Automatic Surveillance System

A Project Report Submitted
in Partial Fulfillment of Requirements
for the Degree of

Bachelor of Technology

by

Nikhil Nandyala (2015csb1020)
Allu Krishna Sai Teja (2015csb1005)

Department of Computer Science & Engineering

Indian Institute of Technology Ropar

Rupnagar 140001, India

April 2012

Abstract

Generally, in CCTV Surveillance the camera is fixed in position covering a fixed area which will be monitored. Due to the Limitations of the fixed axis camera, Pan Tilt Zoom (PTZ) Cameras were employed. With PTZ cameras we can change the axis and increase the coverage but we need a human controlling the camera or the camera can be rotated in a static repetitive order irrespective of the environment. In doing so we lose a lot of information. So we designed and implemented a dynamic intelligent scheduling framework for PTZ cameras to overcome the above issues using modern computer vision techniques. This Report contains all the frameworks we explored, implemented, and conclusions reached.

Acknowledgements

We want to thank our BTP guides- Dr.Neeraj Goel and Dr.Mukesh Saini who allowed us to do this wonderful project on the topic-Automatic Surveillance of PTZ camera, which helped us come up with an approach for solving real-life surveillance problems and gave us scope to learn many computer vision algorithms.

Honor Code

We certify that we have properly cited any material taken from other sources and have obtained permission for any copyrighted material included in this report. We take full responsibility for any code submitted as part of this project and the contents of this report.

Nikhil Nandyala (2015csb1020)

Allu Krishna Sai Teja (2015csb1005)

Certificate

It is certified that the B. Tech. project "Automatic Surveillance System " has been done by Nikhil Nandyala(2015csb1020), Allu Krishna Sai Teja (2015csb1005) under my supervision. This report has been submitted towards partial fulfillment of B. Tech. project requirements.

Dr.Mukesh Saini,
Dr.Neeraj Goel
Project Supervisor
Department of Computer Science & Engineering
Indian Institute of Technology Ropar
Rupnagar-140001

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays CCTVs are employed heavily to monitor critical areas like parking lots, banks, railway stations, buildings, malls. Multiple cameras are installed to monitor such kind of areas, but this requires the operator to watch various cams in parallel, by which the operator might miss some activities which are vital as he has to view all the screens simultaneously, also most of the times there would not be any activities which are required to observe. So we can use PTZ (Pan Tilt and Zoom) cameras in such areas. They cover a lot compared to a standard camera as they can move. We are automatically controlling the PTZ camera to reduce the workforce and bring it as close as a human being is monitoring it. General PTZ scheduling algorithms do not consider the anomaly and visit each state after a fixed interval of time. With this kind of algorithms, we might lose potential information as these algorithms are not considering anomalies. There is a tradeoff between information gain and coverage. So our goal is to maximise both of these parameters without compromising any of them. We have suggested many scheduling algorithms for this, and we found randomised implementation as optimal for this kind of a problem.Ferone and Maddalena [2014]

We are dividing the area covered by the camera into several states and scheduling the PTZ camera based on the history of activities that happened in the state.

Also sometimes the user wants to mark a few areas as necessary. So our approach provides a mechanism from which the system can learn based on the user inputs and incorporate these inputs in further scheduling.

Deterministic scheduling approaches do not work in these scenarios because if a camera works deterministically, a person may predict when a camera looks at him and can escape from the view of the camera. It does serve our purpose. So we proposed a combination of deterministic and randomised approach.

Chapter 2

Related Work

2.1 Szoom: A Framework for Automatic Zoom into High Resolution Surveillance Videos

When viewing live surveillance footage, there is an only small part of it that is interesting to the operator. S-zoom by Dr.Mukesh Saini et al is a framework which automatically detects and zooms into the most sensitive areas of fixed view CCTV cameras and concentrates on that area for a fixed amount of time. The framework only digitally zooms, i.e. the camera does not zoom into the area the transmitted video is zoomed to provide more information about the sensitive area without losing any information.

2.1.1 Szoom Framework overview:

- Calculates the sensitivity map for five frames.
- From the Sensitivity map find the region of interest.
- Digitally zoom on to the ROI using cubic spline so that the zoom in is gradual.
- Stay there for five seconds so that the operator can analyse the anomaly.
- If the anomaly moves during the fives seconds track it.

- Zoom back to complete view and repeat the process from step 1 again.

2.1.2 Sensitivity Map Calculation :

Sensitive map contains information about relative sensitiveness of pixels, i.e., how much a pixel is important compared to other pixels at any given time. So they calculate the sensitivity of each pixel as the weighted sum of on the following semantic observations. From Paper Hu et al. [2004] and Kuang et al. [2014] we can see the importance of calculating Sensitivity using the combination of the following features.

- Motion
- Human Detection
- Face Detection

In Surveillance Motion is very important to observe,The presence of a human in the area must be observed, Face Detection is important because the operator need a face to identify any human.

2.1.2.1 Motion Detection :

Motion is detected using background modeling basedmotion detection by Zivkovic. It is Implemented using BackGround Subtractor MOG by Zivkovic and Van Der Heijden [2006] from opencv library.The output of the approach is a binary matrix representing each pixel of frame with '1' if motion detected and '0' otherwise. Motion detection is done over all the five frames.Let the output matrix be named MotionD.

2.1.2.2 Human Detection :

Observing Human presence is very important in Surveillance. Human Presence is detected using HOG feature descriptor based SVM classifier.The output of the approach is a matrix representing each pixel of frame with a value between 0 and 1 which is the probability of human presence in the pixel.Let the output matrix be named HumanD.

2.1.2.3 Face Detection :

Detecting or observing a face is important in surveillance. Face is detected using Haar feature-based cascade classifiers by Viola et al. [2001]. The output of the approach is a matrix representing each pixel of frame with a value between 0 and 1 which is the probability of face presence in the pixel.Let the output matrix be FaceD.

Here time t is measured w.r.t to frames. t+1 means time at which next frame arrives and t+xs means time t plus x seconds.

2.1.2.4 Calculate Sensitivity Map

Sensitivity at pixel(i,j) at frame t is calculated as

$$Sensitivity_{ij}(t) = HumanD_{ij}(t)*ACH+MotionD_{ij}(t)*ACM+FaceD_{ij}(t)*ACF. \tag{2.1}$$

where HumanD, MotionD, FaceD are the corresponding detection Matrices at time t. As these are detectors are not 100 percent accurate , the output of the detectors is multiplied by its corresponding model accuracies ACH,ACM and ACF which are constants. The Sensitivity Calculation can be easily extended other factors depending on the environment. I would be as simple as the feature detection matrix for each frame multiplied by its accuracy.

Sensitivity Map at Time t is Sum of Sensitivies of the previous w frames.

Here Δ w=5.

$$SenstivityMap_{ij}(t) = \frac{1}{\Delta w} \sum_{f=t-\Delta w+1}^{t} Sensivity_{ij}(f) \tag{2.2}$$

Senstivity Map is a Matrix of frame size with values between '0' and '1'.

2.1.3 Finding Region Of Interest(ROI)

: In the Sensitive Map pixels with values less than the set threshold are made zero. We find out the contours in new Sensitivity map. ROI is the Contour with highest sum. We draw a Rectangle around the contour with some extra padding and zooms in that area slowly.

We digitally zoom into the ROI for 5 seconds and Track the identified object using Mean shift Algorithm by Comaniciu et al. [2000] if it moves during the time window. The we zoom and out repeat the process again.Sometimes we will always concentrate on a particular area, So a Gaussian Penalty is levied on the ROI which reduces with time. The Penalty matrix is multiplied to the Sensitivity Map before finding the ROI.

2.2 Updated Szoom :

We updated the Szoom framework to improve it's accuracy.

2.2.1 Motion Detection :

The Background Subtractor used in the earlier algorithm was MOG we replaced it with updated MOG2 Background Subtractor which is an updated version of MOG and has more accuracy in detecting motion, MOG2 has more accuracy, precision and and less processing time.Comparison between is present in L. A. Marcomini [2014].

2.2.2 Face Detection :

The Haar Cascades have less accuracy to compared new Deep Neural Networks.We updated it with face_detector/res10_300x300_ssd_iter_140000.caffemodel which is a pretrained RESNET from opencv library.

2.2.3 Speed as new feature :

An object in running state may be of more interest to a security operator.So including speed while calculating sensitivity would improve our anomaly detection . From the MotionD matrix detected by BackGround Subtractor in the first frame we find out three biggest contours.Storing the objects in the contours and detecting in the next w-2 frames would be computationally expensive. So we use a Multitracker which Tracks Multiple Objects through out the rest of the w-2 frames and gives its location in the last frame of calculation of sensitivity map . Since it is for a small duration of time we take euclidean distance between the start location and final location of the object irrespective of their trajectory.Since it is for the same amount of time distance is taken as relative speed. To make the speed value relative speed i.e. the euclidean distance is divided by the frame diagonal length. SpeedD is a matrix with values between '0' and '1' where the contours of the top three object moving are filled with their relative speed values. Unlike the other features speed is calculated for all the five frames.

2.2.4 Updated Sensitivity Map

The Updated Sensitivities at time t will be

$$Sensitivity_{ij}(t) = HumanD_{ij}(t)*ACH+UMotionD_{ij}(t)*ACM+UFaceD_{ij}(t)*ACF. \tag{2.3}$$

where UMotionD, UFaceD are the corresponding detection Matrices at time t that are detected using the updated techniques.

UpdatedSensitivity Map at Time t is Sum of Sensitivies of the previous w frames plus SpeedD which is calculated by tracking the previous w frames. Accuracy is considered to be one for speed.
Here Δ w=5.

$$SenstivityMap_{ij}(t) = \frac{1}{\Delta w}(\sum_{f=t-\Delta w+1}^{t} Sensivity_{ij}(f) \quad) + Speed_{ij}(t) \tag{2.4}$$

Senstivity Map is a Matrix of frame size with values between '0' and '1'

Chapter 3

Proposed Frameworks :

Multiple Frameworks were proposed to solve the problem. Considering the drawbacks of each framework we finalised to a Stochastic scheduling based framework.One of the Proposed frameworks uses the updated Szoom framework to calculate the sensitivity map. One of the implemented frameworks is explained below.

3.1 Fixed Principal axis and tracking :

Define a principal axis(A) of the camera based on your requirement.Here Principal axis is the axis in which the center of the camera lens is directed.Principal axis can be selected by the user such that it covers his most important area.

Once the user fixes his principal axis The Proposed framework is as follows

3.1.1

1. SET Camera in principal axis , penaltyfactor = 1
2. Calculate the Sensitivity Map for 5 Frames.
3. If Camera is in Principal axis :
    - PAS=Sum of all the pixels in the sensitivity map.

4. If Camera not in Principal axis :
    - CAS=Sum of all the pixels in the sensitivity map.
5. If any detected object with sum of sensitivities above a defined threshold is moving out of the are covered by the current axis.
    - Perform pan and tilt operations such that the object becomes center of the new axis.
    - Zoom on the center of the frame for next 5 seconds so that object is more clearly observed.
    - IF $PenaltyFactor * CAS > PAS$:
        - $PenaltyFactor = \alpha * PenaltyFactor$ where 0< α <1.
        - Return to Step 2.
    - Else :
        - Return to Step 1.
6. Else :
    - Zoom on to the Calculated ROI of the frame for next 5 seconds so that object is more clearly observed.
    - Return to Step1.

This Framework is based on general operator activity. Generally a Security operator watches over a principal area and if any considerable object/human moves out the area he rotates the camera to track him but eventually after observing it for some time t he comes back to his principal area . The time t depends on the anomalies observed in the new area.So to achieve this increasing Penalty is levied, which multiplied by the Sensitivity of current axis is compared to the most recent principal axis Sensitivity. When the penalised sensitivity of the current axis is less than the principal axis the camera return to principal axis.

3.1.2 Drawbacks :

This algorithm does not improve that much over general PTZ Camera Control, This is more similar to a tracking algorithm.It does not Increase Coverage and does not properly serve our aim. In this Framework the principal axis is fixed, s it spends most of the time in a single area. This framework works fine when there is principal area where a operator must concentrate.

Chapter 4

Final Proposed Framework

4.1 Setting The States

We have divided the surveillance area into 9 parts. We can cover the complete area with the help of these 9 states. The states are defined as follows :
Let's say the camera is its normal position without any pan and tilts. Name it as state 1.
1. The camera covers some area in this state. Now there are 8 places one right to this state where it views completely to the right of the state 1 and similarly, other states are defined as continued.
2. The right of state 1 is named as state 2
3. The left of state 1 is named as state 3
4. The top of state 1 is named as state 4
5. The right top of state 1 is named as state 5
6. The left top of state 1 is named as state 6
7. The right bottom of state 1 is named as state 7
8. The left bottom of state 1 is named as state 8
9. The bottom of state 1 is named as state 9

All the states are defined with respect to the centre state.The user can preset these state as per his convenience on the camera. The camera has the support of preset. This is just a convention the number of states can be defined by the user himself based on his his requirement. The states are set so that is no or minimum overlap between them i.e there should be no area of intersection between them.

4.2 Learning Phase

Initially, the system knows nothing about the environment. So it needs to collect information about its surroundings. This phase is known as the leaning phase where it tries to learn about the environment. Each state is associated with some information which depends on the anomalies present in the state. This is computed by visiting all the states initially one after the other. Once we visit a state we compute the sensitivity map and update its corresponding information. Initially, the information of the states is set to 0 and the information eventually keeps on updated till the learning phase is complete. At the end this phase we have some information about the information all the states.

4.3 Proposed Sampling based Scheduling

We will start by picking a random state in the beginning.Another challenge with PTZ camera scheduling is to update the importance map. The importance map can be updated only when a state is visited. In other words, we have the opportunity to update the importance map every time we visit a state. However, the new importance map should be a function of older importance map values as well as the recently calculated importance map. In our proposed approach, we consider both past importance score as well as the current importance score of that state to calculate a new importance map for a state. The information gain probability when state importance map is updated is denoted as IGP_U_i

. Once we visit the state we will update its information score based on the below equation.

$$\Psi(S_i^t) = (1-\alpha)\cdot\Psi(S_i^t) + \alpha\cdot\Psi(S_i^{t-1}) \tag{4.1}$$

where $\Psi(S_i^t)$ on the LHS is the Updated Information score after visiting state i
$\Psi(S_i^t)$ on the RHS is the current observed information gain.
$\Psi(S_i^{t-1})$ is the information score of state i when it was previously visited

α is the learning rate, it is the weighted average of the current score and its previous score since current score should be given more importance, α is chosen 0.2.

So the final information score at state i depends on the all the scores it computed whenever it visited that state.
After updating the information we define the probability information score at state i as

$$IGP_i = \frac{\Psi(S_i^t)}{\sum_{n=1}^{N}(\Psi(S_n^t))} \tag{4.2}$$

Since, the IGP_i is used to select the next states, if IGP value is very low for a state, that state will never be visited. Therefore, we also propose to use a modified IGP_i called IGP_B_i, where each state will have a minimum base probability. Thus, if a system is using IGP_B_i, there would be a certain probability that a state will be visited. Let's say the minimum base probability is B. IGP_B_i is computed as:

$$IGP\_B_i = \begin{cases} B, & \text{if } IGP_i \leq B \\ IGP_i * (1 - \sum_{n=1}^{N} max(0, (B - IGP_n))), & \text{otherwise} \end{cases} \tag{4.3}$$

After calculating the information score, we will stay in the state for next 5 to 6 seconds. Meanwhile in that particular state, if the most important anomaly in the state is moving, we will track it by keeping our camera as its center until it crosses the state. If the anomaly is static we will digital zoom into the most important anomaly, so that the end user only sees where the anomaly is happening. The end user can disable this mode if he wants to see the whole area. So when we are in a state we are tracking the most important anomaly. After the timer expires we will have to choose the next state. The information we previously collected helps us to make decision of choosing the state. We will choose the next state based on the probability information score. We will use random sampling algorithm to pick the state. This algorithm is based on a random number generated between 0 and 1, and we will map each state to a number between 0 and 1. Based on the random number generated we will choose the next state. We will continue this algorithm.

4.4 State Selection Algorithm

We propose a stochastic scheduling algorithm, in which next state depends on information score probability of the states. The algorithm dictates proposed state selection algorithm step by step. The algorithm uses a factored sampling strategy for generating the most probable state for each pass. The input to the algorithm is states, pixels, and information gain probabilities of all states. It calculates the cumulative state wise interval using information gain probabilities, as described in state 3 of the algorithm. Then a random number is generated between 0-1 to select the most potential state for the next move of the camera. We take the interval score in which this random number lies and the corresponding state is considered the winner. By doing so systems ensure that highest importance score possessing state gets more probability compared to others. It also ensures that state with less score may also get a chance as it depends on what random number we get in the pass.

Algorithm 1 State Selection Algorithm

```
N(# states), M(# pixel in a state), [$S_1$, ... $S_N$](State), [$IGP\_U_1$ ... $IGP\_U_N$] (information gain probabilities of states) $S_p$: most probable state for transition
for $n = 1$, $n$++, while $n <= N$    calculate $IGP\_U_n$
Assign interval to states as:
$LowerBound_1 = \mathbf{0}$
$UpperBound_1 = IGP\_U_1$
for $n = 2$, $n$++, while $n <= N$
$LowerBound_n = UpperBound_{n-1}$ $UpperBound_n = UpperBound_{n-1} + IGP\_U_n$
Generate a random number between $0 - 1$
Output state = interval in which the random number lies
```

The complete procedure is pictorially represented in the below figure

4.5 Re-initializing Phase

After a long time, if the environmental conditions changes like day and night, some of the least important states may not be visited at all. So we have to reinitialize the states so that it learns all the states once again and performs better.

z

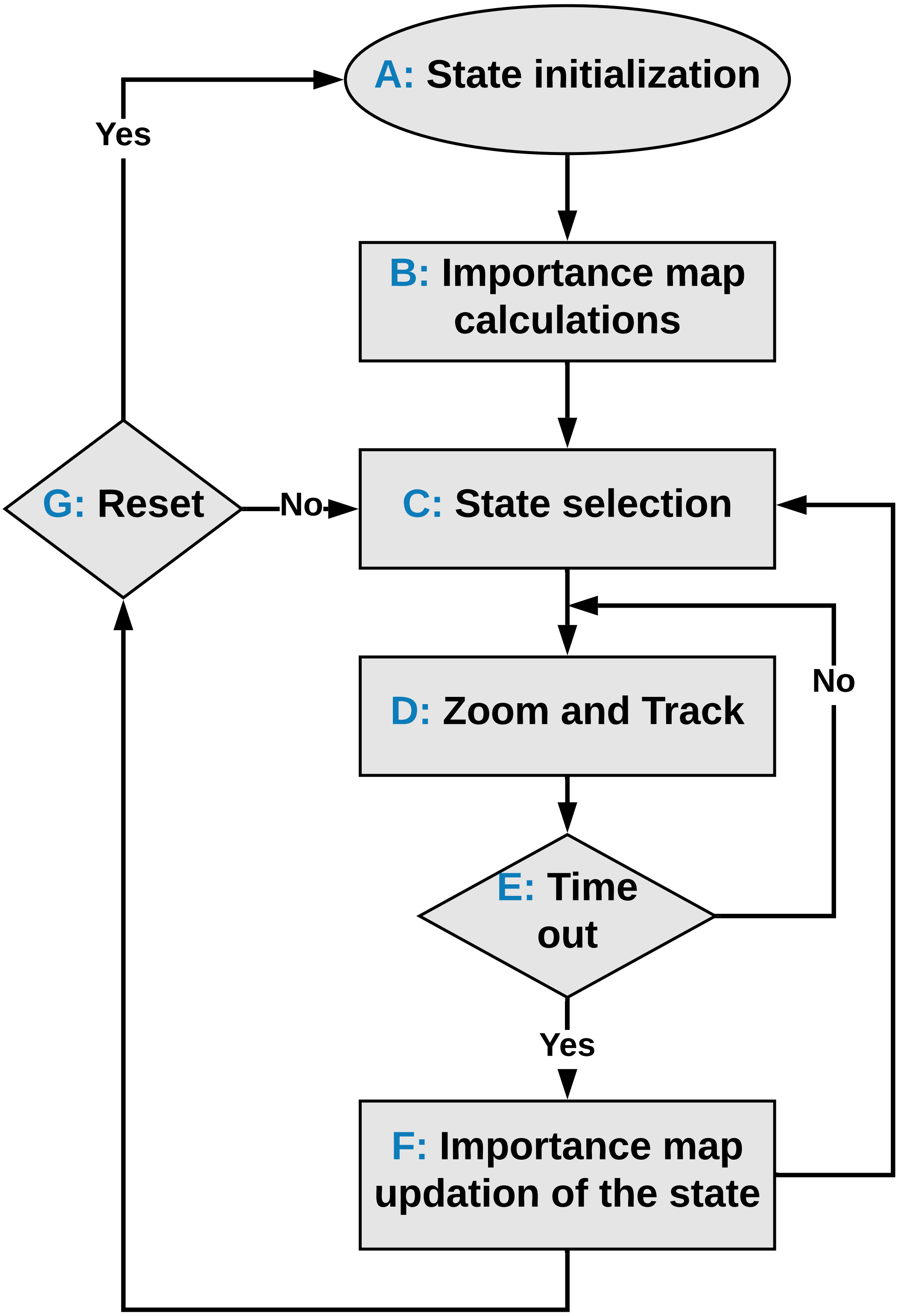


Figure 4.1: Pan-Tilt-Zoom Framework. Action are shown in order.

4.6 Handling User Input

In order to handle the user input, we take inputs from the user. The user knows all the states and can give commands to the system when he feels certain states as important. Our system should be adaptive and try to learn the inputs from the user the user can give a certain state as important. The algorithm execution pauses and gives priority to the user input. If the user selects a state i, the camera immediately turn to the state. Also the information score for the state is incremented with some value so that the state gets more important.For this we created a web server which streams the output video and has a input box where a user can give inputs. Once the user gives input, the web-server accepts the input and sends the input to the main code.

4.7 Implementation :

This Framework is implemented in C++ using Opencv. We used Wanscam Hw00051 PTZ camera to showcase the pa-trolling strategy. It has a resolution of 1280 X 720.We fixed 6 number of states on the camera.Our Implementation Constantly requests frames from the PTZ Camera detects the next state using the above mentioned framework and sends onvif commands the ptz camera regarding the state information.When the camera receives the Onvif command it changes its state to current detected state.During the zooming and tracking our implementation constantly request the frames and using the frames we digitally zoom in frame and the Zoomed frames are constantly streamed to url where the user can see the output of the zoomed in cctv footage i.e. the final output of our framework . The User can also input a state which he wants to see more or is more important. When the user enters the state number that state probability are increased as the user thinks the state is important.

Video frame rate(frames per second) will depend upon internet speed. For our experiment, it was on average 7. In initial setup phase camera uses 5 frames to calculates Sensitivity in each state. State visiting time is 10 seconds i.e. the camera will spend 10 seconds in any state after this it goes to other states intelligently.

Chapter 5

Experimental SetUp and Experiments

5.1 Experimental Setup

5.1.1 Camera And Its API

We used Wanscam Hw00051 PTZ camera. In order to fetch images and give commands to the IP camera, we explored the html and js files provided by the camera company. Once we type the IP and its username and password for the camera, it directs us to its home page where we can view the video. The camera API is as follows
for fetching hd image: http//username:password@IP/web/tnpfs/snap.jpg
for fetching normal image: http//username:password@IP/web/tnpfs/auto.jpg
States can be set in the camera itself. To invoke a State : http//username:password@IP/cgi-bin/hi3550/param.cgi?cmd=preset-act=goto-number=STATENUMBER

Since at any given the camera is present only in one state, So we do not know the complete ground truth. To Compute the ground truth we set up a 6 camera network which covers all the states all time.
We also explored the CISCO IP camera. It has its own problems associated with it while we were exploring. It took too long for the instructions to reach the camera as the camera was situated some where else on the network. Network

calls were taking a lot of time. They did not directly expose their API for fetching the live stream and also to control their camera. This camera maintains sessions unlike the web scam one. The camera has ssh access. So we think the user should be able to know the API via ssh access. Since its a public camera we were not supposed to modify these files which required special permissions.

5.1.2 Streaming the Output Video

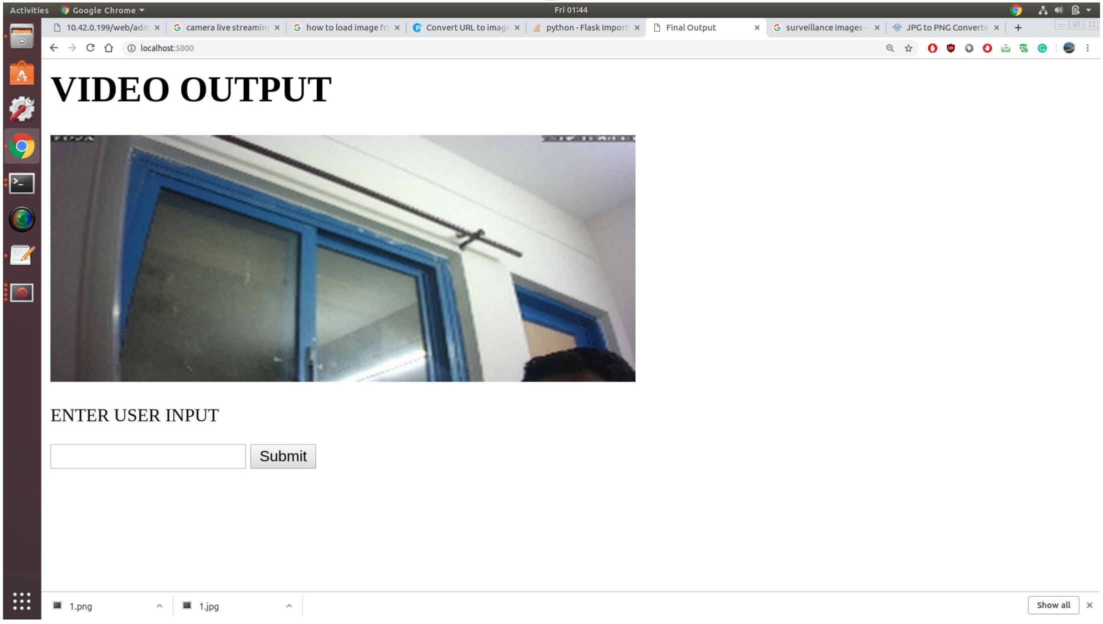
We used a web streaming server which was developed already. The method that works well with the streaming feature of Flask is to stream a series of independent JPEG pictures. This is called Motion JPEG and is used by many IP security cameras. This method has low latency, but the quality is not the best since JPEG compression is not very efficient for motion video. The streamer uses flask framework. Flask provides native support for streaming responses through the use of generator functions. A generator is a special function that can be interrupted and resumed. Flask uses this characteristic of generator functions to implement streaming.

This application imports a Camera class that is in charge of providing the sequence of frames. The frames are written by the main code which finally generates the zoomed images into a common directory shared by the two processes. The streaming code is in python as it is easy to implement and the main code is in C. The final images are written in this common directory and to generate frames, the camera folder access the most recent file written, To avoid the read-write problem, we access the most recent 10th image so that both codes can work in parallel. As the main code keeps on writing images, the memory might overflow. As we no more require the previous images, we can delete them. The python code takes the responsibility of removing the images.

5.1.3 Compatible Support for Other Devices

The website is stream able on any device and also supports mobile devices. The should be on the same network as that of the server. The server IP address is required for this streaming. Once you type the servers IP address, it automatically directs you to the home page of the website. Also the user should disable any

Z



VIDEO OUTPUT

ENTER USER INPUT

Figure 5.1: Sample look of our web site streaming video and handling user input.

firewalls.

5.1.4 Handling User Input

The web page of the URL has a box below for the user to give his desired relevant state. Once the user enters the state and clicks the submit button, the user input is written in a common text file which can be accessed by both the programs. The main code checks each time in a loop if there is a user input in the file. If it finds some user input, it considers it and does the required action. The main code itself removes the value in the file once it reads the value. The next immediate state taken by the main code is the state given by the user.

5.2 Experiments :

We compare our framework with current PTZ camera control methods and other variations of our framework to see which works better. We computed information gain using sensitivity map for each of the methods over a duration of T=20 minutes on 2 different set of videos representing six states.

Different Frameworks that we are comparing :

- Round Robin: The states are selected are selected in a round robin fashion without any other calculations.
- Greedy Information gain Approach: The state that has highest Information gain at the current time is selected.
- Our Proposed Approach.
- Ideal Approach: Assuming a Framework knows the next state with maximum Information gain. And it visits the all the states optimally.Generally the Information gain would be one for this approach.
- N-Camera Setup : A Setup where all the N-states can be seen all the time. We have done experiments in two different environments. We used 6 phones for settings the states. The phones are oriented such that they were covering 6 different directions. One experiment is done at Dubey Cafe and another one done in Sutlej hostel balcony. For the Dubey cafe, there are different orientations in which few people were walking in one state and few people were sitting in the cafe in one state and few were playing games in some state. All the states were almost horizontally placed on one of the cafe table and the videos were recorded. In the second setup we choose the recording from the top of Satluj hostel. This is a perspective from the top, where we view what is happening in the bottom. There are 2 entrances where people enter and leave from these entrance, we fix these as the states and also there is a state where people play. Here the top vertical state has some significance because if a normal camera turns up it sees the sky which is of no use. Here as the camera is viewing from the to, its vertical view covers some far places. All the phones used were nearly set to record the same resolution video of 1280x720p

We compare information gain among variations of our proposed framework.

Table 5.1: Comparison of information gain of various scheduling strategies on two different videos.

Scheduling Framework	Normalized Information Gain (video-1)	Normalized Information Gain (video-2)
Round Robin	0.4991	0.5612
Greedy Information Gain Approach	0.4859	0.5507
Ideal Approach	1	1
Our Approach	**0.5512**	**0.6230**
N-Camera setup	2.9643	2.7945

Base Probability Approach :Some times some states have very less sensitivity for a period of time than the probability of that nears zero and our algorithm may never go to that state ever.So we introduce base probability for each state such that probability of each state will never reach zero. All the States are given a base probability b¿0. Probability of each state is max(b,Probability calculated by the Information Gain Approach).

Retraining Approach : Sometimes the framework may be stuck in some states if the states have very high probability.To avoid this we retrain the framework after every t mins.

As we can see the results the our Approach works better than other other approaches because it takes a probabilistic approach combined with historical knowledge of state.

5.2.0.1 Quantifying Coverage

To quantifying coverage, we run the stochastic scheduling algorithm on the PTZ camera - with base information gain probability model and information gain probability with updates. To generate coverage map for ΔT seconds window, we need to perform the state selection algorithm for $x = (\Delta T * fps)/\Delta F$ times . Where ΔF is the frames taken in execution of the the state selection algorithm. Figure 5.2, 5.3, and 5.4 compare coverage map against corresponding importance map. For these all experiments, importance map, and coverage map were calculated for a duration of 10 minutes i.e. we run our scheduling framework on the mentioned

Table 5.2: Comparison of information gain based various possible stochastic approaches.

Scheduling Approach	Normalized Information Gain
base approach	**0.5512**
base probability(B=0.025)	0.5124
base probability(B=0.04)	0.5153
base probability(B=0.05)	0.4791
retraining after each t=2min	0.5044
retraining after each t=4min	0.4990
retraining after each t=6min	0.4977
retraining after each t=8min	0.5176

camera for 10 minutes of duration. Experiments in figure 5.2 and 5.3 were done for N=8 states and that in figure 5.4 was done with N=7. We see that in all figure 5.2, 5.3, and 5.4 coverage is in proportion with information gain throughout the state. This validates our coverage definition. Some state which has negligible(not zero) information gain show negligible(not zero) coverage. We do not see them in the figure because values were too low to be visible for a low duration of 10 minutes. Figure 5.3 shows little inconsistency in state 5 (S5), this may happen because it uses probability-based state selection so for a short duration map it is showing biases. We observe that the value of mean square error (Eq 6) is within 2% for all the experiments.

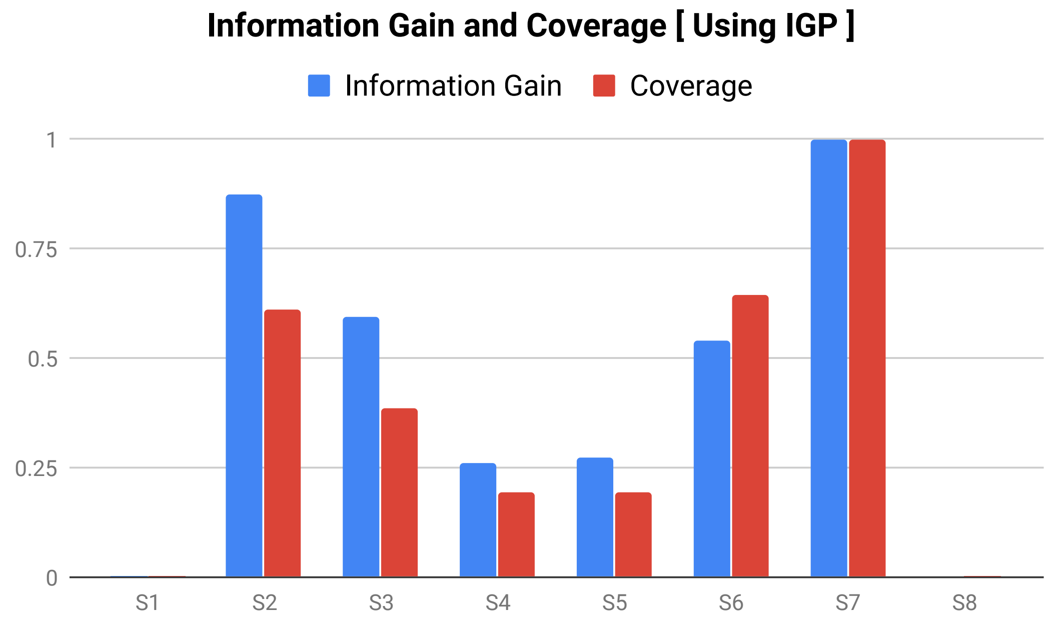


Figure 5.2: Comparing coverage and information gain using IGP_i. Parameters: N=8, T=10 minutes.

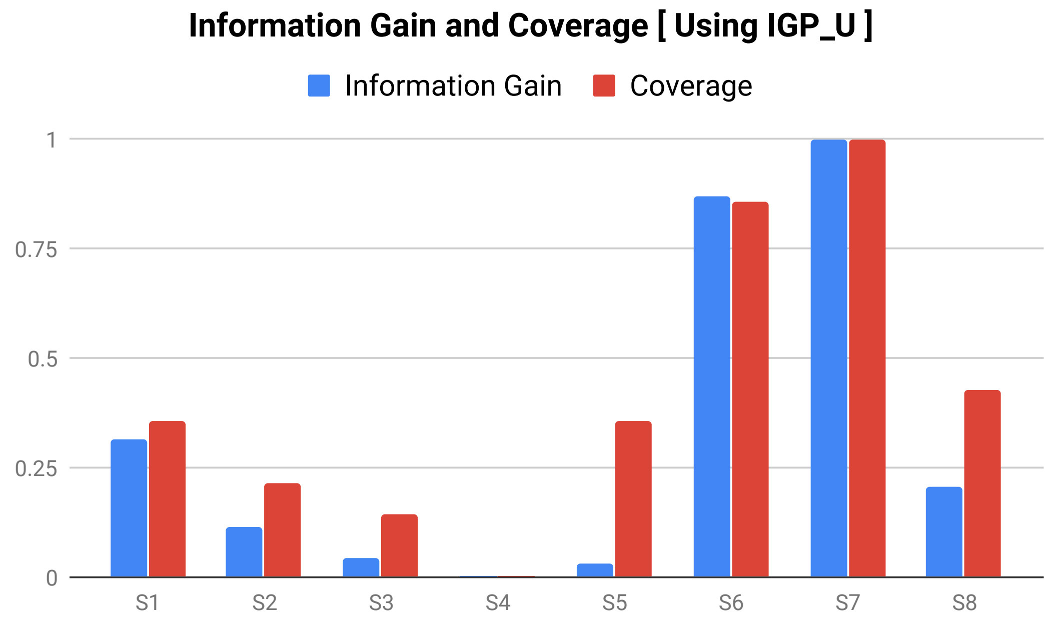


Figure 5.3: Comparing coverage and information gain using IGP_U_i. Parameters: N=8, T=10 minutes.

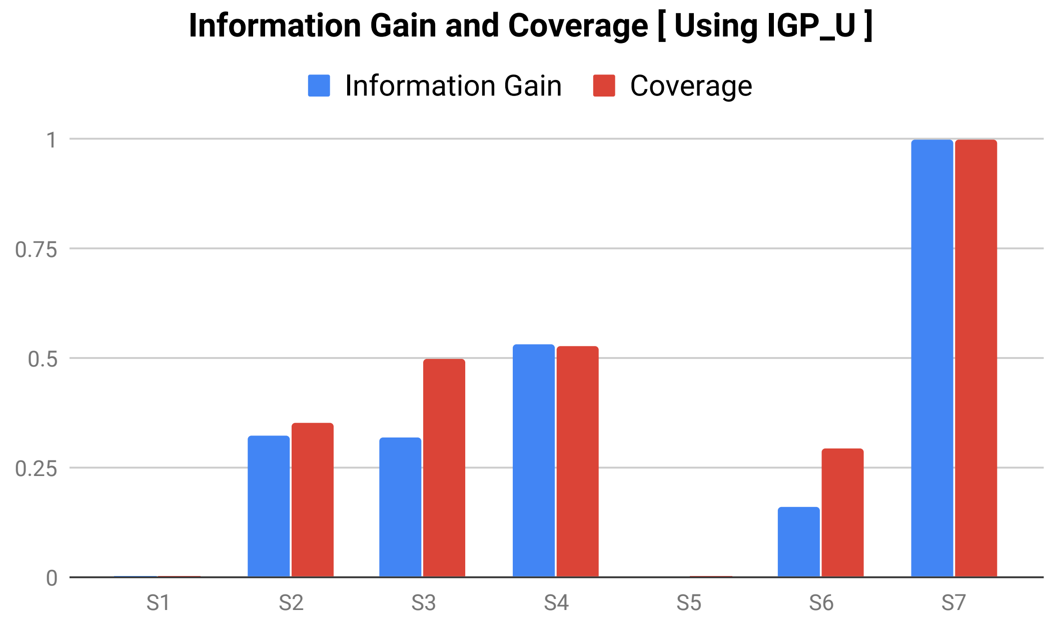


Figure 5.4: Comparing coverage and information gain using IGP_U_i. Parameters: N=7, T=10 minutes.

Chapter 6

Conclusions and Future Work

In this project we implemented a Intelligent Surveillance System. It maximizes the utilization of pan-tilt-zoom capability of the camera.This framework is best suitable be used in areas where there is activity across a large area but not happening simultaneously.This Approach overcomes the issues of the general fixed path camera rotation techniques as it chooses it states based on real time information and approximates to act as N- Camera Setup by concentrating on the important areas .And Randomization included in the framework helps the operator because people cannot predict the camera motion pattern. There this proposed framework will effectively utilize PTZ camera for surveillance. With the help of Dr. Mukesh Saini and Dr. Neeraj Goel and doctoral student Prathibha this framework "Dynamic Scheduling of an Autonomous PTZ camera for effective Surveillance" has been submitted to ACM Multimedia Conference 2019 and is under review .

The Drawbacks of this method is that it required additional System for analysing the frames constantly. This framework is Computationally expensive.

This Framework can be improved more by tuning the learning rate dynamically based on the real time environment.

Basically there are two main improvements which can be thought. One of them is including audio processing. With the help of audio processing the camera has more information of the states and can choose a perfect state. Another major improvement is to come up with idea of linking consecutive states. Lets say state a and b are consecutive which means if a person is in state a is moving then he will reach state b soon. We are not taking any advantage of this information. If we take advantage of this information of consecutive state, we think we can achieve more than this. Basically we thought this at the end of the semester. Few more feature works include more robust way of choosing the states, instead of user setting the states, the camera should be able to figure it out itself by moving around. This reduces the burden to the user.

References

Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, pages 142–149. IEEE, 2000. 6

Alessio Ferone and Lucia Maddalena. Neural background subtraction for pan-tilt-zoom cameras. *IEEE transactions on systems, man, and cybernetics: systems*, 44(5):571–579, 2014. 1

Weiming Hu, Tieniu Tan, Liang Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Trans. Sys. Man Cyber Part C*, 34(3):334–352, August 2004. 4

Hao Kuang, Benjamin Guthier, Mukesh Saini, Dwarikanath Mahapatra, and Abdulmotaleb El Saddik. A real-time smart assistant for video surveillance through handheld devices. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 917–920. ACM, 2014. 4

A. L. Cunha L. A. Marcomini. A comparison between background modelling methods for vehicle segmentation in highway traffic videos. 2014. 6

Paul Viola, Michael Jones, et al. Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, 1:511–518, 2001. 5

Zoran Zivkovic and Ferdinand Van Der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006. 4