CROWD BEHAVIOR ANALYSIS

PRANSHU GUPTA [13493] AND **LAVISHA AGARWAL** [13375]
MENTOR: **Prof. Vinay P. Namboodiri**

ABSTRACT

Crowd behavior analysis is an important field of research in modern world. It has wide applications in surveillance and public safety which are one of the prime social concerns. One way to analyze crowd behavior is obtain crowd movement data and then find out outliers in the individual trajectories to infer any abnormal behavior in the crowd.

INTRODUCTION

We have implemented a system that takes a set of trajectories obtained from crowd data and detects the outliers in that set. A trajectory is a sequence of points (x, y, t), where x, y are the ground co-ordinates of the person at time t.

METHODOLOGY

We have implemented the Trajectory Outlier Detection algorithm as proposed in [1]. The algorithm detects the outliers from a given set of trajectories. There are three phases in this algorithm:

1. Partition
2. Detection
3. Marking

NOVEL APPROACH

We have incorporated temporal data along with spatial data for outlier detection. The similarity between trajectories in [1] is found entirely through geographical location but in our approach we have added the speed component also for computing similarity.

PARTITION PHASE

All the trajectories are partitioned into line segments and all the segments are collectively sent (L) to the detection phase. These line segments are called t-partitions. The technique used for partitioning a trajectory in smaller line segments is based on the principle of Minimum Description Length [3].

We aim at finding the points in the trajectory where the behavior of the trajectory changes rapidly.

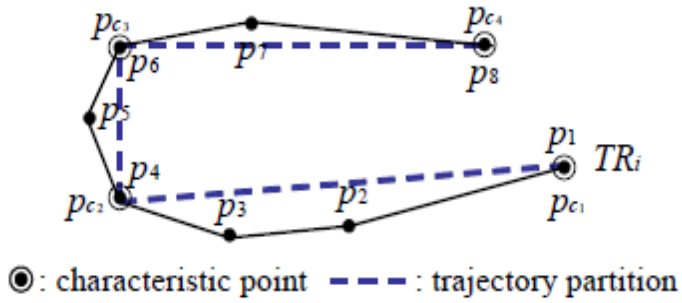


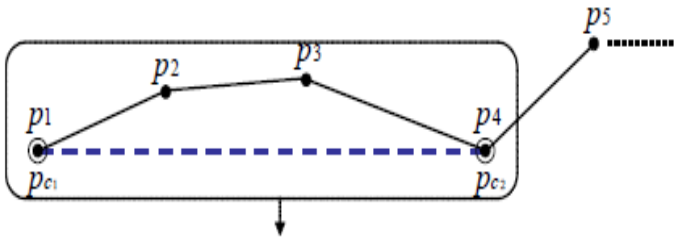
Image taken from [3] ---- Lee Jae-Gil Lee, Jiawei Han, K.Y. Whang

These points are called characteristic points. Then, the trajectory is partitioned at every characteristic point found in it and then represented by a set of line segments that join two consecutive characteristic points.

THE MINIMUM DESCRIPTION LENGTH PRINCIPLE [3]

The optimal partitioning of a trajectory should possess two desirable properties: preciseness and conciseness. Preciseness means that the difference between a trajectory and a set of its trajectory partitions should be as small as possible. Conciseness means that the number of trajectory partitions should be as small as possible. This is called the MDL principle. **[3]**

The cost of MDL consists of two terms L(H) and L(D|H). Here, L(H) represents the sum of the length of all trajectory partitions. L(D|H) represents the sum of the difference between a trajectory and a set of its trajectory partitions.



$$L(H) = len(p1p4)$$

$$L(D|H) = distance(p1p2, p1p4) + distance(p2p3, p1p4) + distance(p3p4, p1p4)$$

Image taken from [3] ---- Lee Jae-Gil Lee, Jiawei Han, K.Y. Whang

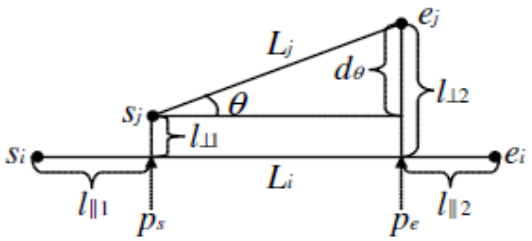
To get the optimal trajectory partitioning we minimize the MDL cost using the following approximate algorithm [3]

INPUT: A trajectory $TR_i = p_1 p_2 p_3 \cdots p_j \cdots p_{len_i}$
OUTPUT: A set CP_i of characteristic points
ALGORITHM:
01: Add p_1 into the set CP_i; /* the starting point */
02: $startIndex := 1$, $length := 1$;
03: **while** $(startIndex + length \leq len_i)$ **do**
04: $currIndex := startIndex + length$;
05: $cost_{par} := MDL_{par}(p_{startIndex}, p_{currIndex})$;
06: $cost_{nopar} := MDL_{nopar}(p_{startIndex}, p_{currIndex})$;
/* check if partitioning at the current point makes the MDL cost larger than not partitioning */
07: **if** $(cost_{par} > cost_{nopar})$ **then**
/* partition at the previous point */
08: Add $p_{currIndex-1}$ into the set CP_i;
09: $startIndex := currIndex - 1$, $length := 1$;
10: **else**
11: $length := length + 1$;
12: Add p_{len_i} into the set CP_i; /* the ending point */

MDL_{par} is the MDL cost of partitioning the trajectory at the previous point during the iteration and $MDL_{no\text{-}par}$ is the MDL cost if we don't partition the trajectory at the previous point.

DISTANCE MEASURE

The distance between two t-partitions is defined as weighted mean of the perpendicular distance, parallel distance, angular distance and the speed difference between the partitions.



$$d_\perp = \frac{l_{\perp 1}^2 + l_{\perp 2}^2}{l_{\perp 1} + l_{\perp 2}}$$

$$d_\parallel = \mathrm{MIN}(l_{\parallel 1}, l_{\parallel 2})$$

$$d_\theta = \|L_j\| \times \sin(\theta)$$

Image taken from [1] ---- Lee Jae-Gil Lee, Jiawei Han, Xiaolei Li

DETECTION PHASE

For each t-partition we find out the set of trajectories that are close to it $CTR(Li, D)$. A trajectory is said to be close to a t-partition if the length of the part of the trajectory similar to the t-partition is greater than the t-partition's length. Two t-partitions are said to be similar if the distance between them is less than a threshold (D). Now, if the product of a density parameter and number of trajectories close to a t-partition is less than a given fraction $(1-p)$ of the total number of trajectories, then we say that the t-partition is an outlier.

Density parameter $density(L_i)$ for a t-partition is the number of t-partitions which are within a distance equal to the standard deviation of all the pairwise distances between the t-partitions;

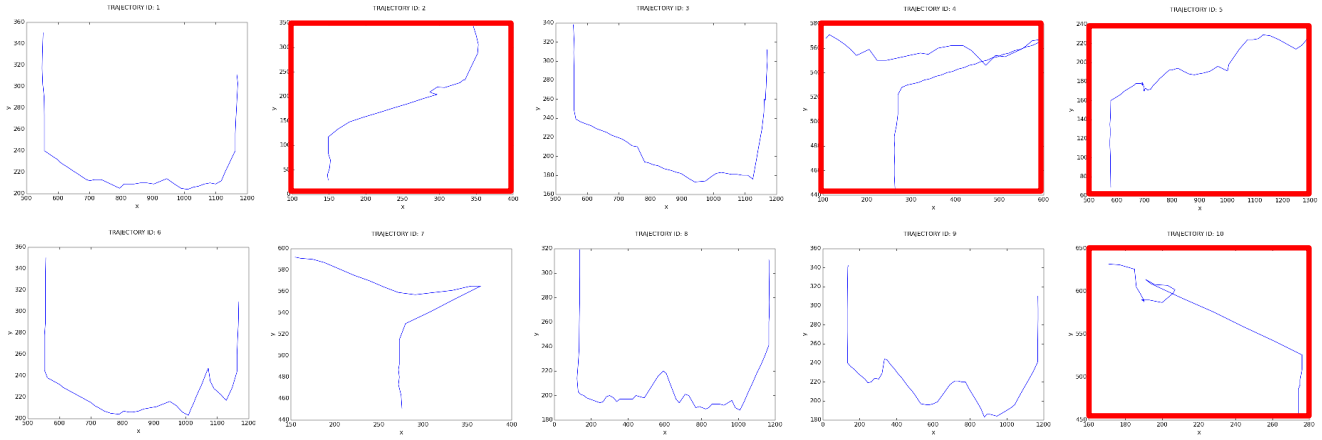
Adjusting coefficient $Adj(L_i)$ for a t-partition is the ratio of the average density to the density of the t-partition.

MARKING PHASE

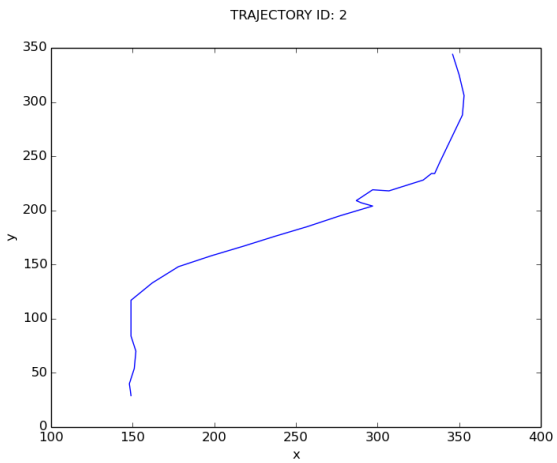
If the ratio of total lengths of outlying t-partitions in a trajectory to the length of the trajectory itself is greater than a given threshold (F) then we say that the trajectory is an outlier. The complete algorithm is as follows [1]

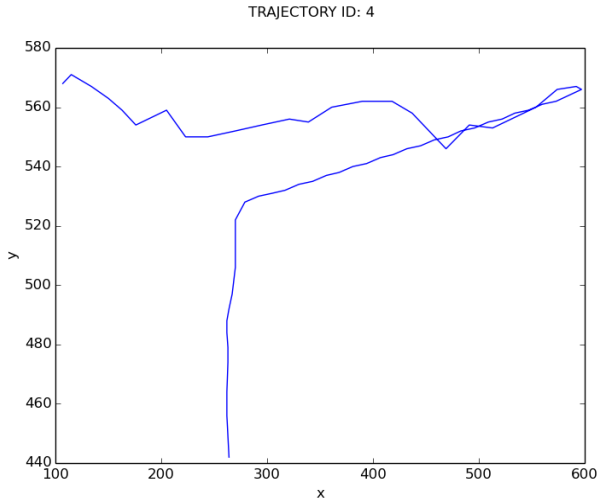
/* I. PARTITIONING PHASE */
01: **for each** $TR_i \in \mathcal{I}$ **do**
02: Partition TR_i at a base unit;
/* II. DETECTION PHASE */
/* $\mathcal{L}$ denotes the set of t-partitions */
03: **for each** $L_i \in \mathcal{L}$ **do**
/* Definition 1 */
04: Count $|CTR(L_i, D)|$ by computing $dist(L_i, L_j)$,
/* $TR(L_i)$ means the trajectory enclosing L_i */
where $L_j \in \mathcal{L}$ and $TR(L_i) \neq TR(L_j)$;
/* Definition 2 */
05: **if** $\lceil |CTR(L_i, D)| \cdot adj(L_i) \rceil \leq \lceil (1-p)|\mathcal{I}| \rceil$ **then**
06: Mark L_i as outlying;
07: **for each** $TR_i \in \mathcal{I}$ **do**
/* Definition 3 */
08: **if** $Ofrac(TR_i) \geq F$ **then**
09: Output TR_i with its outlying t-partitions;

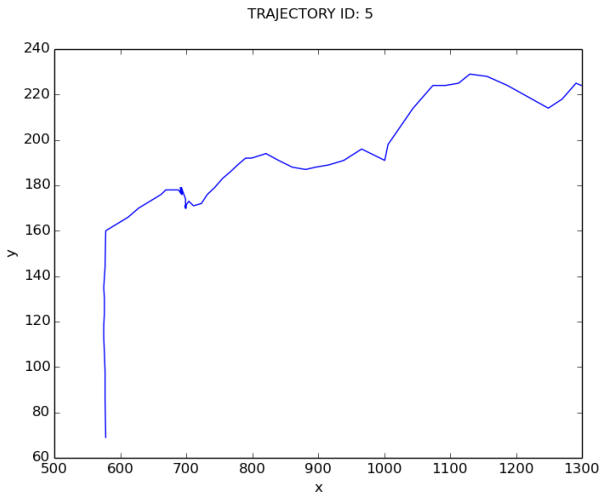
RESULTS

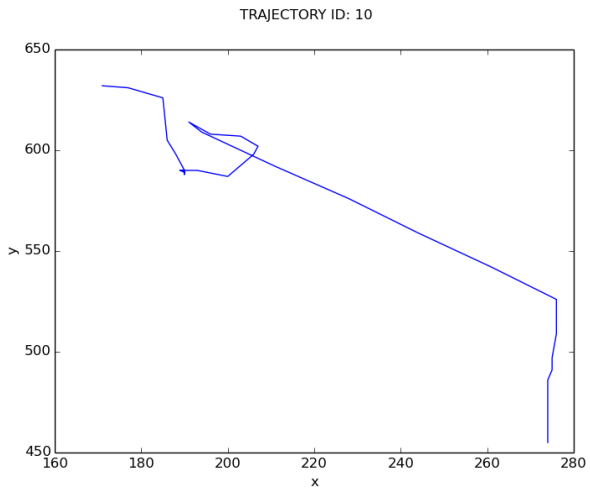


These are plots of 10 trajectories, out of which 4 have been detected as outliers (the ones in red). The parameters used for this experiment were **D = 37, p = 0.99, F = 0.4**. As we can see, the trajectories that have been marked as outliers have different **shape** as well as are more **chaotic**. A trajectory with just a different shape but not chaotic (*e.g.* T7) is not marked as an outlier (which is what we expect). Below are the individual plots of the 4 outlying trajectories.









The parameters D, p and F can be adjusted in order to get more specific outliers as per the need.The trajectories above are the trajectories of first 10 pedestrians in dataset 'al_position2013-02-06.csv'. (Here is a sample image).

1	2013-02-06T07:00:05:196;PIW;78230;20890;1
2	2013-02-06T07:00:05:296;PIW;78272;20764;1
3	2013-02-06T07:00:05:396;PIW;78293;20763;1
4	2013-02-06T07:00:05:496;PIW;78316;20728;1
5	2013-02-06T07:00:05:596;PIW;78337;20729;1
6	2013-02-06T07:00:05:696;PIW;78338;20633;1
7	2013-02-06T07:00:05:796;PIW;78356;20513;1
8	2013-02-06T07:00:05:896;PIW;78363;20439;1

CONCLUSION

We have successfully implemented an outlier detection system for trajectories which can be used for any kind of trajectories (trajectories obtained from pedestrians or vehicles or animals or hurricanes etc. the list is endless). Also, our system is capable of detecting outliers in any number of trajectories (limited only to resources available on the machine).

FUTURE WORK

Our outlier detector which is based on a completely algorithmic approach can be integrated with a **supervised** machine learning system so that it can learn the parameters D, p, and F from a **labelled** data-set. Then the resulting system can be used as a component of an anomaly detector or any other more abstract crowd behavior analysis tool.

DATA & SOURCE CODE

The data that we have used [2] can be obtained by sending a request to:
mailto:alex.email@gmail.com
The source code for the project is available on github.com at:
https://github.com/Pranshu258/cba

REFERENCES

[1] Trajectory Outlier Detection: A Partition & Detect Framework, ***Jae-Gil Lee, Jiawei Han, Xiaolei Li.***
[2] Socially-aware Large-scale Crowd Forecasting, ***Alexandre Alahi, Vignesh Ramanathan, Li Fei-Fei.***
[3] Trajectory Clustering: A Partition-and-Group Framework, ***Jae-Gil Lee, Jiawei Han, K.Y. Whang***