Detecting and Tracking Coordinated Groups in Dense, Systematically Moving, Crowds

James Rosswog* Kanad Ghose†

Abstract

We address the problem of detecting and tracking clusters of moving objects in very noisy environments. Monitoring a crowded football stadium for small groups of individuals acting suspiciously is an example instance of this problem. In this example the vast majority of individuals are not part of a suspicious group and are considered as noise. Existing spatio-temporal cluster algorithms are only capable of detecting small clusters in extreme noise when the noise objects are moving randomly. In reality, including the example cited, the noise objects move more systematically instead of moving randomly. The members of the suspicious groups attempt to mimic the behaviors of the crowd in order to blend in and avoid detection. This significantly exacerbates the problem of detecting the true clusters. We propose the use of Support Vector Machines (SVMs) to differentiate the true clusters and their members from the systematically moving noise objects. Our technique utilizes the relational history of the moving objects, implicitly tracked in a relationship graph, and a SVM to increase the accuracy of the clustering algorithm. A modified DBSCAN algorithm is then used to discover clusters of highly related objects from the relationship graph. We evaluate our technique experimentally on several data sets of mobile objects. The experiments show that our technique is able to accurately and efficiently identify groups of suspicious individuals in dense crowds.

1 Introduction

Monitoring large crowds of people for small groups of individuals acting in a suspicious manner is a problem that modern security forces face. This task is often done by human operators and is quite demanding, making the detection prone to many mistakes. In this paper we propose a technique to accurately detect and track small groups of individuals moving in a coordinated manner in the midst of a large, dense crowd. As an example, consider the entrance area to a football stadium before a game. Thousands of people will pass through this area on the way to their seats. The security personnel must monitor this large crowd and detect individuals whose behaviors pose a threat to others and intervene before any harm is done. A group of people ganging up and attacking another person is an example of a hostile behavior. In some parts of the world, today's shipping vessels face a similar problem when operating in congested ports, harbors and coastlines. Small terrorist boats often masquerade as part of the normal sea traffic and when opportunity arises, several such vehicles group together to attack and take over a targeted vessel. Quickly detecting and tracking these swarms of small terrorist boats is critical to protecting the larger ships against this type of attack. In this case, the relationships among these boats can be detected using their relative proximity and/or from detecting radio signals used by the terrorists as they converge on the target.

Automated techniques are needed to identify the coordinated activities or movements of suspicious groups of objects within a large crowd of objects. Such a system must minimize both false positives, which incorrectly identify individuals as being part of a suspicious group, and false negatives, which fail to identify the individuals of a suspicious group. Even a small false positive rate can cause serious problems in the above scenarios because the number of individuals that are not part of a suspicious group (the noise objects) is much larger than the number of individuals that are part of a suspicious group. The inability to identify the clusters of suspicious objects and their membership has consequences. For example, in the Stadium example, incorrectly identifying many innocent individuals as suspicious will force the security personnel to track a large number of individuals, causing long delays for individuals entering the stadium. On the other hand, not detecting the individuals of suspicious groups could compromise the security of the entire stadium.

There are several clustering algorithms that are capable of detecting and tracking clusters of moving objects in noise free environments [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. The Dynamic Density Based Clustering (DDBC) algorithm, introduced in [13], attempts to address this problem in noisy environments, but has some serious limitations. DDBC is capable of detecting and tracking groups of objects moving in a coordinated fashion only when the noise objects are moving randomly. DDBC's problem arises from the assumption that the noise objects will be moving in a random, Brownian mo-

*Department of Computer Science, Binghamton University. Email: jim.rosswog@binghamton.edu

†Department of Computer Science, Binghamton University. Email: ghose@cs.binghamton.edu

tion. In real situations this assumption does not hold as seen from the stadium example described above. The noise objects are the thousands of fans entering the stadium. They are moving from a starting location, the entrance gate, to a goal, their seat. In the maritime example given earlier, the noise objects are the innocent vessels. When DDBC is faced with these types of realistic data sets, it suffers from a high false positive rate (see section 6 on experimental results), which reduces its utility.

1.1 Systematic Noise In most realistic situations that involve crowds, the individuals in the crowd are not moving randomly. The individuals of a crowd are typically moving systematically toward a goal (seat in a stadium, exit, etc). Detecting a small group of individuals in a crowd that is moving systematically is a much harder problem than detecting a small group of individuals moving in a coordinated fashion in the midst of randomly moving individuals. This problem is made even more difficult when the small group is moving in the same general direction as the individuals in the rest of the crowd. Consider the stadium data set illustrated in figure 1(a). Each gray and black box represents an individual entering a stadium. The black boxes represent individuals that are part of a small organized group. These are the individuals that the security personnel are trying to identify in the crowd. The gray boxes represent the innocent individual fans entering the stadium, and are considered as noise by the system attempting to find the groups of suspicious individuals. All individuals (group members and noise) enter the area being observed from the gate area on the left side of the area and move toward the seating area on the right side of the area.

Figure 1(b) shows the results of running the DDBC algorithm on the data from figure 1(a). In this case, boxes colored black are the individuals that DDBC identified as being part of a suspicious group. As can be seen, DDBC correctly labels the members of the right most group, but also incorrectly labels many noise objects as being part of the group. DDBC assumes that the noise objects are moving in a Brownian motion and can therefore use the distance between the objects over time to estimate the strength of the relationships between the individuals. When the noise objects are moving systematically, DDBC is not able to differentiate the suspicious groups from the rest of the crowd.

1.2 Scope of This Work and Contributions We propose a technique called SVM Dynamic Clustering (SVM-DC) that quickly and accurately detects and tracks small groups of individuals moving in a coordinated fashion within a large dense crowd. The goal of SVM-DC is to quickly and accurately detect and track groups of related moving objects in the presence of a dense and systematically moving crowd of other objects. In this work we focus on detecting groups of objects that are moving as a group. SVM-DC if not limited to detecting groups of this type, and can be trained to detect other relationship. Figure 1(c) shows the results of running SVM-DC on the data from figures 1(a). SVM-DC correctly identifies the members of the right most groups without reporting any false positives. The left most group was not identified by either DDBC or SVM-DC because it has just entered the observation area and sufficient data has not been collected on it yet. Detailed experimental results will be presented in section 6. SVM-DC combines the relationship graph from DDBC with Support Vector Machines to simultaneously minimize both false positive and false negative errors. A SVM is capable of reducing both sources of error simultaneously. The SVM is used to differentiate strong stable relationships between the moving objects from weak, transient relationships. The Relations Graph and the modified DBSCAN algorithm proposed by DDBC are used to efficiently maintain these relationships and detect clusters of strongly related objects. The main contributions of this work are:

- We propose a dynamic spatio-temporal clustering algorithm that is capable of accurately detecting and tracking groups of moving objects in the presence of dense, systematically moving noise.
- We experimentally evaluate the performance of the dynamic spatio-temporal clustering algorithm. The performance of the new algorithm is directly compared to the performance of the current state of the art spatio-temporal clustering algorithms.
- We introduce a set of metrics for evaluating dynamic spatio-temporal clustering algorithms that measure all sources of error.

2 Related Work

Clustering data sets that contain static objects is a well studied field [14, 15]. These techniques can be applied to data sets that contain moving objects, by periodically running the static clustering algorithm on snapshots of the dynamic data set. This approach may yield good results when the clusters are well separated, but will tend to merge the clusters when they come near each other. Clustering techniques are needed that consider the dynamic nature of moving objects and the overlap of clusters in space.

The problem of finding clusters of moving objects has been directly addressed in several recent works. Kalnis et al. treats time as a set of discrete intervals [16]. The positions of the objects at each time interval are treated as independent static data sets. A traditional static clustering algorithm is then used to cluster the objects at each time interval. The clusters are then correlated with the clusters created in adjacent time intervals to create a moving set of

(a) Truth Data (b) DDBC Results (c) SVM-DC Resutls

Figure 1: Figure 1(a) shows individuals enter the observation area from the gates on the left side and exit into the seating area on the right side. The black objects represent individuals that are part of an organized group. The gray objects are innocent fans entering the stadium. In figure 1(b), the black objects represent groups identified by the DDBC algorithm. DDBC suffers from a high false positive rate when faced with systematically moving "noise". In figure 1(c), the black objects represent groups identified by the SVM-DC algorithm. SVM-DC is able to detect and track the coordinated group without producing false positives.

clusters. This technique is not capable of tracking moving clusters through periods where they overlap with each other or when noise objects are present.

Li et al. propose a method for detecting moving clusters based on the idea of Moving Micro-Cluster [1]. Moving Micro-Clusters are an extension of the idea of micro-clusters used in the BIRCH clustering algorithm [2]. This technique has several limitations. The definition of micro-cluster limits the algorithm to only finding spherical clusters. The algorithm is also not able to keep clusters separated during periods of overlap with other clusters and objects not belonging to the cluster. Several recent works extend the idea of Moving Micro-Clusters, but share the same set of limitations [3, 4, 5].

Bernkert et al. present several techniques for detecting flocks in sets of moving objects [6]. A flock is defined as a set of objects that are within a radius of each other for a specified number of consecutive time steps. This definition of flock restricts the algorithm to only finding spherical flocks. Zhang et al. propose an efficient technique for continuously clustering moving objects using k-Means [7]. The proposed algorithm makes no attempt to track the moving cluster through time.

Trajectory data mining techniques divide the time dimension into discrete intervals and represent a moving object as a list of its positions at each time interval. Clustering techniques are used to find groups of similar trajectories [8, 9, 10, 11, 12]. These techniques must store the past locations of each moving object. The similarity function used to compare two trajectories must also consider all past positions, making clustering a computationally complex process. Like our proposed scheme, the work of [10] uses density-based clustering as its basis. However, [10] makes use of explicit positional information in the form of inter-trajectory distances and makes use of an optimal time interval for isolating the likely clusters to improve the overall quality of clustering.

In contrast, the dynamic clustering technique proposed in this paper incrementally updates a relationship graph at each time step, using a SVM to track relationships accurately. The relationship graph is a sparsely connected weighted graph that represents the relationships between the moving objects. The number of edges in the relationship graph is expected to be far fewer than the number of position measurements stored and processed by the trajectory data mining techniques. The clustering process can then operate on the relationship graph eliminating the need to store and continuously processes all of the past positions. This allows our technique to utilize the position history of the objects implicitly to increase the accuracy of clusters created without the computation and storage demands of trajectory data mining techniques.

Sequential pattern mining methods attempt to find common sequences of actions performed by moving objects [17, 18, 19, 20, 21, 22, 23, 24]. Finding commonly used driving routes is an example of sequential pattern mining. Sequential pattern mining is fundamentally different than clustering moving objects in that sequential pattern mining does not require that similar sequences occur at the same time. A cluster of moving objects is a set of objects that are moving together in both space and time where a sequential pattern is a set of events that frequently occur in the same sequence.

Dynamic Density Based Clustering (DDBC) efficiently uses the position history of the moving objects to detect and track cluster of moving objects [13]. DDBC is able to detect clusters in presence of randomly moving noise objects. DDBC assumes the noise objects will be moving a Brownian manner. The limitations of DDBC were described earlier in section 1.

3 SVM Dynamic Clustering

The goal of this work is to improve upon existing dynamic clustering techniques, by reducing the false positive rate while maintaining the ability to detect and track clusters of moving objects. We decompose the problem of detecting and tracking clusters of moving objects into 2 subproblems: Relationship Estimation and Cluster Detection. Relationship Estimation is the process of determining the strength of the relationships between the individual moving objects. The strengths of the relationships are maintained in a weighted graph called the Relationship Graph. The Relation Graph contains one node for each object. The edges in the graph connect the nodes that represent objects that may be related. The weight of an edge represents the strength of the relationships. The relationship strength is an estimate of the likelihood that two objects are actually related. Cluster detection identifies groups of related moving objects and tracks the groups' movement through the environment. Cluster detection works on the Relationship Graph and finds clusters of densely connected nodes.

3.1 Relationship Estimation Relationship Estimation is the process of determining which moving objects are related, and how strong the relationship is. Relationships between objects are fundamental to the formation of clusters. In particularly, we are interested in discovering relationships that are stable over a period of time. Transient relationships should not influence the clustering. In order to accurately distinguish between true, stable relationships and false, transient relationships, we must minimize both the false positive and false negative errors. Prior attempts to solve this problem are not capable of minimizing both sources of error and are therefore either incapable of detecting the true relationships, or suffer from a high false positive rate. In order to address this problem we use a SVM to estimate the strength of the relationships between objects. An SVM was chosen for its ability to simultaneously minimize all sources of errors.

The relationship estimation phase of SVM-DC creates a relationship graph where the vertices represent the data objects and the edges represent relationships between the objects. Initially the relationship graph contains one vertex per object and has no edges. The relationship graph is maintained by the following periodic process:

1. Measure the Position of Each Object
2. Estimate the Relationships Between the Objects Using the SVM
3. Update the Relationship Graph

The positions of all the objects in the observation area are measured periodically. Objects that are within each others' Region Of Interested (ROI) are used to update the relationship graph. The ROI is an area around each object that defines other objects that it may be related to. The ROI is used to limit the number of pairs of objects that need to be processed at each time step. There is no reason to estimate the relationship strength between objects that are far away from each other when we are looking for groups of object that are moving together. In our examples the ROI is defined as a physical area around an object. In another application the ROI could be an area defined in an abstract feature space. The object positions are maintained in an R-Tree to improve the efficiency of finding pairs of objects that are within each other's ROI. The relationship strength estimates along with features used by the SVM (see section 3.1.1) are stored on the edges of the relationship graph. The runtime complexity of the maintaining the relationship graph is $O(n \log n)$, assuming that the graph is sparsely connected. The number of edges maintained in the relationship graph is expected to be small.

3.1.1 Feature Selection In order to use a SVM for relation estimation, we had to select a set of features for the SVM to process. We selected a set of features that describe the movements of the two moving objects being examined, and the environment they are moving through. The following features are computed at each time interval, for each pair of objects that are within each others ROI:

- **Observation Time:** This is the number of time steps that the two objects have been within each others ROI.
- **Distance Mean:** The mean distance between the two objects during the observation time.
- **Distance Variance:** The variance of the distance between the two objects during the observation time.
- **Distance Rate of Change:** The rate of change of the distance between the two objects during the observation time.
- **Low Local Density:** The density of objects within the ROI of both moving objects are computed. This is the smaller of the two densities.
- **High Local Density:** The larger of the two local densities
- **Global Density:** The density of the objects in the entire observation area.

Observation time, distance mean, and distance variance are used to measure the stability of the relationship. Distance rate of change is used to detect periods of transition, where relationships are forming or being broken. Low local density, high local density, and global density are included as indicators of the overall activity of the crowd the objects are moving through and allow a trained SVM to be used in several different scenarios.

3.1.2 Parameter Selection In order to utilize a SVM we need to choose a kernel function and the parameters for the kernel. We used an Evolutionary Programming (EP) technique based on [25] to determine the best parameters to use. The EP process encodes a set of SVM parameters as a chromosome. An initial population of chromosomes is randomly generated and their fitness estimated. The chromosomes are mutated over n-generations. After each generation, the fitness of each chromosome is estimated. This fitness is used when selecting which chromosomes will survive to the next generation. After the final generation the chromosome with the best fitness is selected and used as the parameters for training the SVM.

The EP process begins by generating an initial population of chromosomes. Each chromosome contained the following genes:

- **Kernel Function:** Linear, Polynomial, Radial Basis Function, Sigmoid
- **C:** Controls the softness of decision boundary [26]. A larger value of C allows for a softer margin.
- **γ:** The parameter for the RBF Kernel
- **Degree:** The degree of the polynomial kernel function
- **Coef 0:** The coefficient for the linear and polynomial kernel functions

Once the initial population is created the chromosomes are evolved through a series of generations. During each generation the fitness of each chromosome is evaluated using the fitness function described bellow. A tournament selection algorithm is used to select the top 40% of the population. These chromosomes are then mutated. The chromosomes and their mutants are then added to the next generation. The remaining 20% of the population of the next generation consists of new randomly generated chromosomes.

To evaluate the fitness of each chromosome, a SVM is trained and evaluated using the parameters from each chromosome. Five fold cross validation is used to train and evaluate the SVM. The area under the Receiver Operation Characteristic (ROC) curve is used as the fitness estimate

Chromosomes are mutated to explore the parameter space around the chromosomes with the highest fitness. Each gene in the chromosome is independently mutated with probably 0.75. If the kernel gene is selected to be mutated, one of the possible kernels is randomly selected. The C, γ, and Coef0 parameters are mutated by adding a normally distributed random number with mean 0 and variance 1. The degree parameter is mutated in similar manner, but the random number is rounded to the nearest integer. This process was run for 500 generations. The Radial Basis Function (RBF) was selected as the kernel, C was set at 3.89845, and γ was set at 0.00424.

3.1.3 Relationship Graph Information about the relationships between objects is maintained in the relationship graph. The relationship graph contains one vertex for each moving object being tracked. Weighted edges represent relationships between the objects. At each time interval the SVM described above is used to evaluate the relationship between each pair of objects that are within each others ROI during the time interval. The weight of the edge is equal to the result of the SVM. The relationship graph tracks the evolution and stability of relationships by collecting the feature information and using the SVM described above to evaluate the strength of the relationship between two objects. *This allows clusters to be constructed from the relationship graph without making direct use of the objects' history*. The relationship graph is more efficient than techniques that directly use history information because only the features are stored and the graph is expected to be sparsely connected.

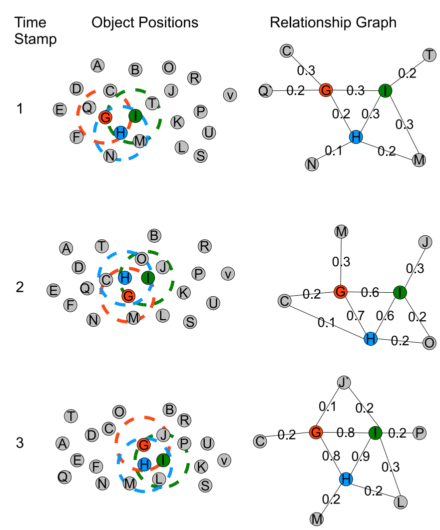


Figure 2: Illustration of how the relationship graph is maintained over time.

Figure 2 illustrates the process of maintaining the Relationship Graph. Objects G, H, and I are part of group moving through a densely cluttered area. At each time period the positions of the objects are shown on the left and the resulting relationship graph on the right. Only the edges adjacent to objects G, H, and I are shown. The ROI for objects G, H, and I are shown as dashed circles. The relationship graph contains no edges prior to time 1. At time 1, an edge is added between each pair of objects that are within each others' ROI. At time 2 the weight of existing edges are updated using the SVM and if two objects are within each others' ROI and an edge does not exist between the objects, an edge is created. Edges with a weight less than or equal to 0 are removed from the graph. The same process is repeated with the position measurements from time 3.

3.2 Density Based Clustering Once the relationship graph has been updated, the modified DBSCAN algorithm is used to find clusters of highly related vertices. The DBSCAN definition of cluster is based on the idea of density-connected objects [27]. A pair of objects p and q are density-connected if there exists a third object r where p and r are density-reachable and q and r are density reachable. An object p is said to be directly-density reachable from object r if p and r are within each others Eps-Neighborhood and r's Eps-Neighborhood contains at least minPoints (a parameter to the algorithm) objects. Two objects p and r are density reachable if there is a series of objects $p_1, ..., p_n$ where $p_1 = p$ and $p_n = r$ and each p_{i+1} is directly density reachable from p_i. A cluster is then defined as as set of objects that: 1) are all mutually density-connected and 2) all other objects that are density-connected to any point in the cluster.

We use the same definition of cluster, but replace the Eps-Neighborhood with the RST Relationship Neighborhood. The RST Relationship Neighborhood of an object e is the set of objects connected to e with an edge that has a weight greater than the Relationship Strength Threshold (RST). The SVM computes the weight of the edges at each timestep as described in section 3.1.3. RST defines the point at which we are confident that an edge in the graph represents a true relationship.

DEFINITION 1. *Relationship Strength Threshold: Only relationships in the relationship graph stronger (greater than or equal to) than RST are considered by the clustering algorithm. Relationships that are weaker (less than) than RST are considered to be caused by noise.*

DEFINITION 2. *RST Relationship Neighborhood: The RST relationship neighborhood of vertex $v \in V(G)$ denoted $N_{RST}(v)$ is defined as:*

$$N_{RST}(v) = \{\forall u \in V(G) | \exists E(v,u) \wedge weight(E(v,u)) \geq RST\} \tag{3.1}$$

In the original DBSCAN algorithm, a spatial search through all the objects was required to find the Eps-Neighborhood of an object. In the Modified DBSCAN algorithm, the RST Relationship Neighborhood can be found by examining only the edges adjacent to the object.

4 Measures Of Performance

To effectively detect and track clusters of moving objects in a noisy environment, an algorithm must minimize the following sources of error:

- **False Positives:** Occur when an object that is not part of an organized group is identified as part of a group.
- **False Negatives:** Occur when an object that is part of an organized group is not identified as part of a group.
- **Clustering Errors:** Occur when an object that is part of organized group, is identified as belong to a different organized group.

False positives are very important when the ratio of the number of objects that are not part of an organized group to the number of objects that are part of an organized group is high. In the stadium example from above, even a modest false positive rate would result in falsely identifying many innocent individuals as being suspicious. This will cause the unnecessary deployment of security personnel, potentially causing them to miss the truly suspicious individuals and causing long lines and delays entering the stadium. On the other hand, false negatives can result in potentially dangerous individuals making their way into the stadium. Clustering errors can make it difficult for security personnel to track and monitor groups, causing them to be less effective.

To evaluate the performance of the clustering algorithm we used the following metrics: Precision, Recall, and Average False Positives per time step. Precision and recall are often used to measure the accuracy of pattern recognition and information retrieval systems, but neither will identify false clusters created from noise objects. To track this source of error, the average number of false positives per time step is presented. Table 1 shows how these metrics map to the source of errors identified above.

	False Positives	False Negatives	Clustering Error
Precision	V		X
Recall		X	X
False Positives	X		

Table 1: Mapping of the sources of error in dynamic spatio-temporal clustering to the metrics used to measure the error. An X indicates that the metric measures the error and a V indicates the metric partially measures the error.

Precision and Recall are often used to measure the accuracy of pattern recognition and information retrieval systems. Given a query over a set of data, precision is defined as the fraction of returned data instances that truly match the query.

$$\frac{|RelevantData \cap ReturnedData|}{|ReturnedData|} \tag{4.2}$$

Recall is defined as the fraction of the data instance that truly match the query that are returned by the query.

$$\frac{|RelevantData \cap ReturnedData|}{|RelevantData|} \tag{4.3}$$

To utilize precision and recall to measure the accuracy of dynamic clustering algorithms we must define a query and account for the performance of the algorithm over time. We cannot simply use all moving objects identified as being part of a group as the query because there may be multiple groups present and we are interested in detecting and tracking each group individually. We propose using a single moving object as the query. The result of the query will be all other moving objects that have been placed in the same group as this object by the clustering algorithm being evaluated. Precision and recall are then computed for each group member. The precision and recall for the group is defined as the average precision and recall of the group members.

Let G be the set of truth groups: The precision (p) and recall (r) of group $g \in G$ at time t is defined as:

(4.4)

$$p = \frac{\sum_{\forall o \in g} \frac{|g \cap cluster(o,t)|}{|cluster(o,t)|}}{|g|}, r = \frac{\sum_{\forall o \in g} \frac{|g \cap cluster(o,t)|}{|g|}}{|g|}$$

Where $cluster(o,t)$ returns the set of objects the clustering algorithm assigned to the same cluster as moving object o at time t.

In order to compute a precision and recall measure over all time steps we replace $cluster(o,t)$ with $cluster(o)$. This function returns the set of moving objects that the clustering algorithm assigned to the same group as object o during at least one time interval. We call these measures overall precision and overall recall.

The overall precision (op) and overall recall (or) of group $g \in G$ is defined as:

(4.5)

$$op = \frac{\sum_{\forall o \in g} \frac{|g \cap cluster(o)|}{|cluster(o)|}}{|g|}, or = \frac{\sum_{\forall o \in g} \frac{|g \cap cluster(o)|}{|g|}}{|g|}$$

5 Dynamic Spatio Temporal Data Sets

Real spatio-temporal data sets are extremely hard to find. Many entities that collect such data sets do not publicly release the data sets due to privacy concerns. To the best of our knowledge there are no publicly available spatio-temporal data sets that can be used as benchmarks to measure the performance of spatio-temporal clustering techniques. To evaluate the proposed clustering algorithms, we generated several spatio-temporal data sets. These data sets can be obtained for future research by contacting the author. The datasets are intended to represent a large crowd of sports fans entering a stadium. Moving objects that represent fans enter the area being observed through a series of gates along the left side of the area. Figure 1(a) illustrates one such data set. Each fan then moves through the area to a randomly generated point along the right edge of the area. The point along the right edge represents the fan's seat in the stadium. The fans move in a realistic way, avoiding each other and the walls of the stadium. The crowd generation algorithm described in [28] was used to control their movements. We were able to change the density of the fans in the observation area by controlling the rate that fans were let in through the gates. In addition to the fans, small groups of coordinated objects were added to the area. These groups entered through the same gates as the fans, but moved as a swarm toward a common location in the seating area of the stadium. These groups are intended to represent organized groups that may pose a threat to the safety of the other fans. The fan objects are noise to an algorithm attempting to detect groups of coordinated objects. Four stadium data sets were created to evaluate how the dynamic clustering algorithms are able to scale as the number of noise objects and the number of groups of coordinated objects is increased.

The first 3 data sets each contains 8 organized groups of objects that move through the area. On average there are two groups in the area at the same time. Each of these groups consists of 10 objects. We generated 3 data sets with varying densities of fan objects in the observations areas. These data sets will be called stadium-low noise (which contains on average 188 fan objects), stadium-medium noise (which contains on average 343 fan objects), and stadium-high noise (which contains on average 474 fan objects). One final data set was created that contained an average of 4 organized groups and 512 noise objects in the observation area at a time. This data will be called stadium-high noise+. Table 2 summaries the key features of these data sets.

In addition to these data sets, two data sets with randomly moving noise objects that were presented in [13] are also used. The noise objects in these data sets are moving randomly in a Brownian motion. The first of these data sets contains one organized group of 20 objects and 150 noise objects. The final data set contains 6 organized groups of individual, each with a different number of objects, and 200 noise objects. Like the noise, the organized groups are also moving in a Brownian manner. The key attributes of these data sets are summarized in table 2. SVM-DC is directly compared to three other dynamic clustering algorithms. DDBC, a trajectory mining algorithm based on [10] and a moving cluster mining technique based on [16] were also run on the data sets.

Data Set	Avg. Number of Groups in Observation Area	Avg. Group Size in Observation Area	Avg. Number of Noise Objects in Observation Area
Stadium-Low Noise	2	10	188
Stadium-Med Noise	2	10	343
Stadium-High Noise	2	10	474
Stadium-High Noise +	4	10	512
Random Data Med Noise	1	20	150
6 Cluster Random Data High Noise	6	18	200

Table 2: Summary of the Data Sets Used to Evaluate the Dynamic Spatial Temporal Clustering Algorithms

6 Experimental Results

Below are the results from running SVM-DC, DDBC, Trajectory Clustering, and Moving Cluster Mining on the data sets described above. The results clearly show that SVM-DC is the only algorithm evaluated that is capable of detecting and tracking coordinated groups of moving objects in dense noise. DDBC performs well when the noise objects are moving randomly, but suffers from a high false positive rate on the crowd data sets where the noise objects move systematically.

Gama	1.00000	sa7	sa8	sa9
	0.50000	sa4	sa5	sa6
	0.00424	sa1	sa2	sa3
		2.89845	3.89845	4.89845
			C	

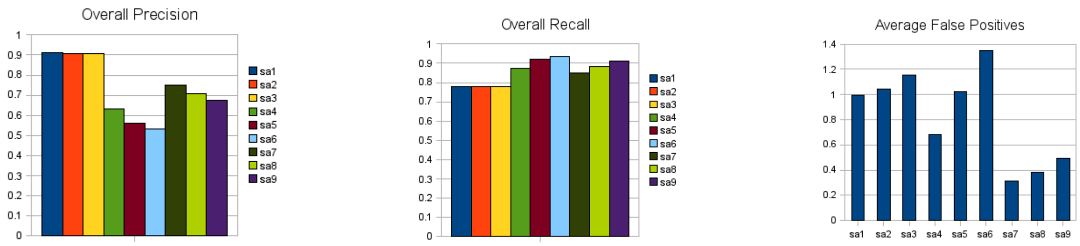
Figure 3: Values of C and Gamma that were evaluated.

6.1 Training The SVM In order to use SVM-DC, the SVM used during the relationship estimation phase must be trained. We generated one set of training data for the random data sets and one training data set for the stadium data sets. The random data set contained noise objects that randomly moved around in the observation area. The density of the noise varied and included the densities of the noise in the test data sets. The crowd training data set contained systematically moving noise objects. Like the random data training sets, the density of the noise was varied and included the densities used in the test data sets. Both data sets contained groups of organized objects that moved through the area. The number of objects in the groups and the densities of the groups were varied. The SVM was trained on each data set independently to produce two models. One model was used for all of the random data set experiments and the other was used for all of the stadium data set experiments. In both cases the RBF kernel was used with C set to 3.89845 and γ set to 0.00424. These parameters were selecting using the processes described in section 3.1.2. To validate the result of the EP process, we evaluated several sets of parameters on the Stadium-High Noise data set. Figure 3 shows the values of C and Gamma that were evaluated.

Figure 4(a) shows the overall precision achieved using the RBF kernel and the parameters described in figure 3. The performance of SVM-DC is more sensitive to changes in Gamma than to changes in C. The three parameter combinations where Gamma was set to 0.00424 outperformed all other parameter combinations tested. Figure 4(b) shows the overall recall achieved with the nine parameter combinations. At first glance, recall appears lower than was expected. Upon closer inspection it was discovered that the last group to enter the observation area was not present long enough for the algorithms to detect it before the simulation was stopped. When this group is removed, the recall for all parameter combinations is very close to 1.0. This is explained in more detailed in section 6.3. The parameter combinations where Gamma is set to 0.5 and 1.0 perform better in this measure because they are able to detect some of the members of the last group before the data set ends. This detection speed comes at the expensive of overall precision.

All parameter combinations tested were able to control the number of false positives. Figure 4(c) shows the average number of false positives per time step for each parameter combination. Considering overall precision, overall recall, and average number of false positives, it is clear that Gamma should be set to 0.00424 and that the value of C is less important. This confirms that the EP process described in section 3.1.2 selected a good set of parameters for the SVM.

6.2 Random Noise Data The random noise sets are included in this study to show that SVM-DC performs well on data sets that have been used in past studies. DDBC performs very well on these data sets because the noise objects are moving randomly. Figure 5(c) shows the average number of false positives generated by the DDBC, SVM-DC, Moving Clusters, and Trajectory Mining algorithms. Both DDBC and SVM-DC successfully control the number of false positives generated. Figure 5(a) shows the overall precision of the four algorithms. DDBC and SVM-DC achieve near perfect precision. Figure 5(b) shows the overall recall of the four algorithms. All algorithms perform very well on this



(a) Overall Precision

(b) Overall Recall

(c) Average False Positives

Figure 4: Overall precision, overall recall, and average false positives for several values of C and Gamma.

metric. This indicates that all algorithms identified all members of the cluster, but as shown by the metrics, only DDBC and SVM-DC were able to do so without a producing a large number of false positives.

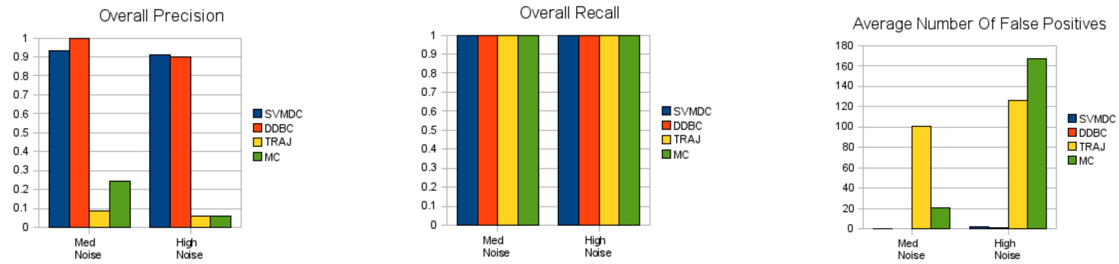
6.3 Crowd Data The crowd data sets introduce noise objects that are moving systematically through the area. These data sets simulate a large crowd entering a stadium. All objects enter the area from the left side and move to a randomly assigned goal point on the right side of the area. The goal point represents a seat in the stadium. The groups of objects move in the same way, they start from the left and move to a goal on the right. It is much more difficult to detect and track the groups of moving objects in the crowd data set because the groups and noise object are moving in a similar manner. Figure 6(c) shows the average number of false positives produced by SVM-DC, DDBC, and Trajectory Mining on the four random crowd data sets. The Moving Cluster algorithm produced an order of magnitude more false positives than the Trajectory Mining algorithm and has been omitted from Figure 6(c). SVM-DC is the only algorithm that does not produce a large number of false positives. DDBC and Trajectory Mining produce almost as many false positives as there are objects that are actually part of a group. As discussed above, a high false positive rate significantly reduces the utility of a surveillance system.

Figure 6(a) shows the overall Precision for SVM-DC, DDBC, and Trajectory Mining on the four crowd data sets. The Moving Cluster algorithm is again omitted because of its poor performance. SVM-DC is able to able to out perform the other algorithms in is category. As described in section 4 above, Precision does not account for clusters that are produced from purely noise objects (this is accounted for in the False Positive metric). The Precision results combined with the False Positive results shows that SVM-DC is the only algorithm that can accurately detect groups of moving objects in a dense, systematically moving crowd.

Figure 6(b) show the overall Recall for SVM-DC, DDBC, and Trajectory Mining on the four crowd data sets. Both SVM-DC and DDBC perform well in the category. At first glance recall appears lower than was expected. We expected both SVM-DC and DDBC to achieve a recall very close to 1.0. Upon closer inspection it was discovered that the last group to enter the observation area was not present long enough for any of the algorithms to detect it before the simulation was stopped. SVM-DC and DDBC achieve a recall of nearly 1.0 when this group is removed. DDBC achieves a slightly better recall than SVM-DC on the final data set. DDBC is able to detect some of the members of the final group in the dataset where SVM-DC is not. This comes at the expensive of a high number of false positives. The recall results combined with the false positive results show that SVM-DC is the only algorithm capable of detecting groups of moving clusters in a dense, systematically moving crowd without producing a large number of false positives.

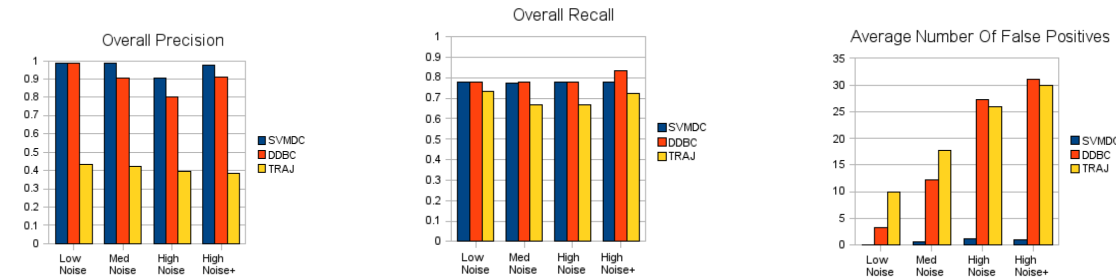
7 Conclusion

Detecting and tracking coordinated groups of moving objects in the presence of dense, systematically moving noise is a problem that security and surveillance personal face today. This is a very difficult problem because the number of suspicious individuals is very small compared to the number of non suspicious individuals, and the hostile individuals will be attempting to blend into the crowd. Existing techniques suffer from a high false positive rate when faced with systematically moving, dense noise. Even a low false positive rate will results in many more false detections than there are actual suspicious individuals when the ratio of non suspicious objects to suspicious objects is large. In this paper we show that existing spatio-temporal clustering algorithms are not capable of detecting and tracking groups of objects in the presence of dense, systematically moving noise. We present an algorithm that we call SVM-DC that is designed to identify groups of related individuals in these types of situations. SVM-DC uses a support vector machine and a Relationship Graph to detect and track relationships between individual moving objects. The SVM allows the algorithm to



(a) Overall Precision (b) Overall Recall (c) Average False Positives

Figure 5: Overall precision, overall recall, and average false positives on the random data sets.



(a) Overall Precision (b) Overall Recall (c) Average False Positives

Figure 6: Overall precision, overall recall, and average false positives on the crowd data sets.

simultaneously reduce the number of false positive and false negative errors. A modified version of the DDBC clustering algorithm is then run on the Relationship Graph to identify groups of related objects from the individual relationships.

We define a set of metrics for evaluating dynamic spatio-temporal clustering algorithms and measure all sources of error in this complex environment. We also define a set of data sets that simulate a crowd of individuals entering a sports stadium. These datasets are available to evaluate other dynamic spatio-temporal clustering algorithms. We compare SVM-DC with three other algorithms. The results show that SVM-DC is the only algorithm capable of accurately detecting and tracking coordinated groups of individuals in dense, systematically moving crowds. SVM-DC successfully detected and tracked all coordinated groups and reduced the number of false positives by more than 90% when compared to the current state of the art spatio-temporal clustering algorithms.

References

[1] Y. Li, J. Han, and J. Yang, "Clustering Moving Objects," *Proc. of the 10th ACM SIGKDD Int'l. Conf. on Knowledge Discovery and Data Mining*, 2004.

[2] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," *Proc. of the 1996 ACM SIGMOD Int'l. Conf. on Management of Data*, 1996.

[3] R. Nehme and E. Rundensteiner, "SCUBA: Scalable Cluster-Based Algorithm for Evaluating Continuous Spatio-temporal Queries on Moving Objects," *10th Int'l. Conf. on Extending Database Technology*, 2006.

[4] J. Chen, C. Lai, X. Meng, J. Xu, and H. Hu, "Clustering Moving Objects in Spatial Networks," *Lecture Notes in Computer Science*, vol. 4443, 2007.

[5] C. Jensen, D. Lin, and B. Ooi, "Continuous Clustering of Moving Objects," *IEEE Trans. on Knowledge and Data Engineering*, 2007.

[6] M. Benkert, J. Gudmundsson, F. Hübner, and T. Wolle, "Reporting flock patterns," *Computational Geometry: Theory and Applications*, 2007.

[7] Y. Y. T. A. Zhang, Z. and D. Papadias, "Continuous k-Means Monitoring Over Moving Objects," *IEEE Trans. on Knowledge and Data Engineering*, 2008.

[8] S. Elnekave, M. Last, O. Maimon, Y. Ben-Shimol, H. Einsiedler, M. Friedman, and M. Siebert, "Discovering Regular Groups of Mobile Objects Using Incremental Clustering," *5th Workshop on Positioning, Navigation and Communication*, 2008.

[9] S. Gaffney and P. Smyth, "Trajectory clustering with mixtures of regression models," *Proc. of the fifth ACM SIGKDD Int'l. Conf. on Knowledge discovery and data*

mining, 1999.

[10] M. Nanni and D. Pedreschi, “Time-focused clustering of trajectories of moving objects,” *Journal of Intelligent Information Systems*, vol. 27, no. 3, 2006.

[11] M. Vlachos, G. Kollios, and D. Gunopulos, “Discovering similar multidimensional trajectories,” *Data Engineering, 2002. Proc.. 18th Int'l. Conf. on*, 2002.

[12] B. Takács and Y. Demiris, “Balancing Spectral Clustering for Segmenting Spatio-temporal Observations of Multi-agent Systems,” *Proc. of the IEEE Int'l. Conf. on Data Mining*, 2008.

[13] J. Rosswog and K. Ghose, “Efficiently detecting clusters of mobile objects in the presence of dense noise,” *Proc. of the 2010 ACM Symposium on Applied Computing*, 2010.

[14] P. Berkhin, “Survey of clustering data mining techniques,” *Technical Report, Accrue Software*, 2002.

[15] A. Jain, M. Murty, and P. Flynn, “Data clustering: a review,” *ACM Computing Surveys*, vol. 31, no. 3, 1999.

[16] P. Kalnis, N. Mamoulis, and S. Bakiras, “On discovering moving clusters in spatio-temporal data,” *Proc. of 9th Int'l. Sym. on Spatial and Temporal Databases*, 2005.

[17] H. Cao, N. Mamoulis, and D. Cheung, “Mining frequent spatio-temporal sequential patterns,” *Fifth IEEE Conference on Data Mining*, 2005.

[18] M. Celik, S. Shekhar, J. Rogers, J. Shine, and J. Yoo, “Mixed-Drove Spatio-Temporal Co-occurrence Pattern Mining: A Summary of Results,” *Proc. of the Sixth Int'l. Conf. on Data Mining*, 2006.

[19] M. Celik, J. Kang, and S. Shekhar, “Zonal Co-location Pattern Discovery with Dynamic Parameters,” *7th IEEE Int'l. Conf. on Data Mining*, 2007.

[20] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, “Trajectory pattern mining,” *Proc. of the 13th ACM SIGKDD Int'l. Conf. on Knowledge discovery and data mining*, 2007.

[21] Y. Huang, Z. L., and P. Zhang, “A Framework for Mining Sequential Patterns from Spatio-Temporal Event Data Sets,” *IEEE Trans. on Knowledge and Data Engineering*, 2008.

[22] D. Sacharidis, K. Patroumpas, M. Terrovitis, V. Kantere, M. Potamias, K. Mouratidis, and T. Sellis, “On-line discovery of hot motion paths,” *Proc. of the 11th Int'l. Conf. on Extending database technology: Advances in database technology*, 2008.

[23] B. Morris and M. Trivedi, “Learning and classification of trajectories in dynamic scenes: A general framework for live video analysis,” *Proc. of the IEEE Int'l. Conf on Advanced Video and Signal Based Surveillance*, 2008.

[24] C. Piciarelli, C. Micheloni, and G. Foresti, “Trajectory-based anomalous event detection,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2008.

[25] W. Land Jr, D. McKee, R. Velazquez, L. Wong, J. Lo, and F. Anderson, “Application of support vector machines to breast cancer screening using mammogram and clinical history data,” *Proc SPIE 2003*, 2003.

[26] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, 1995.

[27] M. Ester, H. Kriegel, J. Sander, and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, 1996.

[28] N. Pelechano, J. M. Allbeck, and N. I. Badler, “Controlling individual agents in high-density crowd simulation,” *Proc. of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007.