TJHSST Computer Systems Lab Senior Research Project
Tracking in Persistent Surveillance
2009-2010

Adam Mounts

January 28, 2010

Abstract

The development of a program that can track targets is a crucial development in security and/or surveillance systems. A tracker can be used in the event of a crisis situation to follow potential suspects or targets from the scene of a crime, or to find where these targets originated from, all based on aerial imagery. By using a program to do this, quick, real-time analysis is feasible rather than having humans toil over movies at a later time.

Keywords: tracking, persistent surveillance, urban surveillance, image analysis.

1 Introduction

1.1 Scope of Study

The goal of this project is to create a tracker that can follow a certain object, whether it be a human, a vehicle, or some other moving target, and trace its path through a series of images. The extent of this project is quite variable, due to that there is no apparent limit to how detailed it can become. By adding more and more noise or increasing the number of targets, the problem will increase in complexity and become harder and harder. In addition, I will

also change and compare algorithms that I use to track the targets to further develop my project.

1.2 Expected Results

Ideally, the initial result of this project will be a tracker that can successfully track a target or multiple targets through a simulated terrain and through actually aeriel imagery taken of an urban area. By the end of the project, I will hope to have quantitative and qualitative data and comparisons between different tracking algorithms, those being pixel subtraction and Kalman Filtering.

1.3 Type of Research

My project is a project of pure applied research. I am not seeking new fundamental understanding of the material, but rather implementing various methodologies such as pixel subtraction. Rather than instituting a brand new theory, I am working to use an established theory and to utilize it, and possibly improve on current models and programs. My research pulls together different algorithms - those mentioned in the Expected Results section, and tries to implement them in Python.

2 Background and Review of Current Literature and Research

My mentor told me about the step in her project of Persistent Surveillance that involves tracking targets through the area of interest. In the office, some of our colleagues are working on posture recognition, and this is partially related to my project. The similarity is that both attempt to analyze images and provide useful information, all by looking at the pixels and edges of the image. In one paper I read, the tracker is being used on thousands of different images, taken from the 4 orthogonal directions. The new tracker generates simulated humans in different postures, and matches the real image to the closest simulated posture. The generated posture is then compared to the edges of the image, by using an edge detection program on the real image. Another paper I discovered gave an introduction into Kalman filtering, which is an algorithm I am using in my project. Other papers I have read also used

Kalman Filtering, and by using these sources in addition to the introduction, I should be able to implement it successfully for tracking.

3 Procedures and Methodology

3.1 Requirements, Overview, Limitations, Development Plan

In order for this project to work succesfully, I will need to have both aerial imagery as well as computer generated imagery. It needs to be able to track targets through real terrains; the reasoning behind having aerial imagery. The computer generated imagery will be used as a starting point, and its flexibility in complexity will allow me to develop my program much more easily, by only adding as much complexity as will most benefit me. I will be working mostly in Python, because it has sufficient image processing capabilities for my project. The limitations are most likely going to be the limitation of usable aerial data. While working at my mentorship, one apparent drawback to using aerial data is the difficulty to accurately orthorectify aerial imagery, or in other words, to orient it in a way so that it is an exact birds eye view and also is to scale. If actual aerial data isn't available, computer generated data might have to be made as close as possible to show that the tracker has real world application.

3.2 Research Theory and Design Criteria

This project has three main sections. The first section was the development of a "movie maker", essentially a program that creates and saves a sequence of images so that the functionality of the rest of the project can be tested. This was done in Python, by taking a simple circle and moving it in a random Brownian-Motion fashion. This allows for a simple and quick qualitative test of my tracking algorithms, but will by no means be the final test.

The second section is the development of a tracking algorithm using Pixel Subtraction. By taking two subsequent images from a movie file, and comparing the two images pixel by pixel, the pixels in which the value has changed are the areas of interest. Prior to the introduction of "noise", or content in the image that isn't the tracker, this theory should be sufficient to track

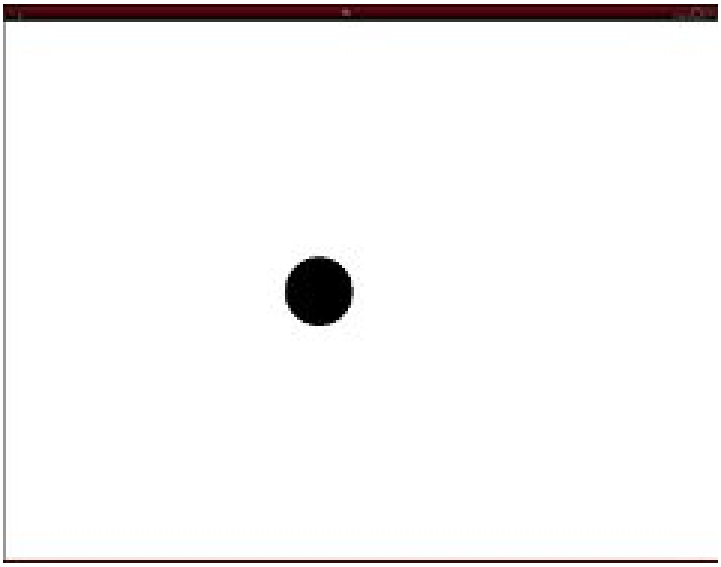

Figure 1: Image of Basic Target and Image Maker

the target. However, as the complexity increases, I will use a filter to help eliminate noise and focus on the target.

The third section is the use of a Kalman Filter to use to help create a tracker. The Kalman filter runs recursively to minimize the square of the error, which means that as time goes on, its estimates will become closer and closer. It does this by first randomly generating possible "solutions", which can change in form accordingly with the scenario of the problem. Next, it changes all of these solutions slightly, and then sees if this effects the solutions in a benefit or a detrimental manner. By maximizing the benefits by making similar moves, and by avoiding moves that had a detrimental effect, it will recursively reach an equilibrium in which the solutions cannot be improved any further. This filter will be extremely useful because it can include data such as velocities, which will help determine between multiple targets if they are to intersect or pass near to each other. The obvious issue with this algorithm is its immense complexity, however I have written some code to run a simple Kalman filter.

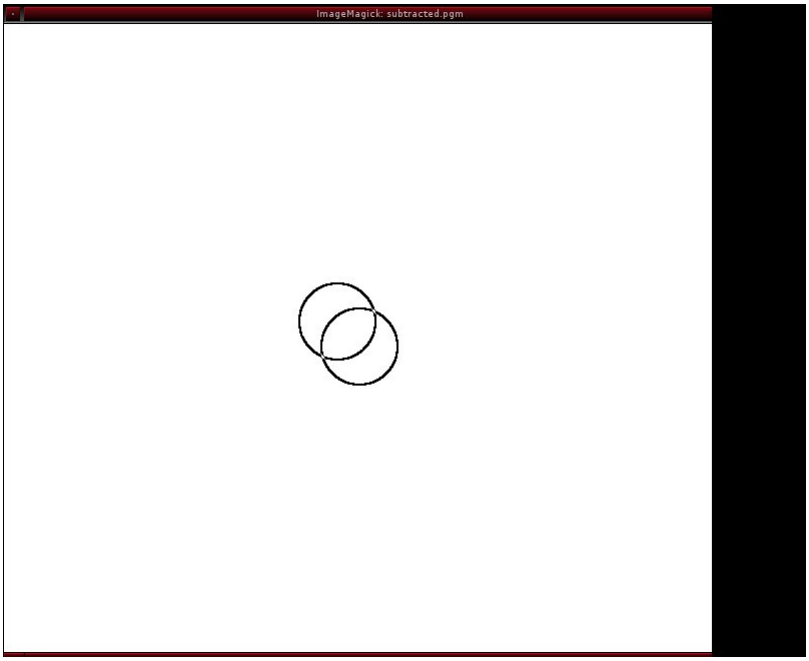

Figure 2: Pixel Subtraction Run on Two videos

3.3 Testing and Analysis

The testing phase of this program will be extremely easy. It is a purely qualitative analysis at this point. Just looking between the input images and the resulting image, whether or not the program ran succesfully will be evident. The way I can fix parts are by improving the algorithms used to follow the different objects. I can always also work to improve efficiency, because with large images streaming constantly, time will become an issue for the actual project. My testing will vary by using different levels of noise, different objects to track, different backgrounds, and varying numbers of tracked objects. The analysis will involve the accuracy of the tracker, in addition to run time.

3.4 Visual representation of data and results

A visual representation of my program will be graphs including graphs comparing run-time with noise, as well as accuracy with noise. A short series of pictures showing my tracker tracing the path taken by the target is a pos-

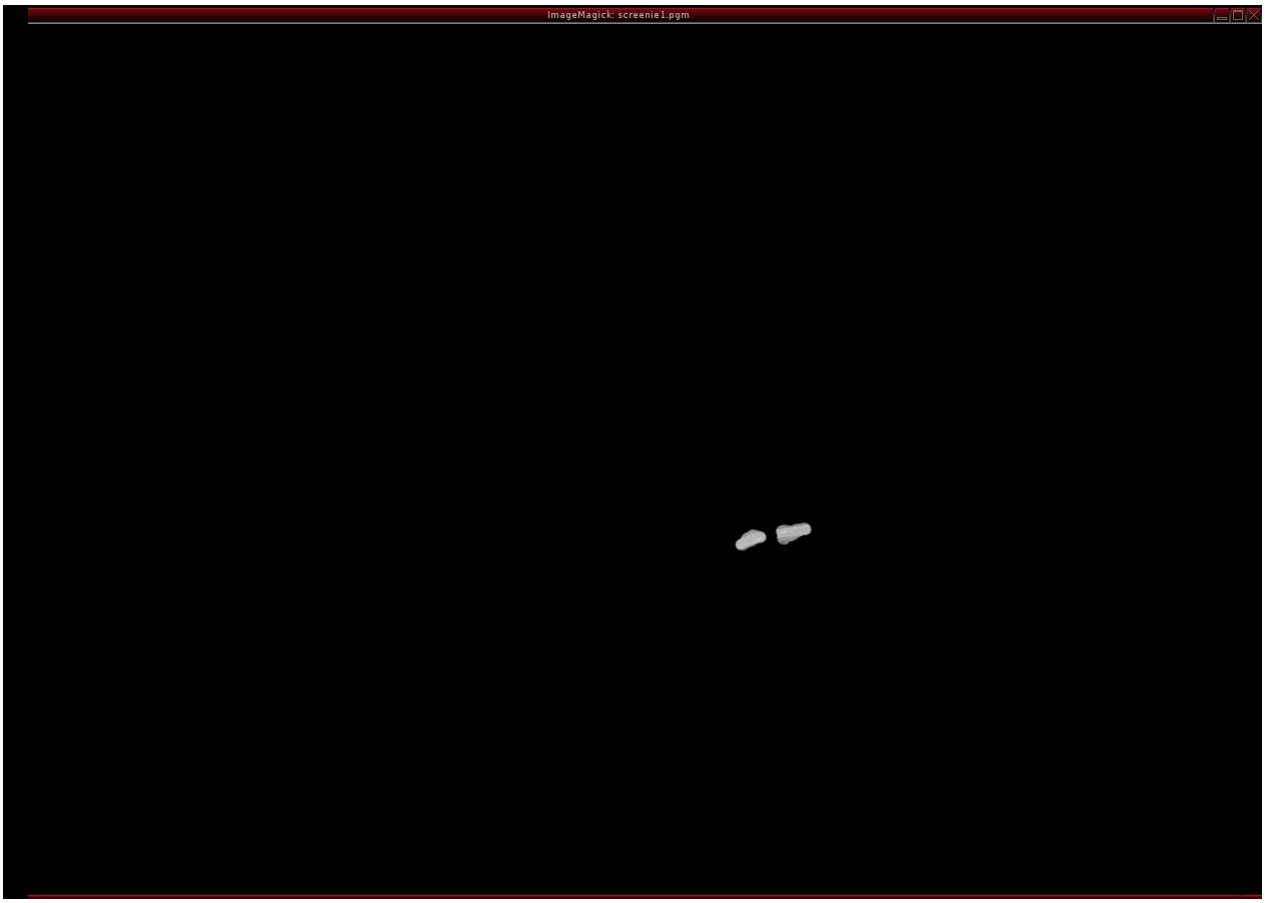

Figure 3: Pixel Subtraction Run on Two Aerial Images

sibility. If I have the time at the end of my project, I will attempt to make short clips of each algorithm running, so that the reader can view them in process and see the differences between accuracy, time, and complexity.

4 Results, Recommendations

4.1 Expected Results

The final result should be able to follow a target or multiple targets through both simulated terrain images as well as real world aerial imagery. I will provide the results most likely in a series of images showing my tracker tracing the path of target(s). I could run the program multiple times, and check if the tracker follows the target throughout its entire path, and graph the percent accuracies depending on the complexity of the image. This project has no apparent end, as the algorithm for tracing and the complexity of the image can always be increased.