ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Σχολή Χρηματοοικονομικής και Στατιστικής

Τμήμα Στατιστικής και Ασφαλιστικής Επιστήμης

Μεταπτυχιακό Πρόγραμμα στην Εφαρμοσμένη Στατιστική

Αποτίμηση Ομοιότητας Ολιστικών Τροχιών Κινούμενων Αντικειμένων

Βασίλης Γεωργάκας

Διπλωματική Εργασία

που υποβλήθηκε στο Τμήμα Στατιστικής και Ασφαλιστικής Επιστήμης του Πανεπιστημίου Πειραιώς ως μέρος των απαιτήσεων για την απόκτηση του Μεταπτυχιακού Διπλώματος Ειδίκευσης στην Εφαρμοσμένη Στατιστική.

Πειραιάς
Σεπτέμβριος 2021

Η παρούσα Διπλωματική Εργασία εγκρίθηκε ομόφωνα από την Τριμελή Εξεταστική Επιτροπή που ορίσθηκε από τη ΓΣΕΣ του Τμήματος Στατιστικής και Ασφαλιστικής Επιστήμης του Πανεπιστημίου Πειραιώς στην υπ’ αριθμό …… συνεδρίαση του σύμφωνα με τον εσωτερικό Κανονισμό Λειτουργίας του Προγράμματος Μεταπτυχιακών Σπουδών στην Εφαρμοσμένη Στατιστική

Τα μέλη της Επιτροπής ήταν:

- Νικόλαος Πελέκης
- Μάρκος Κούτρας
- Ελευθέριος Κοφίδης

Η έγκριση της Διπλωματική Εργασίας από το Τμήμα Στατιστικής και Ασφαλιστικής Επιστήμης του Πανεπιστημίου Πειραιώς δεν υποδηλώνει αποδοχή των γνωμών του συγγραφέα.

UNIVERSITY OF PIRAEUS

School of Finance and Statistics

Department of Statistics and Insurance Science

Postgraduate Program in Applied Statistics

Evaluation of similarity in holistic trajectories

Vasilis Georgakas

MSc Dissertation

submitted to the Department of Statistics and Insurance Science of the University of Piraeus in partial fulfilment of the requirements or the degree of the Master of Science in Applied Statistics

Piraeus
September 2021

Στους γονείς μου και στον αδερφό μου

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, κ. Νικόλαο Πελέκη, για την πολύτιμη βοήθεια, τη στήριξη κα την καθοδήγηση του στην υλοποίηση της παρούσας διπλωματικής εργασίας. Επίσης ευχαριστώ θερμά του καθηγητές, κ. Μάρκο Κούτρα και κ. Ελευθέριο Κοφίδη για την συμμετοχή τους στην τριμελή επιτροπή και υποστήριξη τους.

Τέλος, θέλω να πω ένα μεγάλο ευχαριστώ στην οικογένεια μου για την υποστήριξη και την συμπαράσταση που μου έδωσαν κατά την διάρκεια των σπουδών μου.

Περίληψη

Για πολλά χρόνια η έρευνα πάνω στην αποτίμηση ομοιότητας τροχιών επικεντρώθηκε σε δισδιάστατες ακολουθίες σημείων, λαμβάνοντας υπόψη μόνο πληροφορίες χώρου και χρόνου, που ονομάζονται ακατέργαστες τροχιές. Ωστόσο, η αυξημένη χρήση τεχνολογιών GPS και κοινωνικών δικτύων, οδήγησε πολλές προσεγγίσεις στον εμπλουτισμό αυτών των κινούμενων αντικειμένων με πολλαπλές διαστάσεις συμπεριλαμβανομένης της σημασιολογική διάστασης. Ως αποτέλεσμα, οι πιο πρόσφατες προσεγγίσεις έχουν προτείνει μέτρα ομοιότητας που υποστηρίζουν το χώρο, το χρόνο και τη σημασιολογία. Ένας τρόπος σύγκρισης της αποτελεσματικότητας και της στιβαρότητας αυτών των μέτρων ομοιότητας είναι η μετατροπή των τροχιών και ο υπολογισμός της ομοιότητας μεταξύ της αρχικής τροχιάς και των μετασχηματισμένων τροχιών. Στη διπλωματική μας, προτείνουμε μια μέθοδο που εφαρμόζει διαφορετικούς τύπους μετασχηματισμών σε σημασιολογικές τροχιές και δημιουργεί σύνολα κινούμενων αντικειμένων σύμφωνα με προκαθορισμένο συντελεστή *r*. Αυτοί οι μετασχηματισμοί, έχουν την ικανότητα να μεταμορφώνουν τις πληροφορίες του χώρου, του χρόνου καθώς και όλα τα χαρακτηριστικά της σημασιολογικής διάστασης. Για κάθε σύνολο, υπολογίζουμε την ομοιότητα της αρχικής τροχιάς σε σχέση με τις μετασχηματισμένες τροχιές, χρησιμοποιώντας διαφορετικά μέτρα ομοιότητας.

Abstract

For many years the research on evaluation of similarity of trajectories has focused on two-dimensional sequences of points, considering only space and time information, called raw trajectories. However, the increase use of GPS technologies and social media, has led many approaches to enrich these moving objects with multiple dimensions including semantic dimension. As a result, most recent approaches have proposed similarity measures that support space, time and semantics. A way to compare the effectiveness and the robustness of these similarity measures is to transform the trajectories and compute the similarity between the seed trajectory and the transformed trajectories. In our thesis, we propose a method that applies different types of transformations over semantic trajectories and creates sets of moving objects, according to a predefined rate *r*. These transformations, have the ability to transform the information of space, time and as well as all the attributes of the semantic dimension. For each set, we calculate the similarity of the original trajectory in relation to the transformed trajectories, using different similarity measures.

Contents

List of Figures

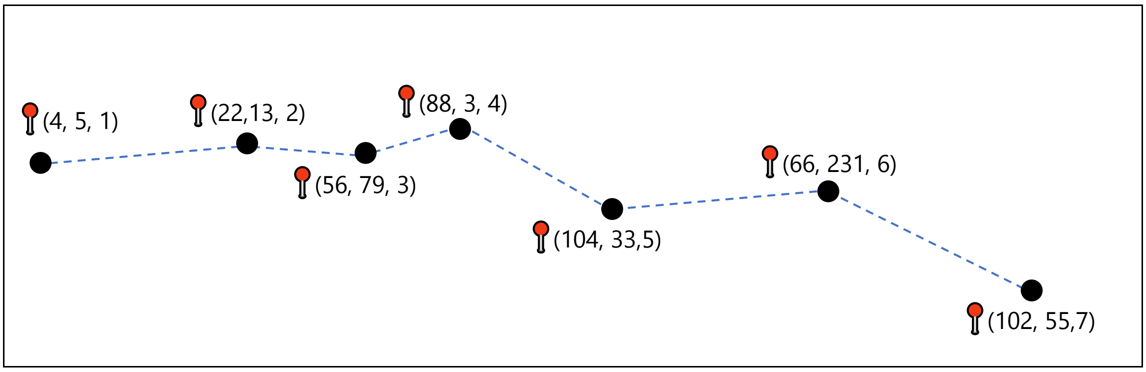
List of Tables

List of Pictures

Chapter 1. Introduction

In the last decades, the research on moving objects has attracted a lot of attention, driven by major developments in the field of technology, due to the use of smartphones with high accuracy GPS-enabled which keeps track of the location of the user, large amounts of data are available, representing the movement history of moving objects, known as trajectories. When an object is moving, its data that describes the information of its motion is collected through its movement in the form of space and time, called raw trajectories. A raw trajectory refers to a sequence of sample points $T = <p_1 , p_2 , \ldots , p_n>$ with $p_i = (x_i, y_i, t_i)$, where x, y represents the position of the object in space and t corresponds to the time dimension. An example of a raw trajectory is illustrated in Figure 1-1. In this figure, assume we have a trajectory *T*, where small circles correspond to the sampled points. The spatial coordinates and the time instants can be seen next to the trajectory points. The last point of the trajectory in the figure is located at the coordinates (102, 55) at time instant 7.

Figure 1-1: Example of a Raw Trajectory



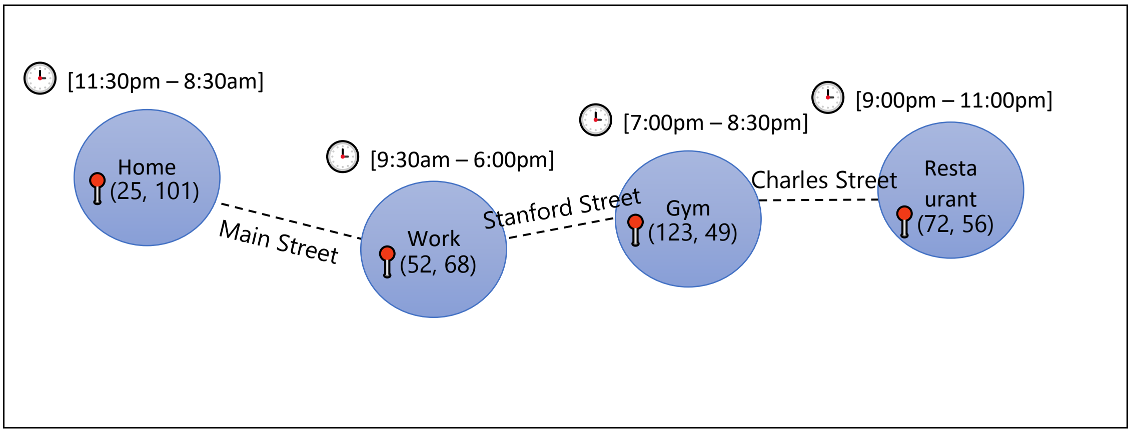
In recent years, trajectory similarity analysis has experienced significant growth, and several measures have been proposed for raw trajectory similarity, basically considering the properties of space-time. Movement similarity measures are useful for several application domains. Typical examples include collecting trajectories in taxicabs for the safety of the passenger, so we can detect if the taxi driver took a different path than the one he is supposed to use. For example, assume that a large number of taxis take the similar routes between two destinations, then we may identify the representative trajectory between these two destinations, and can further infer future locations based on the historical movement of a taxi. More examples are, tracking animals for their migration patterns or identifying their specie by their trajectory and on applications that support friend recommendation based on the paths they follow on their daily life and how similar they are, what is useful in sharing a car or a taxi.

Similarity measures have been proposed for several purposes such as clustering techniques for grouping most similar trajectories such as in (Lee, Han, & Whang, 2007), (Zhao, 2011), in (Pelekis, Kopanakis, Kotsifakos, Frentzos, & Theodoridis , 2011) where they study the effect of uncertainty in Trajectory Databases clustering and SemT-OPTICS for semantic trajectory clustering proposed in (Pelekis, Sideridis, Tampakis, & Theodoridis, 2016), extending the well-known T-Optics (Nanni & Pedreschi, 2006) clustering algorithm that was focused on raw trajectories. Furthermore, more measures proposed for classification of objects according to their trajectories, predicting their location based on trajectories that follows the same paths, outlier detection by identifying the individuals that have different movement from the majority etc. Most state-of-the-art methods on similarity measurement have focused on raw trajectories. These approaches have considered the physical properties of raw trajectories and a summary of these measures is presented in (Ranacher & Tzavella, 2014). Some examples of these approaches include the well-known DTW (Dynamic Time Wrapping) (Berndt & Clifford, 1994), developed for time series, LCSS (Longest Common SubSequence) (Vlachos, Kollios, & Gunopulos, Discovering similar multidimensional trajectories, 2002), EDR (Edit Distance on Real sequence) (Chen, Özsu, & Oria, Robust and fast similarity search for moving object trajectories, 2005), ERP (Edit Distance with Real Penalty) (Chen & Raymond, On the marriage of edit distance and Lp norms, 2004) and UMS (Uncertain Movement Similarity) (Furtado, Alvares, Pelekis , Theodoridis, & Bogorny, 2018). The majority of these measures, compute the similarity score by comparing all points of one trajectory with all points of another to compute their distance in space by using dynamic programming approach (DP). DTW is a distance-based measure and on the other hand LCSS and EDR are ε-threshold based strategy meaning that they use threshold to determine if two sample points match or not. Their main drawback is that they force a match on all dimensions in order to find similarity between trajectory points, not allowing partial similarity. ERP can handle local time shifting, which is essential for time series similarity matching, and is a metric. UMS focus only on the spatial dimension and use ellipses to compute the similarity of trajectory sample points as proposed in (Pfoser & Jensen, 1999). Furthermore, it avoids to use a fixed point threshold which makes it a parameter free method.

All the aforementioned works are focused only on spatio-temporal information of raw trajectories and are not eligible to take into consideration the semantics of a trajectory. More recently, an enormous effort is being made to add more data due the explosion of big data generated on the internet and the increasing use of social media, mobility data can be enriched with several kinds of semantic information, transforming raw trajectory into a semantic trajectory. The notion of semantic trajectory has several definitions and can be found in the literature for a semantic trajectory, such as (Alvares, et al., 2007) and (Bogorny, Renso, Aquino, Siqueira, & Alvares, 2014). Basically, a semantic trajectory is a sequence of locations with semantic data, such as the name and the type of the visited sites by a moving object, and the activities performed at each point. For the sake of simplicity, semantic trajectories are represented as a sequence of visited places called stops and moves, as originally introduced by (Spaccapietra, et al., 2008).

Figure 1-2 shows an example of a semantic trajectory, where trajectory A has four stops (Home, Work, Gym and Restaurant) and three moves (Main Street, Stanford Street, Charles Street). Note that in Figure 1-2 the trajectory is distributed along with the spatial coordinates, the time interval that the stop occurred, the category of the place and the street where the movement object moves. More attributes can be added to the stops and moves such as the activity of the stop, the transportation means etc., but the basic attributes that a trajectory should have are space, time and semantics.

Figure 1-2: Semantic Trajectory *A* with stops and moves



Most measures that were proposed for semantic trajectories does not consider all three dimensions (space, time and semantics) like MSTP (Maximal Semantic Trajectory Pattern Similarity) (Ying, Lu, Lee, Weng, & Tseng, 2010) which considers only the semantic dimension of a trajectory through stay cells. It assigns semantic terms to these cells and defines measure of semantic similarity based on the stay cells of each trajectory. An approach with the same drawback as MSTP (can’t handle all dimensions) since it is focused only on space and semantics, proposed in (Liu & Schneider, 2012) that splits a semantic trajectory into sub-trajectories and based on the longest common subsequence of visited sites, it calculates the semantic similarity of multiple trajectories. Two extensions of DTW that can handle multiple dimensions were proposed, known as MD-DTW (Holt, Gineke, Reinders, & Hendriks, 2007) which is built for multi-dimensional time series and (Shokoohi-Yekta, Hu, Jin, Wang, & Keogh, 2017). Most recently in (Furtado, Kopanaki, Alvares, & Bogorny, 2016) it is proposed the MSM (Multidimension Similarity Measure), which measures the similarity of semantic trajectories in all dimensions (space, time and semantics). In this approach, they manage to compute the distance of each dimension with a different distance function and give different weight for each dimension. However, due to the increase of social media, large amounts of trajectory data are generated by users that allow us to make comparisons and users analysis based on the type and the activities performed at each site. In (Arboleda, Fernández, & Bogorny, 2017) a similarity method was proposed that considers semantic aspects for finding similarity of trajectories, considering visited sited and activities performed in these sites. In (Petry, Ferrero, Alvares, Renso, & Bogorny, 2019) they proposed a new similarity measure called MUITAS (MUltIple-Aspect TrAjectorySimilarity) for multiple-aspect trajectories which considers both dependent and independent and semantically related attributes. In all previous approaches, the measures do not consider both stops and moves of a semantic trajectory. Considering this, in (Lehmann, Alvares, & Bogorny, 2019), they proposed a new semantic trajectory similarity measure that extends MSM, called as SMSM (Stops and Moves Similarity Measure), and takes into account both stops and moves, as well as their space, time and semantic dimensions. Furthermore, it considers the order between stops and gives different weights to stops, moves and dimensions according to the needs of the experiment.

In order to evaluate all the similarity measures many approaches use information retrieval evaluation technique. Specifically, they use the Precision-Recall approach, computing the Mean Average Precision (MAP), as stated in (Manning, Raghavan, & Schütze, 2008) and the Area Under Curve (AUC) values, as stated in (Baeza-Yates, Ricardo, & Ribeiro-Neto, 2011). The MAP value is calculated by considering the average precision in all levels of recall and the AUC value is calculated based on the area under the ROC curve. In our thesis, we will compute the average and the median similarity score of the initial trajectory in relation to other trajectories that we will create.

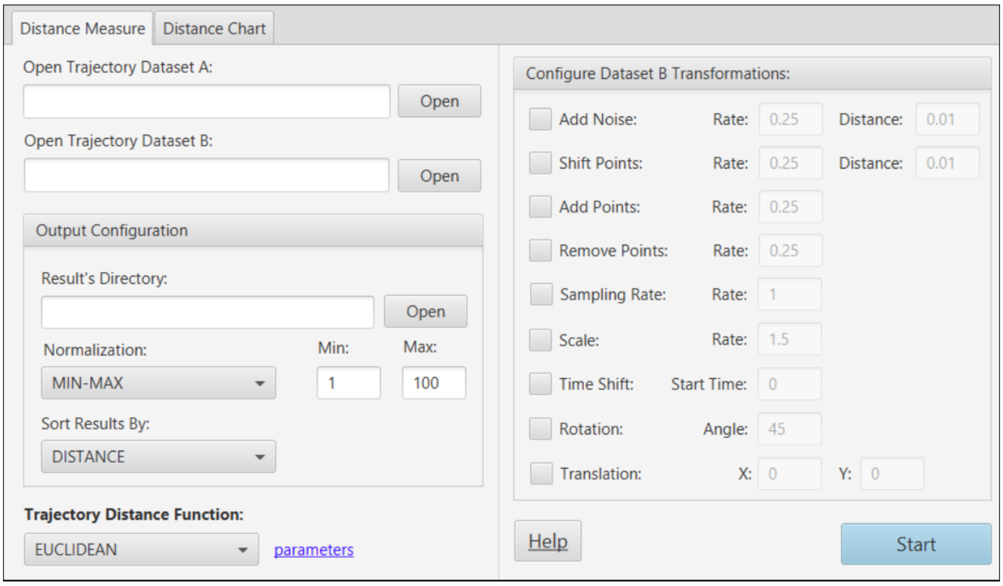
Despite the significant amount of data and research that has become available over the past years, existing trajectory similarity measurements only consider a portion of the information contained in trajectory data and therefore the methods they use may not be interpreted well in both semantic meaning and geographic distributions. As a result, enriching trajectories with semantic geographic information and applying data mining techniques on moving objects proof a good way in discovering behavioral patterns of moving objects that are more accurate and easy to interpret like proposed in (Alvares, et al., 2007), (Chakri & Raghay, 2016), (Wan , Zhou, & Pei, 2017). However, the complexity of moving objects and the lack of real datasets with semantic information, lead to the need of generating realistic datasets that demonstrates the mobility life of a population of moving objects such as Hermoupolis simulator presented in (Pelekis, Sideridis, Tampakis, & Theodoridis, 2016). Nevertheless, our work will not focus on this field of research.

1.1. Objective

In our thesis we will propose a method in which we will apply a set of transformations over a seed trajectory. The trajectories that will be used for our experiment will consider stops and its attributes, and therefore the transformations over the original trajectory will change the information of stops. In addition, the method of the transformations supports multiple dimensions, such as space, time and semantics, meaning that the types of the transformations will consider applying changes on all dimensions, depending on the proposed transformation.

In (douglasapeixoto, 2018) a benchmark system for trajectory similarity/distance measures was proposed with the following functionalities: i) choose well -suited techniques, ii) guide to select appropriate parameters, iii) reduce the development complexity. To support these functionalities, they designed a tool which apply trajectory data transformations, re-implement state-of-the-art trajectory distance measures within a common framework and calculate a mean to evaluate these techniques with different parameters. The languages used to design this tool were mostly Java and HTML, having the following application main GUI window:

Figure 1-3: Application main GUI window



Normalization: MIN-MAX Min: 1 Max: 100

Sort Results By: DISTANCE

Trajectory Distance Function: EUCLIDEAN

□ Add Noise: Rate: 0.25 Distance: 0.01

□ Shift Points: Rate: 0.25 Distance: 0.01

□ Add Points: Rate: 0.25

□ Remove Points: Rate: 0.25

□ Sampling Rate: Rate: 1

□ Scale: Rate: 1.5

□ Time Shift: Start Time: 0

□ Rotation: Angle: 45

□ Translation: X: 0 Y: 0

Some of the type of transformation applied on this work are: i) add noise to the given trajectory, ii) add some extra sample points to the given trajectory, iii) delete sample points from the given points, iv) randomly shift some of the trajectory points, etc. After completing these transformations in order to investigate their effectiveness and compare the similarity of the trajectories, they use some similarity/distance measure such as DTW, Euclidean Distance, EDR, ERP, LCSS, STLCSS (Spatial-Temporal Largest Common Subsequence distance) (Vlachos, Gunopoulos, & Kollios, Robust similarity measures for mobile object trajectories,

2012) and other spatio-temporal similarity measures. The main drawback of this method is that all the transformation are based on the space or/and time information of a trajectory. As a result as, we cannot apply this tool on similarity measures that consider semantic dimension. In our work, we will extend the transformations on multiple dimension trajectories and similarity measures that consider stops and the semantic information of the moving objects.

We follow a similar approach to the one proposed in (Furtado, Kopanaki, Alvares, & Bogorny, 2016), which is based on transforming a seed trajectory into many semantic trajectories according to predefined criteria and rates. In our thesis, we are able to implement controlled transformations over the trajectories, which allows us to compare the impact of each transformation in every similarity measure that we will use.

The transformation that we will apply on the trajectories will be the following:

I. Transformation of adding stops
II. Transformation of deleting stops
III. Replacement of stops
IV. Replacement of possible matching stops (semantic dimension)
V. Position change of stops

In Chapter 3 and Chapter 4 we will analyze in greater depth the methodology and the tools we used for each kind of transformation that was mentioned.

1.2. Scope and Outline

In our thesis we tried to extend (douglasapeixoto, 2018) work and apply controlled transformations on semantic multiple-aspect trajectories, by changing the information of our original trajectory in space, time and semantic dimension according to the type of transformation which we apply. We evaluate and investigate the effectiveness of our method by finding the similarity of the original trajectory in relation to all transformed trajectories.

The rest of our thesis is organized as follows: Chapter 2 describes the preliminary concepts and the related work for this thesis. Chapter 3 presents our proposed method for the controlled transformation of semantic trajectories, the basic concept of the types of the trajectory transformations and the pseudo-algorithms and the techniques applied so we can achieve all the transformations. Chapter 4 presents the evaluation techniques that we will use and the experiments and the results obtained from it. Chapter 5, presents the conclusion of our thesis and lastly, in Chapter 6 we present the bibliography used for our research.

Chapter 2. Preliminary Concepts and Related Work

In this chapter we present the preliminary concepts and related works for this thesis. This chapter is organized as follows: Section 2.1 presents the preliminary concepts. Section 2.2 presents a review of trajectory similarity measures and their characteristics, which are focusing on raw or semantic trajectories.

2.1. Preliminary Concepts

In this section, we first introduce in Subsection 2.1.1 the preliminary concepts about trajectories and we formalize the operations and notations that will be used frequently in the remainder of the thesis. In addition, we present in Subsection 2.1.2 some distance measures and pruning methods and in Subsection 2.1.3 we look at some basic concepts about similarity measures and some evaluation techniques.

2.1.1. Raw and Semantic Trajectories

A trajectory is a sequence of time-stamped point records describing the motion history of any kind of moving objects, such as people, animals, vehicles etc. However, the continuous location record for a moving object is usually inaccurate or not available and therefore a trajectory is a discrete representation of a moving object, as formalized below.

Definition 1 (Trajectory sample point). A trajectory sample point p is a location in d-dimensional space, and t is the time stamp when p is observed.

Definition 2 (Trajectory). Trajectory is a sequence of trajectory sample points, ordered by time stamps t. Trajectory T Is represented by a sequence of trajectory sample points. Therefore, $T = [p_1, p_2, \ldots, p_n]$.

The main research in the field of trajectory similarity measuring in terms of moving objects are focused to deal either with raw trajectories or semantic trajectories. When an individual is moving, its location is collected along time in the form of sequences of space-time points, called raw trajectories, as formalized in Definition 3.

Definition 3 (Raw Trajectory). A raw trajectory is a time-ordered sequence $T = <p_1 , p_2 , \ldots , p_n>$ of points $p_i = (x_i, y_i, t_i)$, where x, y represents the position of the object in space and t corresponds to the time dimension.

An example of a raw trajectory is illustrated in Figure 1-1: Example of a Raw Trajectory. In this figure, assume we have a trajectory T, where small circles correspond to the sampled points. The spatial coordinates and the time instants can be seen next to the trajectory points. The last point of the trajectory in the figure is located at the coordinates (102, 55) at time instant 7.

However, due to the explosion and the increasing use of social media such as Facebook, Instagram, Twitter, etc., internet channels and the facility to enrich trajectories with more context information as linked open data, there are new approaches that have used background geographic information and social media data to enrich these trajectories with a semantic dimension, transforming raw trajectories into semantic trajectories (Alvares, et al., 2007), (Parent, et al., 2013), (Zhang, Han, Shou, Lu, & Porta, 2014). For the sake of simplicity, in this work we consider semantic trajectory to be a sequence of important places called stops, as originally introduced in (Spaccapietra, et al., 2008). Semantic trajectories are more complex and have more data associated than raw trajectories, because they consider space time and semantics. In addition to space and time, a semantic trajectory has data, such as the type or the place of the visited sites by the moving object, the activities performed at each site etc. (Bogorny, Renso, Aquino, Siqueira, & Alvares, 2014). Several definitions can be found in the literature for a semantic trajectory for the sake of simplicity, semantic trajectories are represented as sequences of stops and moves. Stops are the most important parts of trajectories for most applications, representing the geographic space that an object has visited for limited time, and the moves are the trajectory points between stops, which is an extension of the definition originally introduced in (Spaccapietra, et al., 2008).

Definition 4 (Semantic Trajectory). A semantic trajectory $A = <s_1, m_1, s_2, m_2, \dots, s_k, m_k, s_{k+1}>$ is a sequence of stops and moves, where each stop s_i, has a set of attributes $\{d_{s1}, d_{s2}, \dots, d_{sq}\}$ characterizing it according to q-dimensions, and each move m_j has a set of attributes $\{d_{m1}, d_{m2}, \dots, d_{mr}\}$ characterizing it according to r-dimensions.

Figure 1-2, shows an example of a semantic trajectory, where trajectory *A* has four stops. Note that in Figure 1-2, the trajectory is distributed along with the spatial coordinates, the time interval that the stop occurred, the category of the place and the street where the movement object moves. More attributes can be added to the stops and moves such as the activity of the stop, the transportation means etc., but the basic attributes that a trajectory should have are space, time and semantics.

2.1.2. Distance measures and pruning methods

The last few years, many similarity measures were proposed (LCSS, ERP, EDR, UMS etc.) focusing on raw trajectories, which find the similarity between two trajectories, considering only the spatial or/and temporal dimensions. Most recently, with the trajectory semantic enrichment, emerged the need for similarity measures like MSTP, MSM, SMSM etc., that support space, time and semantics. In addition, there are several methods developed for time-series similarity. Most of these measures can be adapted to work with trajectory data. Euclidean distance was proposed as a distance measure between time series. The most commonly used equal-size discrete sequence-only distance measure is Lp-norm distance. It is a distance measure that pair-wisely computes the distance between the points between trajectories. Some of the similarity measures are distance-based like Dynamic Time Warping (DTW) and other are ε-threshold-based. Some approaches that were proposed based on ε-threshold are Longest Common Subsequence (LCSS), Edit Distance on real Sequence (EDR), Multidimensional Similarity Measure (MSM) etc.

In the last years, many people have brought up many pruning and preprocessing methods (FastMap algorithm, lower bound methods) to accelerate the efficiency of many similarity measures. Most similarity measures are implemented using a dynamic programming approach (DP), that has a quadratic *O(nm)* complexity where *n* and *m* are the sizes of the trajectories. In a DP approach, an all pair-wise point-to-point comparison is performed to determine the exact similarity between two trajectories. However, similarity measures need to deal with huge volume of trajectory data, making it a complex issue to deal with the trajectory data and the proposal of fast and accurate measures has an important role in trajectory data analysis. In (Furtado, Pilla, & Bogorny, 2018) a new strategy was presented, called Fast Trajectory Similarity Measuring (FTSM), which focuses on the reduction in the number of element comparisons required in the similarity computing between trajectories. FTSM instead of using DP approach, adopts an approach that takes advantage of distance properties in Euclidean spaces to reduce the number of pair-wise point comparison required to obtain the matching of each element. The advantage of FTSM over DP approaches relies on its ability to prune unnecessary comparisons to determine the matchings, reducing the number of distance operations in the similarity computation and consequently reducing its computational complexity, and can be applied on many similarity measures such as EDR, LCSS, MSM etc.

2.1.3. Similarity measures and evaluation techniques

In order to compare two trajectories we use a similarity measure. In related works and in this thesis, the similarity measure are based on (Lin, 1998), where two trajectories are more similar as the commonality between each other increases, and they are less similar as their differences increase, as formalized in Definition 5.

Definition 5 (Similarity Measure). A similarity measure on two objects *A* and *B* is a function sim : $A \times B \rightarrow [0, 1]$, such that the objects are more similar when the score returned by sim(*A*,*B*) increases.

In Section 2.2 we present the similarity measures for raw and semantic trajectories. We define some operators and symbols, that will be used throughout the remainder of this thesis in Table 2-1.

Table 2-1: Symbol explanation

Symbol	Explanation
p	*A* trajectory sample point
A, B	Trajectories
m, n	Number of points of trajectory *A* and *B*, respectively
$dist(p_i, p_j)$	Distance between two sample points p_i and p_j
dist(A, B)	Distance between two trajectories *A* and *B*
d_i	The *i*th-dimension of data in a point
ε	Distance threshold between two points matching

2.2. Related Works

2.2.1. < Dynamic Time Warping (DTW) >

A well-known method used to measure the similarity between trajectories is Dynamic Time Warping (DTW), developed for time series in (Berndt & Clifford, 1994). DTW finds the best match between the elements of two sequences, creating a matrix with all possible combinations of two elements in the sequences with the distance between them as the entries. The total distance between two sequences is the sum of the entries of the minimum contiguous path in the matrix. The problem of DTW is that it is sensitive to noise, because it finds at least one match for all elements and then sums the distance values. For example, when a trajectory *A* has a stop that is very distant from all the stops of *B*, even if all the other stops of *A* and *B* are close, the distance will be dominated by the distant stop. A formalization of DTW between two trajectories *A* and *B* with lengths *n* and *m* is defined as in Equation 1

Equation 1

$$DTW(A,B) = \begin{cases} 0 & if\ n = 0\ and\ m = 0 \\ \infty & if\ n = 0\ or\ m = 0 \\ dist(Head(A), Head(B) + & \\ \min\{DTW(Rest(A), B), & \\ DTW(A, Rest(B)), & \\ DTW(Rest(A), Rest(B))\} & otherwise \end{cases}$$

An extension of DTW was proposed that can handle sequence of elements that have more than one dimension, called Multi-Dimensional Dynamic Time Warping (MD-DTW) (Holt; Reinders; Hendriks; 2007). The MD-DTW algorithm takes k-dimensions into account when finding the optimal synchronization between two series. MD-DTW normalizes the distance values in the different dimensions and then creates a matrix with entries at the sum of the distance in all dimensions. Finally, it runs DTW over the matrix and find the minimum contiguous path. The MD-DTW algorithm is presented in Table 2-2. However, MD-DTW has the same limitation of DTW, meaning that they find at least one match for all elements and then sum the distance values, so they are both sensitive to noise.

Table 2-2: The MD-DTW Algorithm (Source: (Holt, Gineke, Reinders, & Hendriks, 2007))

Let A, B be two series of dimension K and

- Normalize each dimension of A and B separately to a zero mean and unit variance
- If desired, smooth each dimension with a Gaussian filter
- Fill the M by N distance matric D according to:

$$D(i,j) = \sum_{k=1} |A(i,k) - B(j,k)|$$

- Use this distance matrix to find the best synchronization with the regular DTW algorithm

The distance measure Dynamic Time Warping adaptive (DTWa**)** proposed in (Shokoohi-Yekta, Hu, Jin, Wang, & Keogh, 2017) extends the classical DTW distance measure to multiple data dimensions. DTWa has two possible approaches (DTW_I, DTW_D) and is based on how the DTW computes the distance between two multidimensional sequences. DTW_I is the summed distances of all dimensions independently measured. In DTW_D the independence of dimensions is no longer allowed, and the distance is computed considering mutual dependence between all dimensions. Then, we must decide which approach is more accurate and export an adaptive distance measure DTW_A, by using a training dataset and performing our classification algorithm and some evaluations. Consider a dataset $D = \{T_1, T_2, \ldots, T_M\}$ a collection of M such time series and Q as a M-dimensional time series, the algorithm is presented below in Table 2-3.

Table 2-3: Adaptive classification algorithm (Source: (Shokoohi-Yekta, Hu, Jin, Wang, & Keogh, 2017))

```
Procedure adaptive_Classifier ($Q$, trainData, threshold)
Input: A time series query, $Q$, the labeled data, trainData, a threshold;
Output: An adaptive distance measure to classify $Q$, $DTW_A$;
```

```
$minD$ ← Nearest_Neighbor_Distance_D (Q, trainData);
$minI$ ← Nearest_Neighbor_Distance_I (Q, trainData);
$S$ ← $minD / minI$;
if $S$ > threshold
    $DTW_A$ ← $DTW_1$ ;
else
    $DTW_A$ ← $DTW_1$ ;
end if
Return $DTW_A$
```

2.2.2. < Longest Common Subsequence (LCSS) >

The Longest Common Subsequence (LCSS) (Vlachos, Gunopoulos, & Kollios, Robust similarity measures for mobile object trajectories, 2012) was proposed for raw trajectory similarity measuring, considering the distance of points in space dimension, introducing a matching threshold when looking for the longest common subsequence between two trajectories. In LCSS, given two trajectories *A* and *B*, and a sample point a_1 and b_1 for each trajectory, they match if the distance between them is less or equal to a given threshold ε. In other words, LCSS find all match pairs (a_i, b_i) where $d(a_i, b_i) \leq \varepsilon$, as we present in Equation 2. Therefore, LCSS value is not a parameter free and its effectiveness highly relies on the value of ε. In addition, LCSS is not a metric distance measure because it doesn't satisfy the triangle inequality. This approach reduces the effects of noise by quantifying the similarity between a pair of elements to binary values: 0 if the elements do not match and 1 otherwise. Then, the longest matching sequence is used to calculate the similarity. The longer the common subsequence of matches between two trajectories, the more similar they are. The algorithm of $LCSS(a_1, b_1)$ is given by Equation 3.

Equation 2

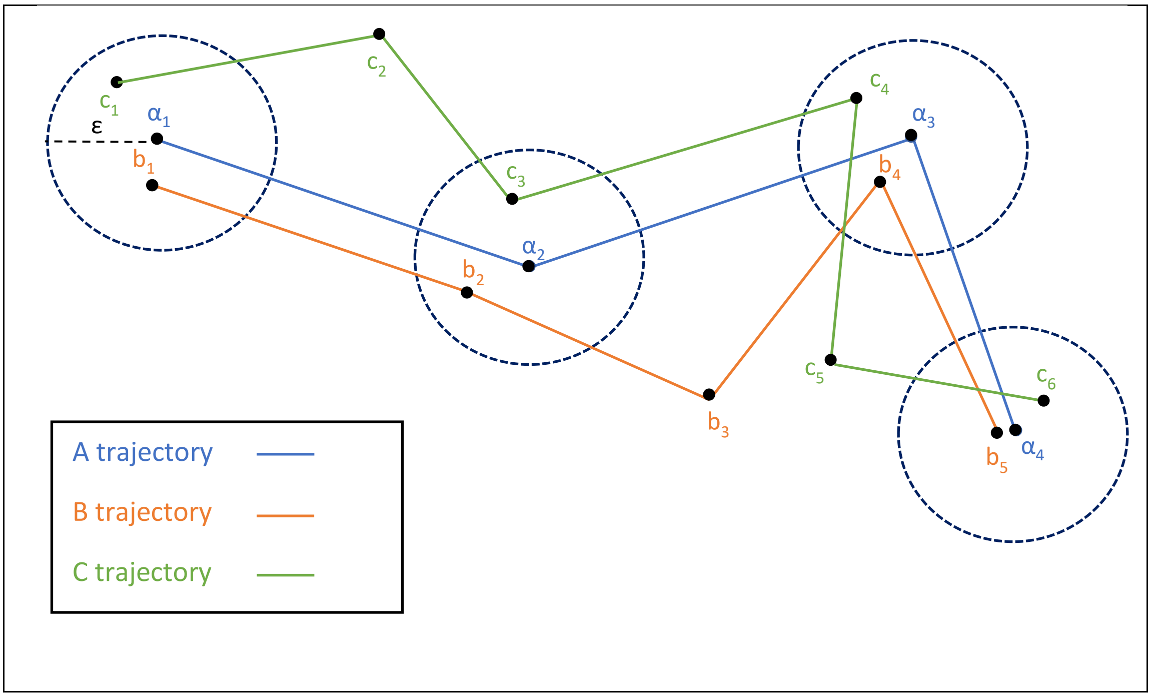
$$match(a,b) = \begin{cases} true & dist(a_{i,x}, b_{i,x}) \leq \varepsilon \\ & and\ dist(a_{i,y}, b_{i,y}) \leq \varepsilon \\ false & otherwise \end{cases}$$

Equation 3

$$LCSS(A,B) = \begin{cases} 0 & if\ n = 0\ and\ m = 0 \\ 1 + LCSS(Rest(A), Rest(B)) & if\ \mathrm{match}(Head(A), Head(B)) \\ \max\{LCSS(Rest(A), B), & otherwise \\ LCSS(A, Rest(B))\} & \end{cases}$$

However, in case where one of the dimensions does not match, the pair of elements is considered as dissimilar. Two main drawbacks of this approach are the need of a match in all dimensions for an element to be considered similar and that it considers only the similar subsequence, ignoring gaps that may vary in size of the sequences, which makes this approach to be inaccurate on many circumstances. For example, in Figure 2-1 given three sequences *A*, *B* and *C* with four, five and six elements, wee distinguish that, four elements of *B* and *C* match with the elements of *A*, since the distance between the points is less than the threshold *ε*. The longest subsequence of matching elements is four in both cases and so the total LCSS similarity of *A* and *B* is the same as the similarity of *A* and *C* even though *B* and *C* have different number of elements that do not match with the elements of sequence *A*.

Figure 2-1: LCSS example for trajectories *A*, *B* and *C*



LCSS measures the similarity of two trajectories and by using the Equation 4 and Equation 5 it can be transformed to LCSS distance. The LCSS distance and the normalized LCSS distance is presented respectively below.

Equation 4

$$dist_{LCSS}\ (A, B) = size(A) + size(B) - 2*LCSS(A, B)$$

Equation 5

$$dist_{LCSS}\ (A, B) = 1 - \frac{LCSS(A,B)}{size(A)+size(B)-2*LCSS(A,B)}$$

Another disadvantage of LCSS is that the value of LCSS relies on the size of compared trajectories. Therefore, when the sampling rate of the trajectories change, the result can be quite different.

2.2.3. < Edit Distance on Real sequence (EDR) >

Edit Distance on Real sequence (EDR) (Chen & Raymond, On the marriage of edit distance and Lp norms, 2004) is an evolution of LCSS, following an approach similar to the one proposed in LCSS. The distance between a pair of elements is quantized to binary values, and a matching threshold is used to reduce the effects of noise. Compared to LCSS, EDR is not only robust to noise, it also assigns penalties according to the sizes of the gaps in between similar shapes, which makes it more accurate. EDR computes the distance of two sequences by adding 0 when the elements match and 1 otherwise. Given a pair of trajectory element vectors a_i and b_j from two trajectories A and B of lengths n and m, respectively, are said to match ($match(a_i, b_j)$ = true) if and only if $dist\left(a_{i,x}, b_{j,x}\right) \leq \varepsilon$ and $dist\left(a_{i,y}, b_{j,y}\right) \leq \varepsilon$, where ε is the matching threshold. EDR uses $subcost(a_1, b_1)$, as follows in Equation 6 to represent the contribution of a_1, b_1 to the value of EDR distance. The algorithm of EDR between two trajectories is defined as Equation 7.

Equation 6

$$subcost(a, b) = \begin{cases} 0 & if\ match(a_1, b_1) = true \\ 1 & otherwise \end{cases}$$

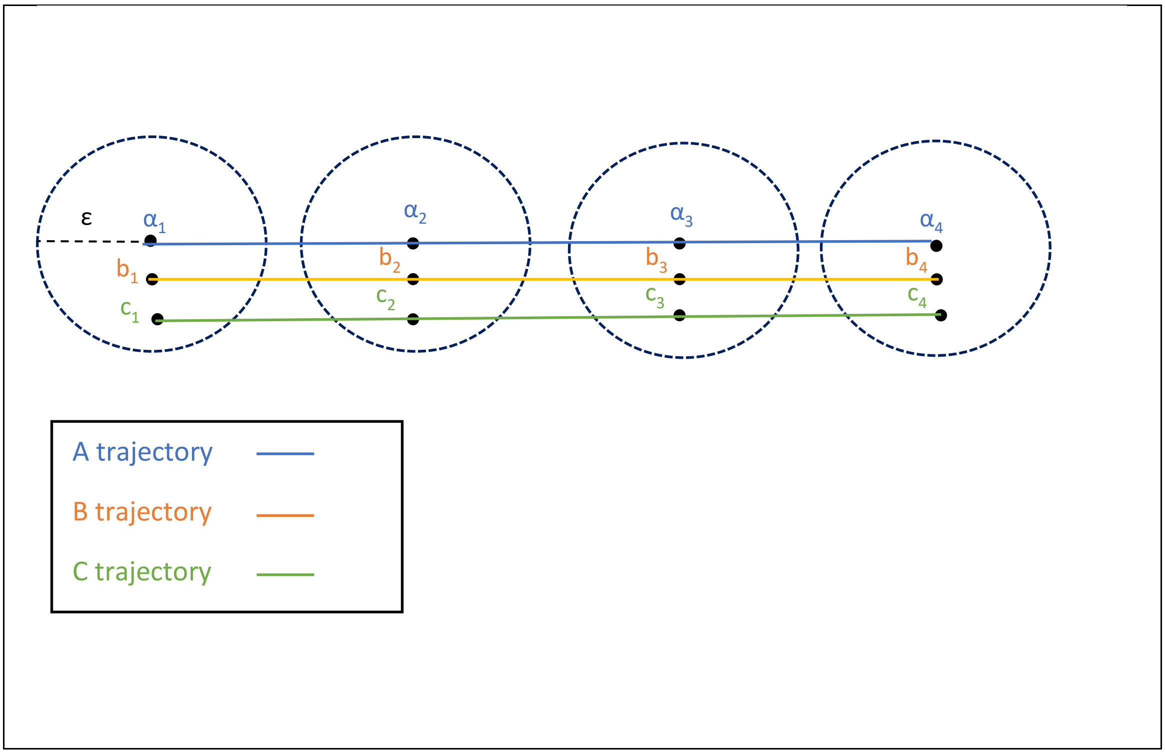
Equation 7

$$EDR(A,B) = \begin{cases} n & if\ m = 0 \\ m & if\ n = 0 \\ \min\{EDR(Rest(A), Rest(B)) + subcost(Head(A), Head(B)), & \\ EDR(Rest(A), B) + 1, EDR(A, Rest(B)) + 1\} & otherwise \end{cases}$$

As we mentioned before, the main drawback of LCSS is that it ignores possible gaps that may vary in size of the sequence, but since this approach increases the distance by 1 when the elements do not match, it solves this problem. Given the previous example that we applied in LCSS in Figure 2-1,the distance between A and B is not the same as the distance between A and C. Therefore, we distinguish that B and C have the same number of elements that matches with the elements of sequence A, but they have different distance score.

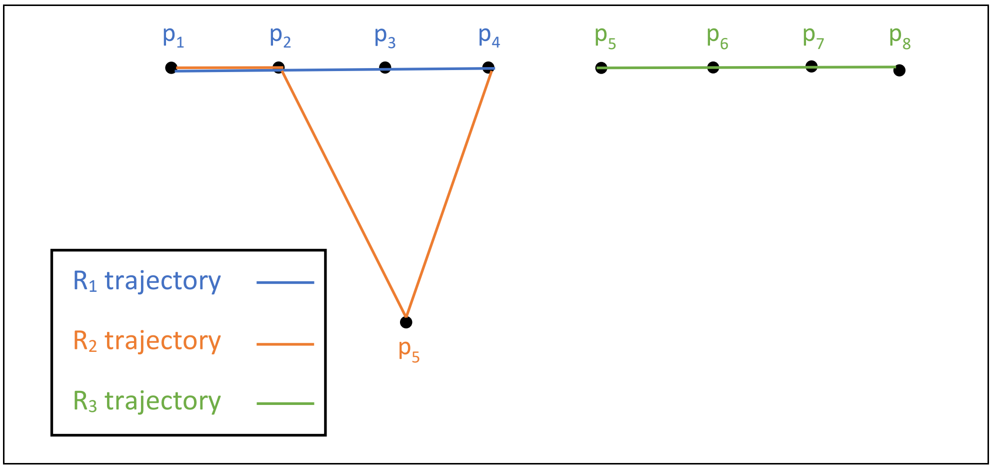
EDR is neither parameter free nor a metric distance measure. Moreover, the limitation of EDR is that the distance value of EDR highly relies on the parameter ε, which it may cause inaccuracy and ineffectiveness in some cases. For example, assume there are three trajectories with the same length A = {a_1, a_2, a_3, a_4}, B = {b_1, b_2, b_3, b_4} and C = {c_1, c_2, c_3, c_4}, as we can see in Figure 2-2. According to the ε that we defined the distance of A and B is EDR(A, B) = 0 same as EDR(A, C) = 0. However, we can clearly see that trajectory A is closer to trajectory B than trajectory C.

Figure 2-2: EDR example for trajectories *A*, *B* and *C*



As we mentioned before LCSS and EDR are robust to noise and solves the problem of DTW of being sensitive to noise. For example, in Figure 2-3, there are three trajectories R_1 = {p_1, p_2, p_3, p_4}, R_2 = {p_1, p_2, p_5, p_4} and R_3 = {p_6, p_7, p_8, p_9}. Assume that R_1 is the trajectory we will compare with the other two trajectories. Obviously, p_5 is the distant point, since it is far away from all the other points. Based on DTW measure the trajectory R_3 is the most similar trajectory to R_1, even though R_2 have 3 points that match with the points of R_1. This is because, DTW is highly effected by the noisy point p_5, making it inaccurate. From the Figure 2-3 we can clearly distinguish that the most similar trajectory to R_1 is the trajectory R_2. Since, LCSS and EDR is robust to noise and is not affected by distant points, they both rank first R_2 in terms to similarity to R_1.

Figure 2-3: Comparison of DTW, LCSS and EDR approaches (Source: *(Su, Liu, Zheng, Zhou, & Zheng, 2020)*)



LCSS and EDR have not been proposed for semantic trajectories, but both measures can be easily extended to handle other dimensions (e.g. semantics). However, both measures demand that all trajectory elements should be homogenous and as a result they can’t always represent semantic trajectories as a sequence of heterogeneous elements. Moreover, in the case we want to consider both stops and moves, the proposed measures are not valid.

2.2.4. < Edit Distance with Real Penalty (ERP) >

Edit Distance with Real Penalty (ERP) (Chen & Raymond, On the marriage of edit distance and Lp norms, 2004) is a distance function proposed for time-series. ERP is a distance measure for trajectories that can be seen as a combination of L_p-norm and edit distance. ERP differs from EDR in avoiding the δ tolerance and it computes the distance between two sequences of points by aligning the sequences, allowing possible gaps in the sequence when there are points that do not match. ERP uses real penalty between two non-gap elements, but a constant value for computing the distance for gaps. Specifically, given two time series *R* and *S* with length m and n, and the elements r_i, s_i, q_i, ERP uses edit distance to get match pairs (r_i, s_i) and then calculates the L_1-norm distance between the elements for every sample point from q_i to a constant. The distance formula between the sample points used by ERP is the following Equation 8:

Equation 8

$$dist_{ERP}(r_i,\ s_i) = \begin{cases} |r_i - s_i| & if\ \ r_i,\ s_i\ not\ gaps \\ |r_i - g| & if\ \ s_i\ is\ a\ gap \\ |s_i - g| & if\ r_i\ is\ a\ gap \end{cases}$$

where g a constant that the user define its value. In this work an appropriate value of g is any value that satisfies the triangle inequality, but it is suggested to pick $g = 0$. The ERP distance between R and S is defined in Equation 9.

Equation 9

$$ERP(R,S) = \begin{cases} \Sigma_1^n |s_i - g| & if\ \ m = 0 \\ \Sigma_1^n |r_i - g| & if\ \ n = 0 \\ \min\{ERP\big(Rest(R), Rest(S)\big) +\ dist_{ERP}(Head(R), Head(S)), & \\ ERP(Rest(R), S) + dist_{ERP}(Head(R), g), & \\ ERP(R, Rest(S)) + dist_{ERP}(Head(S), g)\} & otherwise \end{cases}$$

Furthermore, because of the use the parameter g, ERP is not parameter free method.

2.2.5. < Mining User Similarity from Semantic Trajectories >

In (Ying, Lu, Lee, Weng, & Tseng, 2010), they proposed a novel approach for recommending potential friends based on users semantic trajectories of mobile users. The raw trajectories become semantic trajectories though stay cells. A stay cell represents a place where the user made a stop such as school, gym, restaurant, etc., as we can see on Figure 2-4. .In addition, in the following table (2.4) there are all the information of the semantic trajectories. Based on the Figure 2-4 we have two trajectories, where the subsequence of the pattern are for trajectory *P* = <{School}, {Park}, {Park, Work}, {Coffee}, {Restaurant}> and for trajectory *Q* = <{School}, {Park}, {Gym}, {Coffee}, {Restaurant}>.

Figure 2-4: Example of MSTP for trajectories *P* and *Q* (Source: (Ying, Lu, Lee, Weng, & Tseng, 2010))

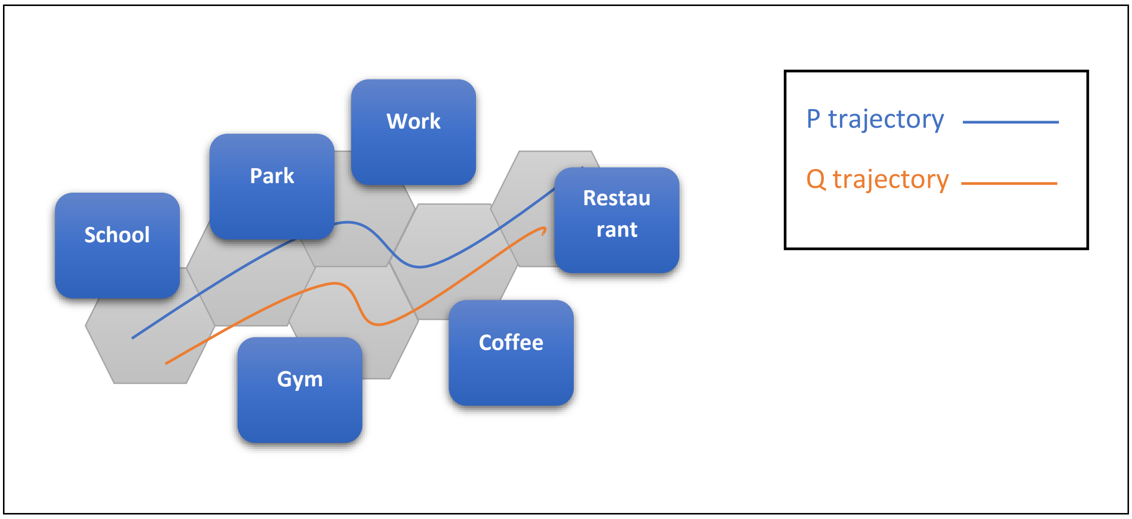


Table 2-4: Moving patterns of trajectories P and Q

Trajectory	Semantic Trajectory
Trajectory *P*	<{School}, {Park}, {Park, Work}, {Coffee}, {Restaurant}>
Trajectory *Q*	<{School}, {Park}, {Gym}, {Coffee}, {Restaurant}>

The core of this framework is a novel similarity measurement, called Maximal Semantic Trajectory Pattern Similarity (MSTP-Similarity), for measuring the similarity between two semantic trajectories based on the stay cells of each trajectory. The more common parts the trajectories have the more similar they are. As a result, it uses the Longest Common Sequence (LCS) between two semantic trajectories to represent their longest common part. The difference from LCSS approach is that, MSTP defines a ratio between each trajectory of the common part to a pattern *P* as follows on Equation 10.

Equation 10

$$ratio(LCS(P, Q), P) = \frac{\Sigma_{i=1}^{|P|}\Sigma_{i=1}^{|LCS(P,Q)|} M(P_i, LCS_j)}{|P|},$$

where

Equation 11

$$M(P_i, LCS_j) = \begin{cases} \frac{|P_i \cap LCS_j|}{|P_i|} & if\ \ LCS_j\ is\ matching\ to\ P_i \\ 0 & otherwise \end{cases}$$

Then, it computes the similarity score of two patterns, by averaging the ratios of the common part to them, avoiding the drawback of LCSS that does not differentiate matching gaps of various sizes. Given P and Q, there are two approaches that calculates the similarity of two patterns (Equal Average (EA) and Weighted Average (WA)), as shown in Equation 11 and Equation 12.

Equation 12

$$MSTP - Similarity_{EA}(P, Q) = \frac{ratio(LCS(P,Q),P) + ratio(LCS(P,Q),Q)}{2}$$

Equation 13

$$MSTP - Similarity_{WA}(P, Q) = \frac{|P| * ratio(LCS(P,Q),P) + |Q| * ratio(LCS(P,Q),Q)}{|P| + |Q|}$$

However, MSTP has huge limitations due to the fact that it’s a measure focusing only on the semantic information of a trajectory, so it can’t handle multiple dimensions.

2.2.6. < Similarity Measurement of Moving Object Trajectories>

The work of (Liu & Schneider, 2012) proposed a novel approach to measure the similarity between trajectories that are focusing on two aspects, the geographic and semantic similarity. Firstly, it splits a trajectory into sub-trajectories, by using a speed ratio to identify their movement patterns. Then, it defines the similarity measurements in geometry, which introduces three concepts we must take into consideration: i) how close in distance are the centroids of the trajectories, ii) the difference of the lengths between trajectories iii) the cosine similarity between the directions of two sub-trajectories. After that, it defines the semantic similarity between trajectories, which is symmetric, by adopting the longest common subsequence algorithm, introduced in LCSS. This approach have many limitations since it’s not considering the time dimension and it demands spatial matching in order to consider similar two trajectories.

2.2.7. < Multidimensional Similarity Measuring for Semantic Trajectories (MSM) >

Multidimensional Similarity Measure (MSM) was proposed in (Furtado, Kopanaki, Alvares, & Bogorny, 2016) which measures the similarity of semantic trajectories or multidimensional sequences. MSM is a multidimensional similarity measure for sequences, which overcomes various drawbacks of the aforementioned approaches when dealing with semantic trajectories, such as the sensitivity to noise, tolerance for possible gaps with different size and the prevalence of the worst dimension similarity when elements of the sequence do not match in all dimensions. In this approach, in order to determine if two elements match in a dimension, each dimension have its own distance function and threshold, as can be seen in Equation 14. For example, two elements can match in one dimension unlike other dimensions that they may not match.

Equation 14

$$match_k(a,b) = \begin{cases} 1 & if\ dist_k(a,b) \leq maxDist_k \\ 0 & otherwise \end{cases}$$

MSM in order to compute the matching score, it considers the dimensions separately and therefore has the ability to give different importance in each dimension D_k. This is accomplished by proposing a pre-defined importance weight w_d that corresponds to the weight of each dimension. The matching score between the elements is computed in Equation 15, where MSM sums the matching value for all D_k dimensions and multiplies it by its weight w_d.

Equation 15

$$score(a, b) = \sum_{k=1}^{|D|}(match_k(a,b) * w_k)$$

To compute the similarity score between two trajectories, MSM tries to find only the best matching score of each element *a* in relation to *B*. For that reason, it calculates the parity between them, as follows in Equation 16, which is the sum of the highest score of all stops a of the trajectory *A*, when compared with all the stop of trajectory *B*.

Equation 16

$$parity(A, B) = \sum_{a \in A} \max\{score(a,b): b \in B\}$$

Finally, MSM calculates the multidimensional similarity measure *MSM(A, B)* between the trajectories *A* and *B* by averaging the parity values of *A* with *B* and of *B* with *A*, as presented in Equation 17.

Equation 17

$$MSM(A, B) = \begin{cases} 0 & if\ |A| = 0\ \vee |B| = 0 \\ \frac{parity(A,B)+parity(B,A)}{|A|+|B|} & otherwise \end{cases}$$

The comparison of two semantic trajectories *A* and *B* with the proposed similarity measure MSM can be seen in Figure 2-5. In this figure, where A = {a_1, …,a_4} and *B = {b_1, …, b_5}* are two semantic trajectories, we have three dimensions for each element of *A* and *B*: Space *(D_1)* – Time *(D_2)* and Semantics *(D_3)*. Specifically, the two semantic trajectories are presented below in Table 2-5. MSM calculates matching score for all dimensions between all pairs of elements of *A* and *B* (with the use of Equation 14 and Equation 15), by using an appropriate distance function and threshold for each dimension. Then, it computes the parity (Equation 16), by summing the best matching score of each stop of trajectory *A* in relation *B* and of each stop of trajectory *B* in relation to *A*. Finally, it calculates the similarity score of *A* and *B* by applying the function describes in Equation 17.

Figure 2-5: MSM example for trajectories A (blue) and B (red)

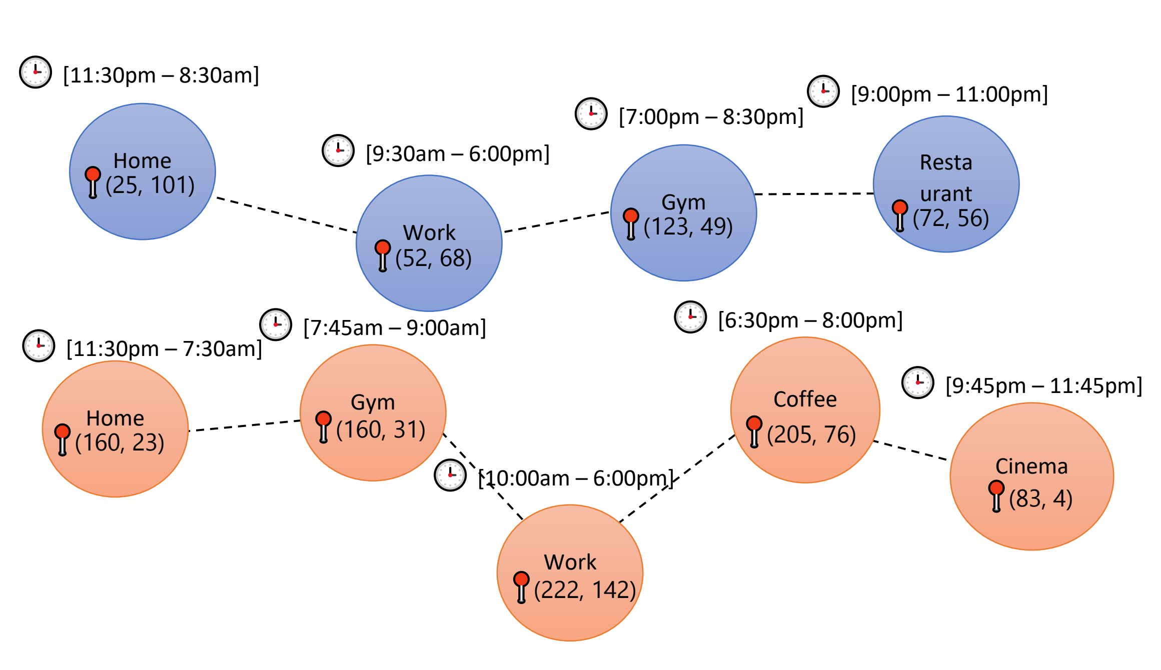


Table 2-5: Information for trajectories *A* and *B*

Trajectory *A*	{((25, 101), [11:30pm – 8:30am], Home), ((52, 68), [9:30am – 6:00pm], Work), ((123, 49), [7:00pm – 8:30pm], Gym)), ((72, 56), [9:00pm – 11:00pm], Restaurant)}
Trajectory *B*	{((160, 23), [11:30pm – 7:30am], Home), ((160, 31), [7:45am – 9:00am], Gym), ((222, 142), [10:00am – 6:00pm], Work)), ((205, 76), [6:30pm – 8:00pm], Coffee), ((83, 4), [9:45pm – 11:45pm], Cinema)}

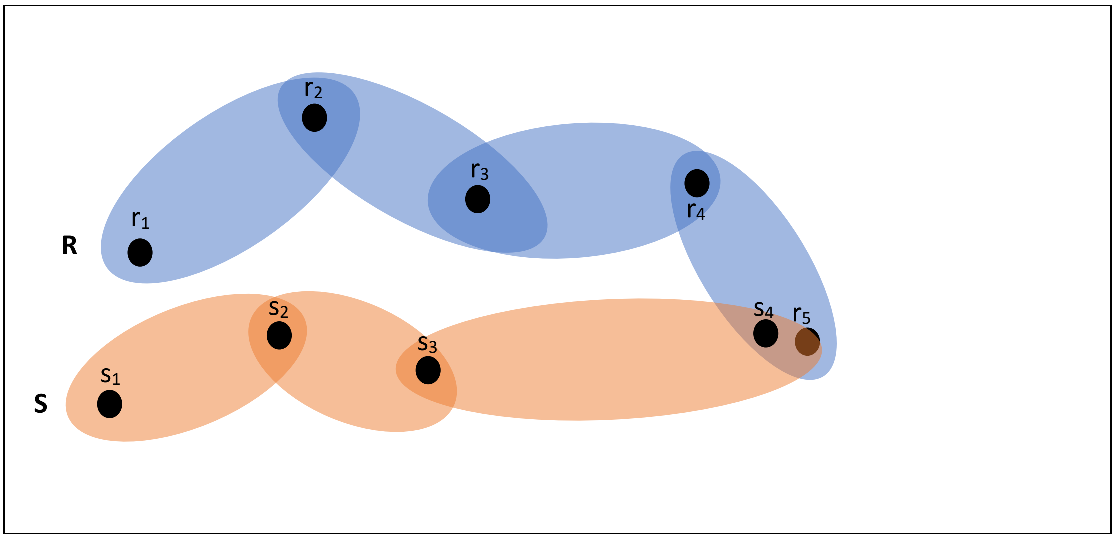
To sum up, this approach considers separately each dimension, such as space, time and semantics and it supports the definition of individual importance weights for each dimension. MSM is more robust and effective than LCSS and EDR in the domain of semantic trajectories, by allowing partial dimension matching and many-to-many elements matching, and by ignoring the order of the stops. However, the sequence of the stops may play a decisive role on some applications, resulting in decreasing the accuracy in trajectory similarity analysis. Furthermore, MSM just like in MSTP ignores the moves between the stops, as it was developed to consider only the stops.

2.2.8. < Unveiling Movement Uncertainty for Robust Trajectory Similarity Analysis (UMS) >

UMS (Uncertain Movement Similarity), a parameter-free trajectory similarity measure for raw trajectories, was proposed in (Furtado, Alvares, Pelekis , Theodoridis, & Bogorny, 2018). UMS is exclusively focusing on the spatial dimension that covers the gaps between trajectory sampled points, where two moving objects considered similar if they share a similar path in space. The main difference of UMS with related similarity measures that were proposed, is that in order to improve the accuracy in trajectory similarity analysis, it uses an elliptical representation of trajectory to compute the distance between two trajectories. This approach doesn’t need a distance threshold or linear interpolation, as the ellipses are dynamically defined according to the distance between two consecutive trajectory points.

An example is illustrated in Figure 2-6, where the trajectories $R = \{r_1, r_2, r_3, r_4, r_5\}$ and $S = \{s_1, s_2, s_3, s_4\}$ are represented as two elliptical trajectories E(R) (Blue) and E(S) (Red) according to UMS, but the shape of trajectories is different. The similarity score of UMS is based on three premises: alikeness, shareness and continuity.

Figure 2-6: Movement ellipses for trajectories *R* and *S*



Alikeness represents how similar are two elliptical trajectories based on their shapes in space, i.e. a high alikeness score indicates that the trajectories have similar shapes. Given two trajectories R and S, a trajectory point $r \in R$ and an elliptical trajectory E(S), the alikeness is computed, as follows on Equation 18.

Equation 18

$$A(R, S) = \frac{\sum_{r \in R} match(r, E(S))}{length(R)} * \frac{\sum_{s \in S} match(s, E(R))}{length(S)},$$

where

Equation 19

$$match(r, E(S)) = \begin{cases} 1 & if\ \ \exists e' \in E(s) | within(r, e') \\ 0 & otherwise \end{cases}$$

Shareness answers the question of how much space covered by the two movement ellipses, share in a common area. Given two trajectories R and S, a trajectory point $r \in R$ and an elliptical trajectory E(S), the shareness is computed, as follows on Equation 20.

Equation 20

$$S(R, S) = \frac{1}{2}\left(\frac{\sum_{r \in R} share(r, E(S))}{length(R)} + \frac{\sum_{s \in S} share(s, E(R))}{length(S)}\right),$$

where

$share(r, E(S)) = 1 - \min_{e' \in E(s)} d_{pnd}(r,\ e')$, and d_{pnd} is the reference point normalized distance.

The third premise is continuity***:*** the movements ellipses order represents individuals traveling in the same direction. Given two trajectories R and S, a trajectory point $r \in R$ and an elliptical trajectory E(S), let U = <first(r_1, E(S)), …, first(r_{n-1}, E(S)), first(r_n, E(S))> and V = *<first(s_1, E(R)), …, first(s_{m-1}, E(R)), first(s_n, E(R))>* be two sequences with the first matching positions of all elements $r \in R$ and $s \in S$, the continuity is computed as follows on Equation 21.

Equation 21

$$C(R, S) = \frac{\sum_{0<i\leq|U|} valid(\mathrm{u_i})}{length(R)} * \frac{\sum_{0<j\leq|V|} valid(\mathrm{v_i})}{length(S)},$$

where

Equation 22

$$valid(u_k) = \begin{cases} 1 & if\ \ (k = 1 \vee \mathrm{u_k} \neq -1) \cup (k > 1\ \mathrm{u_k} \neq -1 \wedge \mathrm{u_k} \geq \mathrm{u_{k-1}}) \\ 0 & otherwise \end{cases}$$

Finally, the similarity score of two trajectories is computed as follows on Equation 23.

Equation 23

$$UMS(R, S) = \frac{(A(R,S)+S(R,S))}{2} * C(R, S)$$

UMS is more robust and precise than related similarity measures to the movement uncertainty, as it solves the problem of variations in the sampling rate caused by the sampling rate and the heterogeneity of this kind of data, but it can’t handle trajectories with higher sample rate, because the movement ellipses will be smaller as the sampling rate grows, making lower the shareness value. In addition, UMS can’t handle multiple dimensions, since it considers only the spatial dimension.

2.2.9. < MUITAS: Towards semantic-aware multiple-aspect trajectory similarity measuring >

A new similarity measure for big trajectory data that involves multiple semantic dimensions was proposed in (Petry, Ferrero, Alvares, Renso, & Bogorny, 2019), called Multiple-aspect Trajectory Similarity (MUITAS). MUITAS is flexible enough to consider both dependent and independent attributes and therefore taking into account the semantic relationship between attributes. In addition, this approach can handle each attribute differently, by allowing the definition of weight and giving the importance needed to each attribute. MUITAS, introduces new terms, called aspect and multiple-aspect trajectory. Aspect is any sort of information annotated to the trajectory such as the weather, the transportation means, etc.

Firstly, we must define an application $\mathcal{A}$ = *(A, D, Δ, F, W)*, where $A = \{a_1, a_2, ..., a_n\}$ is a non-empty set of attributes, $Δ=\{dist_1,dist_2,...,dist_l\}$ is a non-empty set of distance functions, $Δ=\{δ_1,δ_2,...,δ_l\}$ is a none empty set of distance thresholds, $F =\{f_1,f_2,...,f_k\}$ is a non-empty set of features, and $W =\{w_1,w_2,...,w_k\}$ is a non-empty set of weights. $dist_i$ and $δ_i$ are the distance function and threshold of attribute a_i. For each feature $fi ∈F$, we define a corresponding weight $w_i ∈W$ such that $\sum_{i=1}^{|F|} w_i = 1$. Note that, not all applications have the same features, meaning that they may have different distance functions and/or different thresholds. Then, given two trajectories $p ∈P$ and $q ∈Q$ and an application $\mathcal{A}$ = *(A, D, Δ, F, W)*, to compute the similarity for two multiple-aspect trajectories we define a function to calculate the matching score between the *p* and *q*, as follows in Equation 24.

Equation 24

$$score(p, q) = \sum_{k=1}^{|F|}(match_{fi}(p,q) * w_i$$

where

Equation 25

$$match_{fi}(p,q) = \begin{cases} 1 & if\ \forall a_j \in f_i, dist_j(p,q) \leq \delta_i \\ 0 & otherwise \end{cases}$$

In order to proceed forward and propose the multiple-aspect trajectory similarity measure MUITAS we will use a parity function defined by MSM. It calculates the parity between them, as follows in Equation 26, which is the sum of the scores of the best matches of the points of trajectory *P*, when compared with all the points of trajectory *Q*.

Equation 26

$$parity(P, Q) = \sum_{p \in P} \max\{score(p,q) \colon q \in Q\}$$

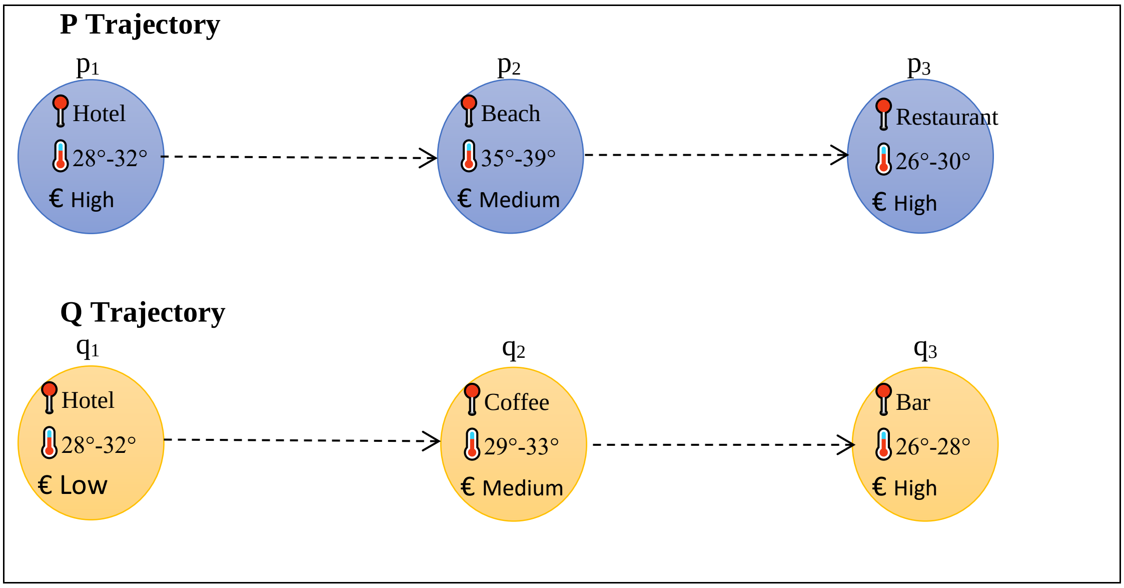
Finally, the similarity of two multiple-aspect trajectories *P* and *Q*, computed by MUITAS, is given by the average parity of *P* and *Q*, in the following Equation 27.

Equation 27

$$MUITAS(P,Q) = \begin{cases} 0 & if\ |P| = 0\ \vee |Q| = 0 \\ \frac{parity(P,Q)+parity(Q,P)}{|P|+|Q|} & otherwise \end{cases}$$

Let us consider the example shown in Figure 2-7, with trajectories *P* and *Q*, where each trajectory has three attributes: the category of place visited, the temperature and the price range of the place (low, medium, high). In this example, trajectory *P* and *Q* visit the same place (hotel) on the first POI, with the same temperature, but different price range. After that, they visit different category of places with slightly different temperature, but with the same price range.

Figure 2-7: Example of MUITAS for trajectories *P* and *Q*



Although MUITAS focused on multiple-aspect trajectories, the proposed similarity can be applied to any type of trajectory and adjust to different applications and scenarios. Furthermore, it overcomes the limitation of state-of-art methods, by allowing partial attribute dependence, being robust to noise and by using different distance function for different attributes and weighting them. However, due to the fact that it needs to use huge amount of heterogeneous data attributes, it makes the trajectory similarity more complex than similar existing methods.

2.2.10. < SMSM: a similarity measure for trajectory stops and moves >

In the work of (Lehmann, Alvares, & Bogorny, 2019) Stops and Moves Similarity Measure (SMSM) was proposed, a similarity measure for semantic trajectories. SMSM is an extension of MSM, which takes into account both stops and moves, considers the sequence of the stops, allows different semantics for the moves and by supporting the definition of weights for stops, moves and dimensions, it provides more or less importance for each part of trajectory. SMSM is the first similarity measure to consider both stops and moves of semantic trajectories and overcomes the limitation of MSM.

A move always start and end with in a stop and can be characterized by different attributes. SMSM introduces a new concept, described as movement element *e = <stopS, move stopE>*, which is the move between two consecutive stop, stopS and stopE. A semantic trajectory will be considered as a sequence of movement elements, as follows: *ST = <*$e_1 = (s_1, m_1, s_2), \ldots, e_n = (s_n, m_n, s_{n+1})$*>*. Given two semantic trajectories *A* and *B*, the similarity of a movement element of trajectory *A* with another movement element of trajectory *B* is divided in two parts, their stops and their moves. To determine if two elements match we define the match function, presented in Equation 28, where it returns 1 if the distance of two movement elements is less than the threshold, and 0 otherwise.

Equation 28

$$match_k(a,b) = \begin{cases} 1 & if\ dist_k(a,b) \leq maxDist_k \\ 0 & otherwise \end{cases}$$

SMSM in order to compute the total score for two movement elements *a* and *b*, it considers the stops and moves separately, therefore it has the ability to give different importance in each of them, depending on the needs of the application. This is accomplished by setting w_{stop} and w_{move}, the weights of the stops and the moves, respectively. The total score for movement elements is computed in Equation 29.

Equation 29

$$score(a, b) = scoreStop(a, b) * w_{stop} + scoreMove(a, b) * w_{move}$$

The functions *scoreStop(a, b)* and *scoreMove(a, b)* are defined in **Equation *30*** and **Equation *31*** respectively, where *r* and *q* are the number of dimensions of stops and moves. This is accomplished by proposing a pre-defined importance weight w_d that corresponds to the weight of each dimension. The score of the stops, computed in Equation 30, is given by the weighted sum of the matching values for all *D_k* dimensions of the start and end stops of two movement elements. The scoreMove highly depends on the function *matchStops(a, b)*.

Equation 30

$$scoreStop(a, b) = \sum_{k=1}^{|r|} (match_k(a_{stopS}, b_{stopS}) + match_k(a_{stopE}, b_{stopE})) \div 2 * w_k$$

Equation 31

$$\text{scoreMove(a, b)} = \begin{cases} \sum_{k=1}^{|q|} (match_k(a_{move}, b_{move}) * w_k & if\ matchStops(a, b) \\ 0 & otherwise \end{cases}$$

To compute the similarity score between two trajectories, SMSM aims at finding only the best matching score of each element *a* in relation to *B*. For that reason, it calculates the parity between them, as follows in Equation 32, which is the sum of the highest score of all stops of the trajectory *A*, when compared with all the stop of trajectory *B*.

Equation 32

$$parity(A, B) = \sum_{a \in A} \max \{score(a, b) \colon b \in B\}$$

Finally, SMSM calculates the stops and moves similarity measure *SMSM(A, B)* between the trajectories *A* and *B* by averaging the parity values of *A* with *B* and of *B* with *A*, over the sum of the number of elements in *A* and the number of elements in *B*, as presented in Equation 33.

Equation 33

$$SMSM(A, B) = \begin{cases} 0 & if\ |A| = 0 \lor |B| = 0 \\ \frac{parity(A,B) + parity(B,A)}{|A| + |B|} & otherwise \end{cases}$$

Let us consider the trajectory shown in Figure 1.2, where trajectory *A* represents the daily routine of a man. Considering the notation stop name (*(x, y)*, [start timestamp - end timestamp]), the man has the following movement behavior: stays at Home ((25, 101), [11:30pm – 8:30am]), then he goes to work via Main street ((52, 68), [9:30am – 6:00pm]), and from there goes to Gym via Stanford street ((123, 49), [7:00pm – 8:30pm]), finishing the day moving via Charles street to the Restaurant ((72, 56), [9:00pm – 11:00pm]).

In conclusion, the main contributions of this approach are that it considers the order between stops, it deals with all dimensions (space, time and semantic), it doesn't ignore the moves between stops and it allows partial dimension matching by not forcing a sequence. For all the aforementioned reasons, SMSM is more robust and flexible than similar measures (LCSS, EDR, MSM, etc.) developed for raw or semantic trajectories.

2.2.11. < Simulating our LifeSteps by Example>

In (Pelekis, Sideridis, Tampakis, & Theodoridis, 2016) was proposed SemT-OPTICS for semantic trajectory clustering, extending the well-known T-OPTICS (Nanni & Pedreschi, 2006) clustering algorithm that was originally designed for raw trajectories. SemT-OPTICS relies on an effective spatio-temporal-textual similarity function over semantic trajectories. The main idea behind this method is to measure the similarity between two timelines and transmit the information to an effective clustering algorithm, so as to divide an SMD (a semantic mobility timeline consisting of a set of timelines) into clusters that contain similar timelines according to a distance measure. However, global timeline clustering may sometimes result in a misleading result, and therefore suggested a novel distance metric that leads into a clustering algorithm (SemT-OPTICS) which can be applied to both timelines and LifeSteps (Definition 6) by selecting the appropriate metric. Mobility timeline is a sequence of LifeSteps and each LifeStep can be abstracted as a pair of values *(θ, κ)*, where θ is a spatio-temporal value that provides an approximation of a portion of the movement of the user, and κ provides a corresponding textual description giving semantics to θ.

Definition 6: (LifeStep). Given a road network G (V, E) (V is a set of vertices, E is a set of edges), a LifeStep ls corresponds to a (raw) sub-trajectory τ' of a moving object, which is valid in G, and is defined as a tuple <ls-id, ls-flag, MBB (Minimum Bounding Box), tags, T-link>, where ls-id is the LifeStep identifier, ls-flag is a flag taking values from set {'Move', 'Stop'}, MBB is a tuple <MBR (Minimum Bounding Rectangle), [tstart, tend]> corresponding to the 3D approximation of τ', with MBR ([tstart, tend]) being the 2D enclosing rectangle of the spatial projection (the 1D interval of the temporal projection, respectively) of τ' in 2D plane (1D timeline, respectively), tags is a set of keywords describing the corresponding activities and semantic annotations related to this portion of movement, and T-link is a link to τ'.

Below we give the definition of the distance between two LifeSteps D_{LS}, which must be set in such a way that leads to an intuitive measure for all possible pairs of LifeSteps types. Note that Stop and Move LifeSteps may have very different sizes and for that reason the defined function should takes this into account in order to be effective.

Definition 7: (D$_{LS}$). Given two LifeSteps *ls_i* and *ls_j*, their distance *D_{LS} (ls_i, ls_j)* is defined by using the following monotone, ranking function with respect to distance proximity of their MBB$_S$ dist$_\theta$, and text relevancy of their sets of keywords dist$_k$:

Equation 34

$$D_{LS}\left(ls_i, ls_j\right) = \lambda * dist_\theta\left(ls_i, ls_j\right) + (1-\lambda) * \ dist_k\left(ls_i, ls_j\right),$$

where

Equation 35

$$dist_\theta\left(ls_i, ls_j\right) = \Sigma_{d\epsilon\{x,y,t\}} w_d * \left(\frac{mbb_d\left(ls_i \cup ls_j\right) - mbb_d\left(ls_i \cap ls_j\right)}{maxdist_d(SMD)}\right)$$

and

Equation 36

$$dist_\kappa\left(ls_i, ls_j\right) = 1 - \left(\frac{\kappa\left(ls_i\right)*\kappa\left(ls_j\right)}{\left|\left|\kappa\left(ls_i\right)\right|\right|^2 + \left|\left|\kappa\left(ls_j\right)\right|\right|^2 - \kappa\left(ls_i\right)*\kappa\left(ls_j\right)}\right)$$

where the textual distance $dist_\kappa$ is measured by Jaccard distance and w_d is used to weight each the three dimensions composing the spatio-temporal component, while *$\lambda \in [0,1]$* is used to tune the relative importance between the two components. In addition, $maxdist_d(SMD)$ works as a normalization factor.

To determine the function that measures the distance between two mobility timelines (D$_{MT}$), which is a metric, they proposed a suitable modification of the Edit distance with Real Penalty (ERP) (Chen & Raymond, On the marriage of edit distance and Lp norms, 2004). Furthermore, D$_{LS}$ is used in order to measure distance D$_{MT}$ and is defined as follows:

Definition 8: The distance D$_{MT}$ between two mobility timelines *mt_i* and *mt_j* of arbitrary length, is given by:

Equation 37

$$D_{MT}\left(mt_i, mt_j\right) = min \begin{cases} D_{MT}\left(R(mt_i), R\left(mt_j\right)\right) + \ D_{LS}\left(ls_{i,1} - ls_{j,1}\right), \\ D_{MT}\left(R(mt_i), R\left(mt_j\right)\right) + \ D_{LS}\left(ls_{i,1} - gap\right), \\ D_{MT}\left(R(mt_i), R\left(mt_j\right)\right) + \ D_{LS}\left(gap - ls_{j,1}\right) \end{cases}$$

$R(mt_i)$ indicates the LifeSteps that remained after we removed the first LifeStep of the i-th timeline ls_i. The value of the gap is similarly determined as defined in (Chen, Özsu, & Oria, Robust and fast similarity search for moving object trajectories, 2005) and usually its value is gap = 0, since it's the first value of the time scale for the time series.

2.2.12. Similarity measures characteristics

In Table 2-6 we summarize some of the characteristics of the most related measures discussed in this thesis. We compare all measures considering the robustness to noise, if the measure uses different distance functions, if the measure compares all pairs of elements (pair-wise similarity), if the measure uses matching threshold, if the measure is able to handle multiple dimensions (space, time and semantics), if the measure takes into account the sequence of the points, if the measure takes into account the stops and the moves of the trajectory, the use of weights for the dimensions and the allowance of partial dimension matching and if it supports multiple-aspect trajectories. A similar table was firstly created in (Lehmann, Alvares, & Bogorny, 2019) where they compared the main characteristics of most related similarity measures in relation to the similarity measure they proposed.

Table 2-6: Similarity measures characteristics (Source: (Lehmann, Alvares, & Bogorny, 2019))

	DTW	LCSS	EDR	ERP	MSTP	Liu	MSM	UMS	MUITAS	SMSM
Robust to noise		✓	✓		✓	✓	✓	✓	✓	✓
Different distance function							✓		✓	✓
Trajectory gaps			✓	✓	✓	✓	✓	✓	✓	✓
Pair-wise similarity	✓	✓	✓	✓		✓	✓		✓	✓
Matching threshold		✓	✓			✓	✓		✓	✓
Space dimension	✓	✓	✓	✓		✓	✓	✓	✓	✓
Time dimension		✓	✓	✓			✓		✓	✓
Semantic dimension					✓	✓	✓		✓	✓
Full sequence	✓								✓	
Partial sequence		✓	✓	✓	✓	✓		✓		✓

No sequence							✓			
Support stops					✓	✓	✓		✓	✓
Support moves										✓
Dimension weighting							✓		✓	✓
Partial matching							✓		✓	✓
Multiple-aspect trajectories									✓	

Chapter 3. A method for the controlled transformation of semantic trajectories

This chapter presents in Section 3.1, the different types of transformations applied to the trajectories, and in Section 3.2 we describe the pseudo-algorithms and the techniques that we will apply in order to transform the semantic trajectories.

3.1. Trajectory Transformations

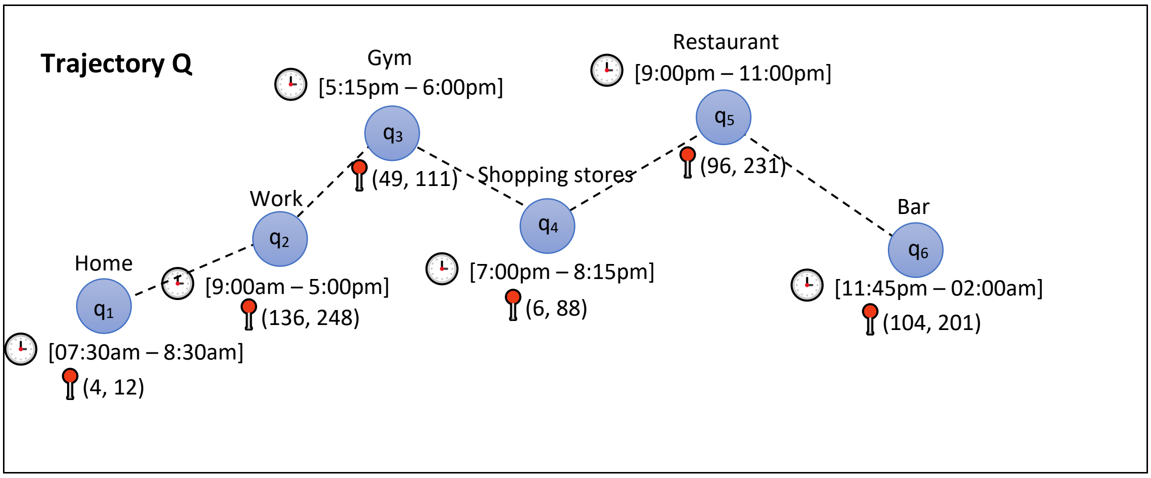
Since the similarity measures are not easy to compare, we can't distinguish easily which method is the best. In order to compare the similarity measures we will conduct an objective experimental evaluation, by using similar approaches that was proposed in the (Wang, Su, Zheng, Sadiq, & Zhou, 2013), (Su, Liu, Zheng, Zhou, & Zheng, 2020). Based upon these, our evaluation procedure works as follows. We firstly define a trajectory as the seed semantic trajectory (original trajectory). Then we perform several types of transformations on the seed semantic trajectory according to different criteria and rates, in a controlled way (by using parameters), resulting in several sets of transformed trajectories. For each transformation, we will calculate the distance between the original and the transformed trajectories, which allow us to see the impact of each transformation over the original trajectory. Therefore, for every similarity measure, the trajectory with a lower degree of transformation should have higher similarity score with the original trajectory, and vice versa. However, this will not necessarily apply to all similarity measures/trajectory transformations, as some similarity measures may not be particularly affected by the rate of the type of trajectory transformation.

These transformations are controlled by four parameters, ratio, sampling frequency, scale and distance. In our work we will only use the parameter ratio, since we want to primarily focus on the part of semantic transformations of the our seed trajectory. This parameter is used to specify the percentage of sample points to be changed in a trajectory. For instance, ratio = 0.1 means that 10% of the sample points need to be changed by the transformation function. Below, we describe in detail all the types on transformations applied over the seed trajectory, which are based on the transformation of point shift.

Point shift transformation means modifying the sample point sequence of a trajectory, while its shape and trend are not modified. A distance measure with the capability of handling point shifting should keep the low distance values between a trajectory and its point shifted counterparts. In our work we present examples and experimental observations of the five types of transformation, applied over the seed semantic trajectory.

In Figure 3-1 we present an example of a semantic seed trajectory *Q* with 6 stops, which details the day of an office worker, where the following movement behavior is : stays at Home ((4, 12), [07:30am – 8:30am]), then he goes to Work ((136, 248), [9:00am – 5:00pm]), and from there he goes to Gym ((49, 111), [5:15pm – 6:00pm]), then he goes to the shopping stores ((6, 88), [7:00pm – 8:15pm]), from there he goes to Restaurant ((96, 231), [9:00pm – 11:00pm]), finishing the day to the Bar ((104, 201), [11:45pm – 02:00am]).

Figure 3-1: Seed trajectory *Q* with 6 stops

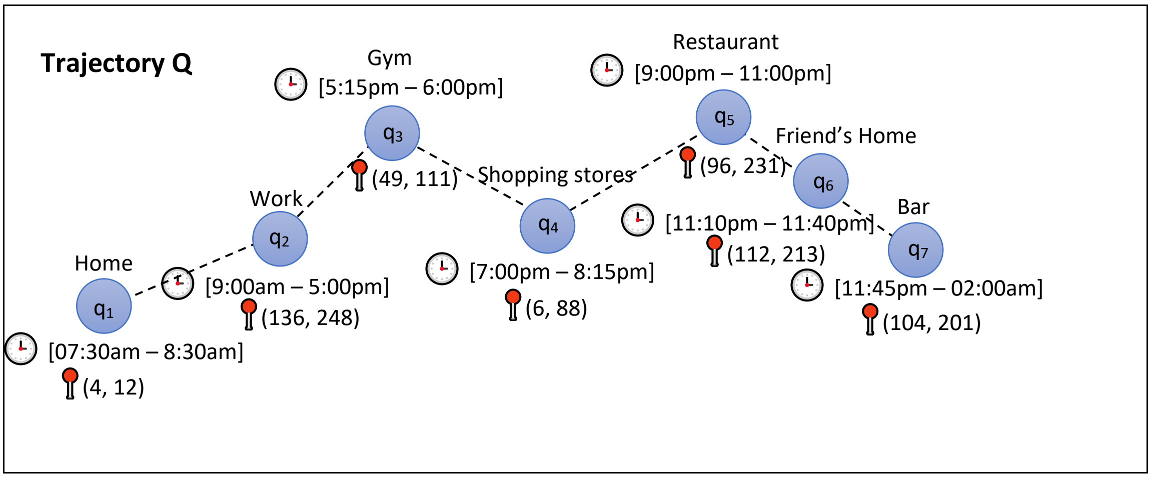


i. Transformation of adding stops

In this transformation, a number of stops (sampling points) are randomly generated and added to the transformed trajectory. The stops that are added to the trajectory, have the same dimensions with the stops of the original trajectory, i.e. the added sample point must have space, time and semantic dimensions if the seed trajectory is multidimensional (time, space and semantic). However, the stops that will be added, will have randomly generated elements that won't necessarily have any matching with the stops of the seed trajectory. In addition, the points that we add *(p_i)*, must be between two continuous sampling points p_{i-1} and p_{i+1}, which is over the seed trajectory, meaning that we can't add a stop which will be the first or the last stop of the transformed trajectory. In order to control the transformation, the user defines a parameter, called ratio *r*. For example, if we have a trajectory *T* with a size of *n*, adding ratio *r* means that we add $n \times r$ sampling points. The added stops will probably not match in the spatio-temporal dimensions. However, the semantic element is randomly selected from all the semantic elements in the dataset used, i.e. if the semantic elements of the dataset used are (Home, Work, Gym, Restaurant, Bank), then the semantic element to be selected will be among the elements we mentioned, and therefore we may have a match on the semantic dimension.

An example of this transformation over the original trajectory is illustrated below in Figure 3-2, where we can see the transformed trajectory. Suppose the parameter *r* has a value of 17%. Therefore, a random stop will be added, which may have a match with the stops of the seed trajectory. In this example, we will add a stop between the sampling points q_5 and q_7. This sampling point *(q_6)* indicates that, after the restaurant the moving object goes to a friend's home ((112, 213), [11:10pm – 11:40pm]). As we can clearly see on the Figure 3-2, the stop q_6 doesn't have any matching on any dimension.

Figure 3-2: Transformation of adding stops for *r* = 17%

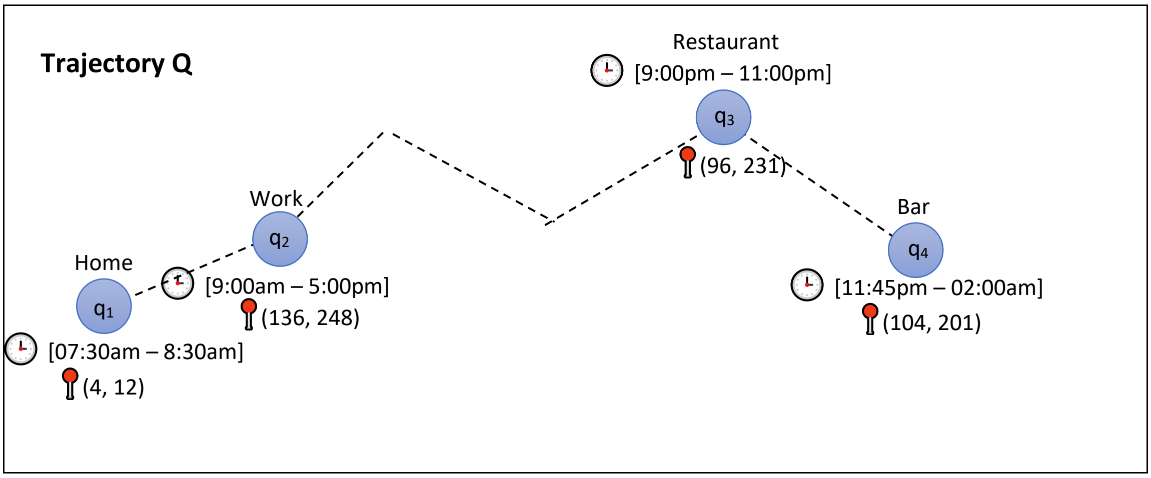


ii. Transformation of deleting stops

In this transformation, a number of stops are randomly removed from the seed trajectory to generate a transformed trajectory. Similar to transformation of adding sampling points, we control the transformation according to the parameter *r*. Since we randomly delete stops from the trajectory, the sequence of the stops is changing, affecting the effectiveness of similarity measures that takes into account the sequence and the order of the stops. As a result, the impact of the transformation over the seed trajectory will be more crucial for similarity measures that are affected by the change of the order of the stops. The distance between the seed trajectory and the transformed trajectory, will be dramatically reduced, as we increase the parameter ratio.

Let us consider an example shown in Figure 3-3, where we apply the transformation of deleting stops over our seed trajectory. Suppose the parameter *r* has a value of 33%. Therefore, two random stops will be deleted from the seed trajectory, resulting in a transformed trajectory which will have 4 stops. In this example, the length of the transformed trajectory is reduced by two stops and the order of the last two stops will change by deleting the sampling points q_3 and q_4.

Figure 3-3: Transformation of deleting stops for *r* = 33%

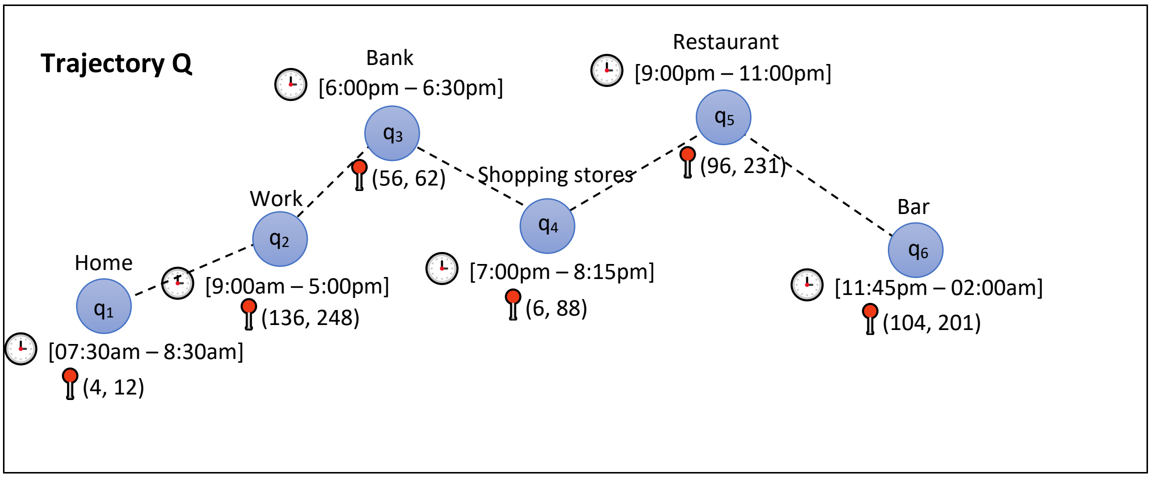


iii. Replacement of stops

In this transformation, the stops of the original trajectory are replaced with randomly generated stops that do not have any matching with the seed trajectory on the semantic dimension. The replaced sampling points of the transformed trajectory have different elements than the stop of the original trajectory on all dimensions (time, space and semantic). Similar to the aforementioned transformations, the number of sampling points to be replaced will be decided by the parameter *r*.

In the Figure 3-4, is presented an example of a transformed trajectory, where we apply the transformation of replacement of stops over our seed trajectory. Suppose the parameter *r* has a value of 17% and therefore one stop will be replaced by a randomly generated stop. More specifically, the sampling point q_3 will be replaced by a stop described as: Bank ((56, 62), [6:00pm – 6:30pm]). We can clearly see that this sampling point have no matching with any of the stops of the original trajectory and it doesn't affect any of the other stops or the sequence of the stops.

Figure 3-4: Replacement of Stops for *r* = 17%

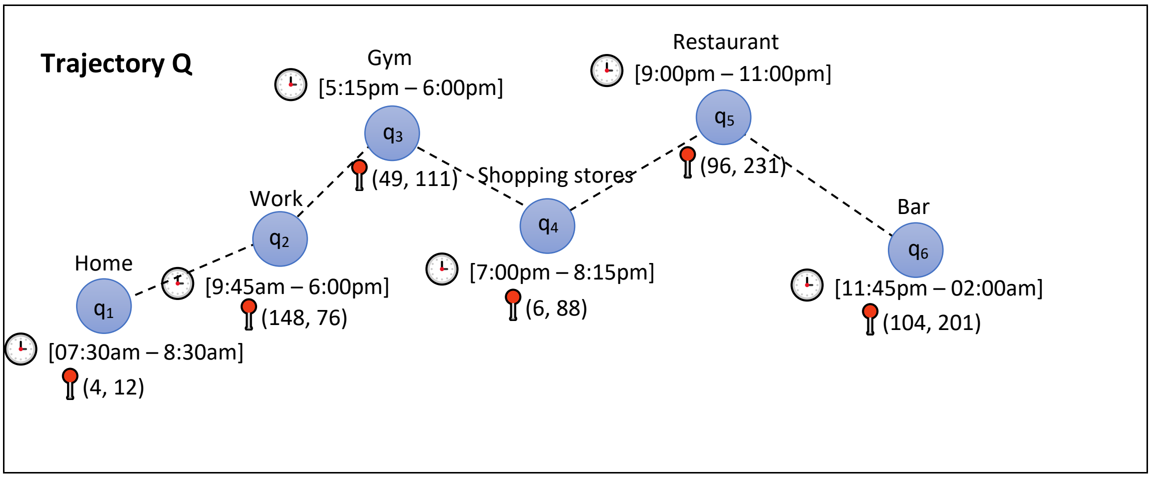


iv. Replacement of possible matching stops (semantic dimension)

In contrast to the previous transformation we mentioned, in this one, the stops of the seed trajectory are replaced by stops with possible matching in semantic dimension with the seed trajectory. Therefore, the stops are randomly generated and replaced according the parameter *r* we define. This transformation doesn't affect any other stop but only the sampling point that has been randomly selected to be replaced.

In the example shown below in Figure 3-5, we see that the sampling point q_2 has been replaced by another stop, which has the same partial matching with the stop of the seed trajectory. More specifically, the semantic dimension remains the same (Work), but the spatio-temporal dimensions have different elements than those in the seed trajectory.

Figure 3-5: Replacement of possible matching stops for *r* = 17%

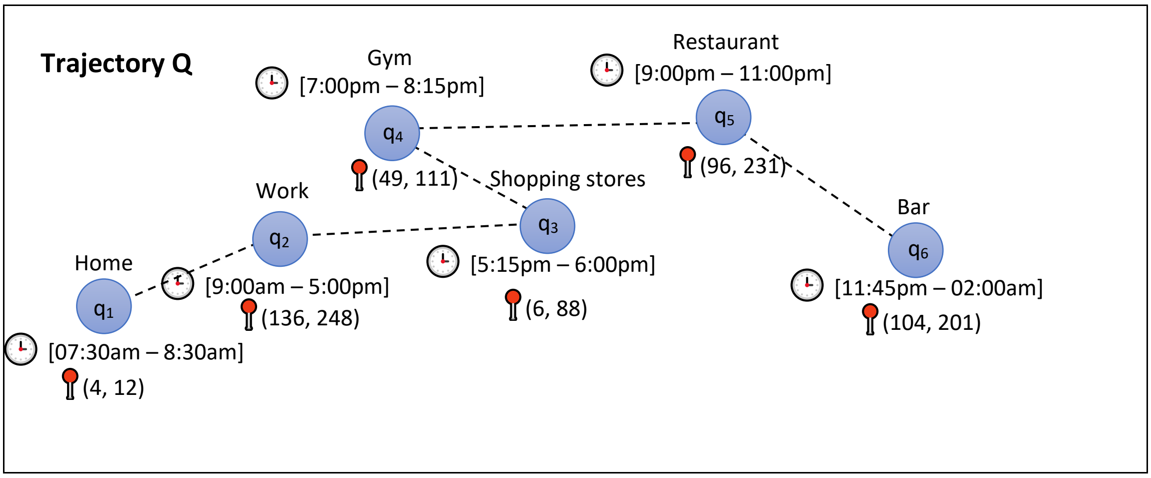


v. Position change of stops

In this transformation, the order of the stops, according to the value of *r*, is randomly changed. In this case, the transformation may affect more stops than those that changed, because the spatial and semantic dimension must change between the stops that we changed their order. For example, in case the value of *r* is 10% over a seed trajectory with 10 stops, the order of one stop will be changed, but this will probably affect the sequence and the dimensions of at least two stops. This transformation, will mostly affect the similarity measures that takes into account the sequence of the stops, unlike other measures (like UMS) that are not affected by the sequence of the stops.

An example of this transformation over the seed trajectory is illustrated below in Figure 3-6, where we can see the transformed trajectory. Suppose the parameter *r* has a value of 17%. and therefore, the position of a stop will change. However, we note that changing the position of the stop q_4, will also affect the sampling point q_3. The following movement behavior is : stays at Home ((4, 12), [07:30am – 8:30am]), then he goes to Work ((136, 248), [9:00am – 5:00pm]), and from there he goes to the shopping stores ((49, 111), [5:15pm – 6:00pm]), then he goes to Gym ((6, 88), [7:00pm – 8:15pm]), from there he goes to Restaurant ((96, 231), [9:00pm – 11:00pm]), finishing the day to the Bar ((104, 201), [11:45pm – 02:00am]).

Figure 3-6: Position change of stop for *r* = 17%

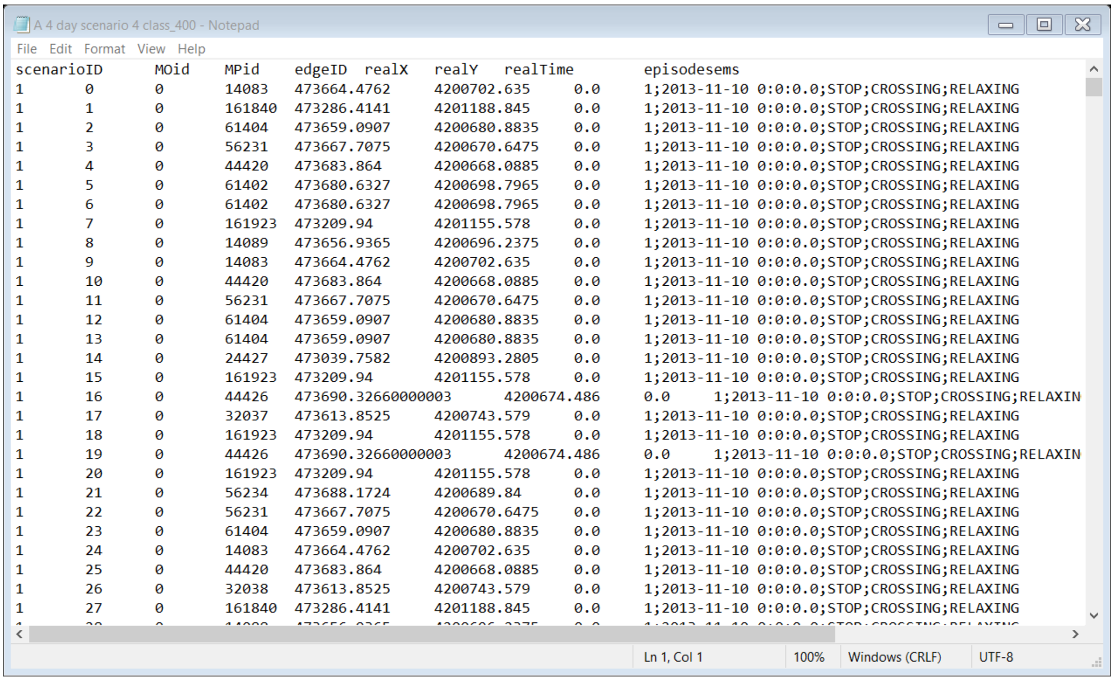


3.2. Algorithms for trajectory transformations

In this section we will describe and analyze the methods and the algorithms used for our datasets in order to achieve the aforementioned types of trajectory transformations. First of all, the techniques and the algorithms we constructed for our thesis are programmed in language R.

The dataset that we will use for the similarity measures MSM and UMS is a 4 day scenario generated by Hermoupolis algorithm. The dataset has 8 columns (scenarioID, Moid, MPid, edgeID, realX, realY, realTime, episodesems) and its form is the following:

Picture 1: A 4 day scenario (Original dataset)



```
scenarioID      MOid    MPid    edgeID  realX   realY   realTime        episodesems
1       0       0       14083   473664.4762     4200702.635     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       1       0       161840  473286.4141     4201188.845     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       2       0       61404   473659.0907     4200680.8835    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       3       0       56231   473667.7075     4200670.6475    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       4       0       44420   473683.864      4200668.0885    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       5       0       61402   473680.6327     4200698.7965    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       6       0       61402   473680.6327     4200698.7965    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       7       0       161923  473209.94       4201155.578     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       8       0       14089   473656.9365     4200696.2375    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       9       0       14083   473664.4762     4200702.635     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       10      0       44420   473683.864      4200668.0885    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       11      0       56231   473667.7075     4200670.6475    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       12      0       61404   473659.0907     4200680.8835    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       13      0       61404   473659.0907     4200680.8835    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       14      0       24427   473039.7582     4200893.2805    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       15      0       161923  473209.94       4201155.578     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       16      0       44426   473690.32660000003      4200674.486     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXIN
1       17      0       32037   473613.8525     4200743.579     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       18      0       161923  473209.94       4201155.578     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       19      0       44426   473690.32660000003      4200674.486     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXIN
1       20      0       161923  473209.94       4201155.578     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       21      0       56234   473688.1724     4200689.84      0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       22      0       56231   473667.7075     4200670.6475    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       23      0       61404   473659.0907     4200680.8835    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       24      0       14083   473664.4762     4200702.635     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       25      0       44420   473683.864      4200668.0885    0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       26      0       32038   473613.8525     4200743.579     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
1       27      0       161840  473286.4141     4201188.845     0.0     1;2013-11-10 0:0:0.0;STOP;CROSSING;RELAXING
```

The meaning of each column is described on the table below:

Table 3-1: Explaining of the columns of the dataset

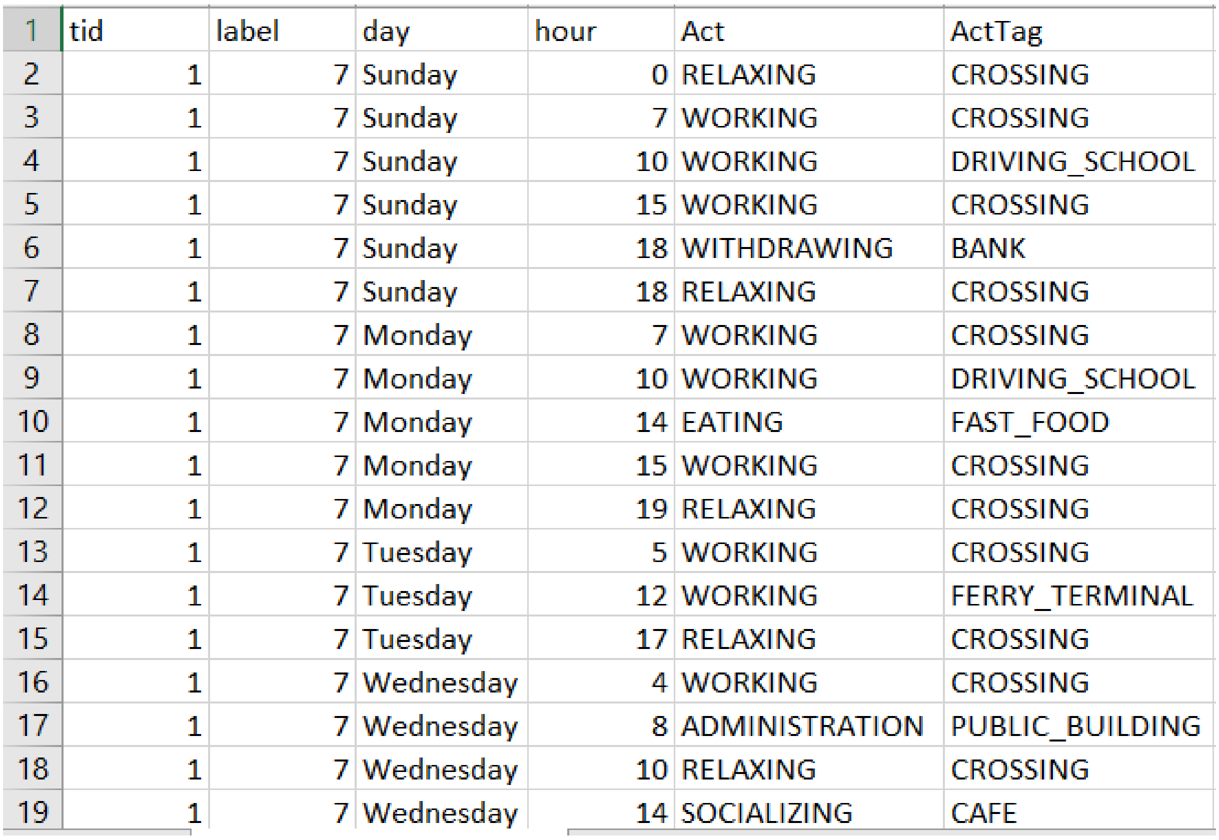
scenarioID	is the id of the mobility scenario run in hermoupolis. Usually is 1 though hermoupolis may run multiple mobility scenarios at the same time.=>1
Moid	is the id of the moving object=>34
MPid	is the id of the mobility profile followed by the corresponding moving object=>2
edgeID	is the id of the network edge moving object is moving on=>161839
realX	is the coordinate X (in cartesian meters)=>473286.4141
realY	is the coordinate Y (in cartesian meters)=>4201188.845
realTime	is the time step of the generator hermoupolis in seconds=>27671.39092144424
episodesems	is the semantics of the current position of the moving object. Consist of the episode id which is the id of the current episode moving object is on, timestamp which is the current timestamp in YYYY-MM-DD HH:MI:SS format, type of episode which can be STOP or MOVE,

	episode tag which is a word describing the episode and activity tag which is a word describing moving object's current activity=>2;2013-11-10 7:35:59.731;MOVE;DRIVE;TRANSPORTATION

Firstly we read the dataset in R and we sort the data by the id of the moving object (Moid). The dataset is clustered in 4 classes 0, 1, 2 and 3 (MPid), so we create a row for each stop and move the moving object made and we split the dataset into 4 datasets, where each one will contain moving objects of only one class. Afterwards, we split the column episodesems into 5 new columns i.e. the id of the episode, the timestamp, the type of episode, the act of the moving object and the activity tag of the moving object. Finally, we name each column and we save the 4 new datasets we created. With the modification we made, our dataset is in the format we desire, so that we can implement our transformations.

To be able to find the similarity score with the help of MUITAS we used the same dataset but in different format. However, the logic behind the algorithms for the transformations are the same as on the other similarity measures . The format of this dataset is the following:

Picture 2: MUITAS dataset



1	tid	label	day	hour	Act	ActTag
2	1	7	Sunday	0	RELAXING	CROSSING
3	1	7	Sunday	7	WORKING	CROSSING
4	1	7	Sunday	10	WORKING	DRIVING_SCHOOL
5	1	7	Sunday	15	WORKING	CROSSING
6	1	7	Sunday	18	WITHDRAWING	BANK
7	1	7	Sunday	18	RELAXING	CROSSING
8	1	7	Monday	7	WORKING	CROSSING
9	1	7	Monday	10	WORKING	DRIVING_SCHOOL
10	1	7	Monday	14	EATING	FAST_FOOD
11	1	7	Monday	15	WORKING	CROSSING
12	1	7	Monday	19	RELAXING	CROSSING
13	1	7	Tuesday	5	WORKING	CROSSING
14	1	7	Tuesday	12	WORKING	FERRY_TERMINAL
15	1	7	Tuesday	17	RELAXING	CROSSING
16	1	7	Wednesday	4	WORKING	CROSSING
17	1	7	Wednesday	8	ADMINISTRATION	PUBLIC_BUILDING
18	1	7	Wednesday	10	RELAXING	CROSSING
19	1	7	Wednesday	14	SOCIALIZING	CAFE

Then, we present the methods we applied, so that we can transform the trajectories according to the type of transformation we define. We distinguish that the dataset “4 day scenario” contains both stops and moves, and also many variables that are not needed to compute the similarity score between the trajectories. As a result, we will delete all the moves and the variable we don’t need for our final dataset.

I. Delete Stops

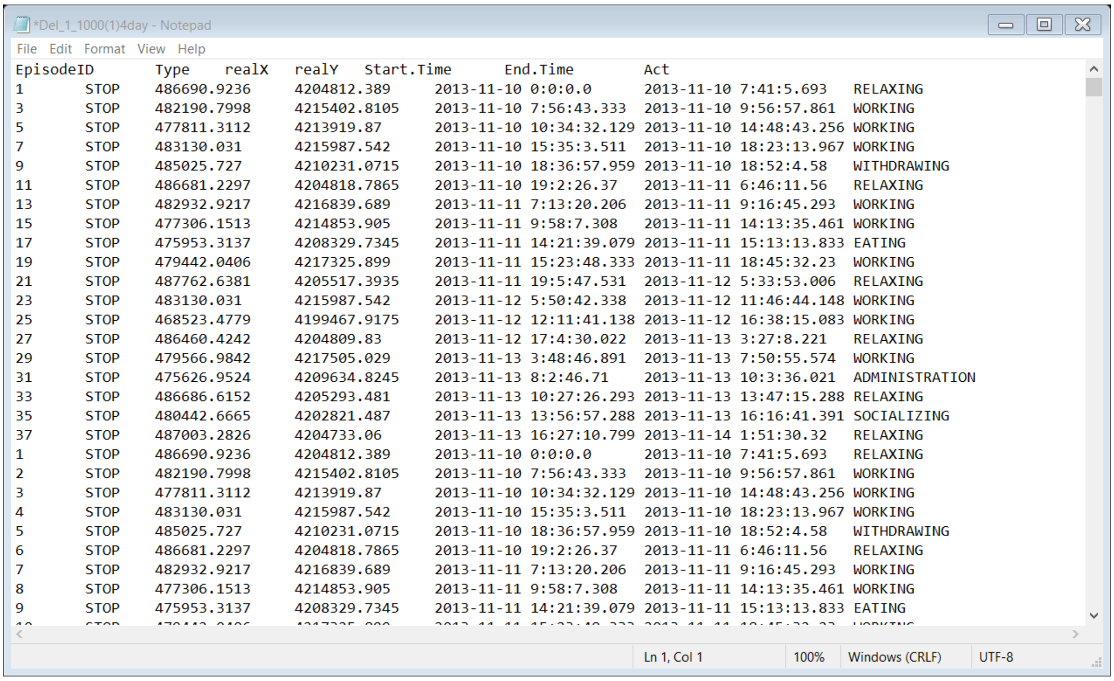
In this transformation we delete sample points according to the predefined rate *r*. Firstly, we read one of our datasets we created before. Then, we randomly select one trajectory and define it as our seed trajectory. After that, we create a variable called “TrajNum”, which defines how many transformed trajectories we will construct. However, our most important variable is called rate and the role of this variable is to control the amount of transformations over our seed trajectory. The following pseudo-algorithm (Figure 3-7) is the core of the transformation of deleting stops and requires the format of the dataset showed on Picture 3 (for MSM and UMS) or the format showed on Picture 2 (for MUITAS).

Figure 3-7: Pseudo-Algorithm for deleting stops

```
#Read the dataset
data ← dataset
#Randomly select a trajectory as our seed trajectory
t ← random trajectory
#define the number of transformed trajectories, the rate r and the number of stops
TrajNum ← n          #user defines the n (i.e. 500)
rate ← r             #define rate from 0 to 1 (i.e. 0.5)
count ← number of stops of the trajectory
#repeat n times, so we generate n transformed trajectories
for (i = 1 to TrajNum) {
#delete stops according to the rate
    delete (rate*count) STOPS
END                  #end the loop
```

The dataset (4 day scenario) we construct with this pseudo-algorithm has 7 columns called EpisodeID, Type, realX, realY, Start.Time, End.Time and Act.. Therefore, our dataset contains the original trajectory and *n* transformed trajectories, which describes the space *(x,y)* of the stop, the start and the end time of the stop and the activity performed by the moving object on this sample point. In addition, the dataset for MUITAS has 6 columns called tid, label, day, hour, Act and ActTag and just like the first dataset it contains the seed trajectory and n transformed trajectories, which describes the time of the sample point and all the attributes of the moving object. Then, we delete the separators (“) from the text document, so we can apply the similarity measures. The format of the text document is illustrated below on Picture 3.

Picture 3: Format of document after deleting stops



```
EpisodeID       Type    realX       realY    Start.Time         End.Time          Act
1       STOP    486690.9236     4204812.389     2013-11-10 0:0:0.0      2013-11-10 7:41:5.693   RELAXING
3       STOP    482190.7998     4215402.8105    2013-11-10 7:56:43.333  2013-11-10 9:56:57.861  WORKING
5       STOP    477811.3112     4213919.87      2013-11-10 10:34:32.129 2013-11-10 14:48:43.256 WORKING
7       STOP    483130.031      4215987.542     2013-11-10 15:35:3.511  2013-11-10 18:23:13.967 WORKING
9       STOP    485025.727      4210231.0715    2013-11-10 18:36:57.959 2013-11-10 18:52:4.58   WITHDRAWING
11      STOP    486681.2297     4204818.7865    2013-11-10 19:2:26.37   2013-11-11 6:46:11.56   RELAXING
13      STOP    482932.9217     4216839.689     2013-11-11 7:13:20.206  2013-11-11 9:16:45.293  WORKING
15      STOP    477306.1513     4214853.905     2013-11-11 9:58:7.308   2013-11-11 14:13:35.461 WORKING
17      STOP    475953.3137     4208329.7345    2013-11-11 14:21:39.079 2013-11-11 15:13:13.833 EATING
19      STOP    479442.0406     4217325.899     2013-11-11 15:23:48.333 2013-11-11 18:45:32.23  WORKING
21      STOP    487762.6381     4205517.3935    2013-11-11 19:5:47.531  2013-11-12 5:33:53.006  RELAXING
23      STOP    483130.031      4215987.542     2013-11-12 5:50:42.338  2013-11-12 11:46:44.148 WORKING
25      STOP    468523.4779     4199467.9175    2013-11-12 12:11:41.138 2013-11-12 16:38:15.083 WORKING
27      STOP    486460.4242     4204809.83      2013-11-12 17:4:30.022  2013-11-13 3:27:8.221   RELAXING
29      STOP    479566.9842     4217505.029     2013-11-13 3:48:46.891  2013-11-13 7:50:55.574  WORKING
31      STOP    475626.9524     4209634.8245    2013-11-13 8:2:46.71    2013-11-13 10:3:36.021  ADMINISTRATION
33      STOP    486686.6152     4205293.481     2013-11-13 10:27:26.293 2013-11-13 13:47:15.288 RELAXING
35      STOP    480442.6665     4202821.487     2013-11-13 13:56:57.288 2013-11-13 16:16:41.391 SOCIALIZING
37      STOP    487003.2826     4204733.06      2013-11-13 16:27:10.799 2013-11-14 1:51:30.32   RELAXING
1       STOP    486690.9236     4204812.389     2013-11-10 0:0:0.0      2013-11-10 7:41:5.693   RELAXING
2       STOP    482190.7998     4215402.8105    2013-11-10 7:56:43.333  2013-11-10 9:56:57.861  WORKING
3       STOP    477811.3112     4213919.87      2013-11-10 10:34:32.129 2013-11-10 14:48:43.256 WORKING
4       STOP    483130.031      4215987.542     2013-11-10 15:35:3.511  2013-11-10 18:23:13.967 WORKING
5       STOP    485025.727      4210231.0715    2013-11-10 18:36:57.959 2013-11-10 18:52:4.58   WITHDRAWING
6       STOP    486681.2297     4204818.7865    2013-11-10 19:2:26.37   2013-11-11 6:46:11.56   RELAXING
7       STOP    482932.9217     4216839.689     2013-11-11 7:13:20.206  2013-11-11 9:16:45.293  WORKING
8       STOP    477306.1513     4214853.905     2013-11-11 9:58:7.308   2013-11-11 14:13:35.461 WORKING
9       STOP    475953.3137     4208329.7345    2013-11-11 14:21:39.079 2013-11-11 15:13:13.833 EATING
```

With the above dataset we are able to find the similarity score of seed trajectory in relation to the transformed trajectories with method UMS and MUITAS. In order to use MSM we need to define space time and semantic threshold. In order to achieve that, we add in the start of our dataset the following 3 lines:

```
SpaceThreshold k1
TimeThreshold k2
SemanticThreshold k3
```

II. Replacement of stops with different elements

In this transformation we replace sample points according to the predefined rate , by following similar approach to the previous method. The below Figure 3-8 presents the pseudo-algorithm used for this transformation which requires the format of the dataset showed on Picture 3 (for MSM and UMS) or the format showed on Picture 2 (for MUITAS).

Figure 3-8: Pseudo-Algorithm of the transformation of replacement of stops with different elements

```
#Read the dataset
```

```
data ← dataset
#Randomly select a trajectory as our seed trajectory
t ← random trajectory
#define the number of transformed trajectories, the rate r and the number of stops
TrajNum ← n          #user defines the n (i.e. 500)
rate ← r             #define rate from 0 to 1 (i.e. 0.5)
count ← number of stops of the trajectory
#repeat n times, so we generate n transformed trajectories
for (i = 1 to TrajNum) {
#replace stops with different elements according to the rate
replace (rate*count) STOPS
END                  #end the loop
```

Initially, we delete the separators (") from the text document, so we can apply UMS and MUITAS and find the similarity score between the seed trajectory and the transformed trajectories. In order to use MSM we need to define space time and semantic threshold, same as before.

III. Replacement of possible matching stops

In this transformation we replace sample points with possible matching stops according to the predefined rate , by following similar approach to the previous method. The below Figure 3-9 presents the pseudo-algorithm used for this transformation which requires the format of the dataset showed on Picture 3 (for MSM and UMS) or the format showed on Picture 2 (for MUITAS).

Figure 3-9: Pseudo-Algorithm of the transformation of replacement with possible matching stops

```
#Read the dataset
data ← dataset
#Randomly select a trajectory as our seed trajectory
t ← random trajectory
#define the number of transformed trajectories, the rate r and the number of stops
TrajNum ← n          #user defines the n (i.e. 500)
rate ← r             #define rate from 0 to 1 (i.e. 0.5)
count ← number of stops of the trajectory
#repeat n times, so we generate n transformed trajectories
for (i = 1 to TrajNum) {
```

```
#replace stops with possible matching elements according to the rate
    replace (rate*count) STOPS
    END                 #end the loop
```

At first, we delete the separators (“) from the text document, so we can apply UMS and MUITAS and find the similarity score between the seed trajectory and the transformed trajectories. In order to use MSM we need to define space time and semantic threshold, same as before.

IV. Position change of stops

In this transformation we randomly change the position between two sample points. As a result, we swap the information of two random sample points. The algorithm of this method is illustrated below which requires the format of the dataset showed on Picture 3 (for MSM and UMS) or the format showed on Picture 2 (for MUITAS).

Figure 3-10: Pseudo-Algorithm for changing the position of the stops

```
#Read the dataset
    data ← dataset
#Randomly select a trajectory as our seed trajectory
    t ← random trajectory
#define the number of transformed trajectories, the rate r and the number of stops
    TrajNum ← n          #user defines the n (i.e. 500)
    rate ← r             #define rate from 0 to 1 (i.e. 0.5)
    count ← number of stops of the trajectory
#repeat n times, so we generate n transformed trajectories
    for (i = 1 to TrajNum) {
#change the position of the stops according to the rate
        for j = 1 to (rate*count) {
    k ← random number of STOP (from 1 to (number of STOPS))
    l ← random number of STOP (from 1 to (number of STOPS) that is different from k)
    $STOP_k$ → $STOP_l$
    $STOP_l$ → $STOP_k$
        END loop 1          #rnd the 1st loop
    END loop 2              #end the 2nd loop
```

Initially, we delete the separators (“) from the text document, so we can apply UMS and MUITAS and find the similarity score of the seed trajectory in relation to the transformed trajectories. Same as before, in order to use MSM we need to define space time and semantic threshold.

V. Adding stops

In this transformation we randomly add some extra points to the given trajectory according to the predefined rate (from 0 to 1), by following similar approach to the previous method. The below Figure 3-11 presents the algorithm used for this transformation which requires the format of the dataset showed on Picture 3 (for MSM and UMS) or the format showed on Picture 2 (for MUITAS).

Figure 3-11: Pseudo-Algorithm for transformation of adding stops

```
#Read the dataset
data ← dataset
#Randomly select a trajectory as our seed trajectory
t ← random trajectory
#define the number of transformed trajectories, the rate r and the number of stops
TrajNum ← n          #user defines the n (i.e. 500)
rate ← r             #define rate from 0 to 1 (i.e. 0.5)
count ← number of stops of the trajectory
#repeat n times, so we generate n transformed trajectories
for (i = 1 to TrajNum) {
#add stops according to the rate
    for j = 1 to (rate*count) {
NewSTOP ← Create a STOP
k ← random number of STOP (from 1 to (number of STOPS))
Add NewSTOP between stops k and (k+1)
    END loop 1           #rnd the 1st loop
END loop 2               #end the 2nd loop
```

At first, we delete the separators (“) from the text document, so we can apply UMS and MUITAS and find the similarity score between the seed trajectory and the transformed trajectories. Same as before, in order to use MSM we need to define space time and semantic threshold.

Chapter 4. Experimental Evaluation

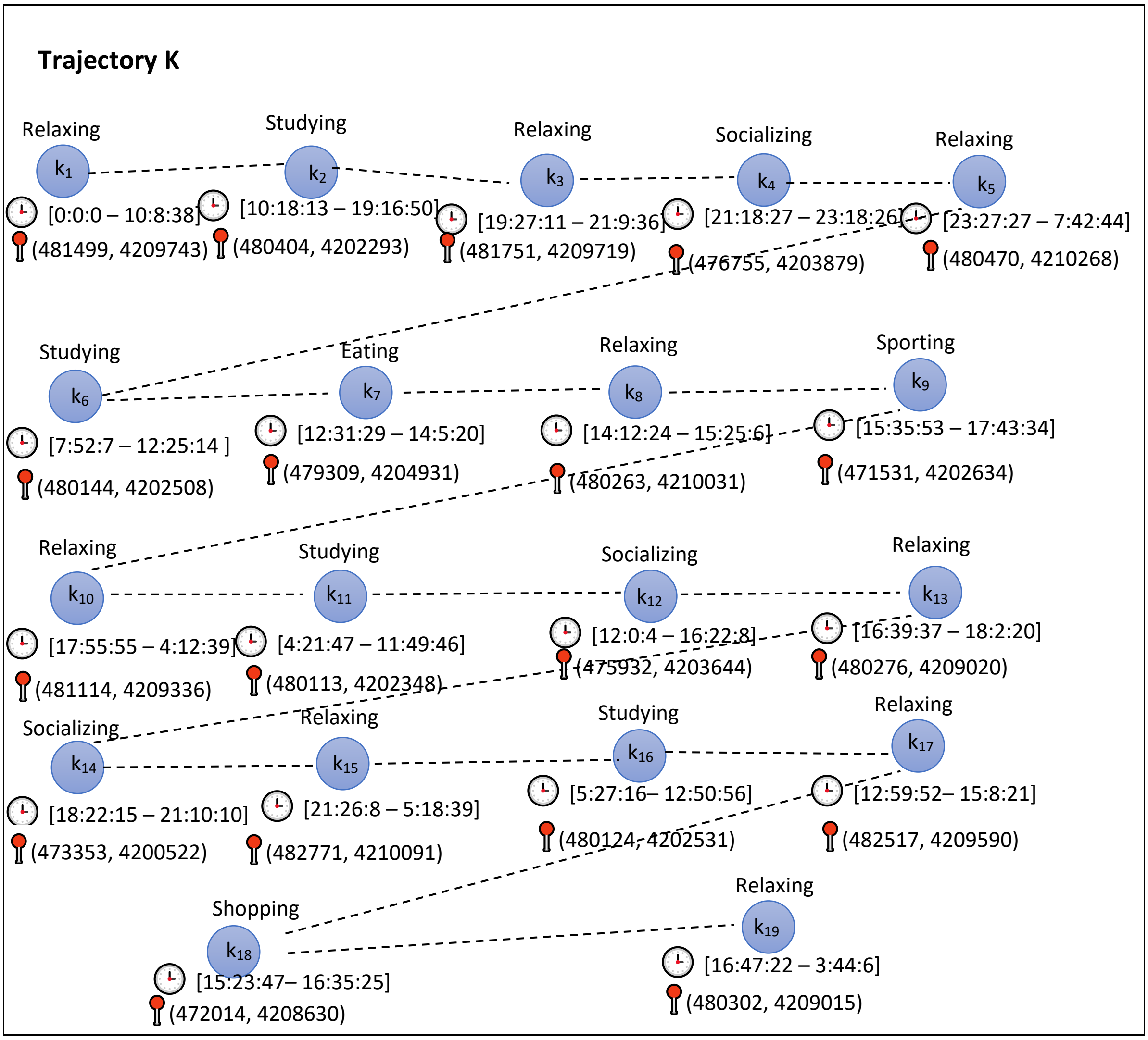
This chapter presents the experiments that will be performed regarding the robustness and effectiveness of the compared similarity measures of semantic trajectories. To test the capability of these similarity measures, we apply a set of transformations over the seed trajectory, computing the similarity between the original trajectory and the transformed trajectories, using all similarity measures, and then we compare the results of all methods. In Section 4.1, we present the steps we follow in order to compare the similarity measures, and in Section 4.2 we present the experiments and analyze the results.

4.1. Evaluation Techniques

In order to be able to compare the similarity measures with each other, we need to follow some specific steps that will result in showing us to what extent each similarity measure is affected by the trajectory transformations we mentioned earlier.

The first step we need to do in this experiment is to randomly select a moving object from the dataset and identify all the elements from its stops. The Figure 4-1 below shows a moving trajectory from the dataset that we will use for the experiments that will be done in Section 4.2. Then, we define this moving object *K* as the seed trajectory which we will find its distance from each of the other generated trajectories.

Figure 4-1: Semantic trajectory *K* with 19 stops



In the second step, we will generate *n* (we define *n* depending on the accuracy and the computing cost we want) different trajectories for each transformation that we will apply over the original trajectory that we have defined. For example, suppose we want to apply the transformation of deleting stops, over our seed trajectory with 9 stops. Then, for the sake of our example, let us define our parameter *r* be equal to 10%. Therefore, we will generate *n* = 500 transformed trajectories, where in each transformed trajectory we will remove a stop from the seed trajectory, and so we will have made 500 transformed trajectories with 8 stops. This process will be followed for all transformations and parameters *r* that we will define. The more trajectories we generate, the more precision we have, but also the more computational cost and vice versa, as for every trajectory we create the computer takes some time so that it can implement the transformation we want. Therefore, we must find a balance between these two factors, which in turn will give us the maximum possible accuracy in combination with low-time consuming speed.

In the next step, we group all the transformed trajectories, and then we use all the similarity measures (MSM, MUITAS and UMS) we want to compare with each other, based on the effects of the trajectory transformations have on these measures. First, we collect all the transformed trajectories we generated in the previous step and group them according to three criteria. More specifically, these criteria are the similarity measure, the type of the transformation and the parameter *r*. For instance, all the transformed trajectories in which we have applied the same type of transformation and defined the same parameter value *r*, are grouped together. As a result, we group *n* transformed trajectories for each different type of transformation and ratio *r*. Afterwards, we calculate the distance between the seed trajectory and the transformed trajectories. Therefore, for each specific group of trajectories we have created, we calculate the distance of each trajectory to the original trajectory we have selected to transform in the previous steps. The method to calculate the distance between the moving objects is by applying the functions and algorithms of the same similarity measure (MSM, MUITAS and UMS) to all the pairs of moving objects, resulting in *n* different results. In order to have an accurate picture of the impact of the transformations on the similarity measure we applied, we calculate the mean and the median of the *n* similarity distances. Eventually, we end up with a mean and a median similarity score for the chosen set of trajectories, which means that, when we transform a trajectory with a certain type of transformation (adding stops, deleting stops etc.) and with a predefined ratio *r*, its similarity distance to the transformed trajectory is approximately equal to the mean/median value we previously calculated.

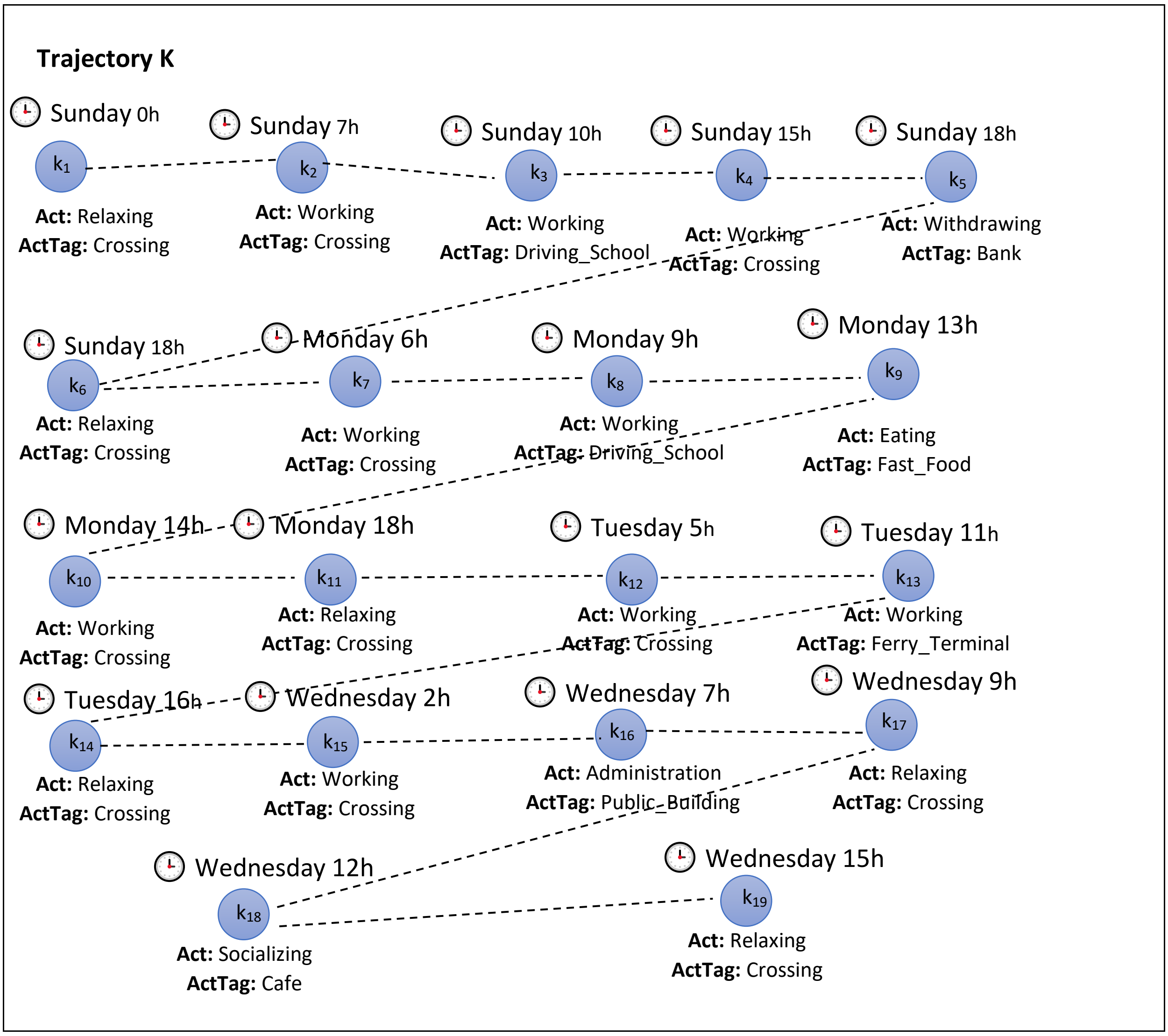
In the final step, we will present diagrammatically the results obtained from the previous step. Each diagram will describe the results (mean and median) for a similarity measure and for a type of transformation. The x-axis will show us the value of the parameter *r* and the y-axis is measuring the value of the mean/median. Nevertheless, in order to be able to comparatively see all the similarity measures at the same time, we will form in the diagram all the resulting curves, by calculating the mean/median distance of the pairs of the trajectories with each similarity measure. The higher the value of the y-axis the more robust the similarity measure is in the type of transformation we test.

4.2. Experiments and Results Presentation

For our experiment we will use a dataset which describes the movement behavior of a moving object for 4 days. We can easily note that this dataset is large enough since it contains the information of a moving object for 4 days. As a result as, we expect the results of the similarity measures to be accurate and reliable for our research. A semantic trajectory is presented in Figure 4-1.

However, in order to calculate the similarity score with the method MUITAS, we will use a different format of this dataset. The difference of this format is that it considers multiple-aspect trajectories that have 2 attributes , the act of the moving object and the activity tag of the moving object. The reason for using this format instead of the one we aforementioned is that it have more attributes which will help us to more effectively discern the impact of the transformations on the similarity measure MUITAS. However, in this format, in order to apply MUITAS similarity function we need to make some more adjustments on the dataset. In the following Figure 4-2, an example of a multiple-aspect trajectory is illustrated with 19 stops and 2 attributes (act of the moving object and the activity tag of the moving object).

Figure 4-2: Multiple-Aspect Semantic trajectory with 19 stops



In this experiment (mainly for MSM), in order to consider whether or not stops match in each dimension, let the corresponding threshold be set to i) 10 for space ii) 0.5 for time iii) 0.5 for semantics.

Just like we explained in Section 3.1 we will apply all the types of transformations i) Transformation of deleting stops, ii) Transformation of adding stops iii) Position change of stops. iv) Replacement of stops v) Replacement of possible matching stops (semantic dimension) and each of them will be presented for all the similarity measures.

4.2.1. Transformation of deleting stops

The results from the transformation of deleting stops from the semantic trajectories and the multiple-aspect trajectories are presented on Figure 4-3, Figure 4-4 and Figure 4-5:

Figure 4-3: Transformation of deleting stops (MSM)

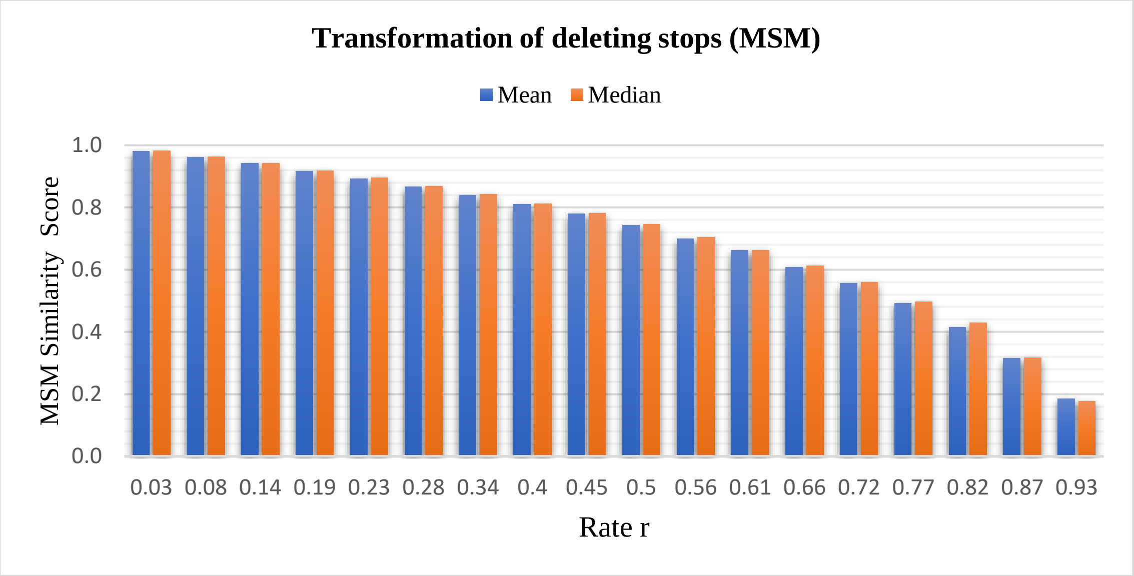


Figure 4-4: Transformation of deleting stops (UMS)

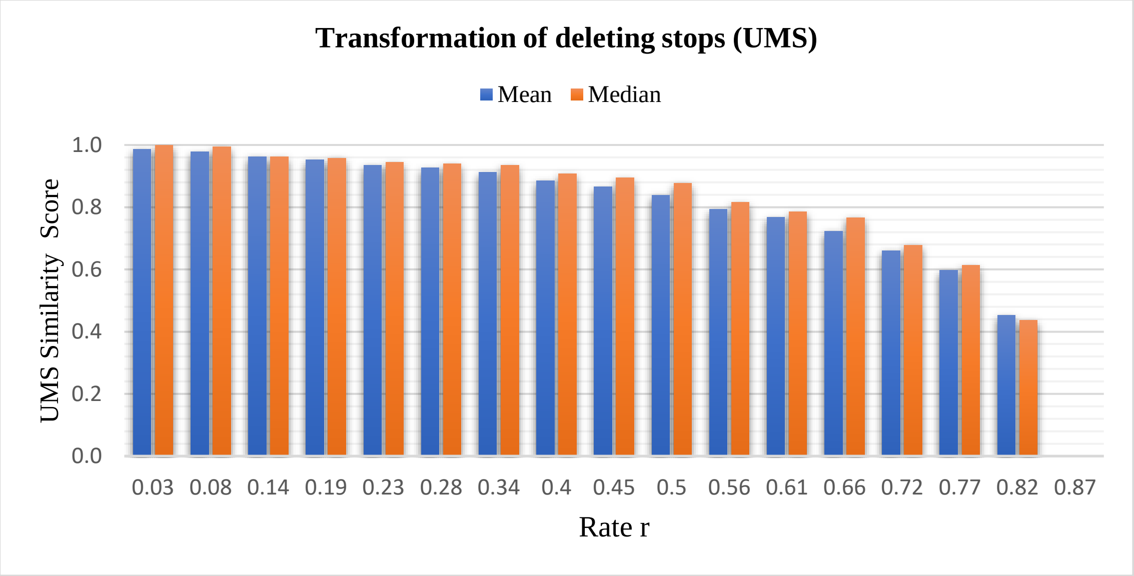
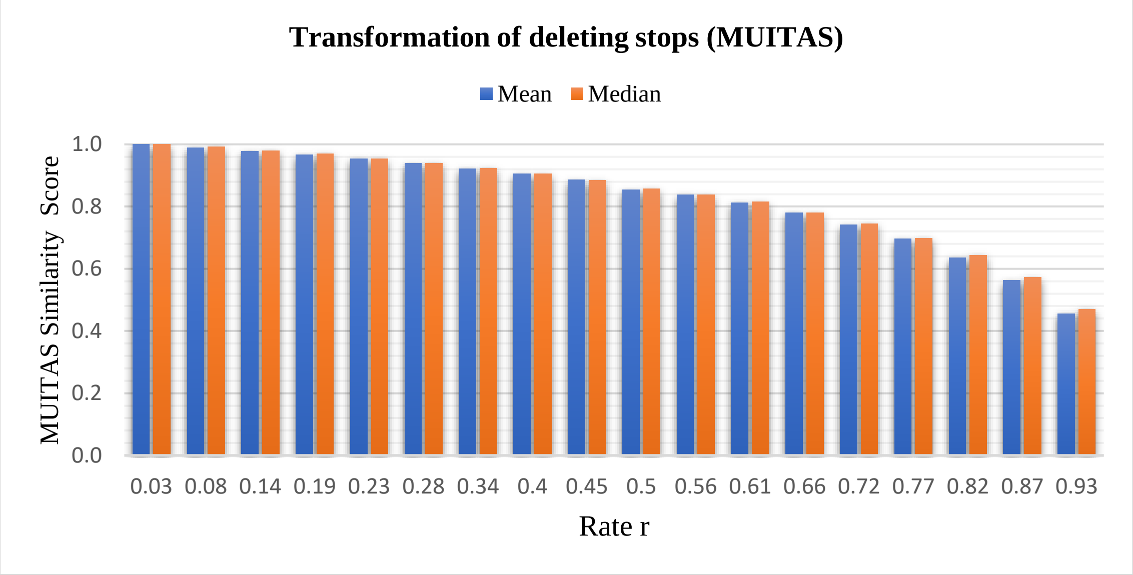


Figure 4-5: Transformation of deleting stops (MUITAS)



In order to better examine the impact of the transformation of deleting stops comparatively on each similarity measure, we will present the mean and the median similarity score of the seed trajectory in relation to the transformed trajectories for each similarity measure on the below figures.

Figure 4-6: Comparison of similarity measures (deleting stops) (Mean)

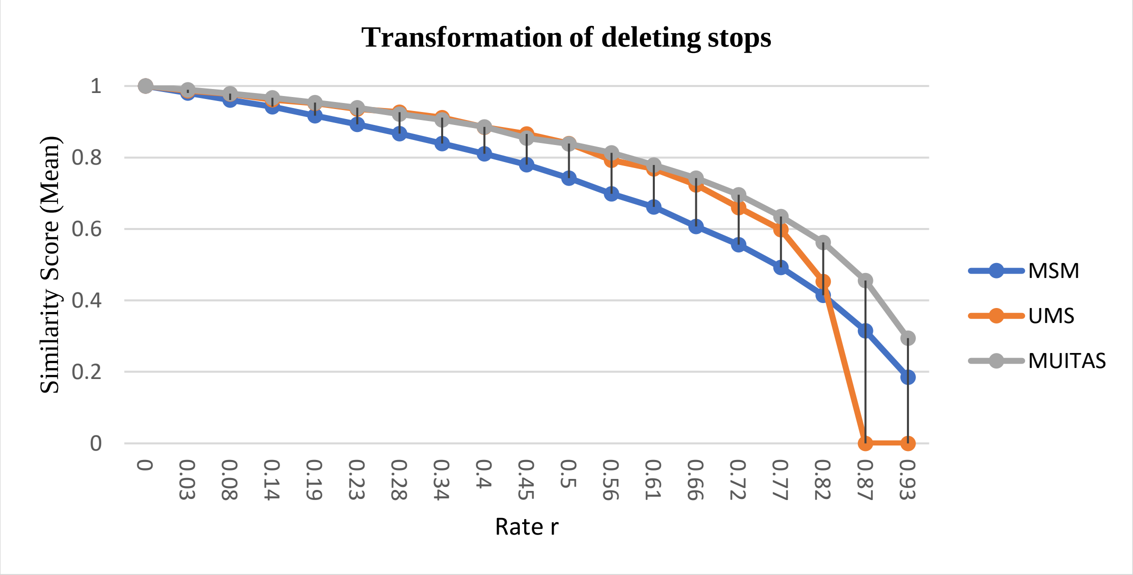
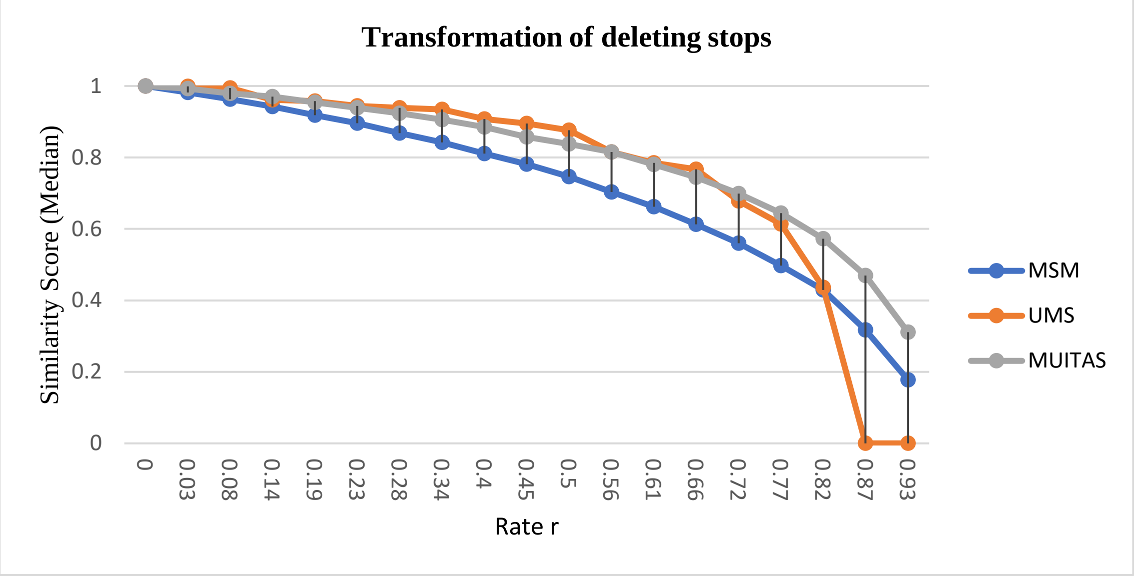


Figure 4-7: Comparison of similarity measures (deleting stops) (Median)



From the two figures above we draw the following conclusions: i) MUITAS is the most robust similarity measure compared to MSM and UMS, as it maintains the highest similarity score throughout the increase of rate r from 0 to 1, ii) we notice that all the similarity measures have almost the same similarity score on most rate values, with MSM getting bigger differences in the similarity score, the more the value of r increases, iii) the similarity score of UMS is the only similarity measure that is zero from rate value 0.87 to 1, unlike the rest measures that has similarity score zero only when all stops are removed from the seed trajectory.

4.2.2. Transformation of adding stops

The results from the transformation of adding stops from the semantic trajectories and the multiple-aspect trajectories are presented on the following figures:

Figure 4-8: Transformation of adding stops (MSM)

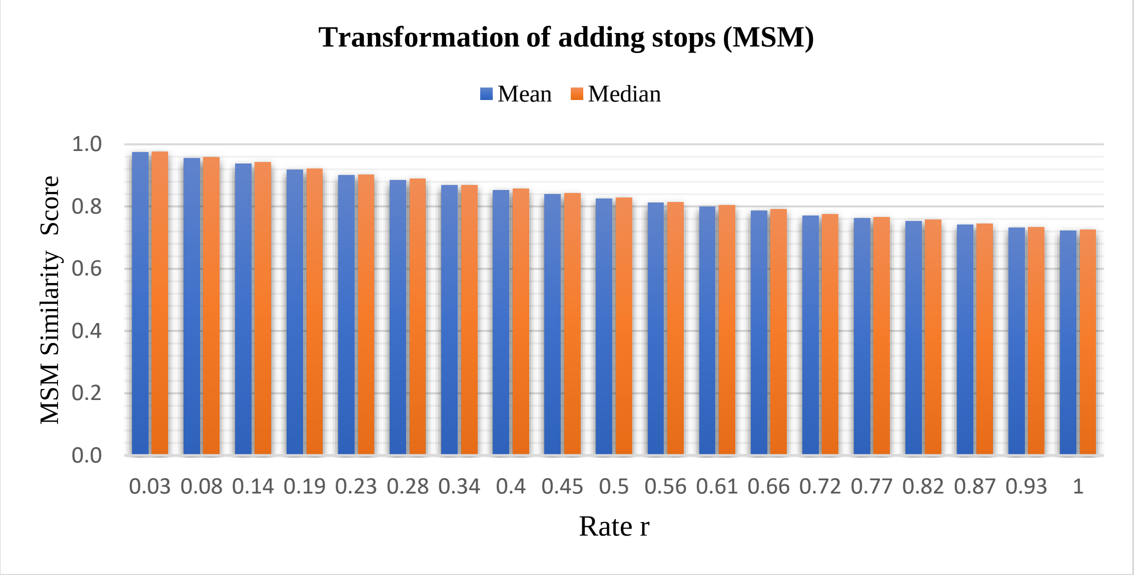


Figure 4-9: Transformation of adding stops (UMS)

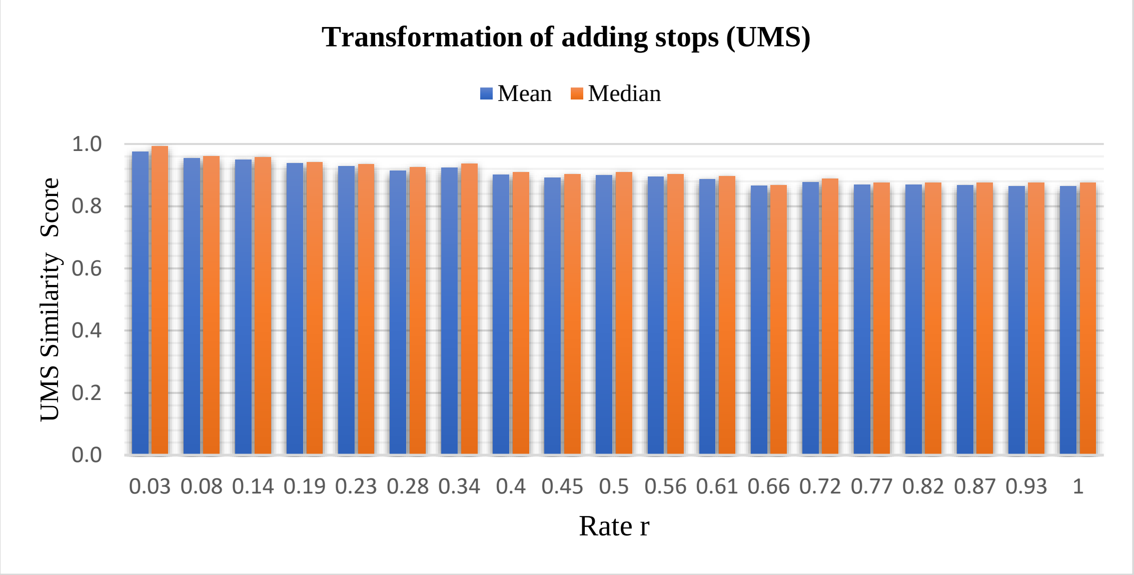
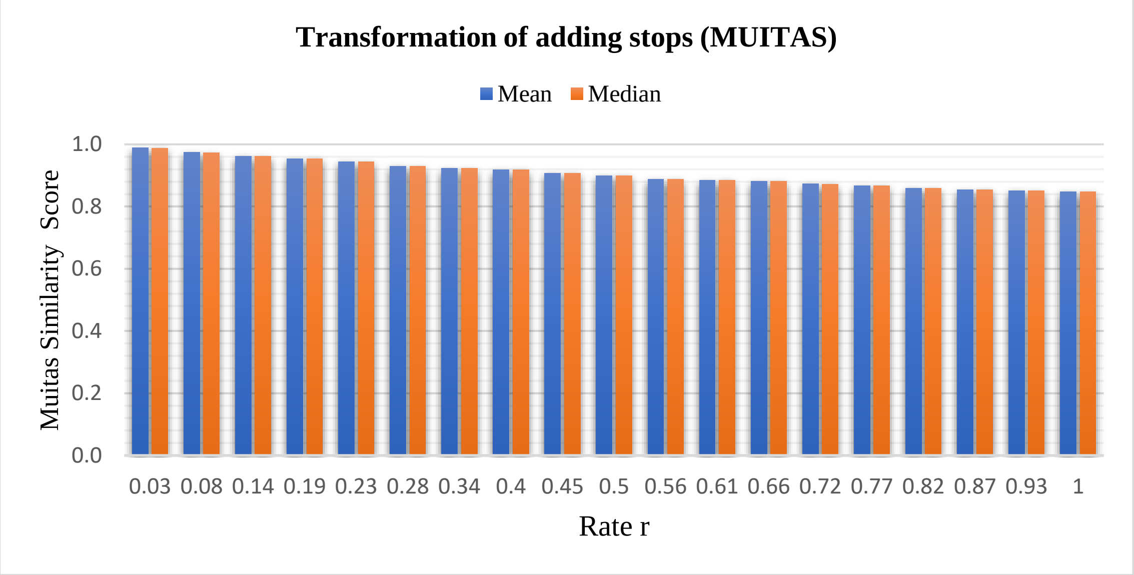


Figure 4-10: Transformation of adding stops (MUITAS)



In order to better examine the impact of the transformation of adding stops comparatively on each similarity measure, we will present the mean and the median similarity score of the seed trajectory in relation to the transformed trajectories for each similarity measure on the below figures.

Figure 4-11: Comparison of similarity measures (adding stops) (Mean)

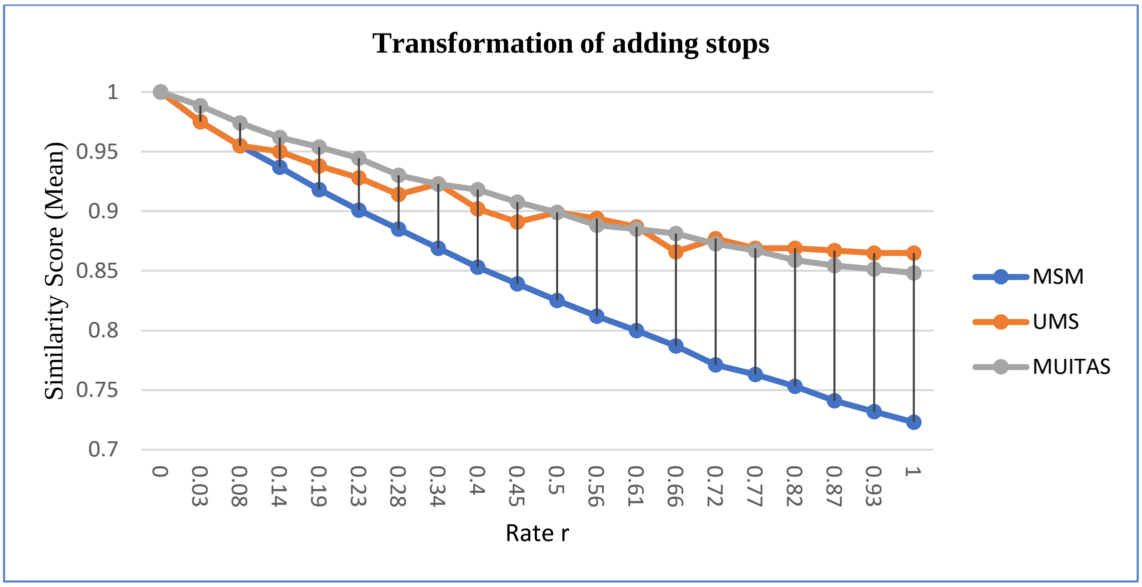
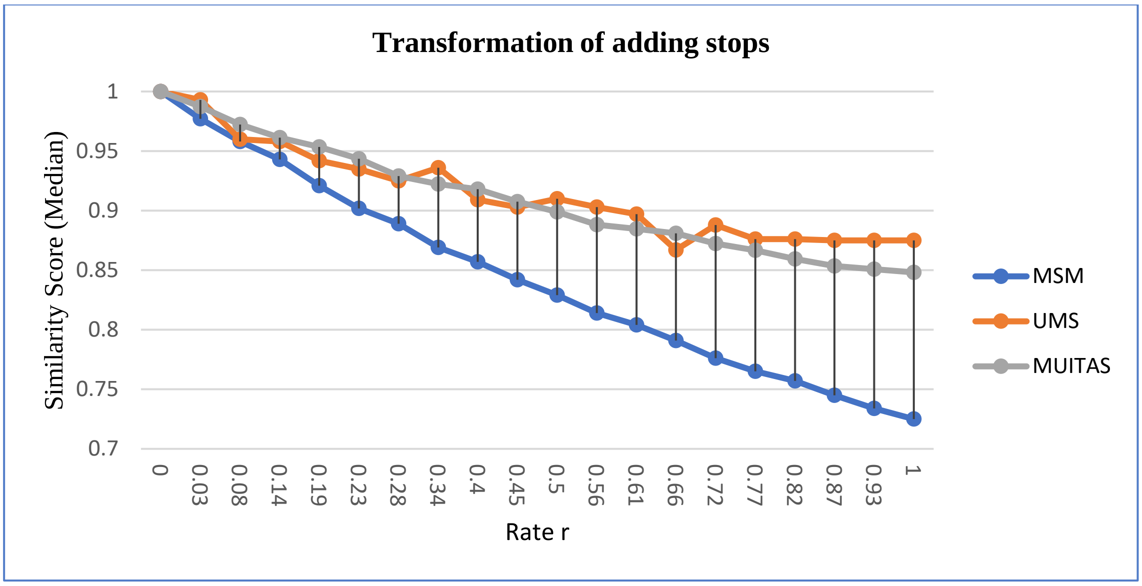


Figure 4-12: Comparison of similarity measures (adding stops) (Median)



From the figures illustrated above we notice the following: i) all the similarity measures are robust to this transformation as they maintain high similarity score with the lowest value being equal to 0.73 when the rate has value 1 and use the similarity measure MSM. However, UMS is the most robust similarity measure on most rate values. ii) we can clearly distinguish that this transformation has the biggest impact on the similarity method MSM and has similarity score difference from 0.05 to 0.15, iii) the similarity measures UMS and MUITAS have almost the same similarity score on all rate values.

4.2.3. Position change of stops

The results from the transformation of changing the position of the stops from the semantic trajectories and the multiple-aspect trajectories are presented on the following figures:

Figure 4-13: Transformation of position change of stops (MSM)

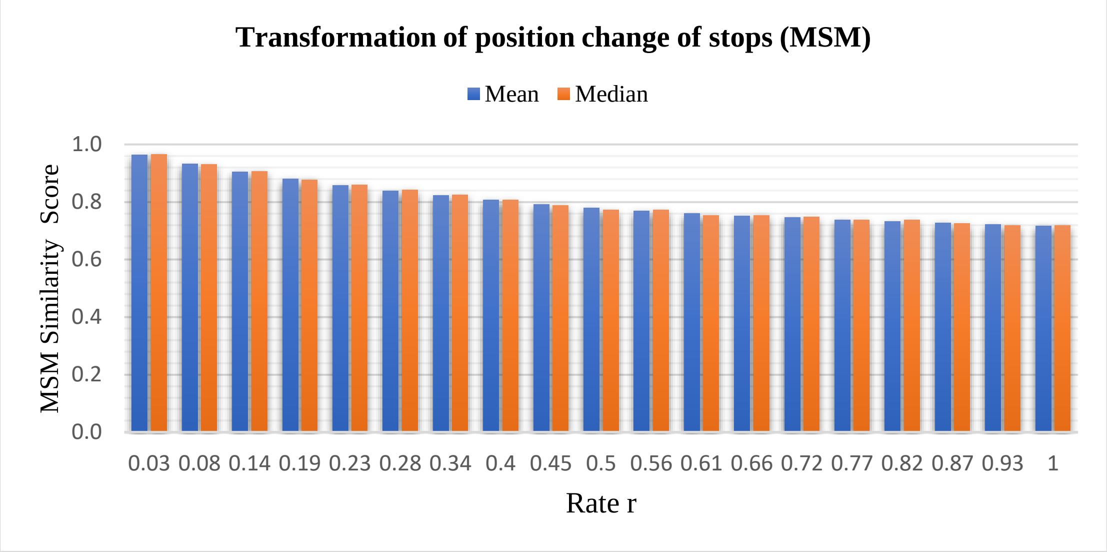


Figure 4-14: Transformation of position change of stops (UMS)

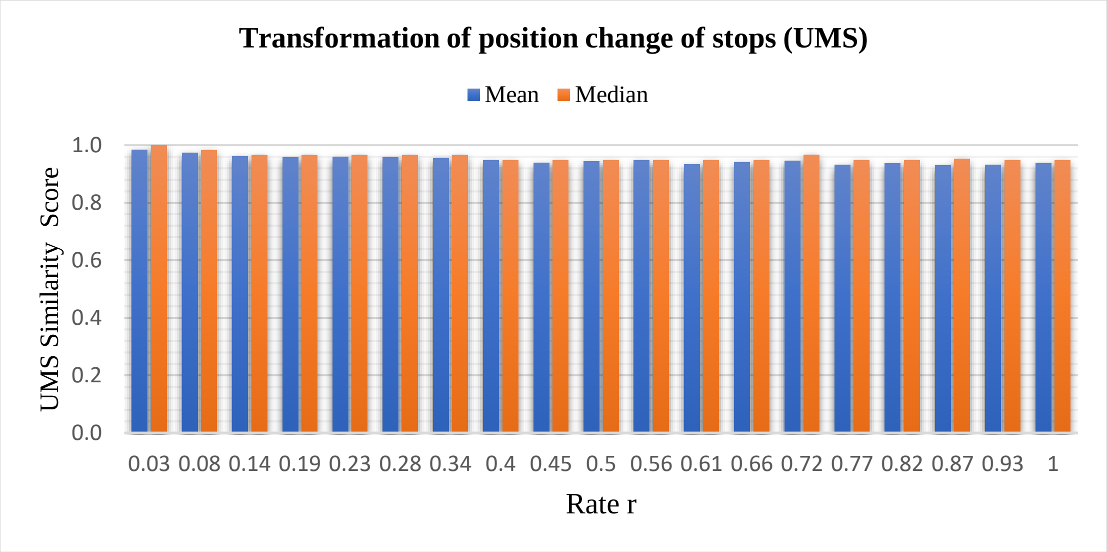
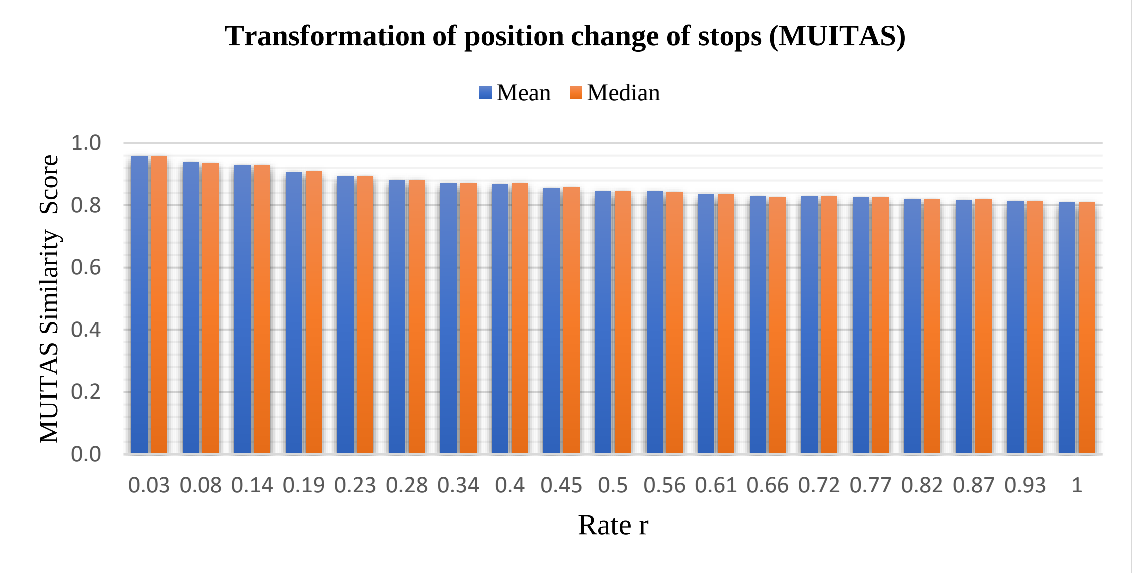


Figure 4-15: Transformation of position change of stops (MUITAS)



In order to better examine the impact of the transformation of changing the position of the stops comparatively on each similarity measure, we will present the mean and the median similarity score of the seed trajectory in relation to the transformed trajectories for each similarity measure on the below figures.

Figure 4-16: Comparison of similarity measures (position change of stops) (Mean)

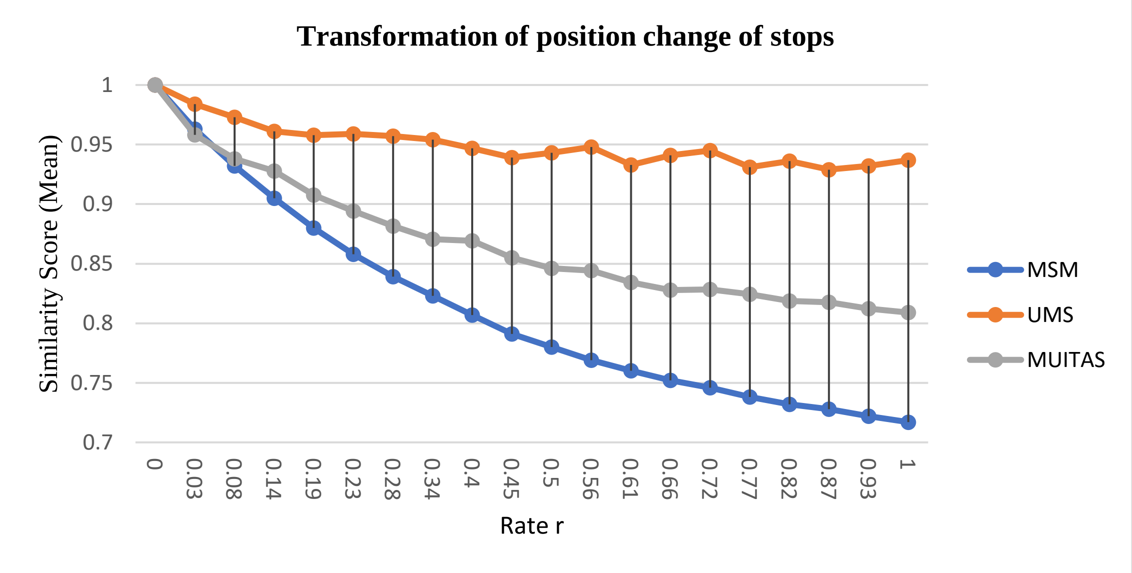
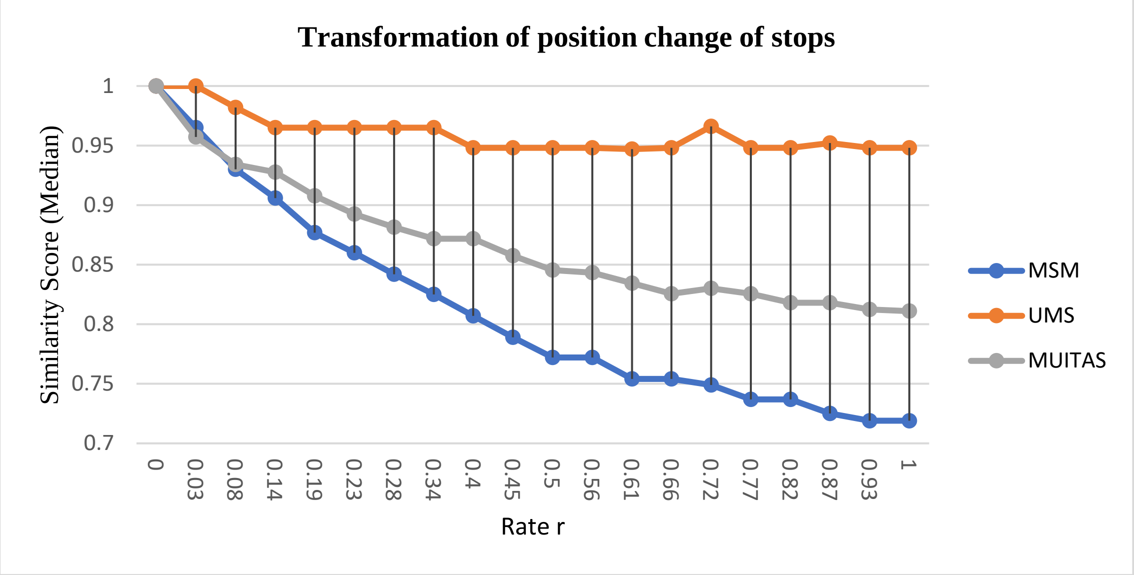


Figure 4-17: Comparison of similarity measures (position change of stops) (Median)



From the figures above we draw the following conclusions: i) this transformation has almost no effect on the similarity measure of UMS regardless of the value of rate r, as we notice that its similarity score fluctuates around 0.95 and therefore UMS has the most robustness on the transformation of position changing of the stops, ii) the similarity measure MUITAS have slightly higher similarity score than MSM on all rate values with a difference in their similarity score around 0.02-0.09 and iii) we notice that this transformation has small impact on MSM and MUITAS, increasing as we increase the value of rate, as they maintain high similarity score.

4.2.4. Replacement of stops

The results from the transformation of replacement of stops from the semantic trajectories and the multiple-aspect trajectories are presented on the following figures:

Figure 4-18: Transformation of replacement of stops (MSM)

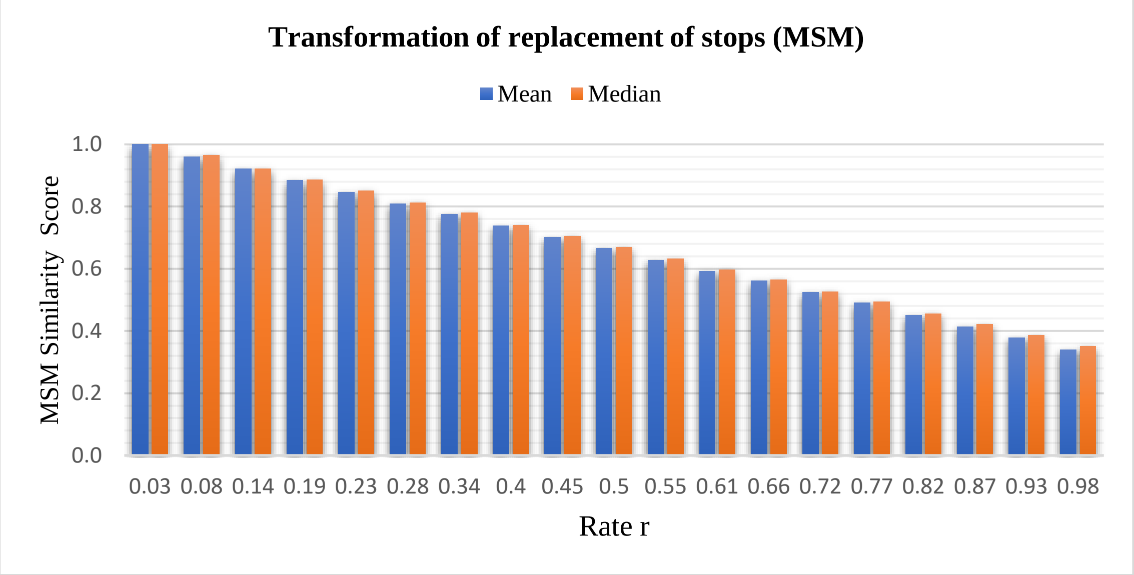


Figure 4-19: Transformation of replacement of stops (UMS)

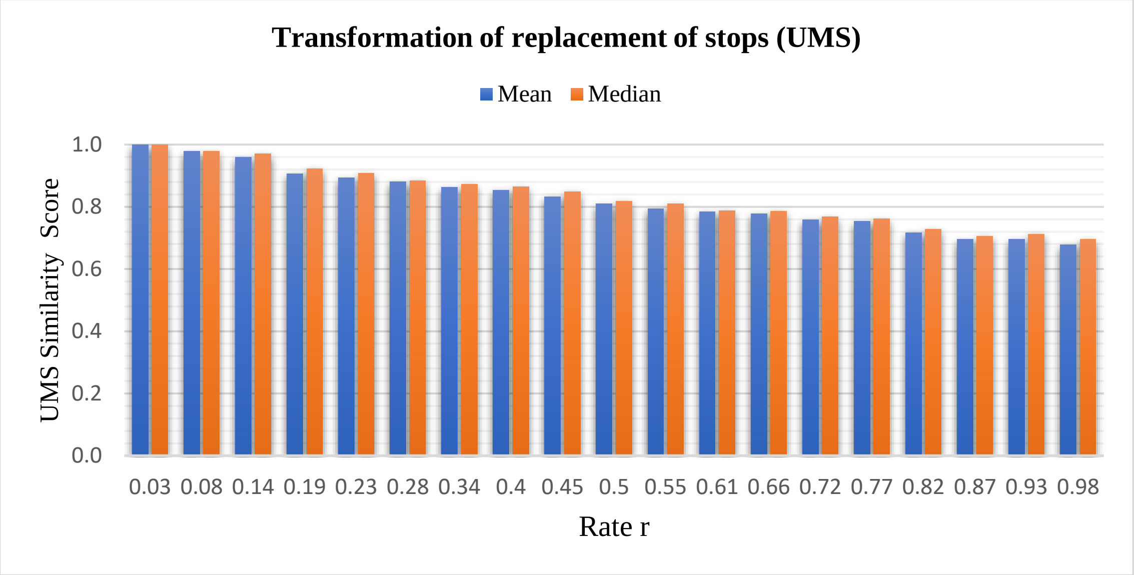
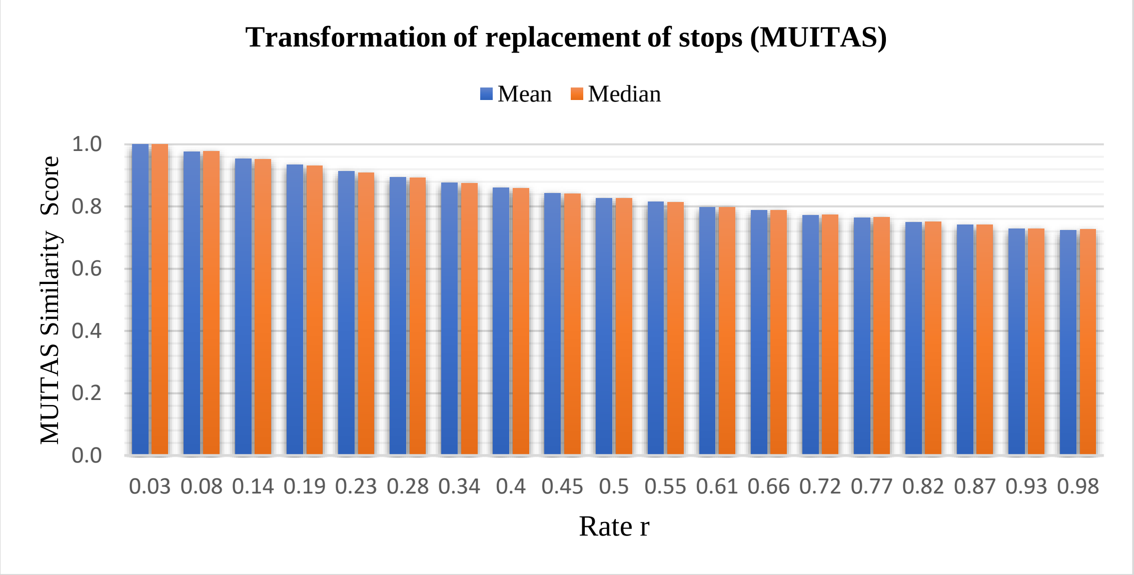


Figure 4-20: Transformation of replacement of stops (MUITAS)



In order to better examine the impact of the transformation of replacement of the stops comparatively on each similarity measure, we will present the mean and the median similarity score of the seed trajectory in relation to the transformed trajectories for each similarity measure on the below figures.

Figure 4-21: Comparison of similarity measures (replacement of stops) (Mean)

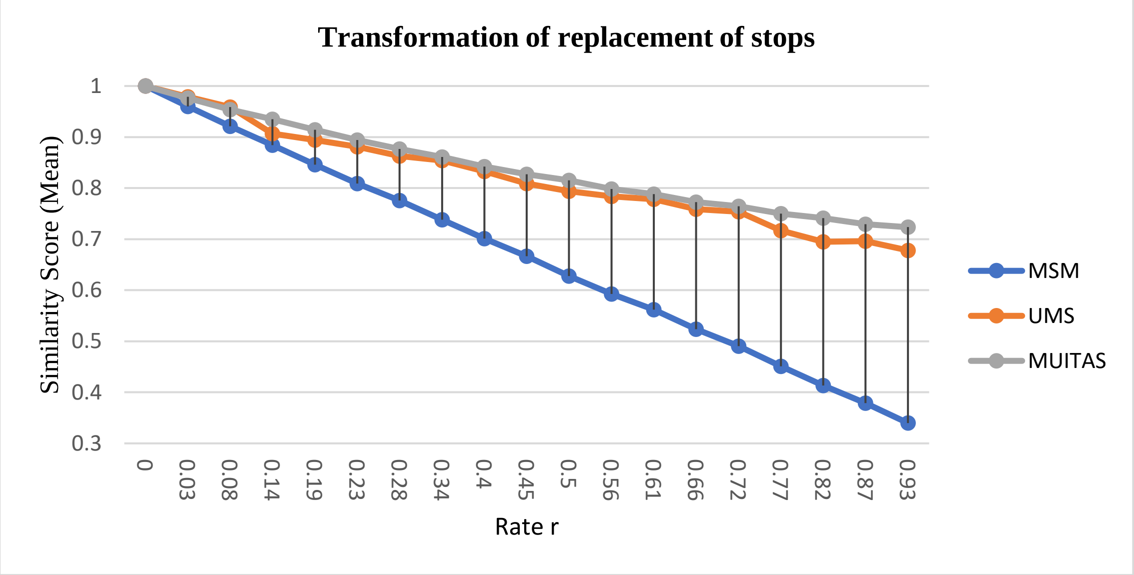
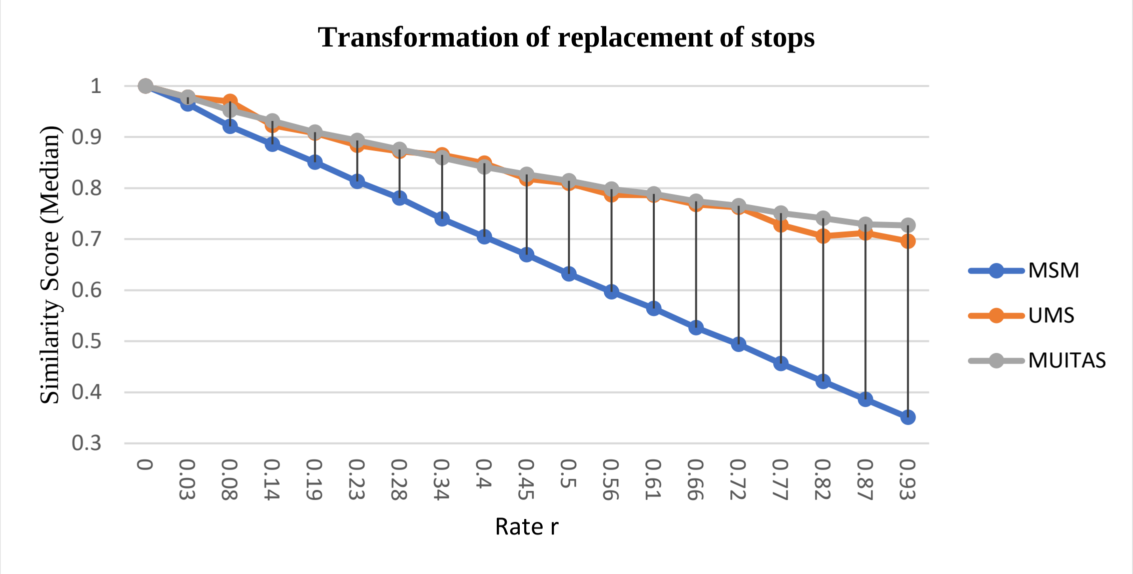


Figure 4-22: Comparison of similarity measures (replacement of stops) (Median)



From the figures illustrated above we notice the following: i) the similarity measures UMS and MUITAS have almost the same similarity score on all rate values with a really small difference and also these methods are very robust, as their lower similarity score is around 0.7 at rate 0.93 and higher, ii) MSM similarity score seems to be steadily decreasing. The higher the value of rate r the smaller the similarity score, which is equal to 0.35 at its lowest score, making this method not so robust when we replace more stops.

4.2.5. Replacement of possible matching stops

The results from the transformation of replacement of possible matching stops from the semantic trajectories and the multiple-aspect trajectories are presented on the following figures:

Figure 4-23: Transformation replacement of possible matching stops (MSM)

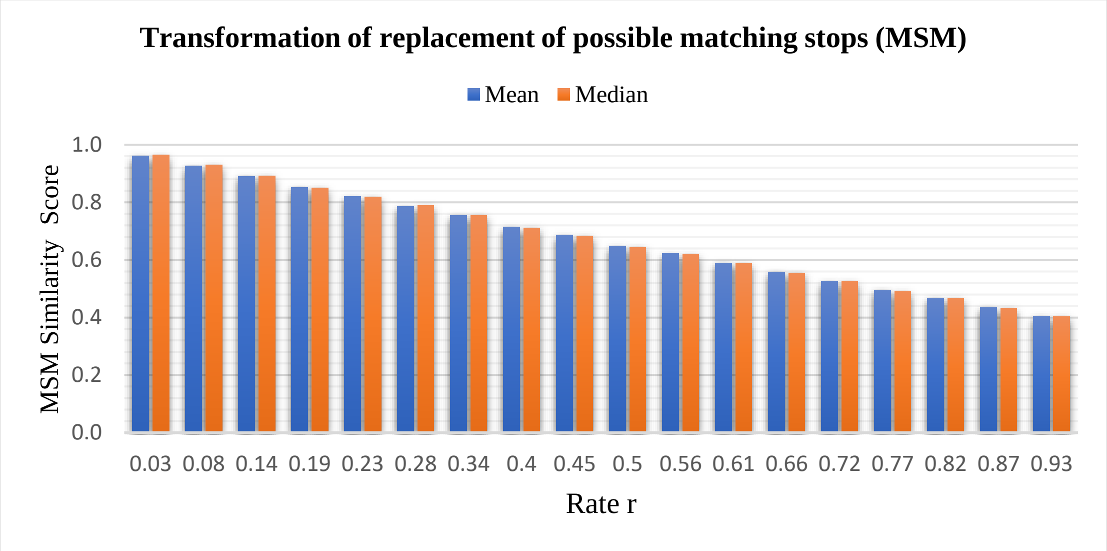


Figure 4-24: Transformation replacement of possible matching stops (UMS)

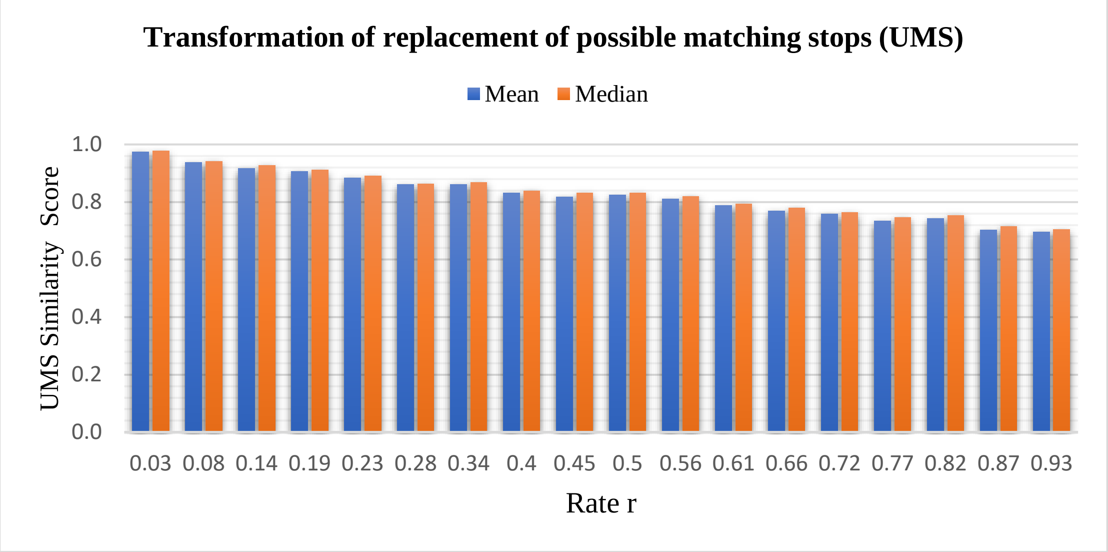
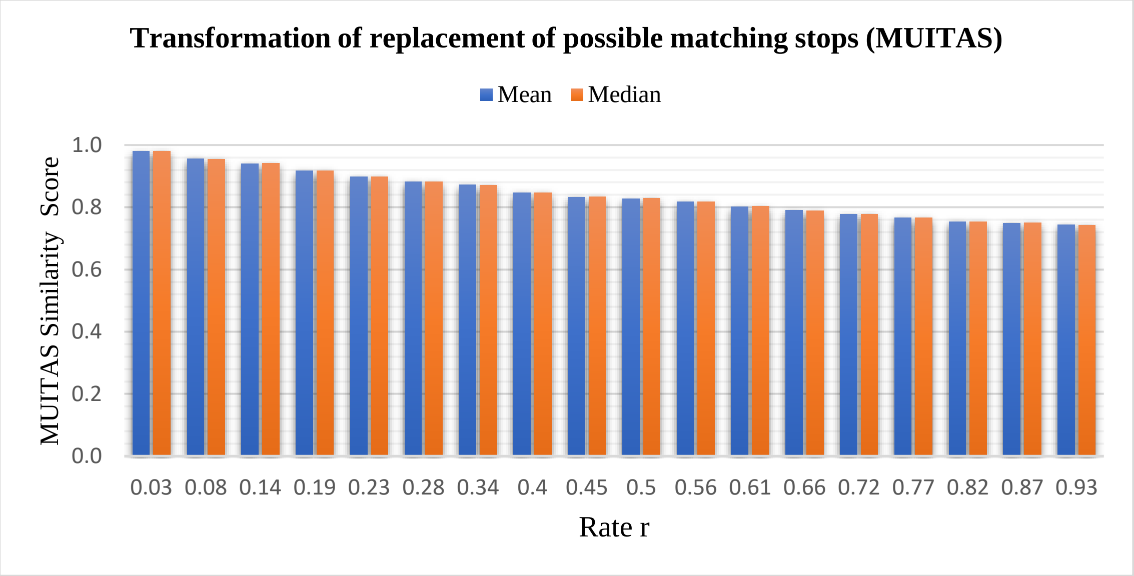


Figure 4-25: Transformation replacement of possible matching stops (MUITAS)



In order to better examine the impact of the transformation of replacement of possible matching stops comparatively on each similarity measure, we will present the mean and the median similarity score of the seed trajectory in relation to the transformed trajectories for each similarity measure on the below figures.

Figure 4-26: Comparison of similarity measures (replacement of possible matching stops) (Mean)

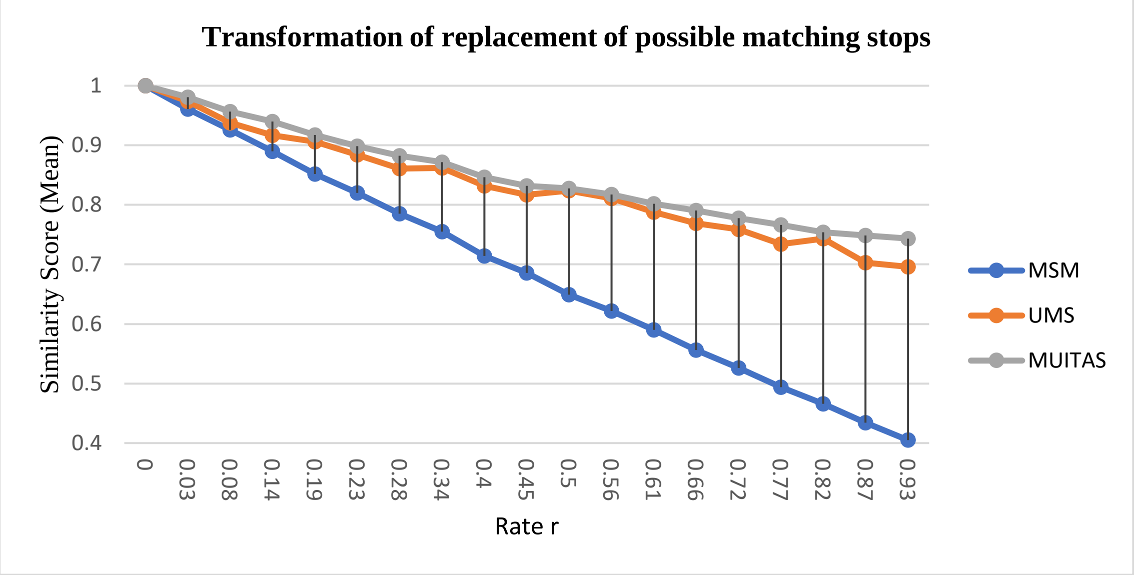
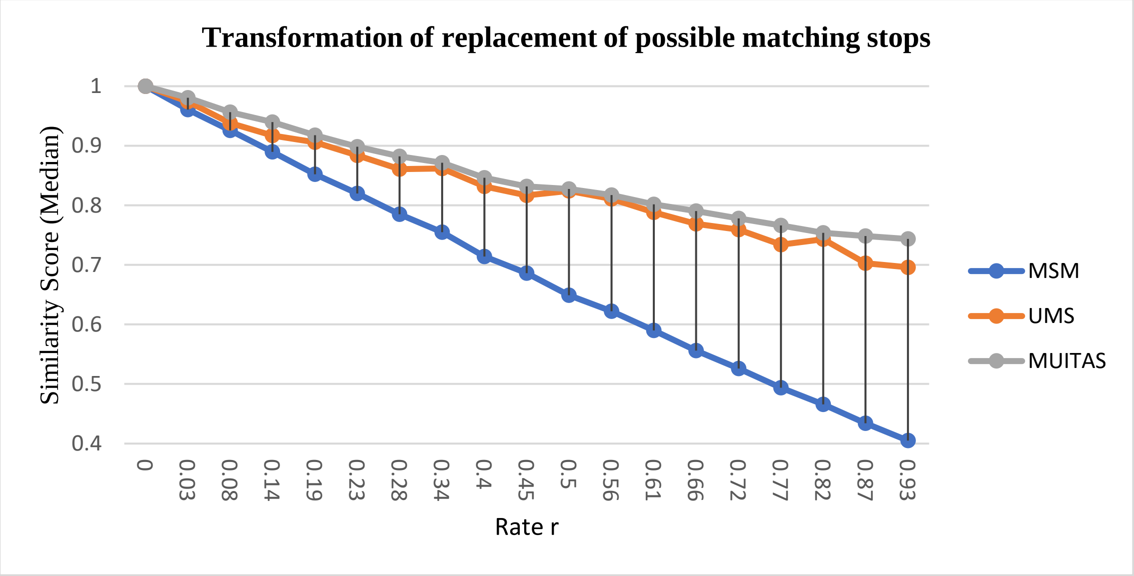


Figure 4-27: Comparison of similarity measures (replacement of possible matching stops) (Median)



The results from this transformation is quite similar to the previous results we had. In more detail from the figures illustrated above we draw the following conclusions: i) the similarity measures UMS and MUITAS have almost the same similarity score on all rate values with a small percentage of a larger difference in relation to the transformation of replacement of stops, and also these methods are very robust, as their lower similarity score is around 0.7 at rate 0.93 and higher, ii) MSM similarity score seems to be steadily decreasing. The higher the value of rate r the smaller the similarity score, which is equal to 0.4 at its lowest score, making this method not so robust when we replace more stops iii) MUITAS is the most robust similarity measure on this transformation with a small difference in relation to UMS.

Chapter 5. Conclusion

In this work we proposed a method which enables us to transform a trajectory in five different ways. The proposed technique is robust enough to consider multiple dimensions, where we can modify and change all the dimensions (space, time and semantics), including the attributes of the stops, for instance, the type of stops, the activity of the moving object, the point of interest, the weather, the price etc. To the best of our knowledge these transformations are the first to be constructed and tested on semantic trajectories that supports multiple dimensions (space, time and semantics) on the programming language R.

As aforementioned our proposed method allows us to apply five transformations over a multiple-dimension trajectory. The types of transformation are the followings: i) transformation of adding stops, ii) transformation of deleting stops, iii) replacement of stops with different elements, iv) replacement of possible matching stops and v) position change of stops. In order to control to what extent we will modify every trajectory we predefine a rate, called r which takes values from 0 to 1. Using these transformations we generated random sets of semantic trajectories or multiple aspect trajectories that were dissimilar from the seed trajectory, in many aspects, such as the time and the coordinates of the sample point, the order of the stop and the semantic information of the sample point.

The experiments performed by using data of a four day scenario of moving objects. We propose a method in which we transform a seed trajectory into many semantic trajectories according to rate r. In our thesis we are able to implement controlled transformation over the trajectories, which allows us to compare the impact of each transformation in every similarity measure that we use. After applying these transformations, we use the similarity measures MSM, UMS and MUITAS in order to test and draw conclusions about the effectiveness and robustness of all the types of transformations.

From the experiments performed in Chapter 4 we noticed that in every type of transformation applied, UMS and MUITAS are the most robust similarity measures. These measures maintain a high similarity score in every transformation, besides when we delete stops from the trajectory, which is logical because the higher the rate r we define the fewer stops the transformed trajectory have. As a result, when we define high rate r, we end up computing the similarity score of our seed trajectory with 19 stops in relation to 500 transformed trajectories with a lot lower stops. It is obvious that the similarity score will fluctuate in low levels for all similarity methods. Except of the transformation of deleting stops, in the rest transformations, MUITAS and UMS maintain a similarity score higher than 0.7, which makes them robust regardless of the value of rate r that we define. In contrast to these similarity, some types of transformations have big impact on the similarity measure MSM, making this method the least robust to all type of transformations. MSM have slightly lower similarity score at almost all values of rate r than MUITAS and UMS. In addition, when we transform the semantic trajectory by replacing the stops regardless if the new stop match or not with our old stop, the impact of this transformation is huge and is increasing as we increase rate r. One more observation we made on the transformations of adding stops and position changing of stops, is that MSM is robust to both of these transformation, as it maintains high similarity score, higher than 0.7 at all levels of r.

A conclusion we drew from the transformation of changing the position of the stops that seems very interesting is that this transformation has almost no impact on the similarity measure UMS. That is explained because UMS when computing the similarity of two trajectories tends to build ellipses and compare the ellipses created of one trajectory with the other one. As a result, when we make a transformed trajectory where we change the position of some stops, the transformed trajectory will have almost the same ellipses and their similarity score will remain at high levels. Furthermore, as we performed our experiment we noticed that UMS similarity score highly depends on the dataset we use. Our dataset which is about a 4 day scenario of a moving object clustered in 4 classes. UMS have small differences of similarity score for every different class we used, which is not something we expect as it doesn’t happen on the rest similarity measures.

Judging by the validity of the algorithms on the aforementioned datasets and by the results we presented on the previous chapters, we can see that the transformations applied on the datasets and as a consequence the random sets of semantic trajectories we generate, give us the ability to evaluate and compare the effectiveness of the similarity measures we desire in a controlled way (in our case MSM, UMS and MUITAS). Some extensions to our work that can be completed with slight changes to the algorithms, are the expansion of our transformations by constructing more types of trajectory transformations or a mix of them in order to compare the robustness of the similarity measures in more aspects, or trajectory transformations that consider both stops and moves.

Chapter 6. Bibliography

Alvares, L. O., Bogorny, V., Kuijpers, B., Macedo, J. F., Moelans, B., & Vaisman, A. (2007). A model for enriching trajectories with semantic geographical information. *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems. 22*, pp. 1-8. Seattle, Washngton: Transaction in GIS. doi:10.1145/1341041

Arboleda, F., Fernández, S., & Bogorny, V. (2017). Towards a semantic trajectory similarity measuring. *Indian Journal of Science and Technology 18*, *10*, pp. 1-14. doi:10.17485/ijst/2017/v10i18/103400

Baeza-Yates, Ricardo, & Ribeiro-Neto, B. (2011). *Modern information retrieval* (Vol. 463). New York: ACM press. doi:10.5555/1796408

Berndt, D. J., & Clifford, J. (1994). Using Dynamic Time Warping to Find Patterns in Time Series. *10*, pp. 359-370. KDD workshop.

Bogorny, V., Renso, C., Aquino, A. R., Siqueira, F. d., & Alvares, L. O. (2014). CONSTAnT – A Conceptual Data Model for Semantic Trajectories of Moving Objects. *18*, pp. 66-68. Transactions in GIS. doi:10.1111/tgis.12011

Chakri, S., & Raghay, S. (2016). Enriching trajectories with semantic data for a deeper analysis of patterns extracted. *International Conference on Hybrid Intelligent Systems*, *552*, pp. 209-218. Springer, Cham. doi:10.1007/978-3-319-52941-7_21

Chen, L., & Raymond, N. (2004). On the marriage of edit distance and Lp norms. *Proc. of 30th. International Conference on Very Large Data Bases' 04*, *30*, pp. 792-803. doi:10.5555/1316689.1316758

Chen, L., Özsu, T. M., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, (pp. 491-502). doi:10.1145/1066157.1066213

douglasapeixoto. (2018). *trajectory-distance-benchmark*. Retrieved from Github: https://github.com/douglasapeixoto/trajectory-distance-benchmark

Furtado, A. S., Pilla, L. L., & Bogorny, V. (2018). A branch and bound strategy for Fast Trajectory Similarity Measuring. *Data & Knowledge Engineering*, *115*, pp. 16-31. doi:10.1016/j.datak.2018.01.003Get

Furtado, A., Alvares, L., Pelekis , N., Theodoridis, Y., & Bogorny, V. (2018). Unveiling movement uncertainty for robust trajectory similarity analysis. *International Journal of Geographical Information Science, 1*, *32*, pp. 140-168. doi:10.1080/13658816.2017.1372763

Furtado, A., Kopanaki, D., Alvares, L., & Bogorny, V. (2016). Multidimensional Similarity Measuring for Semantic Trajectories. *Transactions in GIS, 20*, *2*, pp. 280-298. doi:10.1111/tgis.12156

Holt, T., Gineke, A., Reinders, M. J., & Hendriks, E. A. (2007). Multi-dimensional dynamic time warping for gesture recognition. *Thirteenth annual conference of the Advanced School for Computing and Imaging*, *300*, p. 1.

Lee, J.-G., Han, J., & Whang, K. (2007). Trajectory clustering: a partition-and-group framework. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, (pp. 593-604). doi:10.1145/1247480.1247546

Lehmann, A., Alvares, L., & Bogorny, V. (2019). SMSM: a similarity measure for trajectory stops and moves. *International Journal of Geographical Information Science 33.9*, (pp. 1847-1872). doi:10.1080/13658816.2019.1605074

Lin, D. (1998). An information-theoretic definition of similarity. *98*, pp. 296-304. lcml.

Liu, H., & Schneider, M. (2012). Similarity measurement of moving object trajectories. *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on GeoStreaming*, (pp. 19-22). doi:10.1145/2442968.2442971

Manning, Raghavan, P., & Schütze, H. (2008). Xml retrieval. *In Introduction to Information Retrieval.* Cambridze University Press.

Nanni, M., & Pedreschi, D. (2006). Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, *27*, pp. 267-289. doi:10.1007/s10844-006-9953-7

Parent, C., Spaccapietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., . . . Yan, Z. (2013). Semantic trajectories modeling and analysis. *ACM Computing Surveys (CSUR), 45(4)*, (pp. 1–32). doi:10.1145/2501654.2501656

Pelekis, N., Kopanakis, I., Kotsifakos, E., Frentzos, E., & Theodoridis , Y. (2011). Clustering uncertain trajectories. *Knowledge and information systems 28.1*, (pp. 117-147). doi:10.1007/s10115-010-0316-x

Pelekis, N., Sideridis, S., Tampakis, P., & Theodoridis, Y. (2016). Simulating Our LifeSteps by Example. *ACM Transactions on Spatial Algorithms and Systems*, *2*, pp. 1-39. doi:10.1145/2937753

Petry, L. M. (n.d.). *petry-2019-muitas*. Retrieved from Github: https://github.com/bigdata-ufsc/petry-2019-muitas

Petry, L., Ferrero, C., Alvares, L., Renso, C., & Bogorny, V. (2019). Towards semantic-aware multiple-aspect trajectory similarity measuring. *Transactions in GIS 23.5*, (pp. 960-975). doi:10.1111/tgis.12542

Pfoser, D., & Jensen, C. (1999). Capturing the Uncertainty of Moving-Object Representations. *International Symposium on Spatial Databases. Springer, Berlin, Heidelberg*, (pp. 111-131). doi:10.1007/3-540-48482-5_9

Ranacher, P., & Tzavella, K. (2014). How to compare movement? A review of physical movement similarity measures in geographic information science and beyond. *Cartography and geographic information science 41*, *3*, pp. 286-307. doi:10.1080/15230406.2014.890071

Shokoohi-Yekta, M., Hu, B., Jin, H., Wang, J., & Keogh, E. (2017). Generalizing DTW to the multi-dimensional case requires an adaptive approach. *Data Mining and Knowledge Discovery*, *31*, pp. 1-31. doi:10.1007/s10618-016-0455-0

Spaccapietra, S., Parent, C., Damiani, M. L., Macedo, J. A., Porto, F., & Vangenot, C. (2008). A conceptual view on trajectories. *Data & knowledge engineering*, *65*, pp. 126-146. Amsterdam, Netherlands. doi:10.1016/j.datak.2007.10.008

Su, H., Liu, S., Zheng, B., Zhou, X., & Zheng, K. (2020). A survey of trajectory distance measures and performance evaluation. *The VLDB Journal 29.1*, (pp. 3-32). doi:10.1007/s00778-019-00574-9

Vlachos, M., Gunopoulos, D., & Kollios, G. (2012). Robust similarity measures for mobile object trajectories. *Proceedings. 13th International Workshop on Database and Expert Systems Applications. IEEE*, (pp. 721-726). doi:10.1109/DEXA.2002.1045983

Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. *Proceedings 18th international conference on data engineering. IEEE*, (pp. 673-684). doi:10.1109/ICDE.2002.994784

Wan , Y., Zhou, C., & Pei, T. (2017). Semantic-geographic trajectory pattern mining based on a new similarity measurement. *ISPRS International Journal of Geo-Information 6.7*, (p. 212). doi:10.3390/ijgi6070212

Wang, H., Su, H., Zheng, K., Sadiq, S., & Zhou, X. (2013). An effectiveness study on trajectory similarity measures. *Proceedings of the Twenty-Fourth Australasian Database Conference*, *137*, pp. 13-22. doi:10.5555/2525416.2525418

Ying, J.-C. J., Lu, E. H.-C., Lee, W.-C., Weng, T.-C., & Tseng, V. (2010). Mining user similarity from semantic trajectories. *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, (pp. 19-26). doi:10.1145/1867699.1867703

Zhang, C., Han, J., Shou, L., Lu, J., & Porta, T. (2014). Splitter: Mining fine-grained sequential patterns in semantic trajectories. *Proceedings of the VLDB Endowment, 7(9)*, (pp. 769-780). doi:10.14778/2732939.2732949

Zhao, X. (2011). Progressive refinement for clustering spatio-temporal semantic trajectories. *Proceedings of 2011 International Conference on Computer Science and Network Technology. 4*, pp. 2695-2699. IEEE. doi:10.1109/ICCSNT.2011.6182522