CLEI electronic journal, Volume 25, Number 2, Paper 3, May 2022

Feature-based Trajectory Anomaly Detection

Gerar F. Quispe-Torres, Lauro Enciso-Rodas, Harley Vera-Olivera
Universidad Nacional de San Antonio Abad del Cusco, Department of Informatics
Cusco, Perú, 08000
{ *gerar.quispe, lauro.enciso, harley.vera*} @*unsaac.edu.pe*

and

Germain Garcia-Zanabria
Universidad Católica San Pablo, Department of Computer Science
Arequipa, Perú, 04001
germain.garcia@ucsp.edu.pe

Abstract

The high availability of trajectory data in different fields makes it attractive to analyze and enhance its multiple practical applications. In particular, trajectory anomaly detection has a significant practical value, making it possible identifying trajectories that may indicate illegal and adverse activity in diverse areas such as surveillance, tracking devices, traffic, and people flow. This study presents a methodology to detect anomaly trajectories based on their morphology features. For that, we follow two stages: (1) comparative analysis of the performance of two descriptors to group similar trajectories, and (2) trajectory anomaly detection based on their similarities. We define Wavelet and Fourier transforms as trajectory descriptors to generate characteristics and subsequently detect anomalies ones. Our experiments emphasize the measure of the performance in the description of the coefficient feature space using unsupervised learning, specifically clustering techniques, to create subsets and identify irregular ones. The study's implications demonstrate that it is possible to use descriptors in trajectories for automatic anomaly detection and the use of unsupervised learning to segment required information. Our study's performance and comparative analysis have been demonstrated throughout multiple experiments. We present some quantitative results using synthetic data sets as well as qualitative analysis throughout two case studies considering real data sets that leave evidence of our contribution.

Keywords: Trajectory anomaly detection, trajectory descriptor, feature extraction, trajectory clustering.

1 Introduction

Understanding trajectory dynamics is a challenging problem due to the comprehensive data and Spatio-temporal information. Trajectories analysis play an important role in different areas such as animal tracking [1], air-streams [2], weather prediction [3], traffic flow [4], activity flow [5], sports [6], flight planning [7], and many others. In this field, anomaly behavior may indicate important objects and events in a wide variety of domains [8]. However, this analysis is not a trivial problem due to the sequential analysis, complexity morphology, and parameter calibration of algorithms.

Anomaly trajectory detection is an important problem because it allows identifying trajectories that may indicate illegal and adverse activity. For instance, in video surveillance, it could indicate personal assault, robbery, and infrastructural sabotage. However, it is not a trivial task; the algorithms have to face different problems: the process of cleaning the noise of the trajectories and the extraction of semantic information that involves experimentation and studies transforming raw movements to other kinds of representations [9]. Moreover, the lack of exact metrics to measure the quality of a semantic extractor makes the study of trajectories difficult.

On the other hand, the use of unsupervised learning for anomaly detection is justified and used in the literature [10]. The supervised approaches to anomaly detection are less practical in some contexts, such as

video surveillance applications and automatic motion learning, since labeled training data are not usually available or practical to obtain. That is why techniques are required to learn anomaly activity patterns in an unsupervised manner. Training data will often contain anomalies or outliers that are unusual or infrequently occurring. The learning algorithm must adapt to anomalies and be robust in the presence of noise and occlusion.

This work compares two descriptors for anomaly trajectory detection taking morphology as the main feature. Moreover, the proposed methodologies present experiments to verify that descriptors improve the trajectory analysis process. These experiments will be validated throughout performance comparisons considering some literature data sets. Specifically, we aim to analyze automatic trajectory anomaly detection performance based on unsupervised learning, taking Discrete Fourier Transform and Multilevel Discrete Wavelet Decomposition as principal descriptors. Moreover, we applied our methodology over a real video surveillance data set to identify rare videos based on anomaly trajectories. In summary, our contributions are:

- A methodology to identify anomaly trajectory detection based on Fourier and Wavelet transforms as descriptors.
- Verify the unsupervised learning method as the affinity propagation in the trajectory anomaly detection based on similarity analysis.
- A set of comparative analysis revealing interesting patterns about trajectories considering different data sets and descriptors.
- Two Case studies based on real dataset that demonstrate the usefulness of our methodology for anomaly trajectory detection on video surveillance and Traffic flow dataset.

This paper is an extension of work originally presented in XLVII Latin American Computing Conference (CLEI) entitled, "Trajectory Anomaly Detection based on Similarity Analysis" [11]. In this version, all comments from the conference review have been addressed. In particular, we have improved related works and given more mathematical and computational details about our methodology. We have also improved the presentation of our experiments. In addition, one of the main contributions is an additional case study analyzing traffic route anomaly detection in an realistic scenario. Finally, we have also detailed the software and hardware architecture we used for our entire analytical process.

The rest of the paper is organized as follows: Section 2 introduces the related works organized into three categories. Section 3 presents some relevant concepts to contextualize our approach. Section 4 describes the pipeline of our methodology, including the mathematical and computational mechanisms employed in our analysis. Section 5 describes results obtained with synthetic datasets, mentioning the experiments setup, the evaluation, and the discussion. Besides quantitative experiments, in Section 6, we presented a qualitative analysis throughout case studies. Section 7 presents some details about the software and hardware tools used for our study. Finally, in Section 8, we discuss the limitations and conclusions of this work.

2 Related Works

The literature about trajectory analysis is extensive. To better contextualize our approach, we divide this section into three parts (considering the datasets used for each study): introduction to trajectories, trajectory analysis, and trajectory anomaly detection.

2.1 Trajectories

Kong et al. [12] classified trajectory data as explicit and implicit based on their continuity and structure. Explicit trajectory data provide time and location information, besides a well-structured Spatio-temporal solid continuity. Trajectories generated by GPS data are the most representative ones in this category. On the other hand, implicit trajectory data has weak Spatio-temporal continuity, sub-categorizing this class into signal-based, sensor-based, and network-based data. This study summarizes applications and services that use trajectories, presents an application-based trajectory classification, and also mentions some recommendation system services that use trajectories in their studies. According to this study, our data are within the subcategory of sensor-based data since one of our case studies is related to the monitoring of people.

Depending on the entity that originates the trajectories, they will be subjected to a finite set of classes or trajectory types, from regular movements to highly erratic movements.

2.2 Trajectory analysis

Trajectory modeling is the first and most challenging step in treating trajectories. In the literature, there are different ways to process trajectories; for instance, the algorithm called TRACLUS [13] processes the trajectories using segments, creating a summary of them. On the other hand, some studies restrict trajectories to a road network; it refers to the movement of vehicles following a transport network. In this category, NETSCAN [13], NNCluster [14], and NEAT [15] show analyses restricting the trajectories to a road network. In addition, to being restricted to a road network, NETSCAN, and TRACLUS process trajectories from the segmentation approach without considering characteristics or patterns repeated in different trajectory parts (high-level features). Instead, Naftel et al.[16], Khalid et al.[10] and Annoni et al.[17] modeled trajectories trying to capture information presented in the whole route (individual identity).

2.3 Trajectory anomaly detection

In this part, we presented some works that identify abnormal behavior of the trajectories. For instance, Zhang et al. [18] used Distance-Based Methods to detect the anomaly in feature space. In this work the, trajectories with a long distance from most of them are regarded as abnormal, using clustering to create groups of similar trajectories. We adopt this method to detect anomaly trajectories.

Piciarelli et al. [19] created an algorithm to generate synthetic trajectory data sets. This algorithm generates a thousand subsets automatically generated with sixteen points each. These data sets are used in more academic papers since this study is one of the first works that address detecting anomalous trajectories and sharing their data sets. However, their method does not consider trajectories with different numbers of points. Years later, Laxhammar et al. [8] presented new results considering the algorithms and the datasets generated by Piciarelli. This study emphasized the sequential analysis of incomplete trajectories, which the author termed real-time learning based on an incremental update of the training set. This study was one of the first studies which mention the mean detection delay introducing the run time efficiency in trajectory anomaly detection.

Ergezer et al. [20] presented a trajectory descriptor with a covariance matrix to detect anomalous trajectories using Nearest Neighbors (NN) and Space Representation (SR), besides the use of spectral grouping for the perception of activity. This study also uses the synthetic data set of Piciarelli as part of their results. Also, it performs experiments with real data from the University of California San Diego (UCSD) and the MIT Parking Lot [21] to detect anomalies in videos. In the same vein, Sillito et al. [22] proposed a new framework to detect abnormal trajectories considering the behavior of passersby in terms of trajectory movement. This framework builds a One-Class classifier based on probabilities using the Gaussian distribution. Moreover, they conduct experiments using labeled and unlabeled data sets using two databases such as CAVIAR INRIA [23] and Capark [24]. However, this framework does not consider the morphology of the trajectories. Ma et al. [25] construct a trajectory distance metrics using an autoencoder, and then use the Nearest Neighbors technique to detect various kinds of anomalies. However, despite presenting good results, he had to normalize his trajectories to a fixed number of points, in addition to needing the necessary hardware to be able to experiment with autoencoders.

This section presents some studies in trajectory anomaly detection, which use various methods to extract features from trajectories. However, the use of Wavelet and Fourier transforms as a descriptor focusing on trajectory shape or morphology does not present profound studies.

3 Background

This section presents essential concepts related to our approach to familiarize the reader with our research topic.

3.1 Point, trajectory and sub-trajectory

The following representation of trajectories was inspired by the study of Panagiotakis et al. [26]. For our study a point p is a tuple (x, y, t), where x and y are the position (latitude and longitude respectively) and t is the time-lapse when the position is collected:

$$p_k = (x_k, y_k, t_k), k \in \mathbb{N} \tag{1}$$

and a list of points ordered in time forms a trajectory T_i:

$$T_i = (tid_i, \{p_k\}_{k=1:K}) \tag{2}$$

where tid_i is the identifier with $t_1 < t_2 < t_3 < \ldots < t_K$ in a sequence of points $\{p_1, p_2, \ldots, p_K\}$ and $\{i, K\} \in \mathbb{N}$.
On the other hand, for our study, a sub-trajectory is defined as:

$$T'_s = \{p_k\}_{k=1:K} \tag{3}$$

where T'_s is a set of points $p_k = (c_k, t_k)$, t_k is the time instant in which the component c_k is collected and $c_k \in w_x \vee c_k \in w_y$ defined on Equations 10 and 11 respectively.

3.2 The Discrete Fourier Transform (DFT)

The Fourier transform is a mathematical function that decomposes a waveform, which varies through time, into the frequencies and amplitudes that signal up. The Fourier transform output has real and imaginary parts for positive and negative frequencies. The absolute value of their outputs represents the original function frequencies. Thus, the Fourier transforms allow viewing any function as a sum of simple sinusoids.

The DFT is a discrete transformation used in Fourier analysis [27, 28]. The DFT can be defined as the sampling of a function at a certain frequency, and it requires a finite discrete sequence as input. For instance, these sequences can be generated from the sample of a single section of a signal.

Let $f(t)$ be the signal which is the source of the data and let be N instants separated by sample times denoted by $f[0], f[1], f[2], \ldots, f[N-1]$. The DFT of $f(t)$ can be defined as:

$$F[k] = \sum_{n=0}^{N-1} f[n] W^{kn}, \tag{4}$$

for each $k = 0, 1, \ldots, N-1$. Where $W = e^{-j(2\pi/N)}$ and $j = \sqrt{-1}$ which is an imaginary number. $F[k]$ are the coefficients to each basis function in the linear summation.

3.3 The Multilevel Discrete Wavelet Decomposition (MDWD)

The Wavelet transform is a mathematical function useful in digital signal processing and image compression. In signal processing, Wavelets make it possible to recover weak signals from noisy ones, which is helpful, especially in processing X-ray and magnetic-resonance images in medical applications. The Wavelet and Fourier transform represents a signal through a linear combination of their basic functions. Both decompose signals as a superposition of simple units from which the original signals could be reconstructed. The Wavelet Transform decomposes signals into wavelets, and their base functions are compact or finite in time. This feature allows the Wavelet Transform to obtain time information about a signal in addition to frequency information. The Wavelet transform has a window size that varies frequency scale. This technique is advantageous for analyzing signals containing both discontinuities and soft components. Short high-frequency base functions are needed for discontinuities, while at the same time, long low-frequency ones are needed for the soft components. The Wavelets are a class of functions used to localize a given function in space and scaling.

To analyze non-stationary signals, we need to decompose them into localized units in both time and frequency domains; for this purpose, we use MDWD. According to Wang et al. [29], the MDWD is a wavelet-based discrete signal method that can extract multilevel time-frequency features from a signal by decomposing it as low and high-frequency sub-signals level by level.

For the next explanation we use bold symbols such as $\mathbf{x}$, $\mathbf{a}$ or $\mathcal{X}$ to denote vectors and not-bold a, x or l to scalars. We denote the input for N samples for a signal as $\mathbf{x} = \{x_0, x_1, \ldots, x_{N-1}\}$, and the low and high sub-signals generated in the i-th level as $x^l(i)$ and $x^h(i)$. In the $(i+1)$-th level, MDWD uses a low pass filter $\mathbf{l} = \{l_1, \ldots, l_k, \ldots, l_K\}$ and a high pass filter $\mathbf{h} = \{h_1, \ldots, h_k, \ldots, h_K\}$, $K \ll N$, to convolute low frequency sub-signals of the upper level as:

$$a^l_n(i+1) = \sum_{k=1}^{K} x^l_{n+k-1}(i) \cdot l_k, \tag{5}$$

$$a^h_n(i+1) = \sum_{k=1}^{K} x^l_{n+k-1}(i) \cdot h_k, \tag{6}$$

where $x^l_n(i)$ is the n-th element of the low frequency signal in the i-th level, and $\mathbf{x}^l(0)$ is set as the input signal. The low and high frequency sub-signal $\mathbf{x}^l(i)$ and $\mathbf{x}^h(i)$ in the level i are generated from the 1/2 down-sampling of the intermediate variable signals defined as (7) and (8).

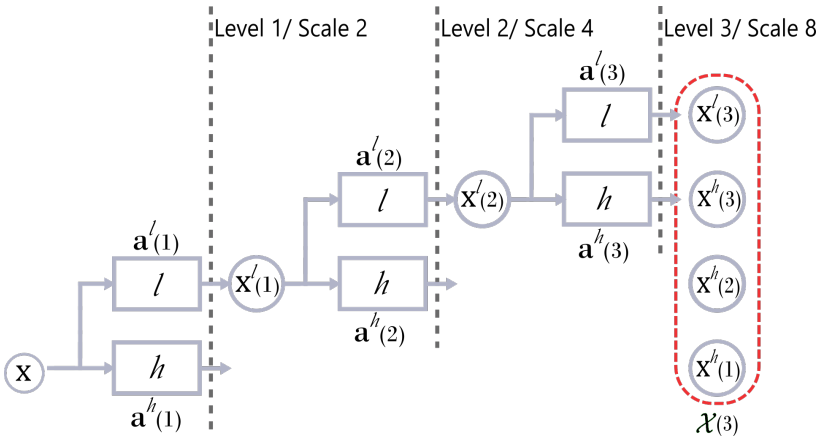

Figure 1: Representative illustration of MDWD to $\mathbf{x}$ with three levels obtaining as results $\mathcal{X}(3)$. Figure based on Wang et al. [29].

$$\mathbf{a}^l(i) = \left\{a_1^l(i), a_2^l(i), \dots\right\} \tag{7}$$

$$\mathbf{a}^h(i) = \left\{a_1^h(i), a_2^h(i), \dots\right\} \tag{8}$$

The sub-signals set:

$$\mathcal{X}(i) = \left\{\mathbf{x}^h(1), \mathbf{x}^h(2), \dots, \mathbf{x}^h(i), \mathbf{x}^l(i)\right\} \tag{9}$$

It is named the i-th level decomposed of $\mathbf{x}$, and it has different time and frequency resolutions. The sub-signals with different frequencies in $\mathcal{X}$ are defined as the MDWD, and it maintains the same order information with the original signal $\mathbf{x}$, and the frequency from $\mathbf{x}^h(1)$ to $\mathbf{x}^l(i)$ is from high to low.

Fig. 1 shows a recreation of MDWD of $\mathbf{x}$ with three levels; each of the pointed lines repairs each level, the first convolution with the initial signal is considered as level 0. Each of the rectangles represent the low or high pass filter giving as a result $\mathbf{a}^l(i)$ and $\mathbf{a}^h(i)$, while $\mathbf{x}^l(i)$ represent the input signal to the next level i and $\mathbf{x}^h(i)$ is added to the solution. The components rounded by lines dotted in red make up the decomposition of the original signal in several levels, which will be used as features for this study. Finally, as a result, we obtain $\mathcal{X}(3)$, which in this case it is composed of four sub-signals.

3.4 Affinity Propagation (AP)

Affinity Propagation clustering [30] is a fast clustering algorithm used especially in the cases of large numbers of clusters. AP can be interesting as it chooses the number of clusters based on the data provided. It works based on similarities between pairs of data points (or $n \times n$ similarity matrix S for n data points) and simultaneously considers all the data points as potential cluster centers (called exemplars).

The AP clustering algorithm has two important concepts: the responsibility $R(i,k)$ and availability $A(i,k)$, which represent two messages indicating how well-suited a data point is to be a potential exemplar. $R(i,k)$ is an accumulated value that reflects how well point i is suited to be the candidate exemplar of data point i and then sends data from the latter to the former, compared to other potential exemplars, point k is the best exemplar. The availability $A(i,k)$ is opposed to $R(i,k)$ and reflects how well-suited it is for point i to choose point k as its exemplars. Based on the candidate exemplar point k, the accumulated message sent to the data point i indicates that point k is more qualified as an exemplar than others.

The sum of $R(i,k)$ and $A(i,k)$ is the evaluation basis for whether the corresponding data point can be a candidate exemplar or not. Once a data point is chosen as a candidate, the rest nearest data points will be assigned to the cluster. The similar value between two data points x_i and x_j $(i \neq j)$ is usually assigned the negative Euclidean distance, such as $S(i,j) = -||x_i - x_j||^2$.

The algorithm uses an initial value called preference, which indicates the preference that the data point can be chosen as an exemplar. It is usually set by the median(s) of all distances.

The algorithm iterates until the cluster boundaries remain unchanged over many iterations or after some predetermined number of iterations. The exemplars are extracted from the final matrices as those whose "responsibility + availability" for themselves is positive.

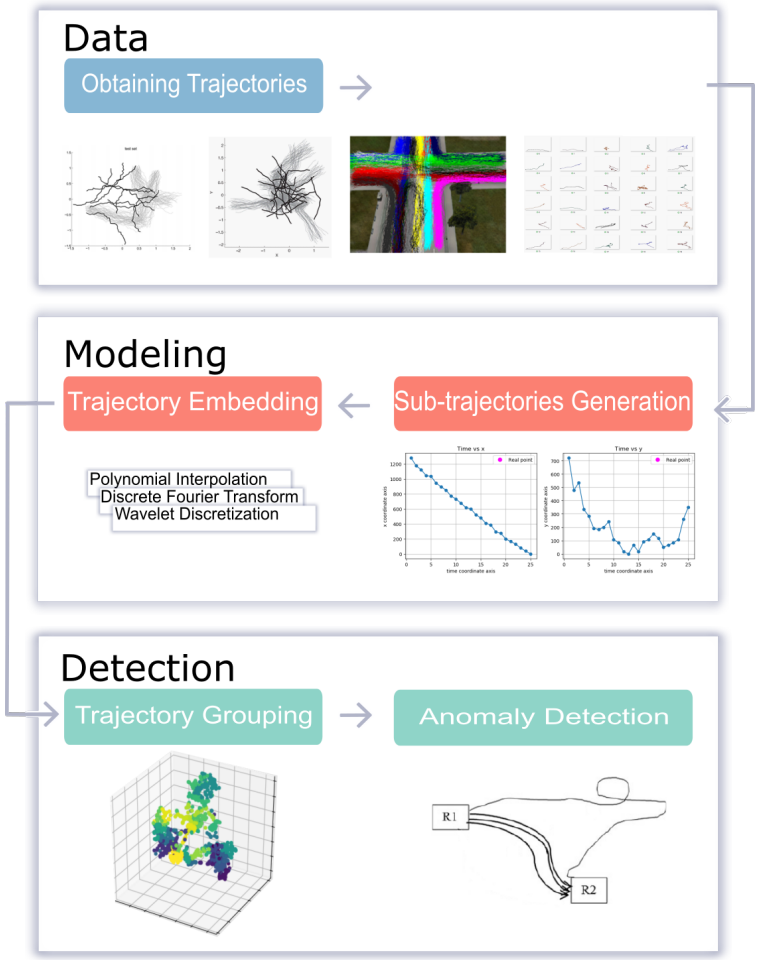

Figure 2: Pipeline overview of the proposed methodology based on three modules: Data, Modeling, and anomaly detection

4 Anomaly Trajectory Detection

This section describes details about the procedures involved in our pipeline. The most important task of our proposed methodology is the description of trajectories; we focus on describing trajectories based on their morphology for similar grouping ones. Fig. 2 illustrates each process applied in our approach. We divide our methodology into three modules: Pre-processing, Modeling, and Detection.

The pre-processing process starts from obtaining trajectories, where sometimes it is necessary to use trajectory transformation algorithms or data cleaning methods. In the real world, trajectories are noisy, and the data are not standardized. The trajectory modeling consists of finding an adequate representation, this representation highlight characteristic that helps to discriminate or classify our data for a specific purpose. Finally, in the third module, anomaly detection, properly this step consists of detecting an isolated point in a hyper-plane since to detect anomalies, in our study, we will use the distance-based methods approach. The trajectories with a long distance from most of them are abnormal and cluster to create similar groups.

4.1 Trajectory data

The module Data is about obtaining and pre-processing the trajectories used in our analyses. For our approach, a trajectory represents an object in motion, having extremes at the beginning and the end. In this work, we used five datasets:

- The Synthetic dataset created by Piciarelli et al. [19] contains about 260 thousand trajectories generated by an algorithm. These trajectories were divided into $1,000$ groups, and each group contains 260 instances with coordinates (x, y). In each group, 250 belonged to a cluster, and the rest ten were labeled as anomalous. These trajectories are 16 points long, with no time information.

- The dataset created by Laxhammar et al. [8] using Piciarelli's [19] algorithm. This new dataset contains

about 200 thousand trajectories, with 100 groups from 10 different clusters; each group contains about 2,000 trajectories.

- The CROSS [5] dataset contains 9,700 trajectories that simulate four-way traffic intersections with various through and turn patterns (including a u-turn). The dataset consists of 9,500 activity paths belonging to 19 clusters and 200 anomaly paths. These paths have different lengths with no time information.
- A dataset with 1,000 trajectories with coordinates (x, y). It contains 970 normal and 30 anomaly paths. These paths have different lengths with no time information. This dataset was obtained from videos that belong to a Laboratory. The videos are used to analyze anomalous events in simple situations [31]; the content is real without forcing any abnormal situation.
- A dataset with 15,013 trajectories with coordinates (latitude, longitude). These routes have different lengths; each point was collected from GPS devices, and it is possible to know the time in which the information was gathered. The data are used to monitor public transport buses, and the content is real without forcing any abnormal situation. The dataset also was used to detect and define traffic jam states and areas.

The first three datasets are already normalized and cleaned; therefore, no pre-processing task was made over them. It is important to mention that our approach supports trajectories of different sizes; it is corroborated by the experimentation of the CROSS, Laboratory, and Traffic flow datasets that contain trajectories with different lengths.

4.2 Trajectory modeling

In the second process of our pipeline, we describe the modeling of trajectories in detail. From once the data sets have been obtained to the generation of feature vectors.

4.2.1 ***Trajectory normalization***

In order to normalize each of our trajectories, we used *feature scaling* method.

Let the following spaces be:

$$w_x = \{x_i \in p_i \mid \forall p_i \in T_j\}, \tag{10}$$

$$w_y = \{y_i \in p_i \mid \forall p_i \in T_j\}, \tag{11}$$

where p_i is a point and T_j a trajectory. For all variables x_i and y_i of T_j, the *feature scaling* formulas are applied:

$$x'_i = \frac{x_i - min(w_x)}{max(w_x) - min(w_x)}, \tag{12}$$

$$y'_i = \frac{y_i - min(w_y)}{max(w_y) - min(w_y)}, \tag{13}$$

Where min and max return the minimum and maximum space values, respectively. After computing each component w_x and w_y with equations 12 and 13, each element has a new assigned value. We can assign the value zero for the minimum and one for the maximum, and the rest of the intermediate values are scales between those thresholds. For instance, for visualization purposes, we multiplied as the maximum values the dimensions of 720 and 1280 for each component respectively (x and y). Fig. 3 shows the result of this process; original trajectory (Fig. 3.a) and normalized trajectory (Fig. 3.b).

4.2.2 ***Trajectory decomposition***

In this study, trajectories' representation is generated by splitting them into 1-D sub-trajectories for x and y spaces, represented as $X = x_i, Y = y_i, i = 1, \ldots, n$ (n is the number of points of a trajectory). X and Y represent the horizontal and vertical movements. Fig. 4 shows an example of two sub-trajectories generated by our modeling. Another interpretation that fits into this process is that these sub-trajectories can behave like some time series representing the variation of each spatial component. Thus, the signals give a parameterized behavior compared to trajectory data, and it aids the description in shape.

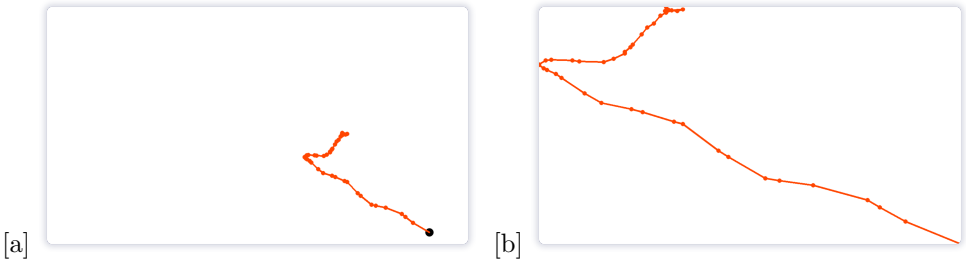

Figure 3: The normalization of trajectories improves our features. (a) The initial trajectory corresponds to the path generated in a video segment. (b) Video dimensions have been used to normalize each component of the trajectory in each component.

4.2.3 ***Feature space representation***

Once the *trajectory decomposition* is finished, we can apply feature extraction methods to describe each sub-trajectory. The descriptor inputs are the two sub-trajectories. We consider two techniques to achieve sub-trajectories description, including discrete Fourier transform and MDWD. The derivation of our feature space representation of sub-trajectories using the three proposed methods is specified as follows:

Discrete Fourier transform The Feature space representation of a trajectory using DFT is similar. The N-points DFT of X (see Section 3), defined as a sequence X_f of N complex numbers ($f = 0, ..., N-1$), is given by:

$$X_f = DFT(X) \tag{14}$$

$$Y_f = DFT(Y) \tag{15}$$

X_f and Y_f are complex numbers with the exception of X_0, Y_0 which are real. As a rule, the DFT sequence is truncated after m terms for X_f and k for Y_f. Formally, let be a_i and $\hat{a}_i$ be the real and imaginary part of X_f, and b_i and $\hat{b}_i$ be the real and imaginary part of Y_f. Since we define working with real instead imaginary numbers, we convert X_f and Y_f into real numbers using equations 16 and 17, respectively.

$$r_i = \sqrt{{a_i}^2 + \hat{a}_i^2}, i = 0, ..., m-1 \tag{16}$$

$$\bar{r}_j = \sqrt{{b_i}^2 + \hat{b}_i^2}, j = 0, ..., k-1 \tag{17}$$

with r_i and $\bar{r}_j$ numbers, we note that some of them appear twice, choosing only one as in equations 18 and 19.

$$R_x = \{r_0, ..., r_i, ..., r_{m-1}\}_{\neq} \tag{18}$$

$$R_y = \{\bar{r}_0, ..., \bar{r}_j, ..., \bar{r}_{k-1}\}_{\neq} \tag{19}$$

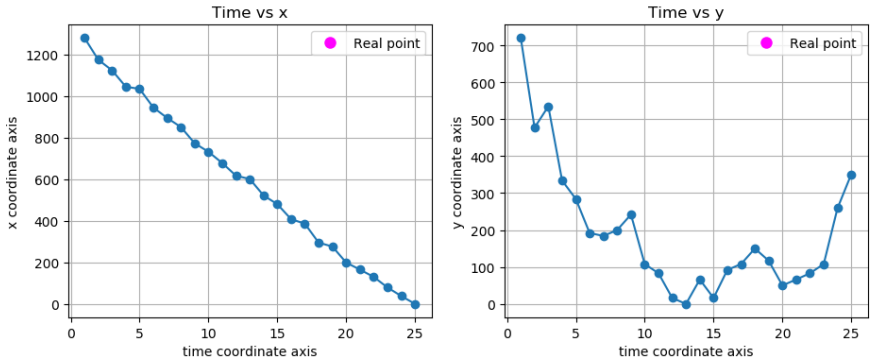

Figure 4: Trajectory decomposition. Our modeling breaks the trajectories down into two 1-D sets of points

where R_x and R_y are set formed by unique elements. We perform discretization or binning with those two sets of variables, transforming the variable length of a set into a constant defined length set. It was using histograms b_q and b'_q. They follow the conditions:

$$|R_x| = \sum_{q=1}^{l} b_q \tag{20}$$

$$|R_y| = \sum_{q=1}^{l} b'_q \tag{21}$$

where b_q and b'_q are functions to count the numbers of observations that fall into each of the disjoint categories (bins). l is the number of bins, $|R_x|$ and $|R_y|$ are the numbers of observation of R_x and R_y respectively. Finally, the trajectory can be represented in the feature space by $\mathbf{F}_{DFT}$ defined as:

$$\mathbf{F}_{DFT} = [\sum_{q=1}^{l} b_q, \sum_{q=1}^{l} b'_q] \tag{22}$$

Multilevel Discrete Wavelet Decomposition For our description process with MDWD, the *Haar* family is used. It presents different levels of frequencies depending on the number of different forms present in the trajectory. Applying MDWD in X and Y, we can obtain:

$$[cA_m, cD_m, cD_{m-1}, \ldots, cD_2, cD_1] = MDWD(X) \tag{23}$$

$$[cA_k, cD_k, cD_{k-1}, \ldots, cD_2, cD_1] = MDWD(Y) \tag{24}$$

The output is a list of coefficients, where m and k denote the maximum useful level of decomposition. Thus, the first element cA_m of the result is the approximation coefficients array, and the following elements $cD_m, \ldots, cD_1$ are detailed coefficients arrays.

We define the feature vector F_x as the concatenation of different levels of coefficients obtained with MDWD for X given by equation 25. A similar expression can be defined for Y as equation 26.

$$\mathbf{F}_x = [cA_m, cD_m, cD_{m-1}, \ldots, cD_2, cD_1] \tag{25}$$

$$\mathbf{F}_y = [cA_k, cD_k, cD_{k-1}, \ldots, cD_2, cD_1] \tag{26}$$

with F_x and F_y, we perform discretization using histograms b_q and b'_q, they follow the conditions:

$$|F_x| = \sum_{q=1}^{l} b_q \tag{27}$$

$$|F_y| = \sum_{q=1}^{l} b'_q \tag{28}$$

Finally, trajectory can be represented in the feature space by $\mathbf{F}$ defined as:

$$\mathbf{F} = [m, k, \sum_{q=1}^{l} b_q, \sum_{q=1}^{l} b'_q] \tag{29}$$

4.3 Anomaly detection

Once the feature vectors were obtained, we performed anomaly detection based on clustering. For that, we used the distance-based methods [18]; the trajectories with a long distance from most trajectories are regarded as anomaly. We segment and separate trajectory information in the clustering process to detect anomalies (located at extremes far from majority groups). As a clustering method, we use *affinity propagation* (AP); this method suits our experiments. Moreover, the AP allows the separation of different trajectories since this clustering method generates more groups than other unsupervised methods. Finally, to *recuperate anomaly trajectories* from clusters, we defined a threshold. It is defined as the maximum number of elements of an abnormal cluster.

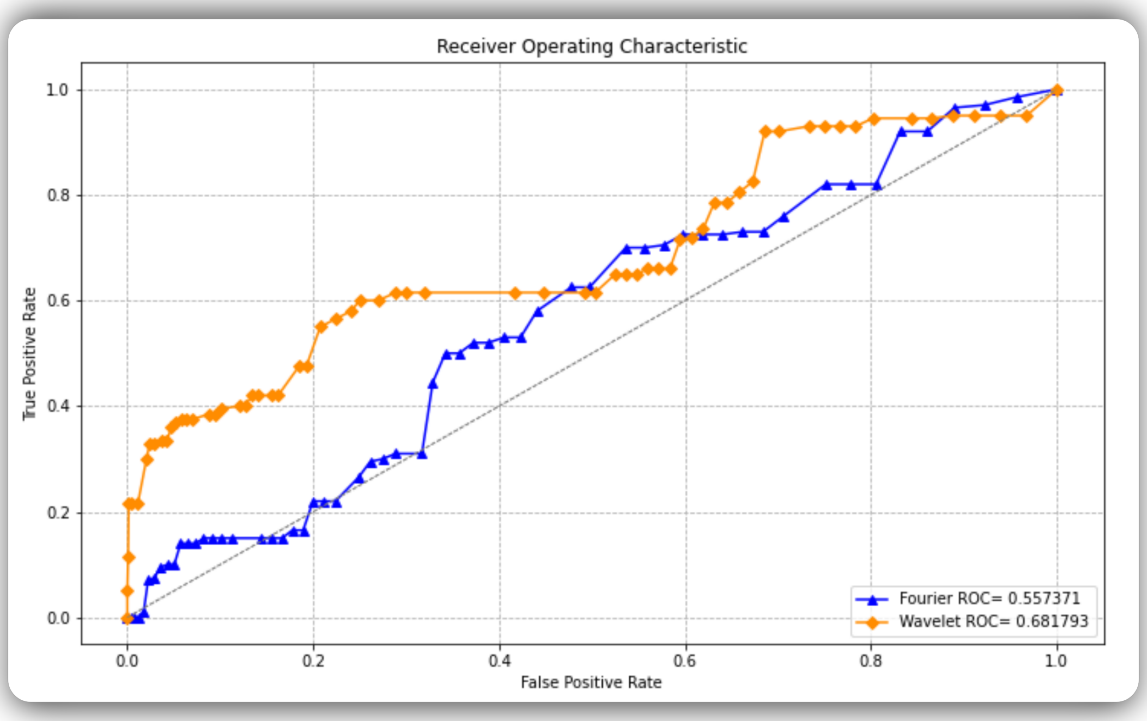

Figure 5: Comparative evaluation performance of DFT and MDWD descriptors using CROSS dataset

5 Results and discussion

In this section, we describe the results of our experiments. Three synthetic datasets perform the quantitative results.

Experimental Setup The hyper-parameters of our experiments are described as follows: For AP, the default parameters are set as the median of input similarities and the damping factor to 0.5, 0.625, and 0.7. In some cases, the maximum number of iterations should be equal to one thousand. In the case of histograms, the number of bins ten, and for obtaining the *average accuracy*, we choose the best *threshold* in each trajectory subset experimentation.

Evaluation In order to evaluate the relative performance of our proposed representation in exhaustive datasets, we perform experiments using synthetic datasets generated in [19] and [8]. Due to the nature of these datasets, we use average accuracy to evaluate performance. Instead, with the CROSS dataset, we use accuracy since it is composed of one set of trajectories. The results are presented in Table 1. We decided to evaluate the performance in this dataset with Receiver Operating Characteristic (ROC) curve, due to the different anomaly threshold and related works [31, 32]. Each point of the ROC curve is obtained with a different anomaly-threshold value, denoting values for True Positive Rate (TPR) and False Positive Rate (FPR) in each detection. This information provides a visual perception of the best threshold using FPR and TPR. We can see our result in Fig. 5. According to it, MDWD descriptors achieve the best results than DFT in this dataset.

Table 1: Quantitative results on three synthetic datasets

	Datasets		
Method	Piciarelli	Laxhammar	CROSS
MDWD	0.9519	0.9780	**0.8884**
DFT	**0.9525**	**0.9848**	0.8825

5.1 Discussion

First, we describe the results achieved with synthetic datasets, and then we explain the results obtained with the CROSS dataset.

For the first two datasets, it is clear that DFT has the highest detection performance. On the third dataset, DFT results were slightly worse than MDWD. Although the accuracy obtained for the CROSS dataset decreased, it presents a great difficulty in processing since the trajectories have variable lengths.

From Table 1, notice that the best score obtained is with the Laxhammar dataset, and we consider that it is competitive with related works [19] and [20]. On the other hand, in the CROSS dataset, we compare our results with experiments performed by Morris et al. [5], which use the technique of [16], which identified 84% abnormalities with a 10% of the false-positive rate. Using TPR, we obtain 64% with a 24% false-positive rate. Obtaining promising results, since our representations take information related to morphology and the CROSS dataset collects shape information in their anomaly definition, we consider that this dataset collects similar information to the proposed objectives.

6 Case Studies

This section presents two case studies that show the effectiveness of our proposed approach in addressing the task of anomaly trajectory detection. The first case study addresses detecting anomalous trajectories generated by people in surveillance videos, while the second is focused on monitoring the public transport bus routes of a city in a Latin American city. These two case studies use our algorithms' same configuration (hyper-parameters). We analyze two types of cases in two different datasets.

6.1 People Flow Anomaly Detection

To assess the performance of our approach, we conducted a case study to identify rare videos based on anomaly detection of people's trajectories. For that, we used SSIG-dataset filmed in a smart sense laboratory door. SSIG-dataset was filmed on a real stage without forcing any abnormal situation, recordings the entrance of the Laboratory.

6.1.1 Dataset

The Laboratory dataset contains people in different situations: pointing in and out of the laboratory, closing and opening the door, stopping and walking outside the laboratory. The criteria for defining "normal" videos are: entering and leaving the laboratory by opening and closing the door; a short amount of time spent in front of the camera (less than 10 seconds); people going through the corridor outside the laboratory; and people leaving and entering the side laboratory. On the other hand, the criteria to define rare behavior are: making several movements to come and go to the laboratory, standing in front of the camera for a long period, and using the key-box located near the door for an extended period.

The dataset consists of $5,025$ videos that last from two seconds to four minutes and twenty-three seconds recorded during two years. For each video, based on *post-estimation*, we selected person's fiducial point, and with the *tracking* method, we connected the *fiducial point* of each frame generating a trajectory. Aiming to explore a considerable number of samples, we randomly selected $1,000$ trajectories. The idea behind this

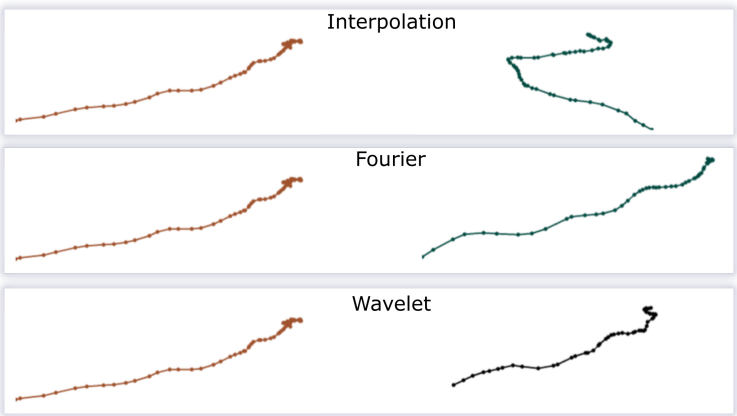

Figure 6: The most similar trajectory for each descriptor

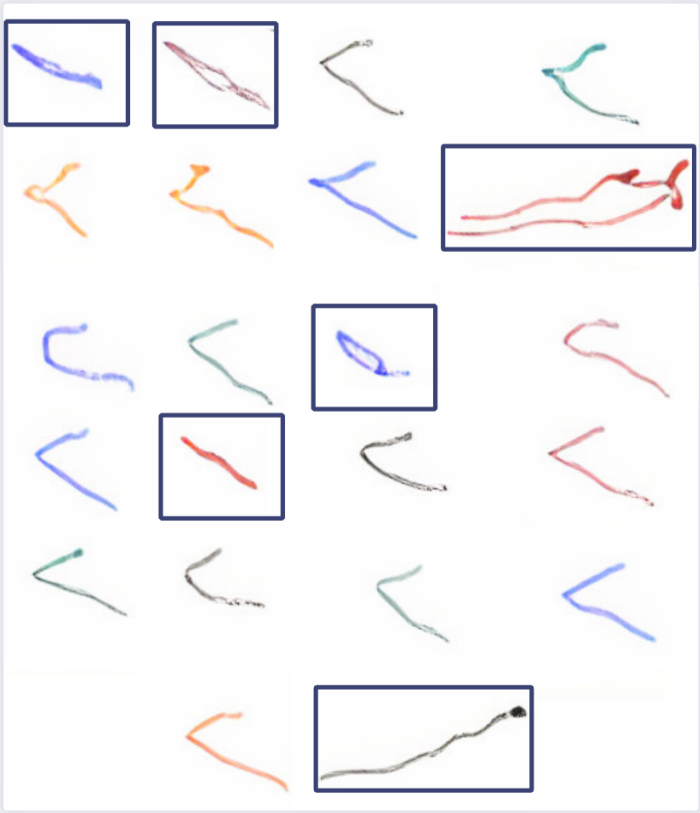

Figure 7: Example of trajectories of a cluster. The boxed trajectories are wrong clustered

data set is to segment a subset that contains abnormal behaviors of a person in terms of their displacement in the video (abnormal trajectories). For that, we manually generate a ground truth considering the conditions for rare and normal behavior.

6.1.2 Feature extraction and clustering

The first step to accomplish this case study is feature extraction. We considered Fourier and Wavelet transform as trajectory feature extractors in this work. Moreover, for this case study, we considered an additional simple descriptor based on interpolation, to have a simple description method to compare. Once the coefficients are found for each trajectory, they are used as a feature vector in the clusterization process. The easiest way to achieve the feature extractor performance is to compare an element with its nearest neighbor. Fig. 6 shows the result of finding the nearest neighbor trajectory using Kernel Density Estimation (KDE). The first column illustrates a random example, while the second column shows the most similar trajectory considering the vector of each descriptor. Notice that Fourier and Wavelet Transform extract better similar neighboring trajectories, demonstrating that they pull characteristics for grouping better than the interpolation method. Thus, these are used as descriptors of the morphology of our trajectories.

In the next step, using *Affinity Propagation*, we clustered our trajectories based on their morphology. To measure the performance of our clustering method, we used the *counting* method. In each cluster, this metric counts how many elements are wrong clustered. The average error is computed based on the number of elements of the cluster and the number of wrong clustered elements. For instance, Fig. 7 shows the trajectories of a cluster; the boxed trajectories are wrong clustered. In this example, the cluster has 22 elements, with six wrong clustered trajectories, having a 27.27% error percentage (e_i). The total error

Table 2: Error percentages for each descriptor

Descriptor	***Affinity Propagation*** *(E%)*
Interpolation	15.67
Fourier	9.20
Wavelet	**6.77**

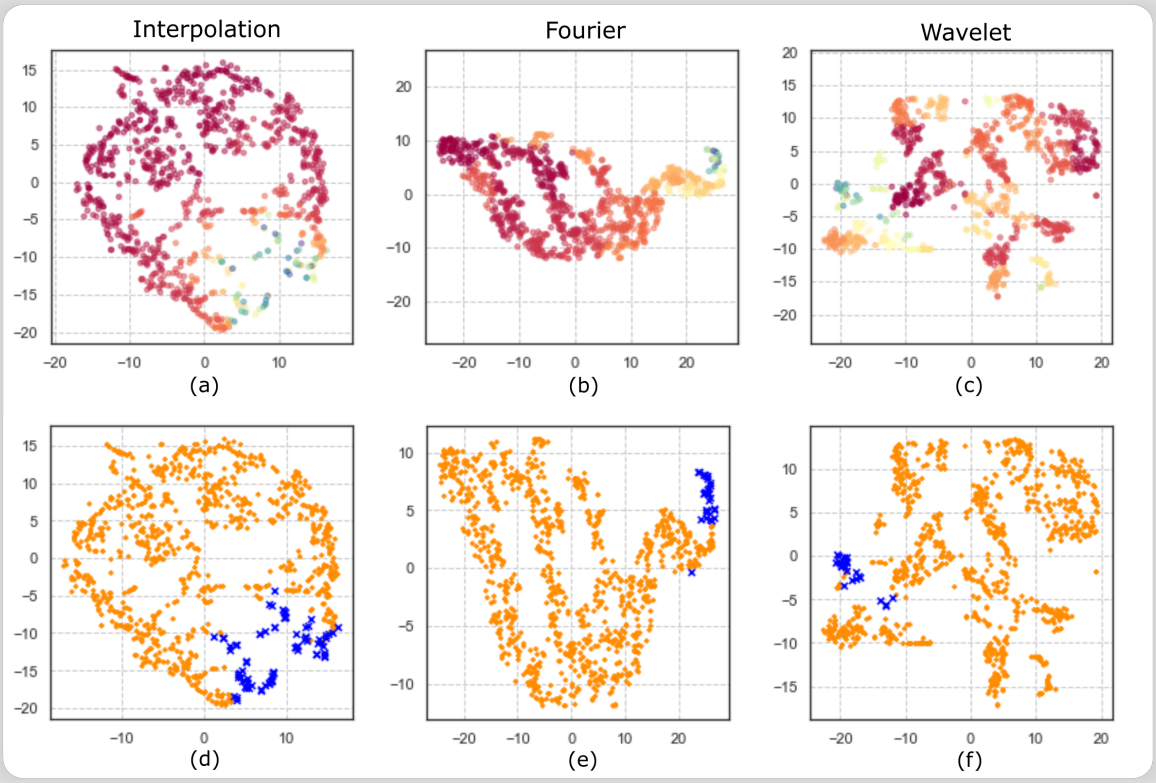

Figure 8: Three feature spaces are generated with different descriptors. They are projected using t-SNE. In (a), (b) and (c) each color represent a different cluster. In (d), (e) and (f) the colored blue x are anomalies defined by our model

percentage is calculated by the average of the percentage of each cluster ($E = \sum^{n} e_i/n$).

6.1.3 Findings

In order to explore each feature extractor method, we conduct an empirical analysis examining the error percentage of each method. Table 2 shows the result of the clustering error percentage of each method. Notice that the Wavelet descriptor has the lowest error percentage, followed by Fourier.

Based on the lowest error percentage in Table 2, in this case study, we use Wavelet transform as the main descriptor. To validate our results is necessary to set a *threshold* defined as the maximum number of elements that a cluster can have to be considered abnormal. Table 3 shows the threshold influence in each quality metric. These metrics were computed considering the ground truth. We can see that 2 is the best threshold, with a 0.97 average for each metric. Note that *Accuracy* values are more significant than 0.989, most of them with 0.99, showing the good performance of our approach.

We present some visual results regarding the qualitative evaluation of our choice (wavelet, Affinity propagation, and threshold 2). Fig. 8 shows the feature spaces generated with descriptors used. Each multidimensional feature space is projected with t-SNE in a 2D plane. We can see that the points distributions are different, and the Wavelet descriptor presents a greater separation of clusters. The plotted blue x's represent the anomalies detected by our approach with different descriptors. Fig. 10 shows anomaly trajectories detected by our approach contained in the rare videos of the ground truth. Fig. 11 shows three clusters generated by our choice. We can see that each cluster groups similar trajectories by their morphology. Finally,

Table 3: Results of gathering anomaly trajectories by our choice

Threshold	***Accuracy***	***Precision***	***Recall***	***Specificity***
1	0.99	1.00	0.70	1.00
2	**0.99**	**0.97**	**0.93**	**0.99**
3	0.989	0.731	1.00	0.98

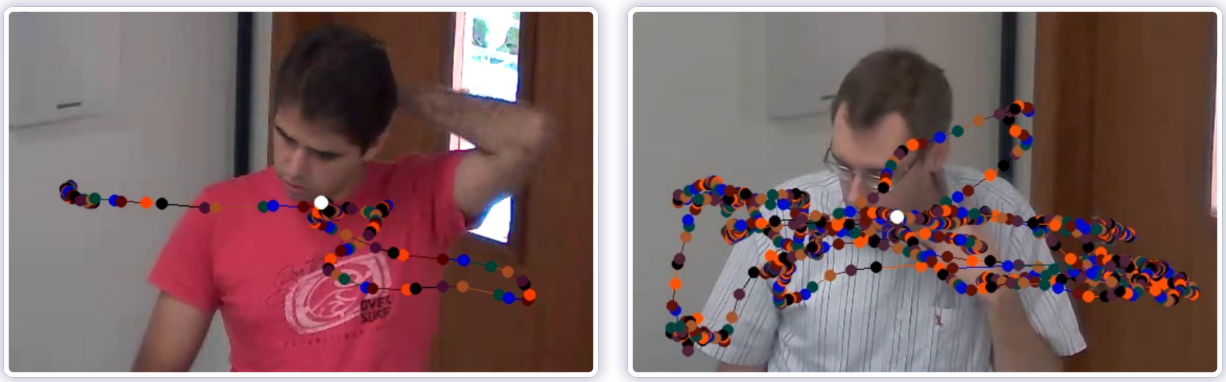

Figure 9: Thumbnails of videos with their respective trajectories. Rare videos with the abnormal trajectories (pronounced deformation). The white point represents the *fiducial point* taken as a reference, while the colored points represent the *fiducial point* in each video frame

Fig. 9 shows two thumbnails after our processing of surveillance videos. The trajectory is represented on the video; each point represents the position in each frame, while the white point represents the point taken for the trajectory's generation. This point allows the observation of the trajectory's direction on each detected point. It could be noticed that the abnormal trajectory presents pronounced deformations (see Fig. 9) while the normal trajectory has smooth chances (see Fig. 11).

6.2 Anomaly Route Detection

The second case study involves identifying anomaly trajectories in the data generated by urban transport buses. The idea is to detect when a transport bus goes out of its route or when the devices used are defective. To achieve it, we used *Intelligent Transportation System* (ITS) Dataset, it was generated from Alvarez et al. [33] and these data are generated by Global Positioning System (GPS) devices. The work-study of Alvarez et al. consisted of developing an intelligent system to monitor urban transport buses.

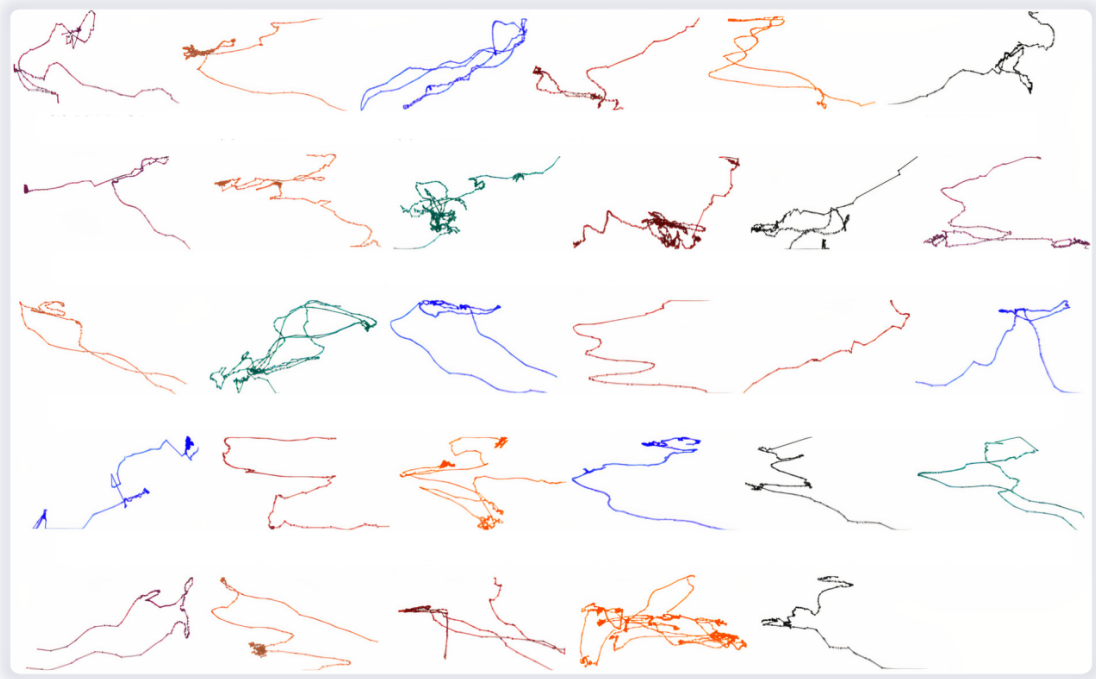

Figure 10: Trajectories detected as anomaly by our approach

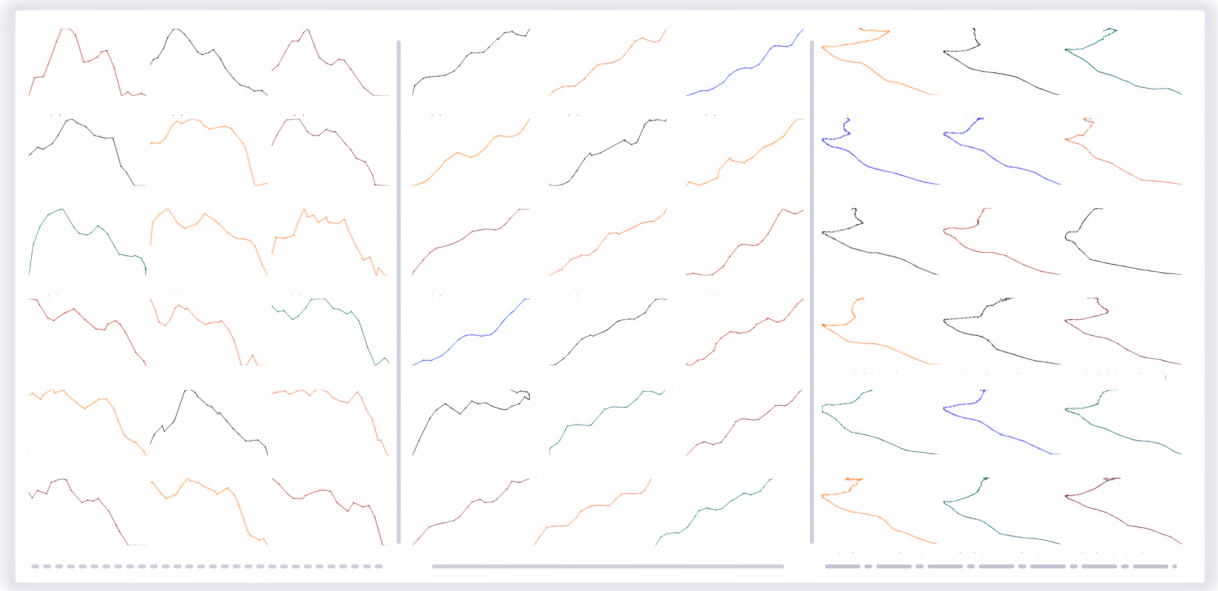

Figure 11: Normal trajectories detected by our approach and three different generated clusters (columns)

6.2.1 Dataset

The Traffic FLow dataset contains different routes of public transport units. It considers the differences between the cycles achieved. One cycle includes the outbound and return routes. The service of monitoring buses is used by more than one transport company. The monitoring process is constant, the GPS devices are running at twenty-four hours, and the data points are produced every 30 seconds. For the study, the criteria to define a "normal" trajectory is given by the route determined by the chosen public transport company. On the other hand, the criteria to define anomaly trajectory are: the route that the transport company does not define, or the route poorly generated by the GPS device errors, due to their poor installation or wear.

This dataset offers a large quantity of monitoring data, so we needed to perform a selection process. The data selection criteria are as follow: we choose just the transport company that passes through the university, to reduce the enormous amount of data. We choose the date from May 02 to June 06 in 2019 because this period corresponds to the last month of the dataset we possess. We select the schedule between five and twenty-two hours to consider the buses' working hours.

The performed selection produced a dataset consisting of $1,107,795$ records, transformed into $15,013$ trajectories. To build the trajectories, the following four attributes provided by the dataset were considered: the day, the bus license plate, the trip number and the time when the records was created. The trip number attribute permits indicate when the bus achieved a cycle, including going back and forth in a trip. With the three first listed attributes, we create subsets of records for them with the time attribute, order the records of each subset, to extract each point chronologically, and build the trajectories. We chose these attributes since any of these buses can perform multiple trips in only one day, and it is necessary to distinguish them.

Finally, once we constructed the trajectories, it was necessary to eliminate noisy points generated by some errors of GPS devices. For this purpose, we achieved two steps: we consider only the points whose the difference in latitude and longitude plane is less than one unit and do not consider trajectories with less than twenty points. The criteria are the following: a) This difference in units was taken because the buses can not move more than one unit of measure in those planes in such a short time, and b) a route cannot be defined by less than twenty points for this dataset.

6.2.2 Feature extraction and clustering

To next steps, we chose only MDWD as the descriptor, since for this case study, the idea is not to compare two descriptors but rather to show results by our model using data from an actual application. We chose MDWD taking into account the previous results obtained in the first case study.

For this second case study we divided the data into four groups, it is caused by the large amount of data. The four data sets have the following distribution: the first set consisting of $3,754$ trajectories and the following three trajectory sets, consist of $3,753$ elements each one. The clustering process is applied once the feature vectors have been generated for each set. With these datasets, we generate 93, 71, 70 and 78

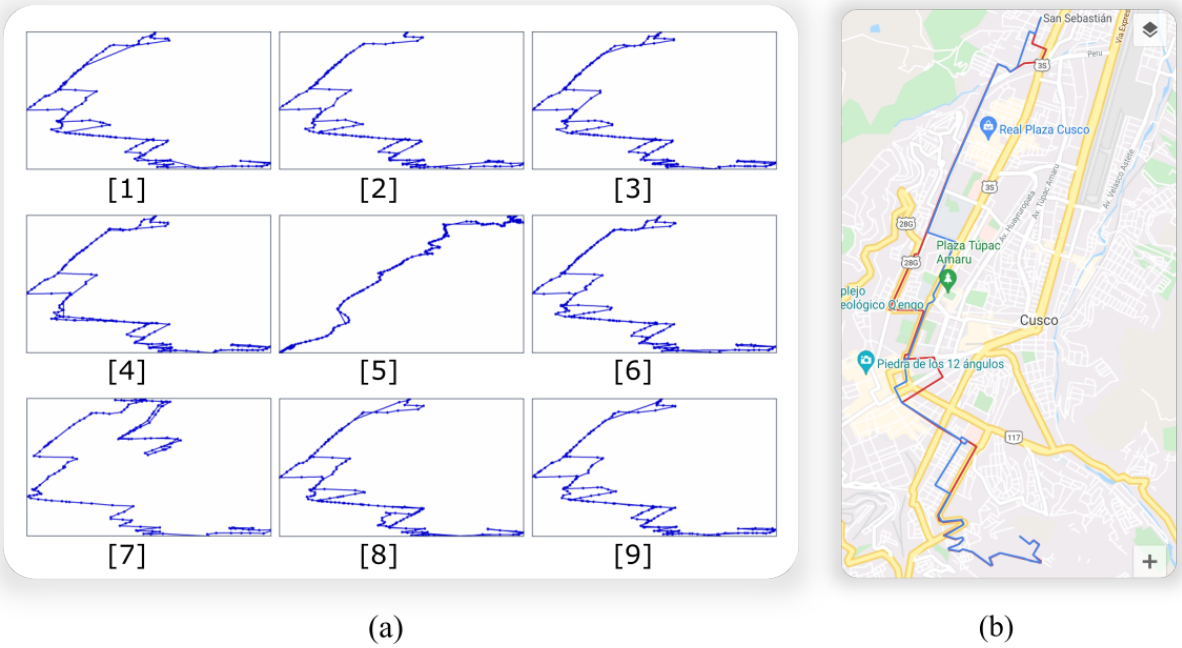

Figure 12: Trajectory anomaly detections in Traffic flow dataset, Fig. (a.5) and Fig. (a.7) are detected as anomalous by our approach. (b) shows the selected route viewed from Google Maps

clusters respectively.

As anomaly threshold, we consider the clusters containing one and two elements. The Fig. 12.a shows the normal trajectories, except for two (Fig. 12.a.5 and Fig. 12.a.7), which are detected as anomalies. According to Fig. 12.a is possible to notice that these trajectories are different from the majority group. The Fig. 12.b shows the specific selected route for this case study. Each of the routes is drawn and available in the application of the intelligent system [33]. This drawing made on Google Maps allows us to visualize where the route is located and distinguishes the outbound and return routes with different colors (red and blue).

6.2.3 Findings

This case study aims to shows a tool to visualize anomaly routes or constant errors in GPS devices. For this reason, we present some results regarding the qualitative evaluation of our choice. With the help of the ITS software administrators and Google Maps, it was possible to compare and validate our results. First, we find when the anomaly trajectories were produced with the date and time attributes. Fig. 12.a.5 shows the anomaly trajectory produced on May 19 from 5h to 21h on Sunday. Asking the ITS software administrators, we found that this trajectory is different because the company on Sundays works irregularly, and these days do not strictly respect the route. For the second anomalous trajectory (Fig. 12.a.7), it was produced on Thursday, May 09, from 07h to 21h. By finding the route in Google Maps and asking the ITS software administrators, we conclude that on this date, due to the festivities, a parade occurred in the city, it produced the driver the need to use alternate routes to finish their trip at the designated interval time.

7 Time and Memory Efficiency

In order to evaluate the performance of the descriptor we measure the running time and memory consuming. The experiments of our were performed on an Intel(R) Core(TM) i7-4710HQ CPU @ 2.50GHz 2.50 GHz Processor with 8GB RAM, Windows 10 Pro x64 bits. The trajectory pre-processing (normalization) was made using C++ language while the anomaly trajectory detection framework uses the following Python libraries: *Scipy* and *Numpy* [34], *Scikit-learn* [35], *PyWavelets* [36] and the process of measure efficiency was performed with *Time* and *Memory-Profile* Python libraries.

For each of the case studies, we sampled different sub-sets. The division of the data set was made taking into account the size variation, from small to bigger. The data distribution is as follows: three trajectories sets were created for each case study, and each of these sets has a different number of trajectories. For the first case study, the number of elements of each set are one hundred, one thousand, and two thousand.

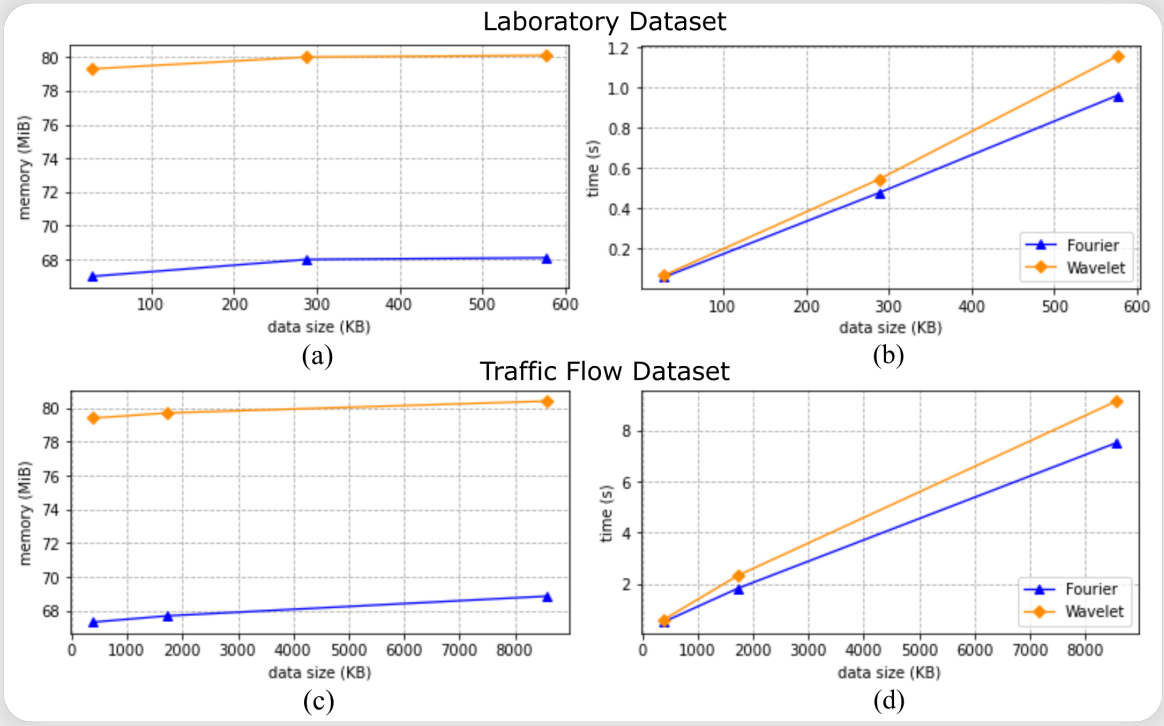

Figure 13: Plots of the mean memory and mean run time dependent on the weights of each input data sets. These performed for each case study

For the second case study, the number of elements of each set are one thousand, two thousand, and fifteen thousand.

We measure the execution time and the memory consumed by each subset. The average values were obtained for each sub-set: In the case of time, seven measurements were averaged to obtain a run-time value, and in the case of memory, three measurements were averaged to obtain a memory-consuming value. Fig 13 shows each of these averaged values for each different trajectory set size.

Just the generation of characteristics was considered for the measurements of time and memory. The reading, writing, or transforming string variables to numbers is not considered in implementing the descriptors since these processes can alter the results, and each descriptor ends up processing $2N$ signals for every N-trajectories because each trajectory is divided into two sub-trajectories.

In order to show the scalability under different sizes of the input data, We plot our results with different data sizes, these measured in kilobytes (KB). Fig. 13.a and Fig. 13.c show the memory efficiency measure in mebibyte (MiB), and Fig. 13.b and Fig. 13.d show the run-time efficiency measure in seconds (s).

Due to the variable number of points of the trajectories in both case studies, we decided to plot these data based on their file weights. According to Fig. 13, we can observe that while the size of the input data grows, the memory used does not scale as much as the execution time. Due to the small memory consumption and short execution time, we can obtain results in a bit of time.

The complexity of our approach is limited by the algorithmic complexity of the AP clustering method since this algorithm has an algorithmic complexity of order $O(N^2T)$ [37], where N is the number of samples and T is the number of iterations until convergence. The AP algorithm has the order $O(N^2)$ in terms of memory complexity. At the same time, the MDWD and DFT algorithms have $O(NlogN)$ complexity order in time, and memory [38]. These complexities are confirmed by our measures obtained in this section.

In this section we show that, it is possible to obtain real-time results to detect anomaly trajectories, even with limited hardware. Due to the complexity of these algorithms, our implementation has an advantage in speed and storage. Additionally, according to the literature, clustering AP can be improved at runtime as the improvement made by Shi [39] for a better performance of the clustering algorithm using Graphics Processing Unit (GPU).

8 Conclusions

This paper presents a comparative analysis of trajectory descriptors using coefficient feature space representation to detect anomaly trajectories. We have also introduced the MDWD as a shape-feature extractor on trajectories, and it yields satisfactory results compared to other descriptors, obtaining greater performance in the detection of anomaly trajectories, due to the better trajectory description provided by this method. Our study was based on an unsupervised learning method using similarity analysis, the validation process took into account various synthetic and real-life datasets.

The usefulness of our approach has been demonstrated throughout a case studies to detect anomaly trajectories in real video surveillance and Traffic data sets, and with these same data, we show the efficiency in run time and memory usage of our descriptors. In clustering, we observe that the used dataset influences the AP algorithm. Whether the number of classes of trajectories increases, the classification precision of the algorithm decrease, and if the trajectories are constantly repeated, the algorithm will consider minor differences.

There is a possible improvement in our approach. For the unsupervised learning process is possible to use the Adaptive AP method [40], to automatically select the preference parameter and find the optimal clustering solution. Using GPU is possible to improve AP performance using the implementation made by Shi [39]. For the tunning of the anomaly threshold, is possible to use the k-Nearest Neighbor technique, in order to automatically define this threshold. Finally, in order to have more details about trajectory similarity measures, we can make experiments with different trajectory patterns and compare their distances.

Acknowledgments

I wish to thank all those who helped me with this study, especially prof. William Schwartz for his advice and ideas during my studies and Smart Sense Lab from UFMG university for providing me with the Laboratory dataset. Part of this work was gathered from my master studies of UCSP university and the other part of the work described in this paper was funded by UNSAAC university. These research teams supported me and help me to finish this work.

References

[1] M. J. Wisdom, N. J. Cimon, B. K. Johnson, E. O. Garton, and J. W. Thomas, "Spatial partitioning by mule deer and elk in relation to traffic." in *In: Transactions of the 69th North American Wildlife and Natural Resources Conference: 509-530*, 2004. [Online]. Available: https://www.fs.usda.gov/treesearch/pubs/24837

[2] M. D. Powell and S. D. Aberson, "Accuracy of united states tropical cyclone landfall forecasts in the atlantic basin (1976–2000)," *Bulletin of the American Meteorological Society*, vol. 82, no. 12, pp. 2749–2768, 2001. [Online]. Available: https://doi.org/10.1175/1520-0477(2001)082⟨2749:AOUSTC⟩2.3.CO;2

[3] Y. Pang and Y. Liu, "Conditional generative adversarial networks (cgan) for aircraft trajectory prediction considering weather effects," in *AIAA Scitech 2020 Forum*, 2020, p. 1853. [Online]. Available: https://doi.org/10.2514/6.2020-1853

[4] J. Kim and H. S. Mahmassani, "Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories," *Transportation Research Procedia*, vol. 9, pp. 164–184, 2015. [Online]. Available: https://doi.org/10.1016/j.trpro.2015.07.010

[5] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 11, pp. 2287–2301, 2011. [Online]. Available: https://doi.org/10.1109/TPAMI.2011.64

[6] F. Turchini, L. Seidenari, and A. Del Bimbo, "Understanding sport activities from correspondences of clustered trajectories," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 43–50. [Online]. Available: https://doi.org/10.1109/ICCVW.2015.103

[7] S. Paul, F. Hole, A. Zytek, and C. A. Varela, "Flight trajectory planning for fixed-wing aircraft in loss of thrust emergencies," *arXiv preprint arXiv:1711.00716*, 2017. [Online]. Available: https://doi.org/10.1109/DASC.2018.8569842

[8] R. Laxhammar and G. Falkman, "Online learning and sequential anomaly detection in trajectories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 6, pp. 1158–1173, 2014. [Online]. Available: https://doi.org/10.1109/TPAMI.2013.172

[9] C. Parent, S. Spaccapietra, C. Renso, G. Andrienko, N. Andrienko, V. Bogorny, M. L. Damiani, A. Gkoulalas-Divanis, J. Macedo, N. Pelekis *et al.*, "Semantic trajectories modeling and analysis," *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, pp. 1–32, 2013. [Online]. Available: https://doi.org/10.1145/2501654.2501656

[10] S. Khalid and A. Naftel, "Automatic motion learning in the presence of anomalies using coefficient feature space representation of trajectories," *Acta Automatica Sinica*, vol. 36, no. 5, pp. 655–666, 2010. [Online]. Available: https://doi.org/10.1016/S1874-1029(09)60028-8

[11] G. F. Quispe-Torres, G. Garcia-Zanabria, H. Vera-Olivera, and L. Enciso-Rodas, "Trajectory anomaly detection based on similarity analysis," in *2021 XLVII Latin American Computing Conference (CLEI)*. IEEE, 2021, pp. 1–10. [Online]. Available: https://doi.org/10.1109/CLEI53233.2021.9639966

[12] X. Kong, M. Li, K. Ma, K. Tian, M. Wang, Z. Ning, and F. Xia, "Big trajectory data: A survey of applications and services," *IEEE Access*, vol. 6, pp. 58 295–58 306, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2873779

[13] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07. New York, NY, USA: Association for Computing Machinery, 2007. [Online]. Available: https://doi.org/10.1145/1247480.1247546

[14] G.-P. Roh and S.-w. Hwang, "Nncluster: An efficient clustering algorithm for road network trajectories," in *International Conference on Database Systems for Advanced Applications*. Springer, 2010, pp. 47–61. [Online]. Available: https://doi.org/10.1007/978-3-642-12098-5_4

[15] B. Han, L. Liu, and E. Omiecinski, "Neat: Road network aware trajectory clustering," in *2012 IEEE 32nd International Conference on Distributed Computing Systems*. IEEE, 2012, pp. 142–151. [Online]. Available: https://doi.org/10.1109/ICDCS.2012.31

[16] A. Naftel and S. Khalid, "Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space," *Multimedia Systems*, vol. 12, no. 3, pp. 227–238, 2006. [Online]. Available: https://doi.org/10.1007/s00530-006-0058-5

[17] R. Annoni and C. H. Forster, "Analysis of aircraft trajectories using fourier descriptors and kernel density estimation," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2012, pp. 1441–1446. [Online]. Available: https://doi.org/10.1109/ITSC.2012.6338863

[18] H. Zhang, Y. Luo, Q. Yu, L. Sun, X. Li, and Z. Sun, "A framework of abnormal behavior detection and classification based on big trajectory data for mobile networks," *Security and Communication Networks*, vol. 2020, 2020. [Online]. Available: https://doi.org/10.1155/2020/8858444

[19] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-based anomalous event detection," *IEEE Transactions on Circuits and Systems for video Technology*, vol. 18, no. 11, pp. 1544–1554, 2008. [Online]. Available: https://doi.org/10.1109/TCSVT.2008.2005599

[20] H. Ergezer and K. Leblebicioğlu, "Anomaly detection and activity perception using covariance descriptor for trajectories," in *European Conference on Computer Vision*. Springer, 2016, pp. 728–742. [Online]. Available: https://doi.org/10.1007/978-3-319-48881-3_51

[21] X. Wang, K. T. Ma, G.-W. Ng, and W. E. L. Grimson, "Trajectory analysis and semantic region modeling using nonparametric hierarchical bayesian models," *International journal of computer vision*, vol. 95, no. 3, pp. 287–312, 2011. [Online]. Available: https://doi.org/10.1007/s11263-011-0459-6

[22] R. R. Sillito and R. B. Fisher, "Semi-supervised learning for anomalous trajectory detection." in *BMVC*, vol. 1, 2008, pp. 035–1. [Online]. Available: https://doi.org/10.5244/C.22.103

[23] I. Labs. (2004) Caviar "INRIA" dataset. [Online]. Available: https://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

[24] H. M. Dee and D. C. Hogg, "Detecting inexplicable behaviour," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2004, pp. 50.1–50.10, doi:10.5244/C.18.50. [Online]. Available: http://dx.doi.org/10.5244/C.18.50

[25] C. Ma, Z. Miao, M. Li, S. Song, and M.-H. Yang, "Detecting anomalous trajectories via recurrent neural networks," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 370–382. [Online]. Available: https://doi.org/10.1007/978-3-030-20870-7_23

[26] C. Panagiotakis, N. Pelekis, I. Kopanakis, E. Ramasso, and Y. Theodoridis, "Segmentation and sampling of moving object trajectories based on representativeness," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 7, pp. 1328–1343, 2011. [Online]. Available: https://doi.org/10.1109/TKDE.2011.39

[27] S. W. Smith *et al.*, "The scientist and engineer's guide to digital signal processing. chapter 8: The discrete fourier transform," 1997, iSBN:978-0-9660176-3-2.

[28] J. Cooley, P. Lewis, and P. Welch, "The finite fourier transform," *IEEE Transactions on audio and electroacoustics*, vol. 17, no. 2, pp. 77–85, 1969. [Online]. Available: https://doi.org/10.1109/TAU.1969.1162036

[29] J. Wang, Z. Wang, J. Li, and J. Wu, "Multilevel wavelet decomposition network for interpretable time series analysis," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2437–2446. [Online]. Available: https://doi.org/10.1145/3219819.3220060

[30] B. J. Frey and D. Dueck, "Clustering by passing messages between data pointssss," *science*, vol. 315, no. 5814, pp. 972–976, 2007, doi:10.1126/science.1136800. [Online]. Available: https://www.science.org/doi/10.1126/science.1136800

[31] R. Colque., E. Cayllahua., V. C. de Melo., G. Chavez., and W. Schwartz., "Anomaly event detection based on people trajectories for surveillance videos," in *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,*, INSTICC. SciTePress, 2020, pp. 107–116, doi:10.5220/0008952401070116. [Online]. Available: http://dx.doi.org/10.5220/0008952401070116

[32] M. L. Dias, C. L. C. Mattos, T. L. da Silva, J. A. F. de Macedo, and W. C. Silva, "Anomaly detection in trajectory data with normalizing flows," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8. [Online]. Available: https://doi.org/10.48550/arXiv.2004.05958

[33] E. Alvarez Mamani, "Sistema de transporte inteligente (sti), para el control y monitoreo del servicio urbano en la ciudad del cusco." Universidad Nacional de San Antonio Abad del Cusco, 2018.

[34] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The numpy array: a structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22–30, 2011, doi:10.1109/MCSE.2011.37. [Online]. Available: https://doi.org/10.1109/MCSE.2011.37

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://doi.org/10.48550/arXiv.1201.0490

[36] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019, doi:10.21105/joss.01237. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.01237

[37] R. Refianti, A. Mutiara, and S. Gunawan, "Time complexity comparison between affinity propagation algorithms." *Journal of Theoretical & Applied Information Technology*, vol. 95, no. 7, 2017.

[38] M. V. Wickerhauser, *Adapted wavelet analysis: from theory to software*. AK Peters/CRC Press, 1996.

[39] X. Shi, "Parallelizing affinity propagation using graphics processing units for spatial cluster analysis over big geospatial data," in *Advances in Geocomputation*. Springer, 2017, pp. 355–369. [Online]. Available: https://doi.org/10.1007/978-3-319-22786-3_32

[40] K. Wang, J. Zhang, D. Li, X. Zhang, and T. Guo, "Adaptive affinity propagation clustering," *CoRR*, vol. abs/0805.1096, 2008. [Online]. Available: http://arxiv.org/abs/0805.1096