to maintain both targets continuously in the active FOV.

Figure 6.8 depicts the J and the input velocities ($u = [\omega_x \;\; \omega_y]^T$) applied to the camera. Notice the initial large value of the shape function J, which is quickly reduced to zero. Figure 6.9 depicts the position X, Z estimated of the two targets. We see

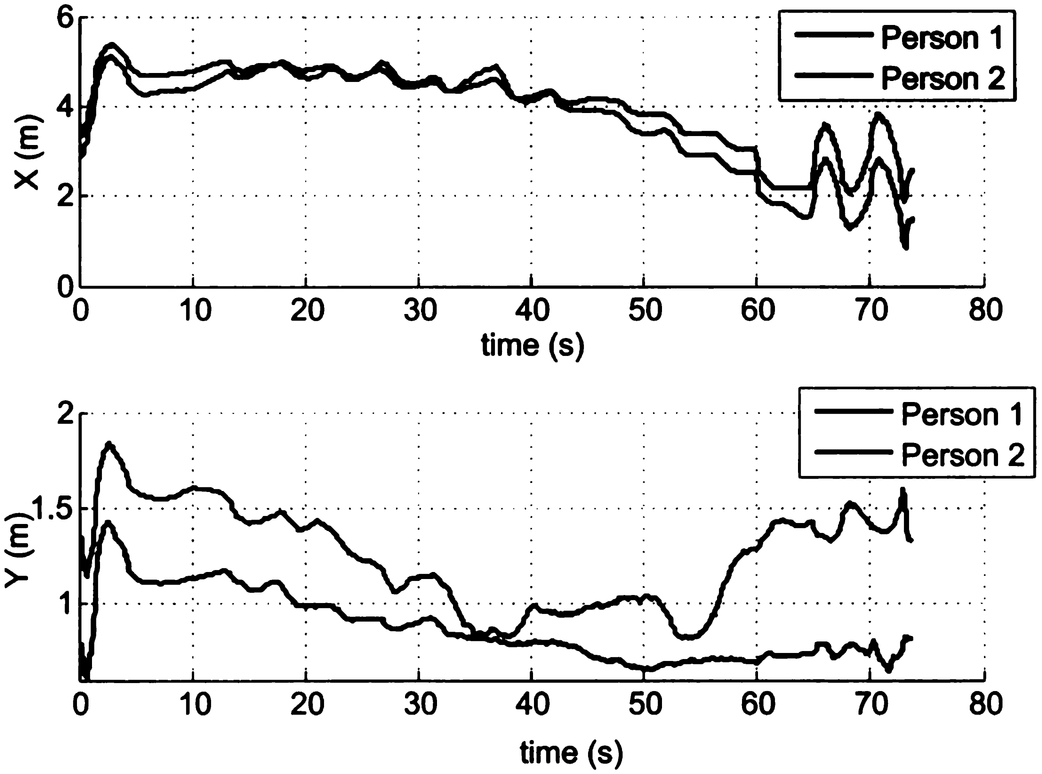
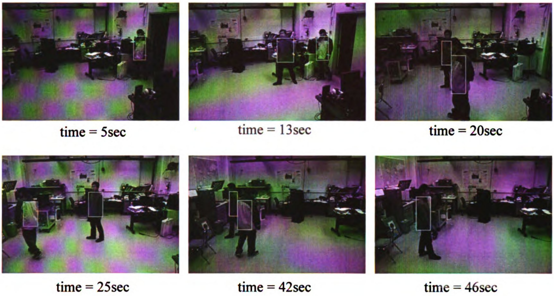


Figure 6.9: Image-based Hausdorff tracking for multiple targets: Estimated position of targets

that despite the seemingly random motion of the two targets, the camera always tries to keep both of them in the active FOV. Further, the energy efficiency of the proposed method is demonstrated by the relatively infrequent input applied to the camera only when one of the objects escapes the active FOV. Figure 6.10 shows the images taken from the camera at various times during the tracking experiment shown in Figure 6.8.

This experiment shows multiple target tracking with a single camera sensor using the Hausdorff tracking method. Tracking multiple targets using a single sensor is advantageous for overloaded networks where the number of targets outnumber the number of cameras available for tracking. The image sequence in Figure 6.10 shows that the targets are moving about randomly and still being maintained in the field of view of the camera sensor at all times, which demonstrates the advantages of



time = 5sec

time = 13sec

time = 20sec

time = 25sec

time = 42sec

time = 46sec

Figure 6.10: Image sequence for tracking multiple targets using Image-based Hausdorff tracking.

Hausdorff tracking over vector space servoing.

6.5 Experimental Implementation for Cooperative Hausdorff Tracking

The cooperation between nodes to continuously track a target by directly tasking various sensors can be done using cooperative Hausdorff tracking as explained in section 3.4.2.

In this experiment, a mobile robot is deployed in order to track a target that cannot be covered by a fixed sensor with adequate resolution. The robot uses the position information received from other sensors in the network to compute a trajectory that will bring the target set within its coverage set.

Figure 6.12 shows the x-z plot, which is the plane of the floor, of the data for both the target centroid and the robot tracking it. The robot's path shows the initial cooperative tracking taking place. As the robot moved its view cone toward the

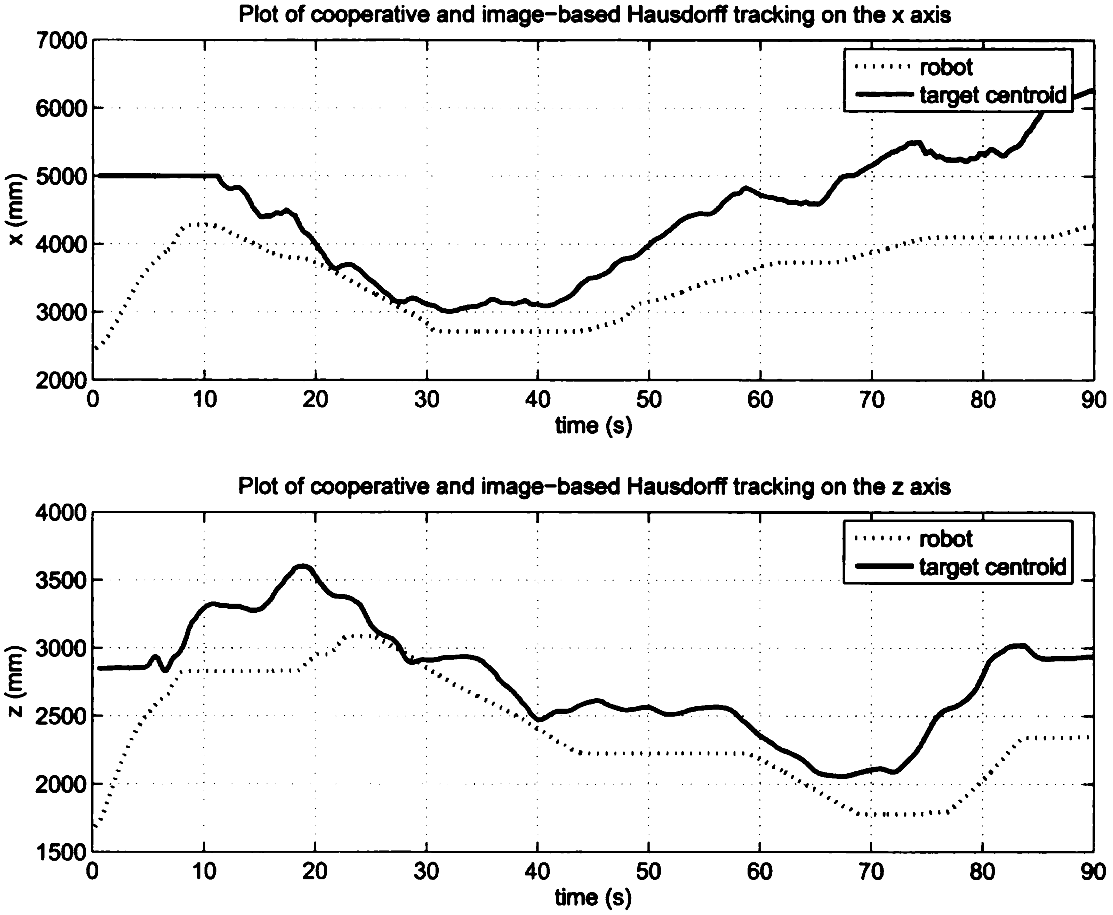


Figure 6.11: Position of robot and target set on x and y axes.

target, the target also began moving. Re-projection of the target onto the view cone caused a slight change in velocity in the robot that can be seen. Once the target set enters the set space of the mobile robot's view cone, image-based Hausdorff tracking begins. Now the robot reacts to changes in the target to maintain optimal resolution. This continues for the remainder of the experiment, as can be seen in the plot.

In Figure 6.11 we can see the data from the individual axes. The robot is quite successful in maintaining the target set within its coverage set, as can be seen by the correlation of the plots of each of the axes.

Finally, in Figure 6.13 we see the plot of the J value for both image-based and

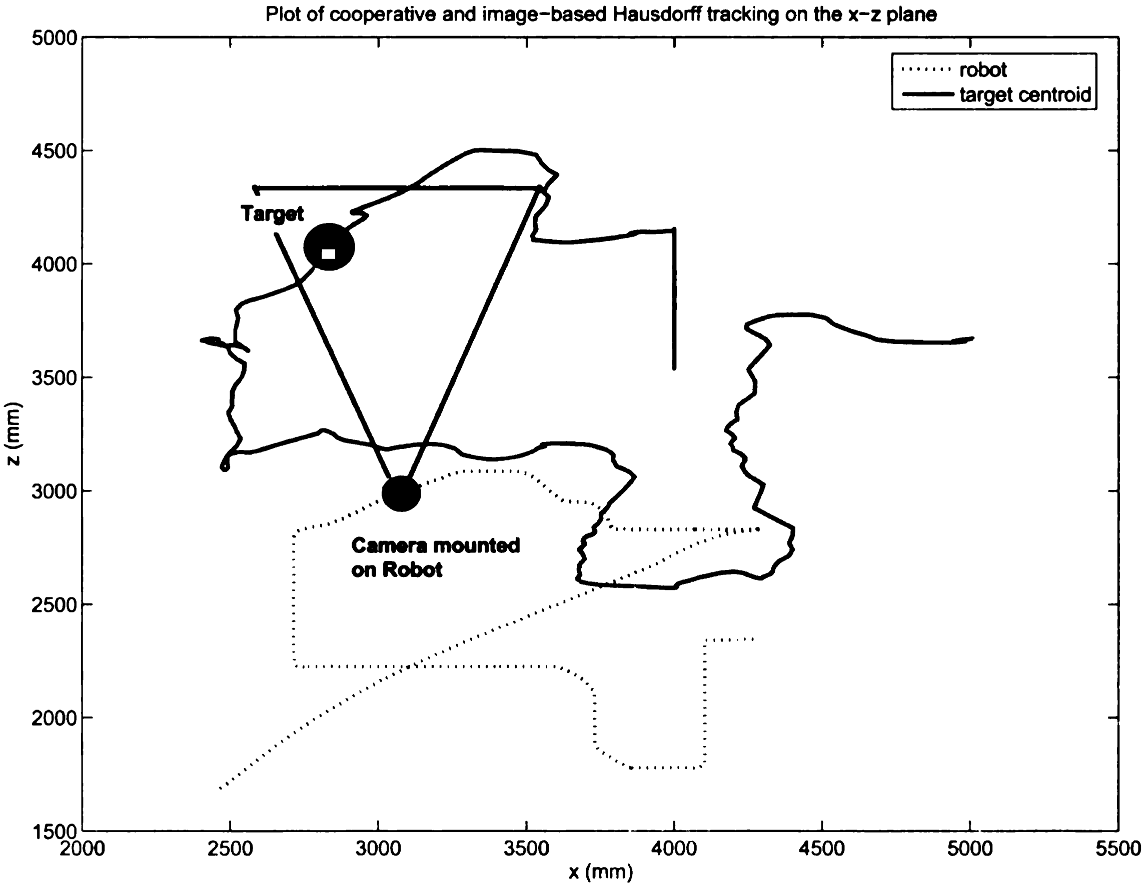


Figure 6.12: Motion of target set and robot on x-z plane. The view cone of the robot is shown at a particular instance in time with the position of the target centroid shown as well.

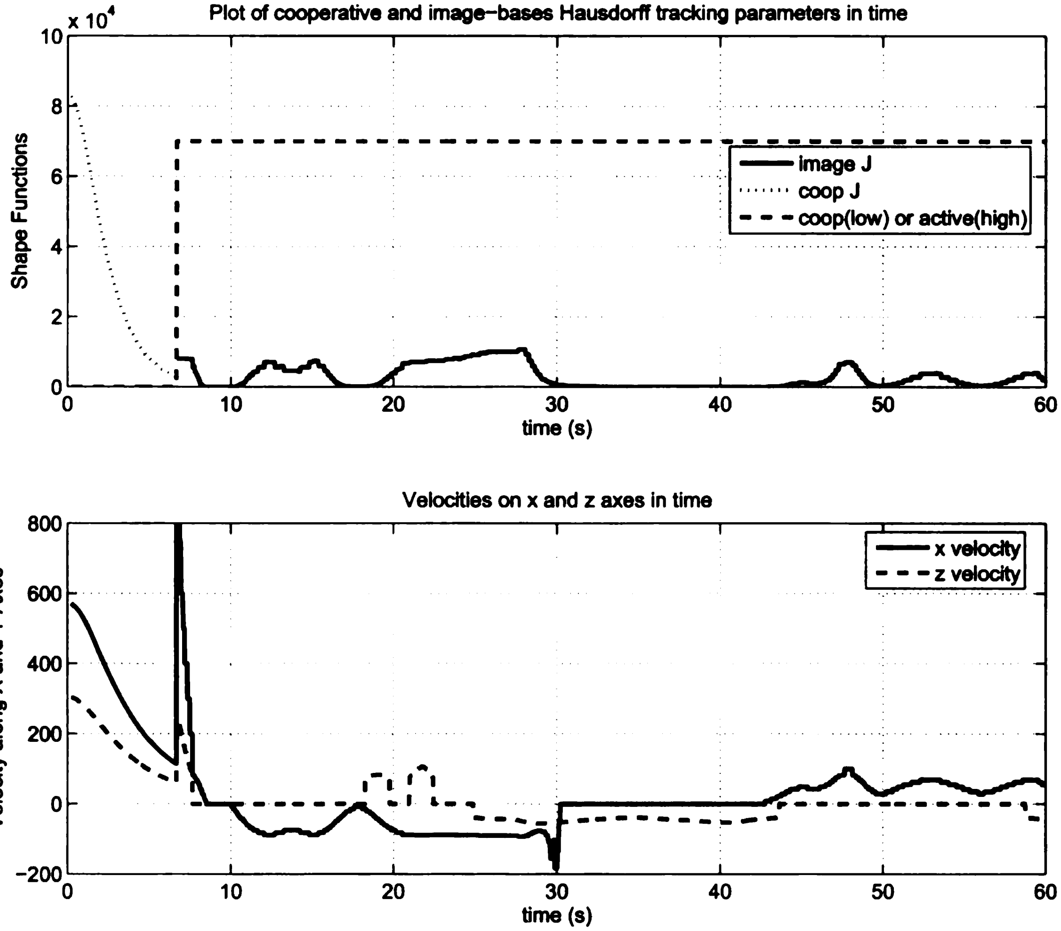
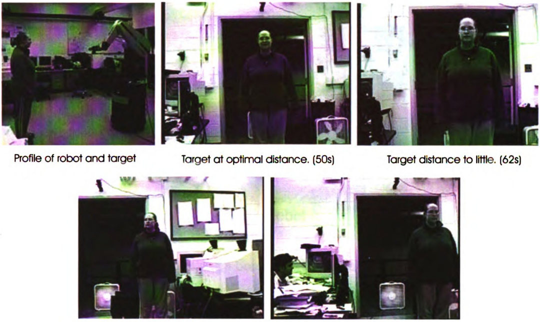


Figure 6.13: Shape functions for cooperative and image-based Hausdorff tracking



Profile of robot and target

Target at optimal distance. (50s)

Target distance to little. (62s)

Target distance to great. (20s)

Target at edge of view. (80s)

Figure 6.14: Robots view of target at several points in time to show several conditions that the tracking system responds to.

cooperative Hausdorff tracking. A demarcation line has been included to clearly mark the switch from cooperative to image-based tracking. This occurs at approximately seven seconds. This agrees with the data from Figure 6.12 since this is the point in time that the values of the axes begin to synchronize. Figure 6.5 shows several images gathered during the experiment. The first image in the series shows a profile of the mobile robot and the target tracked. The remaining images were gathered from the robot's onboard camera. The second image shows the target at the determined optimal distance. This determination is based on the set space of the target instead of simple-distance as is the case with point-based tracking. The third and fourth images show cases where the target is moving toward, and away, from the robot, which then causes the robot to retreat from, and pursue, the target respectively. The final image in the series shows the target strafing left, to which the robot responds by moving to re-center the target set.

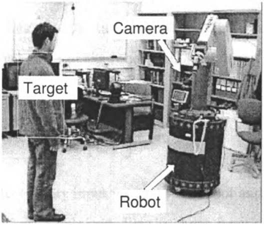


Figure 6.15: Target tracking using a Sony EVI-D30 camera mounted on Robot Nomad XR-4000 mobile robot

This experiment shows that the Hausdorff tracking control can be used adequately for task space servoing, as well. Given the task space location of a target set (possibly multiple disjoint targets) and a model of the coverage set of a camera, the target can be brought into view using cooperative Hausdorff tracking. This experiment also demonstrates the stable switching (due to in-built hysteresis in the switching criteria) from cooperative Hausdorff tracking mode to image-based Hausdorff tracking mode.

6.6 Experimental Implementation for Redundancy Resolution using Optimal Hausdorff Tracking

Based on the discussions in the preceding sections, we can now formulate the task distribution problem for Hausdorff tracking using redundant motion stages. The problem we are considering is the optimal task distribution for a pan-tilt camera mounted on a mobile robot tracking a single target. The experimental setup for the tracking task is depicted in Figure 6.15.

For the experimental implementation, we will consider the redundancy in the robot and camera motion along the X traverse direction. When the target moves along the X direction, either the robot can traverse linearly or the camera can pan. Hence there is a redundancy in the task execution. The model for the Hausdorff tracking task using the camera and the robot combination can be written as:

$$[C_1 C_2] \begin{bmatrix} v_x \\ w_y \end{bmatrix} = -\alpha J \tag{6.2}$$

where $u = [v_x \; w_y]^T$ is the velocity input vector to the robot and camera combination, α is a positive task level scalar gain and J is the shape function which indicates the error in accomplishing the task. The vector $C = [C_1 \; C_2]$ describes the distribution of the task error between the two redundant axes. Equation (6.2) can be derived using the formula provided in [67].

The task indicators used are the energy consumed by the system and the current pan angle of the camera. The current pan angle of the camera affects how the translation in X by the mobile robot affects the task accomplishment. Hence, the objective functions considered for the multi-objective optimization schemes also included energy minimization and pan angle minimization and were represented as:

$$\begin{aligned} D_1 &= q_0^2 = (O - O_0)^2 && (6.3) \\ D_2 &= \|v_x\| + \|w_y\| && (6.4) \end{aligned}$$

where, O is the current pan angle value, O_0 is the desired pan angle value considered as zero pan, $\|v_x\|$ and $\|w_y\|$ are the horizontal X velocity and the angular Y velocity (pan velocity), respectively. Note that confidence parameters are built into the objective functions. For example, when the pan angle is close to zero, we would like to reduce the weight for the objective function D_1, which is exactly what happens.

Various optimization schemes were set up, including one with no optimization involving just a random selection of the null space vector ϵ. The optimization schemes

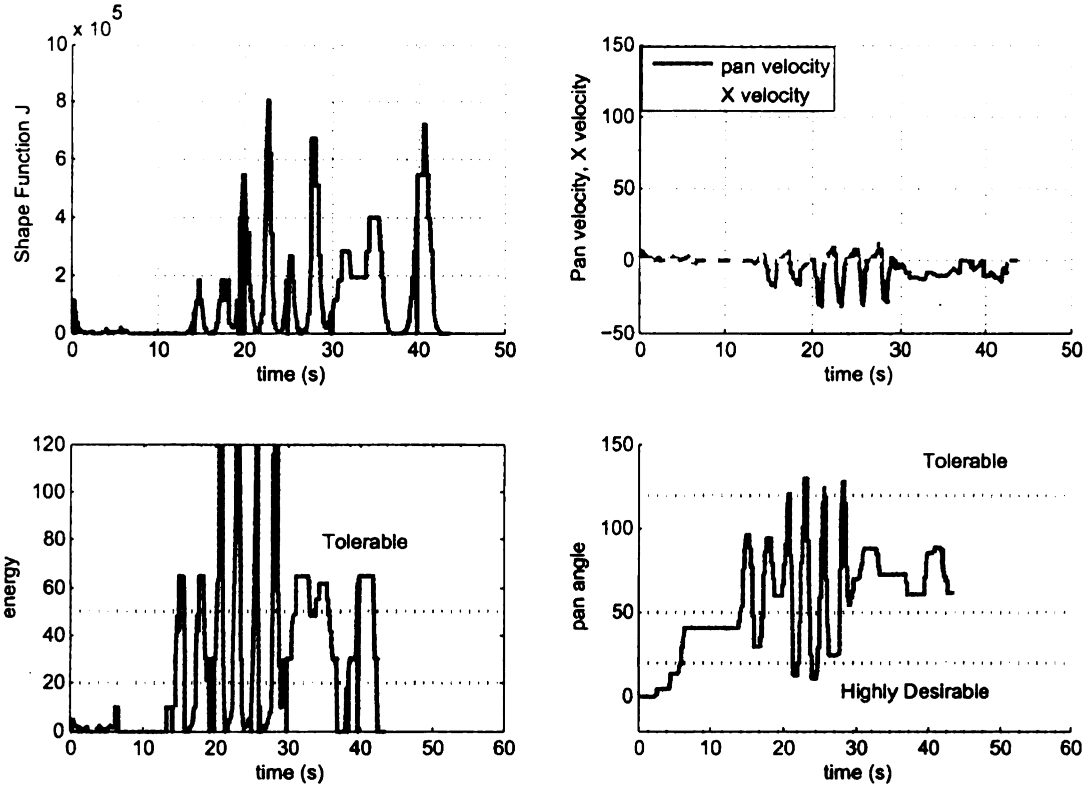


Figure 6.16: Results of the tracking task with optimal task distribution using physical programming.

selected for the experiment included single objective optimization where either the energy minimization or zero pan angle objectives were considered individually. For the multi-objective schemes, both weighted sum and physical programming methods were tested. The results of these tests are tabulated in the following sections. Figure 6.16 shows the results of the experiments carried out using physical programming optimization described above. The figure shows the plots of the shape function, the velocities in the X and pan directions and the objective function values for energy conservation and pan angle zeroing.

Notice that the values of all the objective functions are always maintained at least in the tolerable range by sacrificing the performance of the other cost functions currently under consideration.

The same task was also carried out using a no-optimization scheme with a fixed

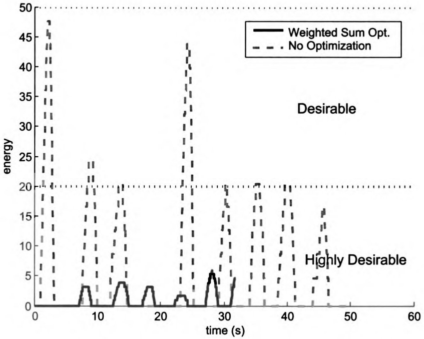


Figure 6.17: The energy consumption for no optimization, and weighted sum.

task distribution between the two redundant axes and also using a weighted sum scheme. As mentioned in section 5.4.1, a weighted sum method with fixed weights without apparent meaning is used in order to combine the various cost criteria. The cost criterion used is:

$$Z = q_0^2 + \frac{(\|v_x\| + \|w_x\|)}{2} \tag{6.5}$$

The two cost functions - i.e., energy and pan-angle maintenance for the no-optimization method and the weighted sum method, are shown in Figures 6.17 and 6.17, respectively.

For the no-optimization case, it can be seen that in the absence of any optimization, the linear velocity transferred to the robot is very small and most of the velocity

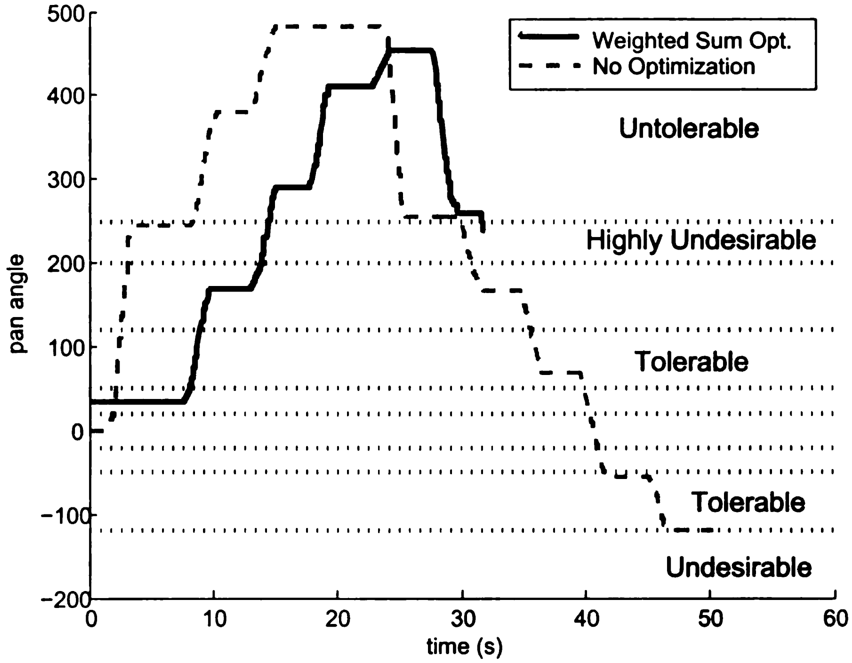


Figure 6.18: The pan angle for no optimization, and weighted sum.

for task execution is allocated to the camera pan.

Using the weighted sum method, the various components comprising the task criterion lack physical meaning. Based on this observation we notice that even when the multi-objective optimization is enabled, the pan angle remains very close to the desired value while the energy conservation objective attains unacceptable values. It should be noted that various tolerance ranges, which are used for the physical programming approach, are not used in the weighted sum approach. Hence, despite the optimization being enabled on the various objectives, the values of the objective functions can attain unacceptable values.

The above results show that the physical programming method provided better results in terms of optimizing multiple objectives while simultaneously accomplishing the task. This is because with the weighted sum method, due to the loss of physical meaning of the weights, dynamic weighting may provide an unacceptable solution. However, for physical programming, since the cost criteria are combined using weights which are attributed a physical meaning, physical programming can calculate solutions on the pareto-optimal frontier.

This experiment further goes to show that the redundancy in the input expressed in Equation (3.30) can be successfully exploited in order to optimize various cost functions associated with the task. Hence, some criteria, such as resolution maintenance of multiple targets. can indeed be removed from the set of constraints and used in the optimization criteria and can provide more flexibility for task specification.

6.7 Experimental Results for Multisensor Cooperation

This section illustrates the experimental implementation and results for the two techniques:

1. Sensor allocation algorithm and
2. Mutational hybrid automata modeling techniques proposed earlier.

6.7.1 Experimental Evaluation for the Sensor Allocation Algorithm

Pervasive surveillance using the proposed multi-sensor coordination mechanism was rigorously tested. The task of tracking a target over multiple sensor coverage regions was carried out. The target moved a large distance, spanning the capable viewing regions of multiple sensors. Figure 6.19 depicts the results of the tracking scenario. Initially, sensor 2 can see the complete target but the target is out of the active FOV of sensor 1, which is depicted by the target visible flag. Based on the solution of the optimization, sensor 1 is tasked to track the object. It uses cooperative Hausdorff surveillance to cover the target. The global coordinates of the target for cooperative Hausdorff tracking are provided by sensor 2. Once it acquires the target, it switches to image-based Hausdorff tracking to ensure target coverage. The target then randomly

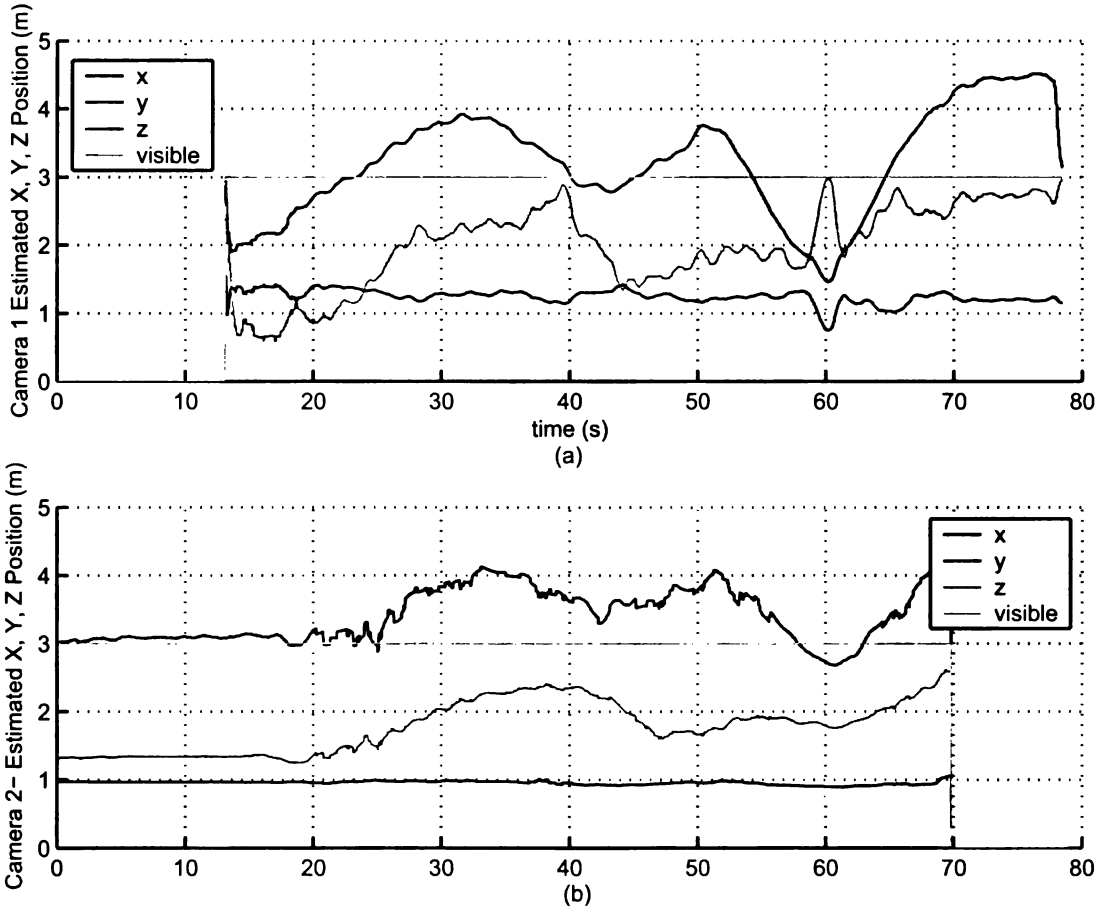


Figure 6.19: Estimated position of targets for multisensor cooperation using sensor allocation algorithm

moves around in the active FOV of both the sensors, which keep continuously tracking it. Notice the robust tracking performance of the image-based tracking scenario despite the minor differences in the estimation of the object location. The target then starts to move out of the tracking range of sensor 2, which again triggers the optimization planning problem, which assigns sensor 1 to keep continuously tracking the target.

This experiment demonstrates the cooperation between multiple sensors for tracking a target. It shows that information from another sensor can be used in order to cooperatively acquire a target by a camera and that the location of the targets can be used to optimally assign the tracking task to the various sensors based on their availability, target priority and the locations of the sensors and targets.

6.7.2 Experimental Evaluation of Multiple Target Tracking using Multiple Formation-Constrained Sensors

Motion is an energy intensive resource for a sensor node and should be minimized. Multiple sensors (maybe with varying sensing modalities) can be mounted on a single motion drive to reduce the energy consumption. Further, the various sensors can be tasked to track different targets simultaneously. Hence the motion of the sensor node should be optimized in order to achieve the various tasks simultaneously. The optimal Hausdorff tracking framework proposed in section 5.4.1 can be used to calculate the optimal sensor input u.

Consider a scenario where two targets are being tracked by two separate formation-constrained sensors as shown in Figure 6.20. The task is to maintain the visibility of targets $\hat{K}_1$ and $\hat{K}_2$ using the sensors cam_1 and cam_2 respectively. The two sensors are mounted on a single motion platform, although sensor cam_1 can be moved independently of sensor cam_2 by using the motion of joint q_0 of the PUMA robot arm.

The shape functions for this task can be expressed as:

$$J_1 = \int_{\hat{K}_1} d_{K_1}(p)\, dp \tag{6.6}$$

$$J_2 = \int_{\hat{K}_2} d_{K_2}(p)\, dp \tag{6.7}$$

The motion inputs to the complete robot system are linear motion along the X axis (v_x) and angular motion about the Z axis of the robot base (ω_z). Angular motion of the robot will not affect the coverage task assigned to sensor cam$_2$ while linear motion along the X axis will affect the task assigned to both the sensors. Using Equation (3.31), the model for Hausdorff tracking for the two-camera system can be written as:

$$Cu = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & 0 \end{bmatrix} \begin{bmatrix} v_x \\ \omega_z \end{bmatrix} \leqslant - \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} \begin{bmatrix} J_1 \\ J_2 \end{bmatrix} \tag{6.8}$$

Figure 6.21 depicts the target tracks of the two targets in the world coordinate frame. The two targets move independently of each other and can cross each other as is shown in Figure 6.24.

Figure 6.22 depicts the shape functions J_1 and J_2 for two cameras cam$_1$ and cam$_2$, respectively, and the velocity profiles of the two inputs v_x and ω_z to the mobile base and robot arm are shown in Figure 6.23. The inputs to the two motion axes (linear velocity v_x and angular velocity ω_z) are calculated using 6.8.

The results of this experiment show that we can use formation-constrained sensors in order to accomplish multiple tracking tasks simultaneously using multiple sensors. This implies that the optimal Hausdorff tracking framework can be leveraged in order to solve multiple target tracking problems.

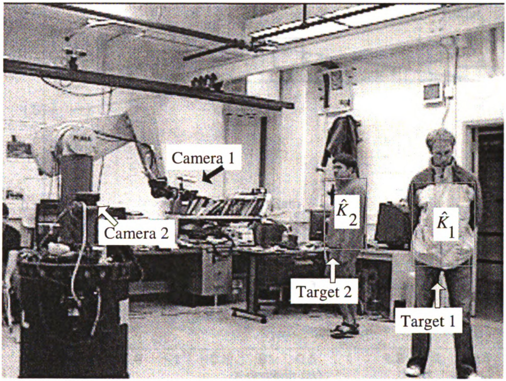


Figure 6.20: Experiment setup for multi-target tracking using multiple cameras.

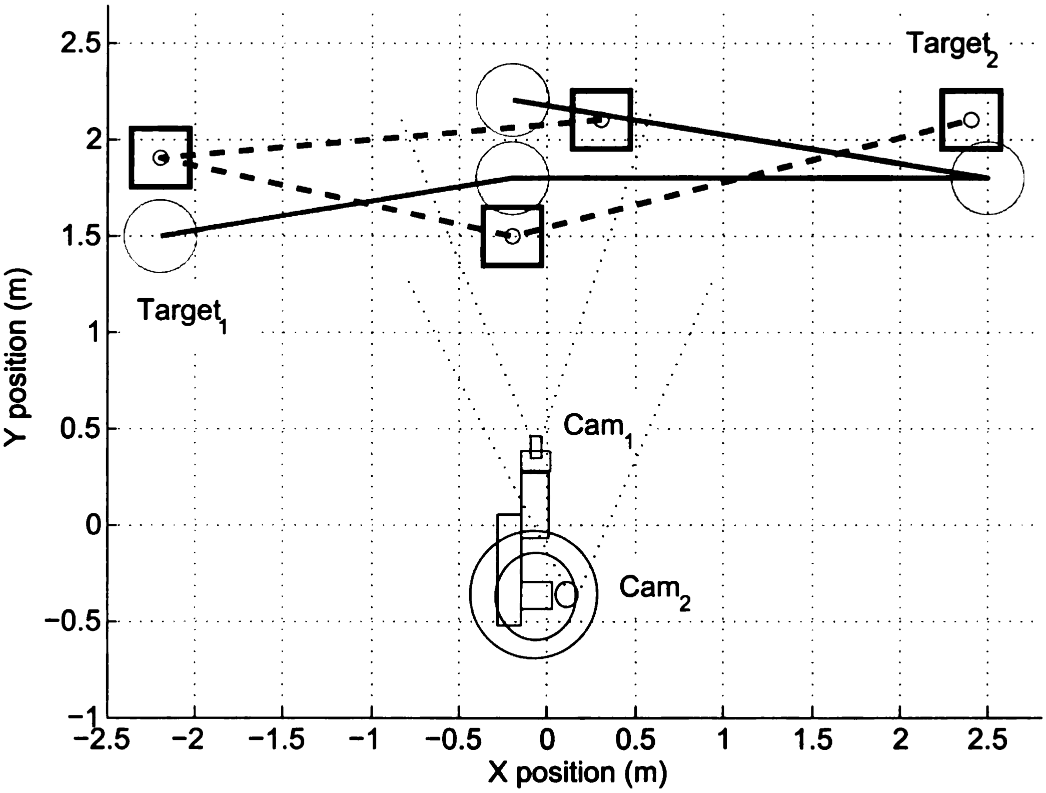


Figure 6.21: Target tracks for multi-target tracking using multiple cameras.

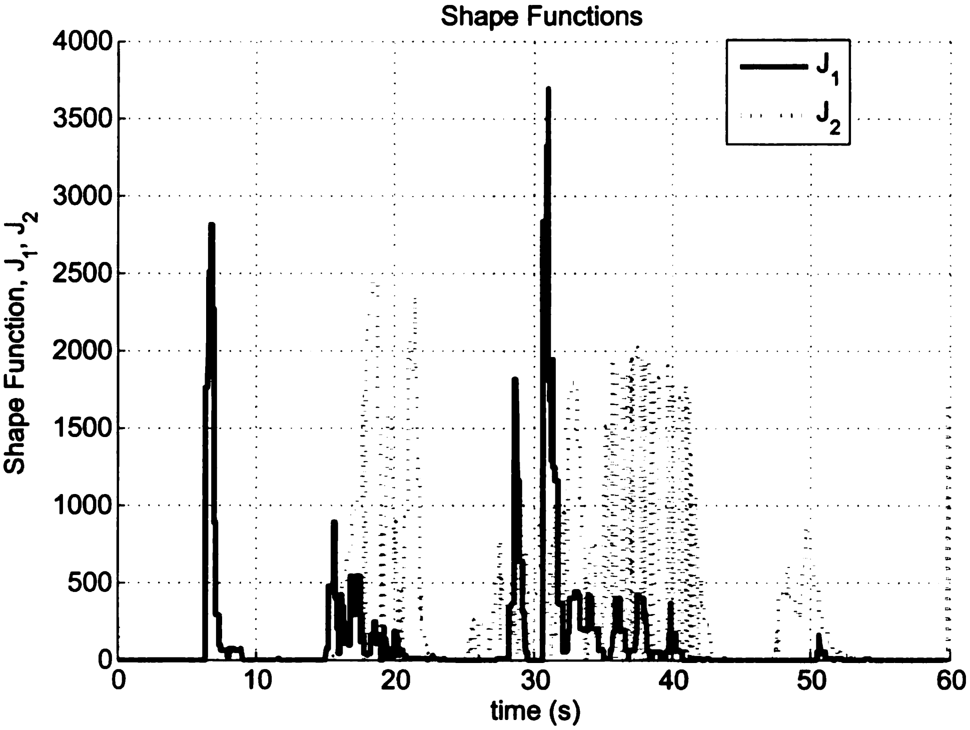


Figure 6.22: Experiment setup for multi-target tracking using multiple cameras.

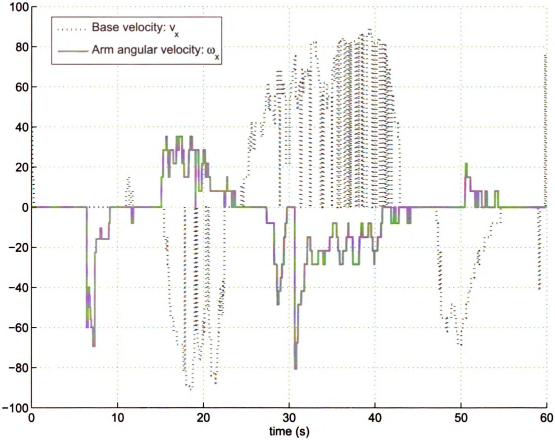
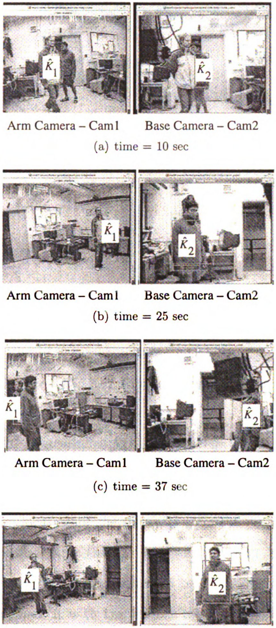


Figure 6.23: Experiment setup for multi-target tracking using multiple cameras.



(a) time = 10 sec

(b) time = 25 sec

(c) time = 37 sec

(d) time = 55 sec

Figure 6.24: Image Sequences taken while tracking the two targets with two formation-constrained cameras

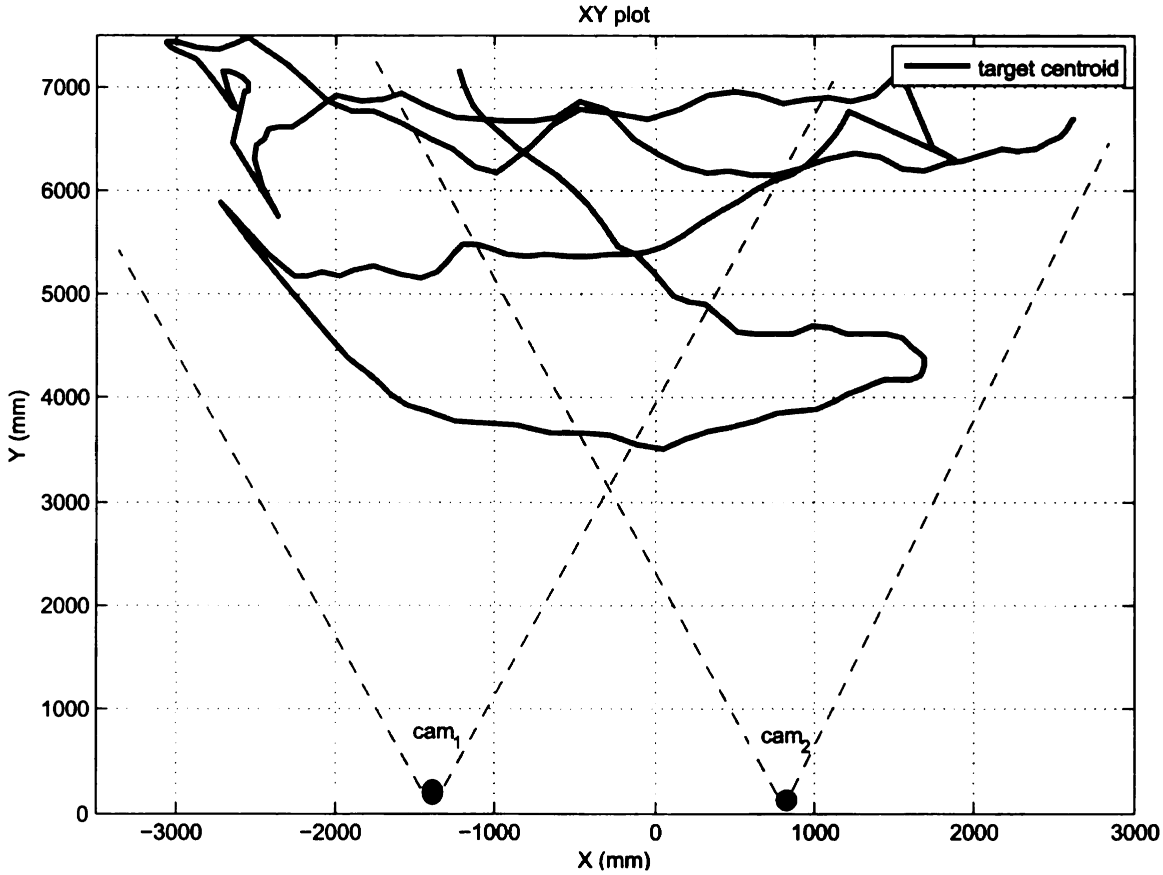


Figure 6.25: X-Y Plot for target motion

6.7.3 Experimental Evaluation for Multi-Sensor Cooperation using Mutational Hybrid Automata Model

For the experiment, the location of the target (human) moving about was captured using a stereo vision system. Note that the location of the target was not used in the feedback control scheme designed, but was used merely to show the random motion of the target and coordinate it with the various mode changes in the MHA model. Figure 6.25 represents the position, in x-y coordinates, of the centroid of the target, located using the stereo camera. The locations of the cameras cam_1 and cam_2 are superimposed on this figure.

Pervasive Target Tracking Using Two Cameras with Continuous Camera Servoing

In this experiment, the MHA model presented in section is implemented.

Figures 6.27 and 6.26 show the results of the experiments conducted using this model. Figure 6.27 represents the data calculated by cam_1 and Figure 6.26 represents the data calculated by cam_2. Figures 6.27 and 6.26 show the value of the shape function J, the pan velocity input to the camera, and the mode $q \in \{q_1, q_2, q_3\}$ (defined in 4.2) that the system is in; and subfigure b depicts the current pan position of the camera and the visibility flag of the target for that camera. The visibility flag indicates whether the target is within the FOV of the camera and is set when the target is recognized by the target detection subsystem.

Figure 6.28 depicts an image sequence captured by the two cameras, with subfigures a, b and c representing the three modes q_1, q_3 and q_2 respectively. Within each Figure 6.28(a), 6.28(b) and 6.28(c), the left and right images correspond to cam_1 and cam_2, respectively. We notice that the target is in the field of view of at least one camera at any time during the course of the entire experiment.

The results of this experiment show that the mutational hybrid automata framework can be used to systematically design a pervasive surveillance network which can be used to track a target as it moves across a wide region covered in parts by individual cameras. It shows that despite the random motions of the target, the surveillance network can, in fact, perform stable tracking without having to estimate the exact location of the target.

Pervasive Target Tracking Using Two Cameras with Controlled Switch

This section presents the results of the experimental implementation of the mutational hybrid automata model with controlled switch shown in Figure 4.3.

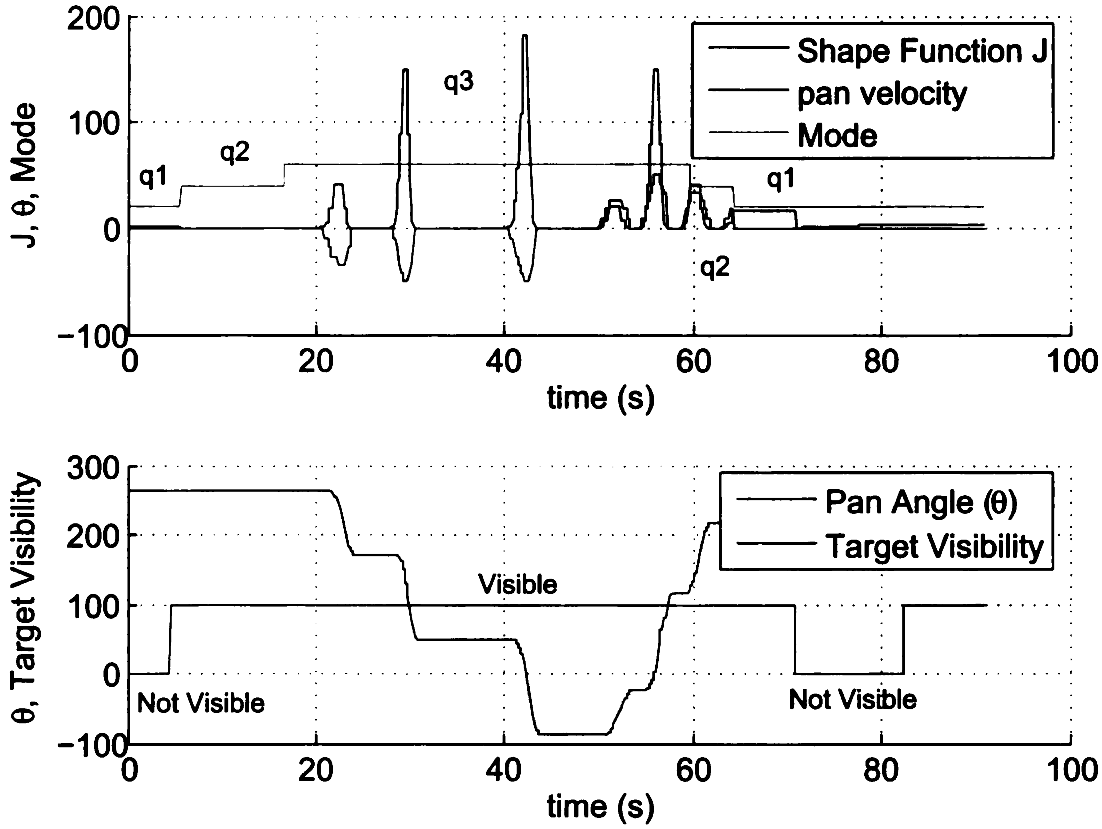


Figure 6.26: Current MHA mode, shape function, velocity, pan angle and target visibility for Cam_1

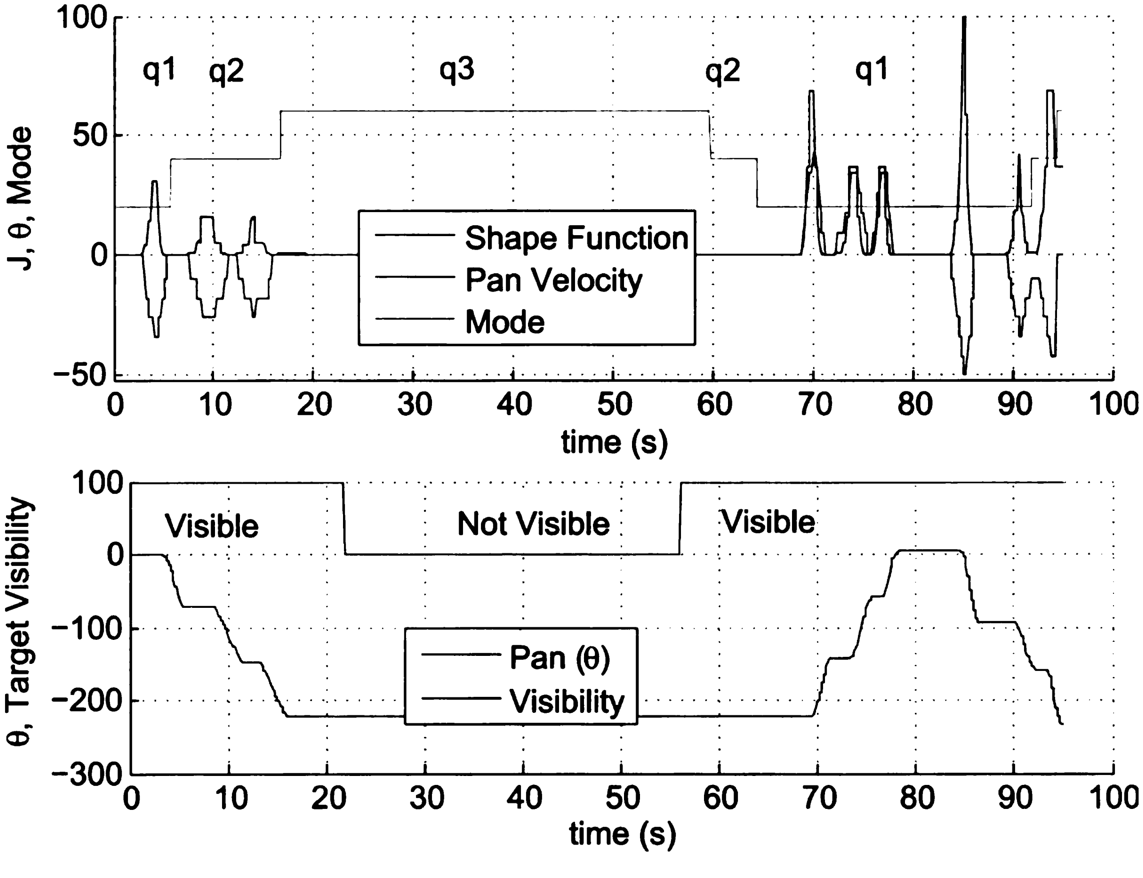
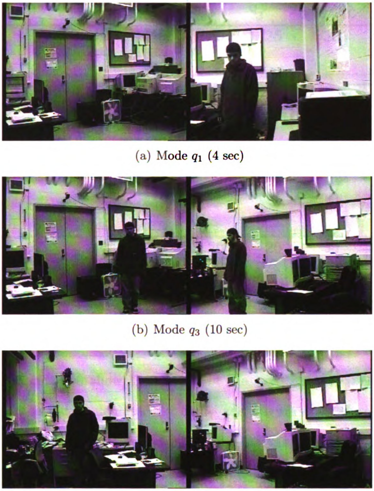


Figure 6.27: Current MHA mode, shape function, velocity, pan angle and target visibility for Cam_2



(a) Mode q_1 (4 sec)

(b) Mode q_3 (10 sec)

(c) Mode q_2 (22 sec)

Figure 6.28: Image Sequences taken while tracking the target over different modes

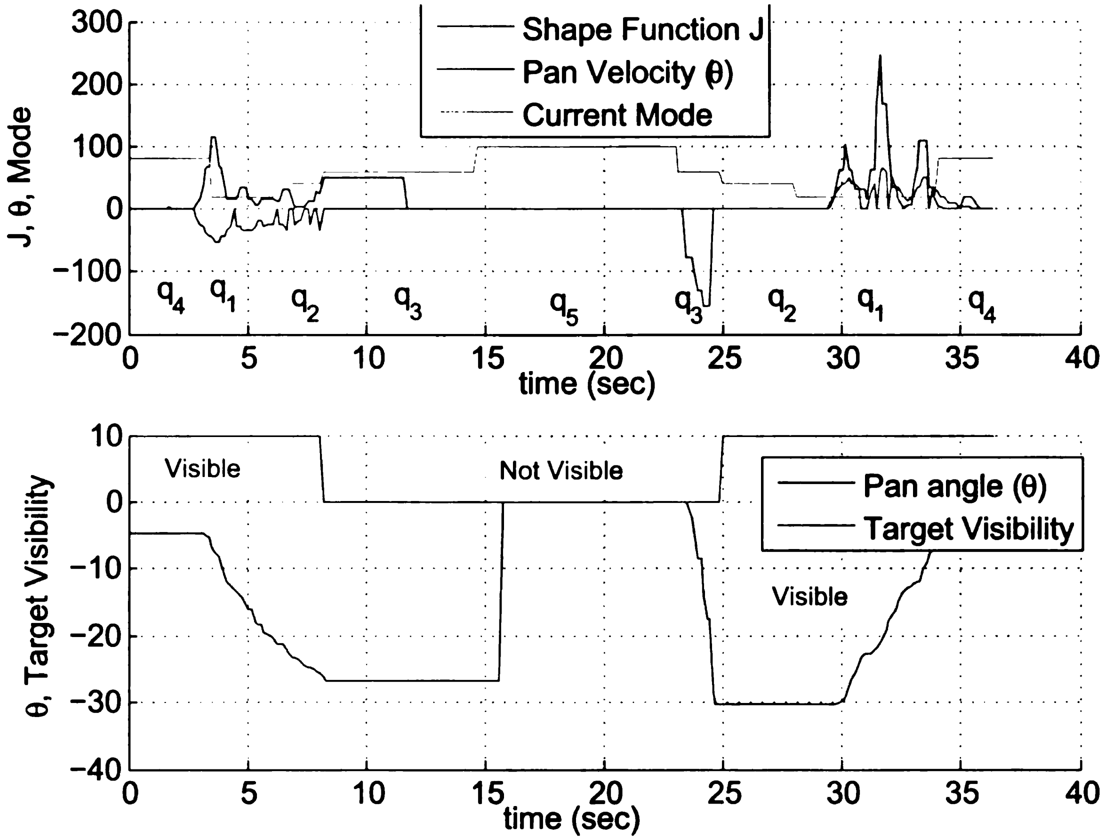


Figure 6.29: Current MHA mode, shape function, velocity, pan angle and target visibility for Cam_1

Figures 6.29 and 6.30 show the results of the experiments conducted. Figure 6.29 represents the data calculated by cam_1 and Figure 6.30 represents the data calculated by cam_2. Figures 6.29 and 6.30 show the value of the shape function J, the pan velocity input to the camera, and the mode $q \in \{q_1, q_2, q_3, q_4, q_5\}$ (defined in 4.3) that the system is in; and subfigure b depicts the current pan position of the camera and the visibility flag of the target for that camera. The visibility flag indicates whether the target is within the FOV of the camera and is set when the target is recognized by the target detection subsystem.

These experiments show that irrespective of the target's current location in the monitored region, visibility of the target is maintained by at least one camera. This

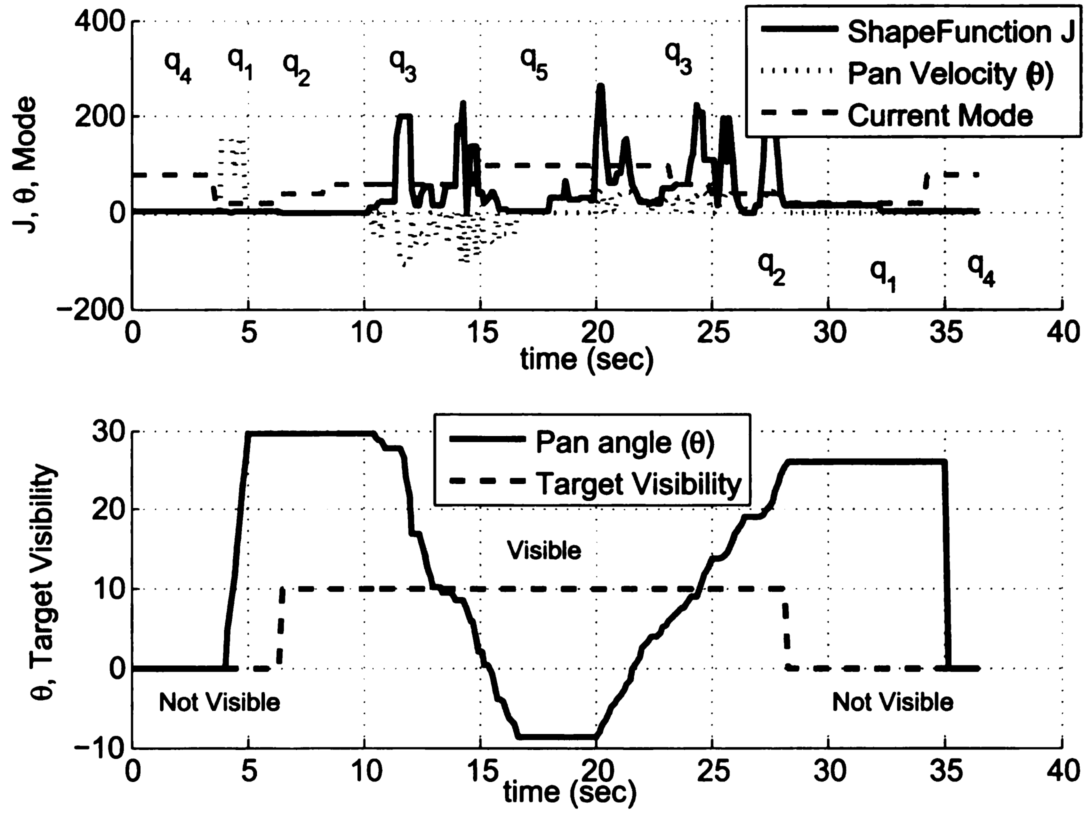


Figure 6.30: Current MHA mode, shape function, velocity, pan angle and target visibility for Cam$_2$

experiment further demonstrates the controlled switch scenario, where, the cameras can be actively turned off, can be used to perform pervasive tracking.

6.8 Performance Analysis of the Video Subsystem

The real-time-performance video subsystem implemented was measured and analyzed for use in the surveillance network scenario. The parameters which affect the performance of a video transport scheme in the context of a switched surveillance scenario are time taken to deliver the video stream to the operator, size and quality of the image, the rate of video frame update (framerate) and the initialization time for the client to receive one complete image frame from the server. We compared the performance of using H.261, H.263 and MJPEG as the encoding/decoding scheme for the visual surveillance tasks. The various choices for the video subsystem implementation have their advantages and disadvantages based on the scenario they are being applied to. We propose a scenario/task based metric which can be used to evaluate the suitability of the various schemes based on their characteristics. The parameters for evaluation include:

1. Codec complexity

2. Bitrate of generated video stream

3. Switched video initialization time

Codec complexity can be represented by the amount of time taken for the capture and compression of an individual image and may be an important factor for implementation on computationally constrained systems such as embedded processors. The bitrate of the generated video stream represents the bandwidth consumed for transporting the video stream to the operator and can be a limiting factor for wireless or power constrained sensor nodes. Switched video initialization time represents the amount of time taken to display the relevant objects (targets or environment) after a video stream switch has occurred. The proposed metric involving these parameters is:

$$M = \frac{K_{LE}}{T_{LE}} + \frac{K_B}{B} + \frac{K_C}{T_C} \tag{6.9}$$

where, K_{LE}, K_B, K_C are gain coefficients chosen on a per-scenario/task basis based on a fuzzy rule. T_{LE} and T_C represent the switched video initialization time and capture and compress time, respectively, and B represents the bitrate of the generated stream.

The surveillance tasks can be divided into two categories - namely, monitoring and tracking tasks. In the monitoring task, the operator must evaluate the actions of the target, which may require the knowledge of the environment w.r.t the target, while the tracking task requires to only keep the target in view. Hence, for the tracking task, in order to calculate the real switching time for the video system, we need to account only for the display of a moving target, while for the monitoring task, we need to evaluate the switching time for the moving target as well as the static environment.

Based on the characteristics of the task/scenario being evaluated, the values of the gain coefficients are chosen based on a fuzzy rule. For example, if the sensor nodes are not constrained by the processing requirements or power consumption (implemented on regular PC's), the codec complexity will not have much effect on the system performance, so we assign a very low value to the gain coefficient K_C. Similarly, if the nodes are networked using a wired LAN (100Mbps ethernet) infrastructure having large enough bandwidth, then we assign a low value to the gain coefficient K_B. In contrast, if the nodes are sharing a low-bandwidth wireless channel, K_B should be assigned a higher value which signifies the importance of the bandwidth-consumed parameter.

The various task/scenario combinations and their respective gain values we have compared for the video system evaluation are tabulated in Table 6.1. In the scenarios

Table 6.1: Scenarios and gain values

#	Scenario	K_{LE}	K_B	K_C
1	Wired network, monitoring task	1	0	0
2	Wired network, tracking task	1	0	0
3	Wireless network, monitoring task	1	1000	0
4	Wireless network, tracking task	1	1000	0

implemented, the sensor nodes are implemented on general purpose PC's running Linux and hence the capture and compress time is not very crucial to the performance comparison of the various schemes. The software implementation of the codecs for the tests was a modified version of the Vic software. The video streams were packetized using the RTP transport protocol.

6.8.1 Performance Analysis for Capture and Processing Time

The capture time and compression time for one image frame for H.263 and MJPEG is shown in Table 6.2. The frame rate is set to 25 fps, the bit rate is set to be 1.5 M and the image size is set to CIF resolution (352×288) for the experiments.

Table 6.2: Capture and Compression Time of the Two Schemes

Scheme	MJPEG	H.261	H.263
Capture & Compression	12.65ms	30.09ms	32.58ms

We noticed that for H.261 and H.263, when the scene has dramatic changes, the compression time tends to increase, and in the case of MJPEG the compression time is quite constant. Since both MJPEG and H.263 are symmetric coding/decoding schemes, the decompression time should be equivalent to the compression time.

6.8.2 Comparison for Video Size and Image Quality

The H.263 standard is very limited in its capability for selection of the frame size, which is limited to 5 standard resolutions, namely, CIF, QCIF, SIF, 4CIF and 16CIF. This may be a limitation when integrating multiple types of camera sensors. On the other hand MJPEG allows for almost all resolutions (limited to multiples of 8, of course) and can be used to transmit very high definition images when required. The viewing quality of the transmitted images can be largely regarded as the same because both schemes use DCT (discrete cosine transform) and quantization for the compression. However, it should be noted that using MJPEG, the complete image is updated at once, while for H.263, the image is updated in parts and may cause a visual degradation of the perceived image quality.

6.8.3 Performance Analysis for Video Bitrate and Frame Rate

Due to the inter-frame (temporal) compression implementation, the video bitrate generated per frame for the H.261/H.263 schemes is lower compared to MJPEG. This makes H.261/H.263 schemes more suitable for video communications over a restricted bandwidth network. The frame rates and bitrates for the two schemes generated for fixed and panning cameras are tabulated in Table 6.3. The quantization 'Q' values [57] used are based on a visual notion of picture clarity and are noted in the table.

Table 6.3: Frame Rate and Bitrate for MJPEG, H.261 and H.263

Frame	MJPEG: Q=30		H.261: Q=10		H.263: Q=10	
Rate	Bitrate(KBPS)		Bitrate(KBPS)		Bitrate(KBPS)	
(fps)	panning	static	panning	static	panning	static
10	668	668	550	25	700	20
20	1300	1300	1000	45	1000	38
25	1700	1700	1500	60	1300	48

6.8.4 Performance Analysis for Server Switching and Camera Hand-off

The initialization time of H.261, H.263 and MJPEG are also measured. The initialization time is defined as the time from when the server starts broadcasting to when the client receives the first image frame. In all three cases, the initialization time is around 10ms.

For H.261 and H.263 schemes, since inter-frame coding is used, there will be long delay when a client joins a session where a server is already broadcasting to other clients. The effect is termed "late entry" and is caused due to the use of inter-frame encoding, where each frame is not independent of the others but must be decoded using some information from other frames. The stream switching delay due to late entry can be large when using these encoding schemes, especially when the scene is relatively static. The late entry problem does not exist for the MJPEG coding scheme because all the frames transmitted are intra-coded and do not rely on information in the previous or subsequent frames.

For the H.261 scheme implementation the server is only required to transmit those master blocks that have changed through consecutive frames. Thus the client has to wait until all the master blocks are transmitted once to be able to see the whole scene. Based on our implementation methodology for the H.261 scheme, a moving target will be displayed almost immediately while the static environment will take a longer time to display after the switch, which implies that the switching time will be low for a tracking task and relatively high for a monitoring task where the static environment needs to be taken into account.

The H.263 standard allows the transmission of a completely intra-coded frame, called the 'I' frame, to be transmitted periodically in order to negate the effects of accumulated errors. In our experiments for the H.263 coded stream carried out at various frame rate settings, the client has to wait for approximately 5-10 seconds to decode and display the first image frame because it can start to decode a stream only after it has received the first 'I' frame. Table 6.4 tabulates the maximum time taken by the client to display all the relevant macro blocks for the first time when the client joins the broadcast session late for both the tracking and monitoring tasks. The maximum switching time due to late entry problem is compared for various frame rates of the video streams with a very large bandwidth.

Table 6.4: Server Switching Time for Monitoring and Tracking Tasks

Frame rate	Monitoring task			Tracking task		
(fps)	MJPEG	H.261	H.263	MJPEG	H.261	H.263
1	1s	32s	9.3s	1s	1s	9.5s
10	0.1s	3.2s	9.8s	0.1s	0.12s	9.7s
25	0.06s	1.3s	9.2s	0.06s	0.05s	9.4s

Note that the times given in Table 6.4 are the measured values for the display to update. We notice that the switching time for the H.261 scheme is significantly lower for the tracking task than for the monitoring task because the macro blocks comprising the moving target will be updated and transmitted almost instantly; however, the static environment blocks will not be updated until they age and expire at the end of their 32-frame update cycle.

It should be noted that for the H.263 coding scheme, all the information needed to initialize the decoder is stored in an intra coded 'I' reference frame, which is transmitted periodically. One method to combat this late entry problem would be to transmit an 'I' frame every time a new client joins the session, and can be implemented using the RTCP (control commands) part of the RTP protocol. This may lead to higher bandwidth consumption, but may be acceptable in wired surveillance applications.

6.8.5 Performance Metric Evaluation for Various Surveillance Network Tasks

Based on the experimental results reported above, the metric M evaluated for the various scenarios and the codec schemes is tabulated in Table 6.5. The video frame rate was held constant at 25 fps with the image quality (Q) being held at 30 for MJPEG and 10 for H.261/H.263. The cameras are assumed to be stationary.

Table 6.5: Metric for Comparison of Various Schemes

Scenario #	MJPEG	H.261	H.263
1: Wired, monitoring	16.67	0.769	0.109
2: Wired, tracking	16.67	20	0.109
3: Wireless, monitoring	17.26	17.43	20.94
4: Wireless, tracking	17.26	36.67	20.94

We notice that the H.261 implementation has best performance for the wired and wireless tracking scenarios, whereas for the wired monitoring scenario, MJPEG has the best performance and the H.263 implementation manages better performance for the wireless monitoring scenario. It must be kept in mind that the performance evaluation for the various schemes are qualified by their implementation methodologies outlined in section 5.5. A brief discussion regarding the advantages and disadvantages of using the various video subsystems is presented here.

The MJPEG system can transmit video frames of various sizes and hence has the advantage of being able to handle heterogenous hardware for video capture. Further it has a low coding and initialization latency and allows for a direct hardware implementation, which is advantageous for using lower processing power on the sensor node. It does not suffer from the late entry problem during switching. However, it requires a consistently high bandwidth, which may limit its application on wireless or low bandwidth communication channels.

H.261 and H.263 encoding schemes consume less bandwidth, allowing for higher frame rates, which is an important characteristic for continuous and responsive real-time monitoring. The drawbacks of these schemes are that they can handle video frames of certain specified sizes only and that both initialization time and switching time are higher than in MJPEG. They also suffer from the late entry problem, which can be detrimental for systems involving video stream switching and nodes linked over unreliable channels. However, the late entry problem can be solved at the expense of transmitting the complete intra-frame-encoded image when requested using the RTCP protocol.

6.9 Chapter Summary

This chapter presents the experimental implementation details of the various algorithms proposed earlier. The image-based Hausdorff tracking method has been implemented for tracking a single and multiple targets simultaneously. This tracking of multiple disjoint targets moving independently is a major advantage of the proposed method, as the current existing image-based tracking techniques, such as visual servoing and gaze control, cannot achieve this. Results for the cooperative Hausdorff tracking are also presented. The two multi-sensor cooperation methods proposed are also validated experimentally. A task-based metric is also proposed for evaluating the suitability of the various video feedback schemes.

Chapter 7

Conclusions

This dissertation has presented an in-depth examination of the various aspects of planning and control of mobile surveillance networks, concentrating particularly in the broad areas of modeling, analysis and design of mobile surveillance networks. This chapter discusses the conclusions and gives a view towards future areas of research.

7.1 Conclusions

The non-intrusive nature of cameras makes them an attractive choice for sensors for various surveillance applications. In recent times, with the decrease in the cost of camera sensors accompanied with increasing processing power and availability of wireless networking capabilities in sensors of exceedingly smaller size, large-scale target tracking systems utilizing a large number of cameras are being increasingly deployed for myriad surveillance and tracking tasks. Sensor mobility accompanied with large scale wireless networking can overcome the limitations due to range, direction and occlusion of visual sensing. However, a large number of sensors with motion capabilities tends to increase the modeling and analysis complexity of such systems.

For successful completion of the surveillance task, it is important to allocate the sensors to the targets that are actively being tracked. Generally, each target should be tracked by the sensor that can view it at an optimal resolution for best visibility of that target. However, when the system is heavily loaded, that is, the target density in a particular region with a limited number of sensors is high, one sensor might need to track multiple targets. Even when the system is under-loaded, multiple target tracking by individual sensors is highly desirable, as viewpoint redundancy will lead to better estimates of the target coordinates and also present the viewer a clearer view of all the targets that are being tracked.

In order to track a target over a wide area with a distributed surveillance network, the tracking sensors need to be successively switched and the tracking task needs to be handed over from one sensor to the next. This switching, while increasing the modeling and analysis complexity, also introduces the design problem of optimally locating the sensors (i.e., configuring the sensors) in order to maximize the tracking performance. Switching the tracking task from one sensor to another tends to be quite disorienting for a human operator and hence needs to be minimized while tracking a target over a large region, while also maintaining adequate resolution of the target to allow discerning various target features for classification and identification of targets.

This dissertation presents various novel approaches to solve the aforementioned problems associated with mobile surveillance networks. The original contributions of this dissertation can be broadly classified into modeling, analysis and design contributions, which are individually addressed below.

Modeling Contributions Using the concepts of continuous and discrete motion of cameras, a generalized modeling framework for conducting pervasive surveillance has been developed. First, we have developed a novel domain-based modeling approach, christened "Hausdorff tracking", for visually tracking multiple targets with a single sensor while simultaneously accomplishing various other sub-tasks such as maintaining adequate resolution of the tracked targets. Hausdorff tracking uses a mutational-analysis-based approach to express the motion of the target and sensor coverage sets with respect to the camera motion input. The various task requirements can be specified using shape functions which can be construed as analogs of vector errors in a vector-based control system.

Using the concept of mutations of domains and shape analysis, the dynamics (change in shape) of the sensor field of view (FOV) and target domains are described and further feedback control mechanisms have been derived to complete the specified task. The assumptions on the control inputs to the camera for stability of the control system can be calculated using the shape Lyapunov theorem. Hausdorff tracking provides a significant advantage over classical vector space servoing as it can express the multiple target tracking succinctly and perform the tracking task in an energy-efficient manner. The methods of image-based Hausdorff tracking and cooperative Hausdorff tracking are introduced to perform target tracking using target information either directly sensed from the camera image plane or acquired from other sensors using the communication module, respectively.

In order to pervasively track a target over a large area covered by multiple sensors, the mutational hybrid automata (MHA) model is proposed. The MHA model is capable of capturing the inherent hybrid (discrete switching with continuous motion) nature of the pervasive surveillance task - i.e., it accounts for both continuous camera motion and discrete jumps in the location of the tracking camera, which indicates a switch of the tracking camera from one sensor to another. The MHA model for pervasive surveillance can be used to model "hand-off" scenarios as the targets move across the coverage regions of multiple networked sensors. It allows modeling of both autonomous and controlled switches, which provides for a rich framework for hybrid control of the various tracking cameras based on the location of the target and other environment inputs.

Analysis Stability properties of the Hausdorff tracking controllers are proved using the shape Lyapunov theorem. Using the shape Lyapunov theorem, it is shown that the shape function (error) converges to zero at least as fast as a stable first-order linear system. Stability for a pervasive tracking scenario implies that when a target is detected by a camera in the network, its visibility will always be ensured while it moves around in the monitored system. Using examples of pervasive surveillance scenarios, a two-step procedure is proposed to show the stability properties of the MHA model for modeling pervasive surveillance tasks. The first step shows that all the modes of the MHA model are internally stable - i.e., continuous evolution within each mode is stable in the sense of Lyapunov and there are no Zeno executions. The next step is to show that a particular set of stable modes in the MHA model forms an invariant set - i.e., if the hybrid execution enters this invariant set, it will remain in this set for all time. Now if we limit the Init set of the MHA model to the invariant set, the execution will never leave that set. This implies that if we can prove that the invariant set is the set of all modes where at least one camera can view the target, then if the target enters the field of view of any one camera, it will be tracked continuously as long as it remains in the monitored region. This procedure is developed for both models with and without controlled switches.

Design A wide area multi-camera location and pose calibration scheme is presented that can calibrate the many heterogenous cameras with respect to a global coordinate system in a short time by simply waving a bright light source in the region observed by the cameras. This method can be used even when all cameras do not have an overlapping field of view. The problem of optimally placing a number of cameras in a given region to facilitate a surveillance task is presented. Metrics for evaluating the performance of the tracking task based on the target resolution and number of switches to pervasively track the target are presented. Another problem we presented is related to camera switching strategy given a full or part of a trajectory path of a moving target. A dynamic-programming-based approach to generate the switching strategy and optimize both the switching and resolution metrics is presented.

The Hausdorff tracking problem is extended to incorporate an optimal control framework for executing multiple subtasks along with the target visibility task for a surveillance network. This optimization framework allows the surveillance task designer to consider various sub-tasks in the planning phase. Various sub-tasks can include minimizing the energy consumed by the network, which will enhance the longevity of the deployed network sensors. Physical programming is used for finding a solution to the optimization problem in order to lend a tangible and physical interpretation to the subtasks being considered.

The design of a switched video feedback system and a sensor node architecture are also presented. The various alternatives for the switched video feedback namely MJPEG, H.261, H.263 and H.264 are evaluated and a task- and scenario-based metric is proposed to analyze their performance. The MJPEG scheme does not suffer from the late entry problem, but the bandwidth requirement is constantly high. H.261 and H.263 schemes consume low bandwidth for relatively static scenes but suffer from the late entry problem when video switching is required. A switched video feedback system using a multiple-client, multiple-server architecture using the H.264 video encoding scheme is implemented.

Implementation and testing A surveillance test-bed consisting of seven heterogenous cameras with motion capabilities for validation and testing of the proposed schemes and algorithms is implemented. The sensors are calibrated using the wide area calibration scheme proposed and various target tracking scenarios are implemented and tested. Experimental results validating the proposed theoretical approaches are presented.

The various theoretical approaches and tools presented in this dissertation will enable the design, analysis and implementation of real-time mobile surveillance systems.

Overall Impacts All together this dissertation presents a collection of modeling analysis and design tools for mobile surveillance networks. The impacts of this dissertation to various fields are summarized below.

Impact on the field of surveillance and sensor networks: This dissertation presents significant advances for modeling analysis and design for the field of surveillance and sensor networks. Modeling multiple target tracking using the Hausdorff tracking method provides a significant advantage for distributed sensor systems as it can significantly improve sensing accuracy due to multiple detections of the same target with different sensors. The novel stability analysis approach presented for guaranteing target coverage using distributed sensors is a significant step forward from the current approaches of probabilistic modeling.

Impact on the field of robotics and control: This dissertation presents a major contribution to the field of robotics and control. Vision based robot control has suffered from the problem of task specificity i.e., the tracking task is generally over specified and also lacked a generalized modeling structure for modeling multi target visibility problems. The method of Hausdorff tracking proposed in this dissertation represents a generalized set based modeling structure for modeling traditional vision based servoing. It can in fact be viewed as the super set of visual servoing as traditional visual servoing tasks can also be readily expressed in this framework.

The use of mutational equations and shape analysis for representing shape based control is a significant step towards extending the rich tool-set of control literature to domain based control. Generalizing a vector based model to a domain based model opens up a lot of avenues for creative design of control algorithms. Many real-world tasks which needed to be parameterized in order to be expressed in a controls theoretic framework can be directly solved using this domain based approach.

Impact on general science and engineering: This dissertation presents various results on suitability of various video streaming approaches for surveillance applications. It provides a method to measure the acceptability of various streaming solutions which will have a significant impact on the implementation of networking and IP-based video transmission.

Impact on society in general: There are many issues up for debate on privacy and intrusion of it thereof. However, these concerns should not preclude the pervasive monitoring and surveillance of critical infrastructure. In this current world scenario, exhibiting a little precaution with regards to safety is, in my opinion, not a bad idea. This dissertation provides tools for conducting pervasive surveillance which will have many potential uses with regards to monitoring critical infrastructure such as airports power plants etc and also many applications in our fuller understanding of the world around us i.e., in environmental monitoring for understanding and preserving natural habitats. However there are also many possible uses for this technology such as continuous tracking of people with express intent of invasion of their privacy. But all technology like a double edged sword has it's advantages and disadvantages. It is behooved on the people who use technology to be fair, just and moral in the application of technology.

7.2 Future Research Directions

Large-scale surveillance systems have only recently been commercialized, and the architecture and capabilities of these systems are still an active research area as many problems still remain to be solved. This dissertation has addressed several issues in modeling, analysis and design of mobile surveillance systems. Certainly more can be done to elaborate on the work presented in this dissertation and even more can be done to address various other issues in mobile surveillance which have not been the topic of this dissertation.

Shape functions for performing visibility- and resolution-based tasks have been developed in this dissertation. However, more general classes of admissible shape functions should be developed in order to provide a rich set of tools to design surveillance tasks.

The mutational hybrid automata model presented uses an edge set in order to describe the discrete switching dynamics of the pervasive target tracking model. It should be noted that the number of edges and modes required to model the pervasive surveillance task increases exponentially as the number of cameras involved in the surveillance task grows. This somewhat limits the applicability of the proposed approach of using edge sets for modeling discrete dynamics. In order to alleviate this problem, a language-based automata model can be used with an alphabet capable of modeling very large-scale camera systems.

The proposed MHA model can be used for ensuring the visibility of a single target moving in a region. The possibility of combining multiple such single-target MHA models in order to track multiple targets simultaneously using the same network should be investigated. Optimal control for the proposed mutational hybrid automata model is still an open problem.

The switched video feedback system is implemented using a baseline encoder. A high profile encoder with object-level support to code the various objects separately would certainly enhance the video feedback system by reducing the bandwidth requirement for the surveillance system. The support of user-defined queries and tasks will certainly enhance the usability of the surveillance system, making it more flexible and adaptable for a real-world surveillance scenario.

There has been relatively little research done on camera deployment and placement issues for tracking applications. The switching-based metric presented in this dissertation serves as a first step towards the final goal of optimal camera placement. A number of avenues can be pursued to carry this work further. A user-friendly camera placement planning system that utilizes the switching metric amongst others would provide a very useful tool for optimal camera placement.

AUTHOR'S PUBLICATIONS

1. A. Goradia, M. Huntwork, C. Haffner, N. Xi, and M. Mutka, "Modeling and Control of Pervasive Surveillance Networks using Mutational Hybrid Systems," in IEEE IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

2. A. Goradia, C. Haffner, N. Xi and M. Mutka, "Optimality Framework for Hausdorff Tracking using Mutational Dynamics and Physical Programming," *submitted to* IEEE International Conference on Robotics and Automation, 2007.

3. A. Goradia, Z. Cen, C. Haffner, N. Xi and M. Mutka,"Sensor Selection Problem using Dynamic Programming," *submitted to* IEEE International Conference on Robotics and Automation, 2007.

4. A. Goradia, C. Haffner, Z. Cen, M. Mutka and N. Xi, "Modeling, Analysis and Design of Mobile Surveillance Networks Using Mutational Hybrid Systems," Submitted to IEEE Transactions on Robotics.

5. A. Goradia, M. Huntwork, C. Haffner, C. Klochko, N. Xi, and M. Mutka, "Pervasive Surveillance using a Cooperative Mobile Sensor Network," in IEEE International Conference on Robotics and Automation, 2006.

6. A. Goradia, Z. Cen, C. Haffner, Y. Liu, B. H. Song, M. Mutka and N. Xi, "Pervasive Surveillance Networks: Design, Implementation and Performance Analysis," *Accepted*, International Journal of Distributed Sensor Networks.

7. A. Goradia, Z. Cen, C. Haffner, N. Xi and M. Mutka, "Design, Implementation and Performance Analysis of Pervasive Surveillance Networks", 19th International FLAIRS conference, 2006.

8. A. Goradia, N. Xi, Z. Cen, and M. Mutka, "Modeling and Design of Mobile Surveillance Networks using a Mutational Analysis Approach," in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.

9. A. Goradia, N. Xi, M. Prokos, Z. Cen, and M. Mutka, "Cooperative Multi-Target Surveillance using a Mutational Analysis Approach," in Proceedings of 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2005), Monterey, California, USA, 2005.

10. A. Goradia, I. Elhajj, and N. Xi, "Internet Based Robots: Applications, Impacts, Challenges and Future Directions," in IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO), 2005.

11. Z. Cen, A. Goradia, Y. Liu, N. Xi, M. Mutka, "Improving the Operation Efficiency of Internet-Based Teleoperation via Overlay Networks," *Submitted*, IEEE Transactions on Robotics.

12. Z. Cen, A. Goradia, M. Mutka, N. Xi, W.-K. Fung, and Y.-h. Liu, "Improving the Operation Efficiency of Supermedia Enhanced Internet Based Teleoperation via an Overlay Network," in IEEE International Conference on Robotics and Automation, Barcelona, Spain, 2005.

13. Z. Cen, M. Mutka, Y. Liu, A. Goradia, and N. Xi, "QoS Management of Supermedia Enhanced Teleoperation via Overlay Networks," in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Edmonton, Alberta, Canada, 2005.

14. A. Goradia, Z. Nichol, Y. Liu, P. Suchyta, M. Prokos, and N. Xi, "Super-Media Enhanced Internet-Based Real-Time Teleoperation," in IEEE International Conference on Mechatronics and International Conference on Hands-on Intelligent Mechatronics and Automation (ICM-HIMA), 2005.

15. A. Goradia, L. Pan and N. Xi, "Dynamic Multi-Objective Optimal Task Distribution for Teleoperated Mobile Manipulators," in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004.

16. J. Tan, A. Goradia, N. Xi, and W. Sheng, "Coordination of Multi-Robot and Human Systems in a Perceptive Reference Frame," International Journal of Vehicle Autonomous Systems, vol. 2, 2004.

17. I. Elhajj, A. Goradia, N. Xi, C. M. Kit, Y.-H. Liu, and T. Fukuda, "Design and Analysis of Internet-Based Tele-Coordinated Multi-Robot Systems," Autonomous Robots, vol. 15, no. 3, 2003.

18. I. Elhajj, A. Goradia, N. Xi, C. M. Kit, Y.-H. Liu, and T. Fukuda, "Tele-Coordinated Control of Multi-Robot Systems via the Internet." in IEEE International Conference on Robotics and Automation, 2003.

19. J. Tan, A. Goradia, N. Xi, and W. Sheng, "Coordination of Human and Mobile Manipulator Formation in a Perceptive Reference Frame." in IEEE International Conference on Robotics and Automation, 2003.

20. I. Elhajj, A. Goradia, N. Xi, W. T. Lo, Y.-H. Liu, and T. Fukuda, "Internet-Based Tele-Manufacturing," in The Seventh International Conference on Automation Technology, Chia-yi, Taiwan, 2003.

21. Amit Goradia, Jindong Tan, Ning Xi, "Hybrid Force/Position Control in Moving Coordinate Frames," International Conference of Control, Automation, Robotics and Computer Vision, Singapore, 2002.

BIBLIOGRAPHY

[1] C. S. Regazzoni, G. Fabri, and G. Vernazza, *Advanced Video-Based Surveillance Systems.* Norwell, MA: Kluwer, 1998.

[2] G. L. Foresti, P. Mahonen, and C. S. Regazzoni, *Multimedia Video-Based Surveillance Systems: Requirements, Issues and Solutions.* Norwell, MA : Kluwer, 2000.

[3] M. Bogaert, N. Chelq, P. Cornez, C. S. Regazzoni, A. Teschioni, and M. Thonnat, "The password project," in *Proceedings of the International Conference on Image Processing*, Chicago , IL, 1996, p. 675 678.

[4] C. S. Regazzoni and A. Tesei, "Distributed data fusion for real-time crowding estimation," *Signal Processing*, vol. 53, p. 47 63, 1996.

[5] E. F. Lyon, "The application of automatic surface lights to improve airport safety," *IEEE AES Systems Magazine*, p. 14 20, 1993.

[6] M. Braasch, M. DiBenedetto, S. Braasch, and R. Thomas, "Laas operations in support of airport surface movement, guidance, control and surveillance: Initial test results," in *IEEE Proceedings on Position Location and Navigation Symposium*, 2000, p. 82 89.

[7] H. Wang, T. Kirubarajan, and Y. Bar-Shalom, "Precision large scale air traffic surveillance using imm/assignment estimators," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 35, p. 255 266, January 1999.

[8] B. Sridhar and G. B. Chatterji, "Computationally efficient conflict detection methods for air traffic management," in *Proceedings of the American Control Conference*, vol. 2, 1997, p. 1126 1130.

[9] G. Donohue, "Vision on aviation surveillance systems," in *Proceedings of the IEEE International Radar Conference*, 1995, p. 1 4.

[10] G. L. Foresti and B. Pani, "Monitoring motorway infrastructures for detection of dangerous events," in *Proceedings of the IEEE International Conference on Image Analysis and Processing*, Venice, Italy, 1999, p. 1144 1147.

[11] J. M. Manendez, L. Salgado, E. Rendon, and N. Garcia, "Motorway surveillance through stereo computer vision," in *Proceedings of the 33rd IEEE Annual International Carnahan Conference on Security Technology*, 1999, p. 197 202.

[12] D. Koller, K. Daniilidis, and H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *International Journal on Computer Vision*, vol. 10, p. 257 281, 1993.

[13] J. Malik, D. Koller, and J. Weber, "Robust multiple car tracking with occlusion reasoning," in *European Conference on Computer Vision*, Stockolm, Sweden, 1994, p. 189 196.

[14] A. N. Ince and E. Topuz, "The design and computer simulation of a maritime surveillance system," in *International Confference on Radar*, 1997, p. 653 656.

[15] J. G. Sanderson, M. K. Teal, and T. J. Ellis, "Target identification in complex maritime scenes," in *6th International Conference on Image Processing and its Applications*, vol. 2, 1997, p. 463 467.

[16] C. A. Rodriguez, J. A. Howell, H. O. Menlove, C. M. Brislawn, J. N. Bradley, P. Chare, and T. Gorten, "Nuclear video image pro-cessing for nuclear safeguards," in *Proceedings of the IEEE 29th Annual International Carnahan Conference on Security Technology*, 1995, p. 355 363.

[17] E. Stringa and C. S. Regazzoni, "Content-based retrieval and real time detection from video sequences acquired by surveillance systems," in *IEEE International Confnference on Image Processing*, vol. 3, Chicago, IL, October 4-7 1998, p. 138 142.

[18] C. Lin and R. Nevatia, "Building detection and description from a single intensity image," *Computer Vision and Image Understanding*, vol. 72, no. 2, p. 101 121, 1998.

[19] L. Vergara and P. Bernabeu, "Automatic signal detection applied to fire control by infrared digital signal processing," *Signal Processing*, vol. 80, no. 4, p. 659 669, 2000.

[20] W. Millesi, M. J. Truppe, F. Watzinger, A. Wagner, and R. Ewers, *Lecture Notes in Computer Science.* Berlin, Germany: Springer, 1997, vol. 1205, ch. Image guided surgery extended by remote stereotactic visualization, p. 813 821.

[21] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Transaction on Pattern Analalysis and Machine Intelligence*, vol. 22, p. 809830, August 2000.

[22] N. M. Oliver, B. Rosario, and A. P. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE Transactions on Pattern Analalysis and Machine Intelligence*, vol. 22, no. 8, p. 831843, August 2000.

[23] D. A. Pritchard, "System overview and applications of a panoramic imaging perimeter sensor," in *Proceedings of the IEEE 29th Annual International Carnahan Conference on Security Technology*, 1995, p. 420425.

[24] U. Oppelt, "New possibilities for video applications in the security field," in *Proceedings of the IEEE 29th Annual International Carnahan Conference on Security Technology*, 1995, p. 426435.

[25] B. Peters, J. Meehan, D. Miller, and D. Moore, "Sensor link protocol: Linking sensor systems to the digital battlefield," in *Proceedings of the IEEE Military Communications Conference*, vol. 3, 1998, p. 919923.

[26] M. T. Fennell and R. P. Wishner, "Battlefield awareness via synergistic sar and mti exploitation," *IEEE Aerospace and Electronic Syststems Magazine*, vol. 13, p. 3943, February 1998.

[27] J. S. Draper, S. Perlman, C. K. Chuang, M. Hanson, L. Lillard, B. Hibbeln, and D. Sene, "Tracking and identification of distant missiles by remote sounding," in *Proceedings of the IEEE Aerospace Conference*, vol. 4, 1999, p. 333341.

[28] G. A. V. Sickle, "Aircraft self reports for military air surveillance," in *Proceedings of the IEEE Digital Avionics Systems Conference*, vol. 2, 1999, p. 28.

[29] K. Toyoma, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintainence," in *International Conference on Computer Vision*, 1999, pp. 255–261.

[30] I. Elhajj, N. Xi, W. K. Fung, Y.-H. Liu, Y. Hasegawa, and T. Fukuda, "Supermedia-enhanced internet-based telerobotics," *Proceedings of the IEEE*, vol. 91, no. 3, pp. 396 – 42, March 2003.

[31] N. Xi and T. J. Tarn, "Action synchronization and control of internet based telerobotic systems," in *IEEE International Conference on Robotics and Automation*, 1999.

[32] C. Regazzoni, V. Ramesh, and G. E. Foresti, "Special issue on third generation surveillance systems," *Proceedings of the IEEE*, vol. 89, Oct. 2001.

[33] R. Collins, A. Lipton, and T. Kanade, "Introduction to the special section on video surveillance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 745–746, August 2000.

[34] J. Tan, N. Xi, W. Sheng, and J. Xiao, "Modeling multiple robot systems for area coverage and cooperation," in *International Cconference on Robotics and Automation*, 2004.

[35] S. Stoeter, P. Rybski, M. Erickson, M. Gini, D. Hougen, D. Krantz, N. Papanikolopoulos, and M. Wyman, "A robot team for exploration and surveillance: Design and architecture," in *Proceedings of the International Conference on Intelligent Autonomous Systems*, 2000, pp. 767–774.

[36] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," in *International Conference on Robotics and Automation*, 2003.

[37] T.Matsuyama and N.Ukita, "Real-time multitarget tracking by a cooperative distributed vision system," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1136–1150, 2002.

[38] T. Boult, R. J. Micheals, X. Gao, and M. Eckmann, "Into the woods: Visual surveillance of noncooperative and camouflaged targets in complex outdoor settings," *Proceedings of the IEEE*, vol. 89, Oct. 2001.

[39] R.T.Collins, A.J.Lipton, H.Fujiyoshi, and T.Kanade, "Algorithms for cooperative multisensor surveillance," *Proceedings of the IEEE*, vol. 89, pp. 1456–1477, 2001.

[40] R. Brooks, C. Griffin, and D. Friedlander, "Self-organized distributed sensor network entity tracking," *International Journal of High Performance Computing*, vol. 16, no. 2, 2002.

[41] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.

[42] C. Brown, "Gaze control with interactions and delays," *IEEE Transactions on Systems Man and Cybernetics*, vol. 20, no. 1, pp. 518–527, 1990.

[43] M. Branicky, V. Borkar, and S. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *IEEE Transactions on Automatic Control*, vol. 43, no. 1, pp. 31–45, 1998.

[44] J. Lygeros, K. Johansson, S. Simic, J. Zhang, and S. S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 2–17, 2003.

[45] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed Diffusion for Wireless Sensor Networking," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, 2003.

[46] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 22–31, 2002.

[47] H. S. Hamza and S. Wu, "RAODV: an Efficient Routing Algorithm for Sensor Networks," *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 297–298, 2004.

[48] R. T. Collins, A. J. Lipton, and T. Kanade, "Introduction to the special section on video surveillance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, p. 745746, August 2000.

[49] A. J. Courtemanche and A. Ceranowicz, "Modsaf development status," in *Proceeding of the 5th Conference on Computer Generated Forces and Behavioral Representation*, Orlando, FL, May 1995, p. 313.

[50] K. Cauble, "The openscene modstealth," in *Proceedings of the Simulation Interoperability Workshop*, Paper 97S-SIW-008 1997.

[51] C. S. Regazzoni, C. Sacchi, and C. Dambra, "Remote cable-based video surveillance applications: the avs-rio project," in *Proceedings of ICIAP99*, Venice , Italy, September 2729 1999, p. 12141215.

[52] P. Rybski, S. Stoeter, M. Gini, D. Hougen, and N. Papanikolopoulos, "Performance of a distributed robotic system using shared communications channels," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 5, pp. 713 – 727, October 2002.

[53] ITU-H.261, "Video codec for audio-visual services at 64-1920 kbit/s," *ITU-T Recommendation H.261*, 1993.

[54] ITU-H.263, "Video coding for low bit rate communication," *ITU-T Recommendation H.263*, March 1996.

[55] I.-T. R. H. . I. 11496-10, "Advanced video coding," *Final Committee Draft, Document JVT-E022*, September 2002.

[56] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "Rtp: A transport protocol for real-time applications," *RFC1889*, July 1997.

[57] L. Berc, W. Fenner, R. Frederick, S. McCanne, and P. Stewart, "Rtp payload format for jpeg-compressed video," *RFC2435*, October 1998.

[58] E. Sontag, *Mathematical Control Theory.* Springer-Verlag, 1990.

[59] J.-P. Aubin, "Mutational equations in metric spaces," *Set-Valued Analysis*, vol. 1, pp. 3–46, 1993.

[60] ——, *Mutational and Morphological Analysis: Tools for Shape Evolution and Morphogenesis.* Birkhäuser, 1999.

[61] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, 2000.

[62] J. Cea, "Problems of shape optimal design," *Optimization of Distributed Parameter Structures vol. I and II*, pp. 1005–1087, 1981.

[63] J. Sokolowski and J.-P. Zolesio, *Introduction to Shape Optimization: Shape Sensitivity Analysis*, ser. Computational Mathematics. Springer-Verlag, 1991.

[64] A. C. Sanderson, L. Weiss, and C. Neuman, "Dynamic sensor based control of robots with visual feedback," *IEEE Transactions on Robotics and Automation*, vol. RA-3, pp. 404–417, October 1987.

[65] L. Doyen, "Shape laypunov functions and stabilization of reachable tubes of control problems," *Journal of Mathematical Analysis and Applications*, vol. 184, pp. 222–228, 1994.

[66] ——, "Mutational equations for shapes and vision-based control," *Journal of Mathematical Imaging and Vision*, vol. 5, no. 2, pp. 99–109, 1995.

[67] A. Goradia, Z. Cen, M. Mutka, and N. Xi, "Modeling and design of mobile surveillance networks using a mutational analysis approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[68] A. Goradia, N. Xi, M. Prokos, Z. Cen, and M. Mutka, "Cooperative Multi-Target Surveillance Using a Mutational Analysis Approach," in *Proceedings of 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2005)*, Monterey, California, USA, 2005.

[69] K. Fregene, D. Kennedy, and D. Wang, "A study of supervisory constraints in a class of coordinated multiagent systems," in *Proceedings of the American control Conference*, 2003, pp. 1044–1049.

[70] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern and Machine Intell.*, vol. 22, no. 11, pp. 1330–1334, 2000.

[71] ——, "Motion and structure from two perspective views: from essential parameters to euclidean motion through the fundamental matrix," *J. Opt. Soc. Am*, vol. 14, no. 11, 1997.

[72] R. D. Z. Zhang, O. Faugeras, "An effective technique for calibrating a binocular stereo through projective reconstruction using both a calibration object and the environment," *Videre: Journal of Computer Vision Research*, vol. 1, no. 1, 1997.

[73] L. Shapiro and G. Stockman, *Computer Vision.* Prentice Hall, 2001.

[74] X. Chen, J. Davis, and P. Slusallek, "Wide area camera calibration using virtual calibration objects," in *Computer Vision and Pattern Recognition. Proceedings. IEEE Conference on*, 2000.

[75] A. Heyden and K. Astrom, "Minimal conditions on intrinsic parameters for euclidean reconstruction," in *ACCV '98: Proceedings of the Third Asian Conference on Computer Vision-Volume II.* London, UK: Springer-Verlag, 1997, pp. 169–176.

[76] M. Grewal and A. Andrews, *Kalman Filtering: Theory and Practice.* Prentice Hall, 1993.

[77] W.-P. Chen, C.-J. Hou, and L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing Special Issue on Sensor Networks*, 2004.

[78] J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao, "Distributed Group Management for Track Initiation and Maintenance in Target Localization Applications," in *Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN)*, 2003.

[79] Q. Fang, J. Liu, L. Guibas, and F. Zhao, "RoamHBA: Maintaining Group Connectivity In Sensor Networks," in *The 3rd international symposium on Information processing in sensor networks (IPSN2004)*, 2004, pp. 151–160.

[80] M. Mauve, J. Widmer, and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," *IEEE Network Magazine*, vol. 15, no. 6, pp. 30–39, 2001.

[81] R. Timothy and J. Arora, "Survey of multi-objective optimization method for engineering." *http://www.ccad.uiowa.edu/ tmarler/rtm/img/survey.pdf*, Accessed March 2005.

[82] B. S. Gottfried and J. Weisman, *Introduction to Optimization Theory.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1973.

[83] A. Mullur, C. Mattson, , and A. Messac, "New decision matrix based approach for concept selection using linear physical programming," in 44th *AIAA/ASME/ASCE/AHS, Paper No. AIAA 2003-1446*, Norfolk, VA, April 2003.

[84] A. Messac, "Physical programming: Effective optimization for computational design," *AIAA Journal*, vol. 34, no. 1, pp. 149-158, 1996.

[85] ——, "From dubious construction of objective functions to the application of physical programming," *AIAA Journal*, vol. 38, no. 1, pp. 155–163, Jan 2000.

[86] R. Fletcher, *Practical Methods of Optimization, Second Edition.* Anchor Brendon Ltd., Tiptree, Essex, 1991.

[87] Trolltech-Inc, "Qt: A cross platform application development framework," *http://www.trolltech.com/products/qt/index.html*, accessed December 2005.

[88] C. Kim and J.-N. Hwang, "Object-based video abstraction for video surveillance systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 12, December 2002.

[89] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 912-925, December 1998.

[90] R. Pal, Nikhil and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277-1294, 1993.

[91] J. Bruce, T. Balch, and M. Veloso, "Fast and inexpensive color image segmentation for interactive robots," in *IROS*, 2000.

[92] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multi-camera self-calibration for virtual environments," *PRESENCE: Teleoperators and Virtual Environments*, vol. 14, no. 4, pp. 407–422, August 2005.