

Automated Casing Event Detection in Persistent Video Surveillance

Daniel T. Schmitt

Stuart H. Kurkowski

Michael J. Mendenhall, Member, IEEE

Graduate School of Engineering and Management

Air Force Institute of Technology

Wright-Patterson AFB, Ohio 45433

Email: [daniel.schmitt, stuart.kurkowski, michael.mendenhall] @afit.edu

Abstract—An increase volume of surveillance video is being collected, by various organizations, which has led to a need for automated video systems in order to reduce reviewing time. Using persistent video gathered from an aircraft overhead, as is done with unmanned aerial systems in Iraq and Afghanistan, we get a birds-eye view of vehicular activity. From these activities we can use a model to detect suspicious surveillance activity (casing). This paper builds a model to detect casing events and tests it using Global Positioning System (GPS) tracks generated from vehicles driving in an urban area to show the effectiveness of the model. The results show that several vehicles can be monitored at once in real-time. Additionally, the model detects when vehicles are casing buildings and which buildings they are targeting.

I. INTRODUCTION

Interest in applying automated computer processes to video data has grown over the past several years. The growth in computer storage space and of digital imaging components has created a large repository of digital images and video. Surveillance applications using these large repositories provide capabilities beyond the typical rewind and playback used prior to computer-centric video processing [1]. With the advent of these large video sources, a need for automated systems to process and extract meaningful information has developed. These automated systems are helpful in some applications, but only to the degree that they accurately detect objects and semantically interpret them in a way that is useful to the user. To make automated systems useful, they must detect objects of interest and identify relationships between them to detect events.

One event that would be of interest to security personnel using persistent surveillance is the detection of terrorist pre-attack surveillance activities. This pre-attack surveillance portion of a terrorist's execution cycle is the most vulnerable to counterterrorist detection and has the greatest probability of disrupting a terrorist event before it occurs [2]. The suspicious activity that we are particularly interested in is a vehicle that is watching a certain location. We will call this activity a "casing event" or "casing."

II. OVERVIEW

A. Problem Definition

The problem we are interested in addressing is the detection of potential casing events track data derived from persistent surveillance video such that there is a high confidence in the detection to avoid false detections. The goal is to detect all casing events in the video sequence and to report meaningful information used to judge whether a detection is accurate or not. The approach is to use GPS data collected in an urban campus area to represent the results of tracking vehicles by overhead persistent surveillance. The GPS data is processed with turn detection and point semantics, and then fed into a Jess reasoner [3] to detect casing events. The performance of the system is measured on the probability of detection (P_D) and the probability given we have a detection that it is a true detection (probability a detection is real). The probability a detection is real is similar to the probability of false alarm (P_{FA}) which is commonly used and is $1 - P_{FA}$.

B. Casing Event Detection System

A block diagram of the Casing Event Detect System (CEDS) is shown in Fig. 1. Along the top of Fig. 1 are the parameters for the CEDS discussed in more detail in Section IV-A. Along the left side of Fig. 1 is the workload discussed in Section IV-C. System components include the subject computer which holds Java code with the CEDS algorithm. Within the Java code are extensions that allow SQL calls to a separate Oracle database, and the Jess reasoner along with the code depicting the rules to model the casing event. Along the right side of Fig. 1 are the outputs of the CEDS. These include whether a detection has occurred or not, the possible buildings involved in the surveillance, and how much time elapsed during the casing event. Table I lists the configuration of components in the CEDS.

III. METHODOLOGY

The overall approach to knowledge generation from persistent video is applying the concept of moving up a pyramid of

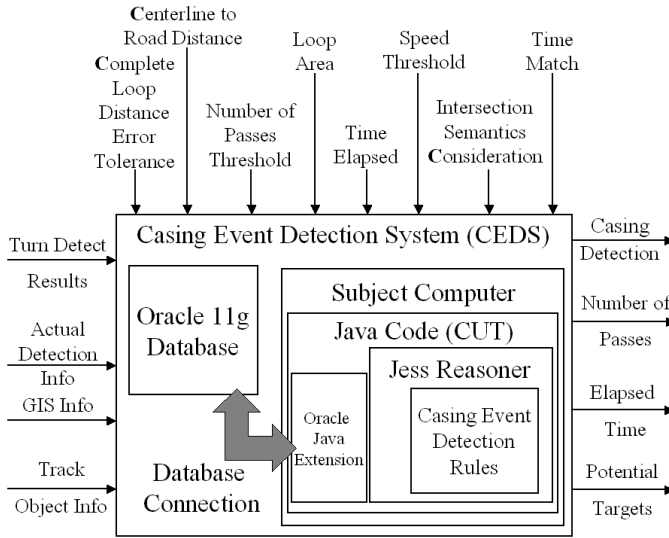


Fig. 1. Block diagram of the Casing Event Detection System (CEDS).

TABLE I
CONFIGURATION OF THE CEDS UNDER TEST

Computer Specifications	Intel Xeon 3.2 GHz, 3 GB RAM
Operating System	Windows XP 64-bit Service Pack 2
Java JDK Version	6
Jess Version	7.1p1 8/6/08
Oracle Version	Oracle Spatial 11g
JDBC Driver Version	5

data fidelity [4] from basic “things” to “worlds” (see Fig. 2). At the base of the pyramid, “things” relate to syntactic detections of vehicle movements. We used additional syntactic processing to turn those detected vehicle movements into detections of whether or not a vehicle has turned left or right in its driving path. This begins crossing into the portion of “knowledge about things” in the model (Level 2). An algorithm was built that processes the detection of turns. Then, the track location waypoints are populated with semantic information related to the contextual interactions of each waypoint with other Geographical Information System (GIS) layers. The turn detections, combined with the point semantics, was reasoned over by an artificial intelligence (AI) reasoning engine to detect more complex events like an event where an adversary was driving circles around a site suspiciously watching that site (i.e. casing).

A. Turn Detection

The first element in detecting a casing event is to detect when a vehicle track has changed course as a result of a deliberate action (i.e. the vehicle has turned). For each waypoint (point of interest), the Turn Detection System (TDS) detects whether a turn has occurred. The TDS balances how far to look forward and backward along the vehicle track to form two vectors with the point of interest as the vertex between them. It also determines the appropriate threshold for the angle formed by the two vectors. The results of turn detection then

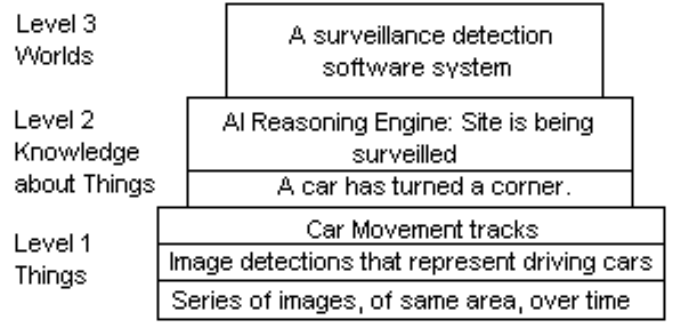


Fig. 2. Evolution of data fidelity as it relates to Persistent Video Surveillance cast in the pyramid of data fidelity in [4].

become part of the workload used to detect casing events.

B. Point Related Semantics

In addition to using output from the TDS, additional context can be added to tracks to further move up the data fidelity pyramid (Fig. 2). By relating a track to objects it is on or near, additional context is created to improve accuracy for detecting higher level events. The purpose of semantic tagging is that each waypoint is updated with additional context of whether that waypoint is turning left or right (from the TDS), is in a parking lot, is on a road, or is in an intersection. With these additional semantics, it is now possible to determine when a vehicle is turning while in a parking lot (*parking* = TRUE, *turn* = TRUE), turning in an intersection (*intersection* = TRUE, *turn* = TRUE), going straight through an intersection (*intersection* = TRUE, *turn* = FALSE), or turning when not in an intersection (*intersection* = FALSE, *turn* = TRUE). These insights allow us to model behavior more accurately (i.e. turning into a parking lot from a road, means something different than turning in an intersection). The semantics augment the turn detection to give better meaning to the behavior of a track, allowing us to ask higher level questions with more context.

C. Defining a Finite Automata Model for a Casing Event

An AI reasoner (Jess) [3] is used to build a finite automata, as shown in Fig. 3, to detect a casing event. Each input of a left turn (L) or right turn (R), will transition from one state to another state. When a series of three turns in the same direction is detected, the system begins tracking forward in time from the point of turn three, and backward in time from the point of turn one, in order to find a past point and future point whose distance is less than the *Complete Loop Distance Error Tolerance* (see Table II). The purpose of tracking forward and backward is to find a location that is revisited in the same track to finish a complete loop. In addition, when canceling turns occur (as in a L followed by a R) the reasoner continues to check simultaneously the combinations of turns (i.e. the sequence LLRLLL will result in two potential matches, one for LLRLL, and one for LLL). To improve performance, the algorithm only looks a set time period in the past and the future, in order to find a portion where the tracks cross. In addition, only

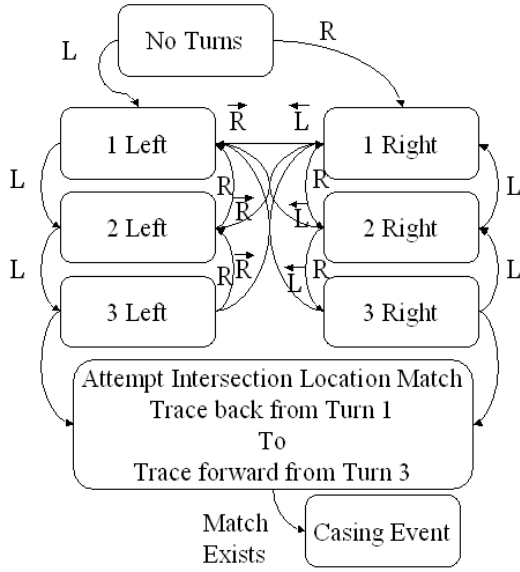


Fig. 3. The finite automata used to detect casing events.

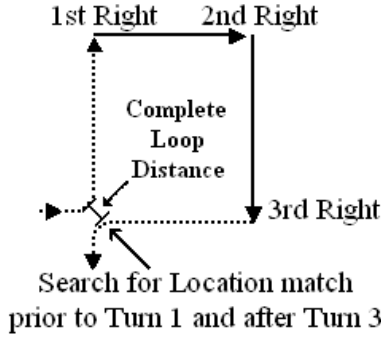


Fig. 4. A three turn approach to detecting casing events. Searching for a location match traces forward from the 3rd turn and backward from the 1st turn comparing all points in intersections until one is found less than the *Complete Loop Distance Error Tolerance*. The *Complete Loop Distance* is depicted as the distance between the past and future tracks.

points that are in intersections are checked, further improving performance. The concept of traceback/traceforward matching is shown pictorially in Fig. 4.

IV. IMPLEMENTATION

A. System Parameters

Table II lists the parameters of the CEDS. For each track waypoint, the system will decide whether or not a casing event has occurred. To understand the *Complete Loop Distance Error Tolerance*, it is first important to understand what the *Complete Loop Distance* is. The *Complete Loop Distance* is the distance between a point of the traceback from the first turn and a point on the traceforward from the third turn. Because we want our CEDS to detect casing events that repeat the same route, not just make turns in the same direction, we need to check the *Complete Loop Distance* when a series of turns is found. If the *Complete Loop Distance* is not sufficiently small, as defined

TABLE II
PARAMETERS THAT AFFECT THE PERFORMANCE OF THE CEDS

Name	Description
Complete Loop Distance Error Tolerance	The distance between the point on the traceback of the first turn, and the point on the traceforward of the third turn, and what distance error is tolerable and still considered to be a complete loop.
Centerline to Road Distance	The distance between a road centerline and a point. If the distance between the centerline and a point is less than the Centerline to Road Distance, the point will be considered as being on the road. Likewise, if the distance between two centerlines and a point is less than the Centerline to Road Distance, the point will be considered as being in an intersection.
Number of Passes Threshold	The number of times a track passes by the point of interest.
Loop Area	The area of the polygon that is formed by the consecutive turns.
Time Elapsed	The elapsed time from the beginning to the end of the casing event.
Speed Threshold	The speed at which the vehicle travels during the casing event.
Intersection Semantics Consideration	Whether or not a turn being done in an intersection is important or not. If it is, only turns that happen at intersections will transition the state of the machine, otherwise, all turns will transition the machine state whether they occur in an intersection or not.
Time Match Threshold	The amount of time that will be traced back from the point of turn 1 and traced forward from the point of turn 3.

TABLE III
LEVELS OF THE *Centerline to Road Distance* FACTOR TO BE TESTED

Level	Value
1	5 m
2	10 m
3	15 m

TABLE IV
LEVELS OF THE *Complete Loop Distance Error Tolerance* FACTOR TO BE TESTED

Level	Value
1	15 m
2	30 m
3	60 m

by the *Complete Loop Distance Error Tolerance*, then it will not be considered a casing event.

One other assumption is that the *Centerline to Road Distance* also defines the boundary of what determines an intersection (an *Intersection Distance*). Thus, an intersection is defined as within the *Centerline to Road Distance* of two roads. An example of this would be if the *Centerline to Road Distance* were set to 15m, then an intersection would be defined as a 30 × 30m square centered on the crossing of those two centerlines (30m represents two lanes each with a width of 15m).

B. Factors

The factors that were varied in this experiment were the *Complete Loop Distance Error Tolerance*, the *Centerline to Road Distance*, the *Intersection Semantics Consideration*, and the *Time Match Threshold* (**bold** in Table II).

Standard roads have a lane width of 11 ft (3.35 m) [5]. It follows then that two lane roads are 22 ft (6.7 m) wide, and six lane roads with a turn lane are 77 ft (23.47 m) wide. As a result of these different possible road sizes, the levels of 5m, 10m, and 15m for the edge of the road to center are appropriate as listed in Table III.

It follows for the size of intersections, that if roads are between 6.7m wide and 23.5m wide as they are in the urban area we tested, that the distance from one corner to the opposite corner would be $\sqrt{2} \times width$ or 9.5m to 42m. As a result, we would expect our *Complete Loop Distance Error Tolerance* to be near those numbers. Adding an additional 50% tolerance to account for sensor error (dropouts in urban canyons, GPS positioning error, etc.) gives us the values that are the top and bottom for the values in Table IV. Table V lists the levels for the *Intersection Semantics Consideration* factor and Table VI lists the levels for the *Time Match Threshold* factor.

TABLE V
LEVELS OF THE *Intersection Semantics Consideration* FACTOR TO BE TESTED

Level	Value
1	Ignore turns not in intersections
2	Consider all turns

TABLE VI
LEVELS OF THE *Time Match Threshold* FACTOR TO BE TESTED

Level	Value
1	5 min
2	3.75 min
3	2.5 min

With regards to the *Intersection Semantics Consideration* factor, it could be argued that casing events that only consider turns in intersections are more accurate, while others might suggest turning into parking lots and looping around buildings are more suspicious. As a result, we tested both configurations.

C. Workload

The workload used is persistent video data over an urban campus area. The video is post-processed to detect moving vehicles denoted as tracks (vehicular tracking in and of itself is a separate research effort outside the scope of this paper). Tracks are listed in a common coordinate system to relate to other static information from a GIS. GPS track data from the same area was used to represent vehicular track input to our model. The GPS data was collected using vehicle-born waypoint logging devices on eight vehicles. Of the eight vehicles, some vehicles were given suspicious scenarios to act out and others were instructed to generate random driving tracks. The GPS data [6] is a good representative of the persistent video data because GPS data logs position coordinates at nearly the same time interval as persistent video images are captured (about 0.5 sec). Both sets of data are also not without fault, as both GPS data and tracks generated from persistent video can have areas of missed detections caused by system errors or occlusions.

The workload consists of the GPS track data, combined with the results of the point related semantics to include turn detection. In turn detection, three vehicles were used for 106 min of total driving time. Testing confirmed that the system performed best when an angle was formed by taking a point of interest, a point behind that point at least 45m away, and a point ahead of that point at least 20m away. A turn is detected when an angle as calculated by the dot product between the two vectors is less than 160 degrees, and the vehicle was traveling 0.5 mph faster out of the turn, than they were going

into the turn. The results show in this configuration, with a 95% confidence interval, that the TDS detects 81.4% to 92.94% of turns. Given that the system detects a turn we can say with 95% confidence that it is a real detection between 84.34% and 95.5% of the time. While these results could be improved upon, they are accurate enough for our casing event detector.

V. RESULTS AND EXPERIMENT DISCUSSION

The methodology from Section IV was applied to produce the results for the Casing Event Detection System (CEDS). Initial results led to a series of experiments that further refined the best performance of the CEDS. These series of experiments are discussed in this section.

A. Experiment #1: Determining Loop Distance and Intersection Semantics

The first experiment tested and varied the factors for a 106 min tracklet. Prior to experimentation, this data set had six casing events identified by a human. After experimentation, with the help of the computer model, it was discovered that there were actually ten casing events in the track data. The first experiment was run setting *Time Match Threshold* to five min. Referring to the results in Table VII, it was discovered early on (compare Line 1 and Line 2) that restricting those turns to only turns at intersections missed half of the casing events. The root cause was that some casing events occurred in areas of the map that were not roads, but were alleyways. Since the alleyways did not have names, they were not tagged as roads so where they met with roads was not considered an intersection.

TABLE VII
RESULTS OF CASING DETECTION VARYING *Loop Distance* AND *Intersection Semantics*

Line No.	Inter-section Dist	Loop Dist	Inter-section On	Time match (min)	P_D (%)	Prob Real (%)	Exec Time
1	15m	30m	Y	5	50	50	7 min
2	15m	30m	N	5	100	83.3	7 min
3	15m	15m	N	5	90	81.8	7 min
4	15m	60m	N	5	100	83.3	7 min

It was further discovered (compare Lines 2, 3 & 4 in Table VII) that reducing the *Loop Distance* to 15m reduced the number of detections, and increasing it to 60m had no effect. In addition, it was discovered that varying the *Intersection Semantics Consideration*, and the *Loop Distance* did not have any effect on execution time. These findings showed that further tests would likely have the best results if the *Loop Distance* is set to 30m, and the *Intersection Semantics Consideration* is turned off (considering all turns whether they were in intersections or not). The latter will account for inconsistencies on how GPS solutions tag roadways and intersections.

B. Experiment #2: Varying Time Match Threshold and Improving Performance Time

The second experiment tested and varied the *Time Matching Threshold* factor for a 106 min tracklet. The results show (see

Table VIII) that decreasing the *Time Matching Threshold* decreased the execution time. Decreasing it too much can have a negative effect on the detection probability. The results for experiment #2 led us to conclude that somewhere between a *Time Matching Threshold* of 3.75 min and 2.5 min we lose detections. For the remainder of the experiments presented in this article, a *Time Matching Threshold* of 3.75 min is used.

TABLE VIII
RESULTS OF CASING DETECTION VARYING *Time Matching*

Line No.	Inter-section Dist	Loop Dist	Inter-section On	Time match (min)	P_D (%)	Prob Real (%)	Exec Time (min)
1	15m	30m	N	5	100	83.3	7
2	15m	30m	N	3.75	100	83.3	4.25
3	15m	30m	N	2.5	60	75	2.15

C. Experiment #3: Varying Intersection Distance and Improving Performance Time

The third experiment tested and varied the *Centerline to Road Distance* factor over a 106 min tracklet. Table IX shows the results of varying the *Centerline to Road distance* (and likewise the *Intersection Distance*). The results show that decreasing the *Intersection Distance* decreases the execution time at the expense of decreasing the detection probability.

TABLE IX
RESULTS OF CASING DETECTION VARYING *Intersection Distance*

Line No.	Inter-section Dist	Loop Dist	Inter-section On	Time match (min)	P_D (%)	Prob Real (%)	Exec Time (sec)
1	15m	30m	N	3.75	100	83.3	255
2	10m	30m	N	3.75	90	81.8	75
3	5m	30m	N	3.75	50	83.3	13

The execution time of the algorithm is driven by the number of distance calculations in matching the past with the future. Decreasing the *Time Matching* and the *Intersection Distance* decreases execution time because fewer points to compare mean fewer distance calculations. Reducing the number of points tagged in an intersection will also result in fewer distance calculations and decrease execution time. The approach taken was tagging all points within 15m of an intersection, and then not comparing those points less than a certain distance inside the 15m. In this way, we compare points only entering or leaving an intersection, but not all the points in the middle of one. The results in Table X show that only using points that fall between 15m and 10m of an intersection yields a four-fold gain in execution time with no change in P_D . Further testing only using points that fall between 15m and 13m yields a nine-fold gain in execution time with minimal loss in P_D .

Applying these results to a real world situation means that we could track approximately 96 vehicles with high accuracy, and 424 vehicles with slightly diminished accuracy in real-time. Further analysis was done to determine the speed that a vehicle would have to be going in order to have a missed detection in the 15m to 10m zone on each side of an

TABLE X
RESULTS OF CASING DETECTION VARYING *Intersection Distance*

Line No.	Inter-section Dist	Loop Dist	Inter-section On	Time match (min)	P_D (%)	Prob Real (%)	Exec Time (sec)
1	15m	30m	N	3.75	100	83.3	255
2	15m-10m	30m	N	3.75	100	83.3	66
3	15m-13m	30m	N	3.75	90	81.8	15

intersection. At a detection rate of one detection every 0.5 sec, the resulting speed is 22 miles per hour (mph). This means that there is a possibility if a vehicle is traveling over 22 mph that a detection may not occur in the 15m to 10m zone. Since it is very likely that vehicles are traveling over 22 mph through an intersection, we will test these parameters over a larger data set to evaluate if any detections are being missed because of the *Intersection Distance* value of 15m to 10m.

D. Experiment #4: Testing against more data

The fourth experiment tested whether the levels for the factors determined for a subset of the data applied to a larger data set. The first data set used for Experiments #1 - #3 was a single track of 106 min in duration and 10 casing events. In the fourth experiment the data set included eight different tracks totalling 17 hours and about four times as many casing events (43 actually). The configuration of this experiment is the same as Line 2 in Table X. The results were that all casing events were detected and only four false positives were found resulting in a 100% probability of detection and a 91.5% probability that a detection was real.

E. Analysis

A binomial analysis was run on the results from Experiment #4 to determine a 95% confidence interval for detection of a casing event. Since all events were detected a confidence interval cannot be found for the probability of detection that meet binomial normality assumptions. The results of the analysis of the probability that a detection is real show that while we estimate the probability of detection at 91.5%, we can, with 95% confidence, say that it is between 83.54% and 99.46%. This means that given another data set, we can expect 95% of the time that the average probability that a detection is real falls between 83.54% and 99.46% on that new data set.

Even though we are only using points that are between 15m and 10m of an intersection, and there are several instances where vehicles are driving through intersections at greater than 22 mph, the number of missed casing events did not increase. As a result, trimming down to only those points within 15m and 10m of an intersection is not as large of a concern as once thought.

VI. EXTENDING CASING EVENTS DETECTION

It is possible to move further up the knowledge layer (Level 2 of Fig. 2) to add specificity to the casing event (what building or address is being cased) and credibility to the detection (number of times the vehicle has passed this location, and how much time elapsed).

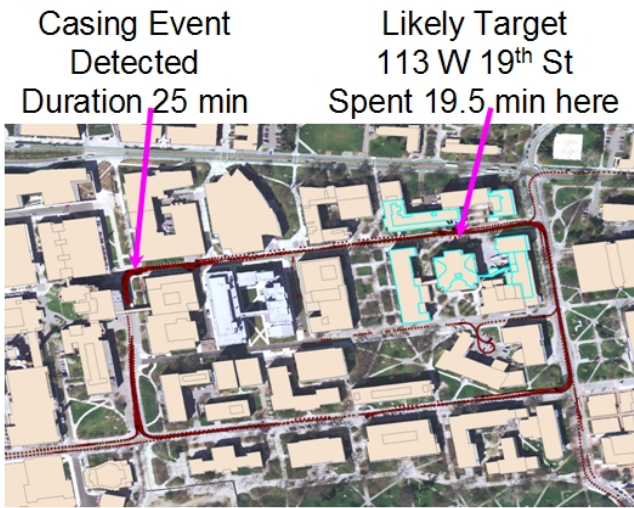


Fig. 5. A screen shot of casing detection and determining which building is the target of surveillance.

A. Time Duration

The output of the casing detection algorithm includes what waypoints are included in the casing event. By comparing the start point to the end point we are able to calculate the duration of the event. This value may be useful in determining whether a casing detection is a true surveillance activity or not. In general, actors conducting surveillance watch targets for a considerable amount of time [7], so a casing event that lasts only a few minutes is not likely a true surveillance activity. We can also use this time duration to help determine what building might be targeted along the route.

B. Target Building

Once we have detected that a casing event has occurred, we can further determine what potential target building was the focus of the surveillance. Since the purpose of surveillance involves the amount of time being spent in view of a target, we calculated the amount of time spent during the casing route near each building. We used this to rank order buildings based on the amount of time spent in front of them. If further screening is required on casing events, we can use both the percentage of time, and the total amount of time in front of a building as discriminators in helping us determine actual surveillance activities from the casing events detected by the system. Fig. 5 shows some results of the casing detector and concludes what group of buildings were most likely the target of the surveillance.

C. Number of Passes

An additional discriminator for determining whether a casing event is indicative of a true surveillance activity is the number of times a vehicle passes by a building of interest. If we have decided that a building is being watched, we can count the number of times that a vehicle has passed by that building and at what times. This information could be aggregated over

time just as the amount of time spent in front of a building could be aggregated to give a more complete view of how much time a target is spending in different locations, and what buildings are potentially threatened by this suspicious vehicle.

D. Address Lookup

Once a building is identified, it is helpful to determine the building address to find it on a map, or relate it to other objects. To get an address, we first calculate the centroid of the building's geometry and get its latitude/longitude coordinates. Next, we use a reverse geocoding lookup web site [8] to translate from latitude/longitude into a street address.

VII. CONCLUSION

In this paper we demonstrated the ability to write and test an algorithm that detects casing events within vehicle tracks. To do this, we used a reasoner to detect three turns in the same direction, then matched points within an intersections back 3.75 min from turn one and forward 3.75 min from turn three. We detected 100% of events and can say, with 95% confidence, that the probability that a detection is real falls between 83.54% and 99.46%. The performance of the system is such that it can process up to 96 vehicles in real-time, or 424 vehicles with a slightly diminished P_D .

Once a casing event was detected, additional analysis was done to provide more information to the user of the system. This included a description of the time involved, a likely target based on where the majority of time was spent, the number of times that vehicle passed the location, and the address of the location of interest. These outputs can be used to help filter out those false alarms of individuals driving loops around the block, that are not actually casing a location.

ACKNOWLEDGMENT

The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

REFERENCES

- [1] H. J. Zhang, Y. Gong, S. W. Smoliar, and S. Y. Tan, "Automatic parsing of news video," in *Int. Conf. On Multimedia Computing and Systems*, May 1994, p. 45.
- [2] "Vulnerabilities in the terrorist attack cycle — stratfor," 10/14/2008. [Online]. Available: http://www.stratfor.com/vulnerabilities_terrorist_attack_cycle
- [3] "Jess, the rule engine for the java platform," 7/28/2008. [Online]. Available: <http://herzberg.ca.sandia.gov/>
- [4] M. C. Daconta, L. J. Obrst, and K. T. Smith, *The Semantic Web*. Indianapolis, Indiana: Wiley Publishing, Inc., 2003.
- [5] "Discussion topic: lane width," 10/16/2008. [Online]. Available: <http://knowledge.fhwa.dot.gov/cops/>
- [6] S. D. Air Force Research Laboratory, "Osu gps data," 28 Oct 2007.
- [7] "Homeland security today - news and analysis - suspicious behavior could indicate terror plotting," 10/14/2008. [Online]. Available: <http://www.hstoday.us/content/view/3932/128/>
- [8] "Geonames," 10/30/2008. [Online]. Available: <http://www.geonames.org/>