SIMILARITY ANALYSIS OF USER TRAJECTORIES BASED ON HAVERSINE DISTANCE AND NEEDLEMAN WUNSCH ALGORITHM

Mohammad Jamhuri[*,**], **Mohammad Isa Irawan**[*], **Imam Mukhlash**[*]
[*]*Department of Mathematics, Faculty of Science and Data Analytics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, m.jamhuri@live.com, mii@its.ac.id, imamm@matematika.its.ac.id*
[**]*Department of Mathematics, Faculty of Science and Technology, UIN Maulana Malik Ibrahim, Malang, Indonesia, m.jamhuri@live.com*

Email Correspondence : m.jamhuri@live.com

Received : March 16, 2021 Accepted : July 8, 2021 Published : December 31, 2021

Abstract: This paper discusses the similarity between two trajectories using the Needleman Wunsch algorithm. The calculation steps are interpolating the trajectory, calculating the distance between the trajectory coordinates, identifying the equivalent length, transforming trajectories into a sequence of alphabetic letters, aligning the sequences, and measuring the magnitude of the similarity based on the alignment results. The similarity obtained is compared directly to the length of the trajectories shared by the two lines. The calculation results show that the accuracy of the alignment method reaches more than 90%.
Keywords: similarity of trajectories; linear interpolation; Haversine distance; global alignment

Abstrak: Dalam tulisan ini dibahas cara perhitungan persentase kesamaan dari dua buah lintasan menggunakan algoritma Needleman Wunsch dan perhitungan secara manual berdasarkan irisan dari lintasan-lintasan tersebut. Pada perhitungan menggunakan algoritma Needleman Wunsch, tahapan-tahapan yang dilakukan adalah menginterpolasi lintasan, menghitung jarak antara titik-titik koordinat dari kedua lintasan, mengidentifikasi jarak yang ekivalen, mengubah lintasan menjadi sekuens huruf alfabet, menyejajarkan sekuens, dan menentukan besarnya kesamaan berdasarkan hasil penyejajaran. Kesamaan yang diperoleh dari metode penyejajaran dibandingkan secara langsung dengan panjang jalur yang dilalui bersama oleh kedua lintasan, hasil perhitungan menunjukkan bahwa akurasi metode penyejajaran mencapai lebih dari 90%.
Kata kunci: kesamaan lintasan; interpolasi linier; jarak Haversine; penyejajaran global

Recommended APA Citation :
Jamhuri, M., Irawan, M. I., & Mukhlash, I. (2021). Similarity Analysis of User Trajectories Based On Haversine Distance and Needleman Wunsch Algorithm. *Elkawnie*, *7*(2), 263-276. https://doi.org/10.22373/ekw.v7i2.9232

Introduction

Everything that changes over time, either its position or its value, is called a moving object. Cars running on the highway, airplanes in the air, rockets fired, satellite travel, stock price movements, temperature changes in an area are examples of moving objects. Recording the movement of these objects will produce a set of

data in the form of coordinate points or values called trajectories (Mello et al., 2019). The trajectory is the history of the movement of an object expressed in an ordered pair set $\{r_1, r_2, \dots, r_m\}$, which represents the location or value at a particular time $\{t_1, t_2, \dots, t_m\}$, so that a trajectory is expressed as $r = \{(r_1, t_1), (r_2, t_2), \dots, (r_m, t_m)\}$ Where m is the number of observed points (Toohey & Duckham, 2015).

The magnitude of the trajectory similarity becomes an actual problem in several application domains. For example, customer trajectory analysis in a supermarket is used to organize product placement on shelves. Another example is to find unusual migration patterns of a group of suspicious birds or the presence of rare trajectories. Other domains include stock and data analysis, a travel route recommendation system, a recommendation system for friends based on interests, hobbies, locations visited, etc. The similarity of the trajectory can also be used to predict future phenomena such as cardinal directions, storms, and earthquakes (Wang et al., 2013).

One of the ways to calculate the similarity of the two trajectories is to use the alignment method as done by Cavojsky in (Čavojský et al., 2020). Cavojsky et al. used the Needleman Wunsch algorithm to align the paths that had previously been converted into alphabetic letter sequences. Sabarish et al. convert the trajectories into a graph and measure the similarity of the route based on edges and vertices (Sabarish et al., 2020). The similarity of the trajectory can be calculated using approximation equations built using a regression model or interpolation. Magdy et al. used three approaches to measure the similarity of pathways: regression, interpolation, and curve barcoding (Magdy et al., 2018).

In Cavojsky (Čavojský & Drozda, 2019), interpolation is carried out to equate the number of coordinate points on the two trajectories. This results in the absence of a maximum distance between the coordinate points in the trajectories, which may cause the distance between the sides of the two trajectories to be categorized as unequal because there are no adjacent points. This paper proposes a slightly different way of applying interpolation. Interpolation on the two paths is done by adding new points if the distance between the coordinate points of the measurement results on the two paths exceeds the specified equivalence distance (ε).

Previously, Chua and Foo (Chua & Foo 2015) used the Needleman Wunsch algorithm to detect suspicious activity on smart home devices and compare the results with the results obtained from the decision tree. Ju et al. (Ju et al., 2018) used the algorithm to identify smart card use and calculate transaction similarity scores to find its relationship with student achievement. Garhwal and Yan (Garhwal & Yan, 2019) used this algorithm to detect watermarks in images. They reported that the method could recognize watermarks in some image data with up to 100% accuracy. Cavojsky and Drozda (Čavojský et al., 2020) say that the Needleman Wunsch algorithm (Needleman & Wunsch, 1970) aligns DNA, RNA, and protein sequences but can also be used for path alignment.

In this paper, the Needleman Wunsch algorithm will be used to calculate the similarity of the two paths. The Needleman-Wunsch algorithm was initially designed to align sequences in the form of letters of the alphabet (Beretta, 2018), (Irawan et al., 2019). To align the trajectory in the form of coordinate points (numerical sequences), it is necessary to make adjustments by transforming them into alphabetic letter sequences. Meanwhile, to convert numeric sequences into alphabetical form, distance calculations are needed, including using Euclid distance (Čavojský et al., 2020), Manhattan distance (Chen et al., 2005) or Haversine distance (Anisya & Swara, 2017), (Sofwan et al., 2019). The distance calculation is needed to identify the equivalent coordinate points on the two paths. Next, labeling the coordinate points is carried out using the same character if the coordinates are identical and with different characters if the coordinates are not equivalent. To improve the accuracy of trajectory similarity, it is possible to interpolate the coordinates of the two tracks using linear interpolation before the transformation is carried out (Boubrahimi et al., 2018).

Material and Method

The main study of this paper is how to determine the similarity of the trajectory of two moving objects based on the coordinates sourced from the GPS (global positioning system) device. Suppose r and s are two trajectories recorded based on the position of moving objects in the form of a set of latitude-longitude coordinates, then r and s can be written as ordered sets $r = \{r_1, \dots, r_m\}$ and $s = \{s_1, \dots, s_n\}$, where $r_i = (r_{i,x}, r_{i,y})$ and $s_j = (s_{j,x}, s_{j,y})$ for $i = 1, \dots, m$ and $j = 1, \dots, n$. To calculate the similarity of the two paths, the completion steps are broadly stated in the flow chart in Figure 1 below. (Čavojský et al., 2020):

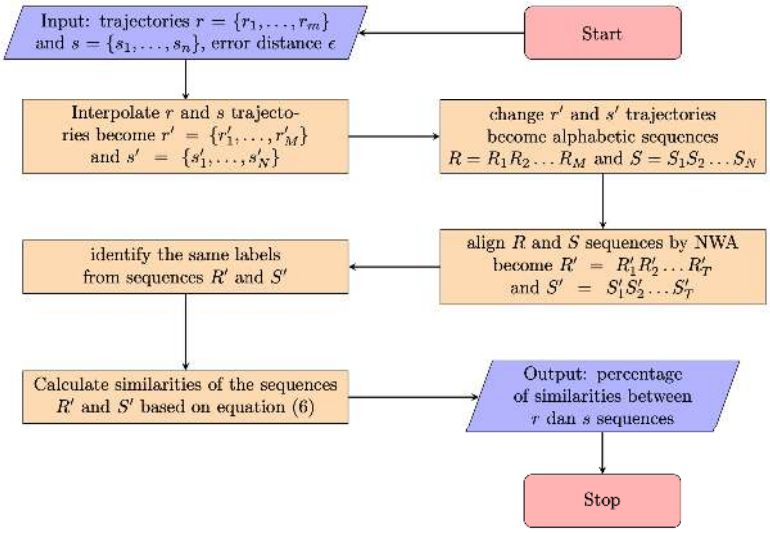

Figure 1. Flowchart to determine the similarity between two trajectories (Čavojský et al., 2020)

The steps for aligning the r and s trajectories in Figure 1 above are 1) interpolation; carried out to add new coordinate points between the existing coordinate points on the track, 2) transformation; changing the numerical sequence of the coordinate points in the path into a sequence of letters of the alphabet, 3) alignment; arrange matching characters or insert alternate characters so that the columns of each sequence contain identical characters, (4) count the number of matching or identical characters, (5) calculate path similarity. An essential concept in interpolation and path transformation is the distance between points, which is used to determine the number of interpolation points and to identify equivalent coordinate points.

Distance between points

The distance between two points in the form of latitude-longitude coordinates can be calculated using the Haversine equation introduced by James Inman in 1835 (Inman, 1849). That is if given two ordered pairs of geographic coordinates $A = (x_a, y_a)$ and $B = (x_b, y_b)$ where x_a,x_b, and y_a,y_b are latitude and longitude, respectively, then the distance between A and B is defined as

$$d(A,B) = 2R \arcsin\sqrt{\sin^2\left(\frac{x_a - x_b}{2}\right) + \cos x_a \cos x_b \sin^2\left(\frac{y_a - y_b}{2}\right)} \text{ …….. (1)}$$

Where R is the radius of the earth, which is ± 6,371 km. In addition to using the Haversine formula, the distance between A and B can be calculated using the Euclidean distance as in (Čavojský et al., 2020), namely

$$d(A,B) = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2} \text{ ……..(2)}$$

The distance obtained from equation (2) is still in geodetic form, so to convert it into kilometers, it must be multiplied by the radius of the earth R. Another method that is almost the same is using the Manhattan distance as used in Chen et al. (Chen et al., 2005), which is

$$d(A,B) = |x_b - x_a| + |y_b - y_a| \text{ ……..(3)}$$

Trajectory Interpolation

Linear interpolation is a method used to find the value of an unknown point from two points that form a linear line known in advance (Chapra & Canale, 1998). Path interpolation is a linear interpolation performed on two consecutive coordinate points in a path with a distance of more than ϵ. Suppose $A = (x_a, y_a)$ and $B = (x_b, y_b)$ are the points to be interpolated, then we can define the interpolation point $C = (x_c, y_c)$ as the midpoint of A and B with

$$x_c = \frac{x_a + x_b}{2},\ y_c = \frac{y_b - y_a}{x_b - x_a}(x_c - x_a) + y_a \text{ ……..(4)}$$

Interpolation is done recursively until all successive points in the path have a distance of less than ϵ. Interpolation steps in a path that has m initial coordinate points in more detail can be seen in the flowchart in Figure 2 below:

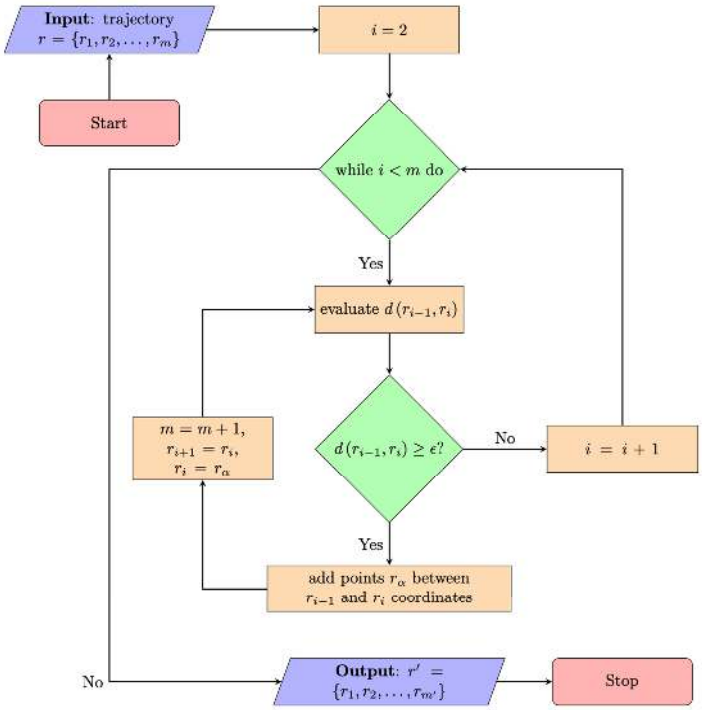

Figure 2. Flowchart Interpolasi Lintasan (Chapra & Canale, 1998)

There is an update on the number of coordinate points in trajectory interpolation, id est the old coordinate points plus the interpolated coordinate points. The number of interpolation points added can be more than one and depends on the distance between the two points and the selected value. For example, consider a path with four coordinates $s = \{s_1, s_2, s_3, s_4\}$ as shown in figure 3 below:

Figure 3. Coordinate points in path s before (left) and after adding interpolation points (right).

The track coordinates in Figure 3 (left) are $s_1 = (-7.9389, 112.5801)$, $s_2 = (7.9389, 112.5799)$, $s_3 = (-7.9388, 112.5799)$, and $s_4 = (-7.9377, 112.5799)$. By using equation (1), it can be calculated the distance between the closest points in the path, namely $d(s_1, s_2) = 30{,}812$ m, $d(s_2, s_3) =$

12.1431 m, and $d(s_3, s_4) = 119.7142$ m. To perform interpolation, select R_+ as the furthest distance between adjacent coordinate points after the path is interpolated, then apply the interpolation algorithm as shown in the flow chart in Figure 2. If we take $\epsilon = 20$, we will get new coordinate points in the path. Such that all the distances between adjacent points are less than 20. The results of the interpolation of path s can be seen in Table 1 below:

Table 1. Result of the interpolation s trajectory.

s	s'	x	y	distance (meters)
				$d(s'_{i-1}, s'_i),\ i = 2, \dots, 12$
s_1	s'_1	-7.9389	112.5801	-
	s'_2	-7.9389	112.5800	15.4060
s_2	s'_3	-7.9389	112.5799	15.4060
s_3	s'_4	-7.9388	112.5799	12.1431
	s'_5	-7.9387	112.5799	14.9643
	s'_6	-7.9385	112.5799	14.9643
	s'_7	-7.9384	112.5799	14.9643
	s'_8	-7.9383	112.5799	14.9643
	s'_9	-7.9381	112.5799	14.9643
	s'_{10}	-7.9380	112.5799	14.9643
	s'_{11}	-7.9379	112.5799	14.9643
s_4	s'_{12}	-7.9377	112.5799	14.9643

In table 1, the second column above, $s'_2, s'_5, s'_6, s'_7, s'_8, s'_9, s'_{10}, s'_{11}$ are the new points added to the path s as a result of interpolation, and in the 5th column, it can be seen that the distance between each successive coordinate is less than $\epsilon = 20$. The number of coordinate points on the path s after interpolation becomes 12 points, that is, four starting points ($s'_1 = s_1, s'_3 = s_2, s'_4 = s_3, s'_{12} = s_4$) plus eight interpolated points. Changes in the coordinate points on the path s can be seen in table 1 above.

In Figure 3 (right), the yellow circles are the original coordinate points in the s path, while the green $\times$ sign is the location of the interpolation points added to the path. Between points s_2 and s_3 no new points are added because $d(s_2, s_3) \geq 20$.

Trajectory Transformation

Interpolation can be done directly on a single path, but to convert the path into a sequence of alphabet letters. An essential part of transforming a path into an alphabetic letter sequence is identifying the equivalent coordinates of the two paths, then labeling them with the same letter if the coordinates are equivalent and with a different letter if the coordinates are not.

Suppose r and s are two paths that will be converted into a sequence of letters of the alphabet. The first step that must be done is to calculate the distance between the coordinate points in r and the coordinate points in s, then look for the coordinate points in r, which is less than from the coordinate points in s, then do the labeling. The flow for converting the path into a sequence of letters of the alphabet in more

detail can be seen in the *lintasantosequence* function in figure 5 and the flowchart in figure 6.

Table 2. Coordinate points of the r (left) and s (right) trajectories.

r	x	y
r_1	-7.9389	112.5801
r_2	-7.9389	112.5799
r_3	-7.9382	112.5799
r_4	-7.9378	112.5799
r_5	-7.9376	112.5799

s	x	y
s_1	-7.9389	112.5801
s_2	-7.9389	112.5799
s_3	-7.9388	112.5799
s_4	-7.9377	112.5799

First, we calculate all the distances between the coordinates in r and s using equation (1) and convert them into meters, while the results of the calculations can be seen in table 3, namely:

Table 3. Distance between the coordinate points in r and s trajectories

$d(r,s)$	s_1	s_2	s_3	s_4
r_1	**3.0182**	29.3034	30.6550	137.6043
r_2	28.4627	**3.1641**	**9.9130**	129.6270
r_3	84.4396	78.4808	66.5822	53.1365
r_4	122.6934	118.4714	106.5713	**13.1431**
r_5	153.5585	149.6993	137.8312	**18.1785**

In table 3 above, the distance between points whose value is less than 20 are marked in bold red letters. It can be said that distances less than 20 are equivalence points, i.e., points to be labeled using the same letter. The labeling process is as follows:

Based on Table 3, perform the following steps:

a) Distance $d(r_1, s_1) \leq 20$ then $r_1 = X_1$ and $s_1 = X_1$.
b) Distance $d(r_2, s_2) \leq 20$ then $r_2 = X_2$ and $s_2 = X_2$.
c) Distance $d(r_2, s_3) \leq 20$, but $d(r_2, s_2) < d(r_2, s_3)$ then $r_2 = X_2$ and $s_3 = S_1$.
d) Because of the distance $d(r_3, s_j) > 20$, for $j = 1, \ldots, n$, then $r_3 = R_1$.
e) Distance $d(r_4, s_4) \leq 20$ then $r_4 = X_3$ and $s_4 = X_3$.
f) Distance $d(r_5, s_4) \leq 20$, but $d(r_4, s_4) < d(r_5, s_4)$ so $r_5 = R_2$ and $s_4 = X_3$

From points a) to f) above, we obtain the following changed

Table 4. Result of the transformation

r	R
r_1	X_1
r_2	X_2
r_3	R_1
r_4	X_3
r_5	R_2

s	S
s_1	X_1
s_2	X_2
s_3	S_1
s_4	X_3

The transformation of trajectories r and s above is done without interpolating first. The path transformation process with or without interpolation is the same way. What differs is the number of coordinate points that must be transformed. In Table 4 above, there are three same letter labels on both paths, namely X_1, X_2, and X_3, meaning that there are three coordinate points in r and s which are equivalent to each other. As an illustration, it can be seen in figure 4 below.

Figure 4. The equivalent coordinates

In Figure 4, the equivalent coordinates are marked using a red circle which contains a green circle (part of the r path) and a yellow circle (part of the s circle). The figure shows that $r_1 \equiv s_1$, $r_2 \equiv s_2$, and $r_4 \equiv s_4$. Actually, $r_2 \equiv s_3$ and $r_5 \equiv s_4$ are also, since $d(r_2, s_2) \leq d(r_2, s_3)$ and $d(r_4, s_4) \leq d(r_5, s_4)$ we pass s_3 and r_5 and are considered not equivalent to another point like a point r_3.

```
function [a, b] = lintasantosekuens(r, s)
E = jarak(r, s)
a = buatlabelR(E)
b = buatlabelS(E)
```

```
function E = jarak(r, s)
for i = 1:m
    for j = 1:n
        d = ||r_i − s_j||_2
        if d ≤ ϵ
            E_{i,j} = 1
        else
            E_{i,j} = 0
```

```
function r = buatlabelR(E)
k = 0, l = 0, r' = ""
for i = 1:m
    if sumrow(E, i) = 0
        l = l + 1
        r' = r' + "R_l"
    else
        n = caribaris(E_i, 1)
        for j = 1:n
            k = k + 1
            r' = r' + "X_k"
```

```
function s = buatlabelS(E)
k = 0, l = 0, s' = []
for j = 1:n
    if sumcol(E, j) = 0
        l = l + 1
        s' = s' + "S_l"
    else
        m = carikolom(E_j, 1)
        for i = 1:m
            k = k + 1
            s' = s' + "X_k"
```

Figure 5. Algorithm of the trajectories transformation

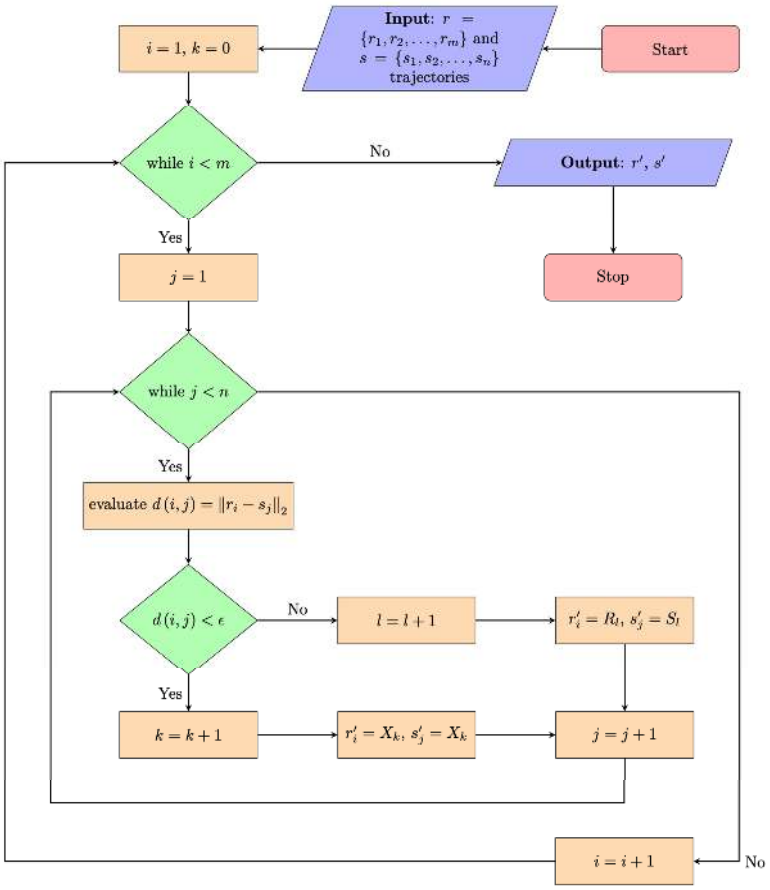

Figure 6. Flowchart of trajectory transformation into alphabetic letter sequence.

The similarity of the trajectories

Trajectory similarity, as defined by Cavojsky (Čavojský et al., 2020), is based on the number of identical coordinate positions and the number of gaps of the two aligned sequences, namely:

Definition (similarity). The paths r and s, which consist of m and n coordinate points, are equal

$$\frac{match}{max(m,n)} \geq \alpha \quad and \quad maxgap \leq \beta \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

Where #match is the number of matching characters and #maxgap is the largest number of gaps in the gap subset.

Using the definition above, We express the percentage of similarity of two paths as

$$similarity\ (\%) = \frac{match}{max(m,n)} \times 100\% \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(6)$$

A match is defined as the same bases in the DNA sequences and located at the exact position of two DNA strands. In paths, a match is the equivalent coordinates on

both paths represented by characters in the same place in both sequences. The parameters m, n in equation (5) are the length of sequence R and sequence S, respectively. So in Figure 4 above, we get $match = 3$, $m = 5$, and $n = 4$, so the similarity of the paths R and S above is

$$similarity\ (\%) = \frac{3}{max(5,4)} \times 100\% = \frac{3}{5} \times 100\% = 60\% \quad \text{.....(7)}$$

Result and Discussion

The number of interpolated points added to a trajectory depends on how much maximum distance (ϵ) is desired after the path is interpolated. The smaller the value of giving, the more interpolation points will be generated, and vice versa. Path interpolation increases the accuracy of the similarity between paths on the same path by estimating the points using the line formed by the points on the path. Interpolation can confirm which parts of a path have the same direction and path.

Example 1

Look at trajectories A and B in Figure 7 below, the length of track A (green color) is 277.98 meters, and the length of track B (yellow color) is 317.90 meters. The similarity between paths A and B is the length of the intersection paths compared to the longest path. Paths A and B are on the same path as far as 107.13 meters, so the similarity value is

$$similarity\ (\%) = \frac{length\ of\ the\ intersection\ trajectories}{max(length\ A, length\ B)} \times 100\ \% = \frac{107.13}{max(277.98,317.90)} \times 100\ \% = 33.7\% \quad \text{.....(8)}$$

The length of the intersection of the paths traversed by the two objects, namely the thick blue line, is visually easy to recognize and find by the human eye but difficult for the computer. Therefore, as described in the previous section above, we need easy methods to do by programs or computers.

Figure 7. Paths A and B Before interpolation (left) and after interpolation (right)

In figure 7 (left) above, the equivalent coordinates of the two paths, which are less than 15 meters away (ϵ), are marked with a red circle, so there are four equivalent coordinate points. The number of points on path A there are seven pieces, while for B, there are ten pieces; based on the definition of similarity in the previous section, the similarity value of the two paths is

$$similarity\ (\%) = \frac{total\ of\ equivalent\ points}{max(number\ of\ points\ in\ A, number\ of\ points\ in\ t\ B)} \times 100\% = \frac{4}{max(7,10)} \times 100\% = \frac{4}{10} \times 100\% = 40\% \quad \text{.....(9)}$$

In figure 7 (right), we interpolated the path using $\epsilon = 20$ so that the number of points on path A becomes 21 and the number of points on path B becomes 24. The equivalent coordinate points are 8 points. The percentage of similarity of paths A and B in Figure 7 (right) becomes

$$similarity\ (\%) = \frac{8}{max(21,24)} \times 100\% = \frac{8}{24} \times 100\% = 33.33\% \quad \text{.....(10)}$$

If the similarity obtained in equations (9) and (10) compare with the similarity in equation (8), the differences are 6.3% and 0.37%, respectively. While the accuracy of similarity before interpolation is 81%, the accuracy increases to 99% after interpolation.

Example 2

Look at tracks C and D in Figure 13 below, the length of track C (green color) is 3897.79 meters, and the length of track C (yellow color) is 4370.64 meters. The value of the similarity or similarity between paths C and D is the length of the path traversed together by the two objects from each path compared to the longest path. Tracks C and D are on the same track for a distance of 2989.65 meters, so the similarity value is

$$similarities\ (\%) = \frac{length\ of\ path\ intersection}{max(length\ of\ path\ A, length\ of\ path\ B)} \times 100\ \% = \frac{2989.65}{max(3897.79,4370.64)} \times 100\ \% = 68.4\% \quad \text{.....(11)}$$

The length of the intersection of the paths traversed by the two objects, namely the thick blue line, is visually easy to recognize and find by the human eye but difficult for the computer. Therefore, as described in the previous section, we need easy methods to do by programs or computers.

In Figure 8 below, the equivalent coordinates of the two trajectories, less than 15 meters apart (ϵ), are marked with a red circle, so there are ten equivalent coordinate points. Meanwhile, there are 19 and 42 coordinate points on paths C and D in Figure 8. Using the definition of similarity in the previous section, the similarity value between the two trajectories is

$$similarities\ (\%) = \frac{number\ of\ equivalence\ points}{max(number\ of\ points\ A, number\ of\ points\ B\ )} \times 100\% = \frac{10}{max(19,42)} \times 100\% = 23.81\% \quad ..(12)$$

Figure 8. Path C (green) path D (yellow), Slice of path C and D (bold line in blue), equivalent coordinate points (red)

Figure 9. The coordinates of path C (green color) and path D (yellow color) after interpolation, the equivalent coordinate points (red color)

In figure 9, we have interpolated the path using $\epsilon = 20$ so that the coordinate points on the C path become 267 and the number of points on the D path becomes 312. There are 186 equivalent coordinate points. So the percentage of similarity of paths C and D in Figure 9 becomes

$$similarities\ (\%) = \frac{186}{max(267,312)} \times 100\% = 59.62\% \quad .....(13)$$

If we compare the similarity obtained in equations (12) and (13) by the similarity in equation (11), the difference is 44.59% and 8.78%, respectively. The

accuracy of similarity before interpolation is 35%; after interpolation, the accuracy increases to 87%.

Conclusion

From the discussion, we can conclude that the accuracy of the similarity value between the two trajectories significantly increased by interpolating. We can calculate the path similarity without prior alignment, that is, by counting the number of the same characters or labels after transforming the sequence, then compared with the most extended sequence length from the path transformation results. Alignment is necessary if we want to use the definition of similarity to express two equal or different paths.

Acknowledgement

We would like to express our gratitude to PMU UIN Maulana Malik Ibrahim Malang for financial support to the continuation of my studies and also indirectly for the completion of this work

References

Anisya, A., & Swara, G. Y. (2017). Implementation of Haversine Formula and Best First Search Method in Searching of Tsunami Evacuation Route. *IOP Conference Series: Earth and Environmental Science*, *97*(1). https://doi.org/10.1088/1755-1315/97/1/012004

Beretta, S. (2018). Algorithms for strings and sequences: Pairwise alignment. In *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics* (Vols. 1–3). Elsevier Ltd. https://doi.org/10.1016/B978-0-12-809633-8.20317-8

Boubrahimi, S. F., Aydin, B., Schuh, M. A., Kempton, D., Angryk, R. A., & Ma, R. (2018). Spatiotemporal interpolation methods for solar event trajectories. *The Astrophysical Journal Supplement Series*, *236*(1), 23.

Čavojský, M., & Drozda, M. (2019). Comparison of user trajectories with the Needleman-Wunsch algorithm. *International Conference on Mobile Computing, Applications, and Services*, 141–154.

Čavojský, M., Drozda, M., & Balogh, Z. (2020). Analysis and experimental evaluation of the Needleman-Wunsch algorithm for trajectory comparison. *Expert Systems with Applications*, *165*(May 2020). https://doi.org/10.1016/j.eswa.2020.114068

Chapra, S. C., & Canale, R. P. (1998). *Numerical methods for engineers*.

Chen, L., Özsu, M. T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 491–502. https://doi.org/10.1145/1066157.1066213

Chua, S. L., & Foo, L. K. (2015). Sensor Selection in Smart Homes. *Procedia*

Computer Science, *69*, 116–124. https://doi.org/10.1016/j.procs.2015.10.012

Garhwal, A. S., & Yan, W. Q. (2019). BIIIA: a bioinformatics-inspired image identification approach. *Multimedia Tools and Applications*, *78*(8), 9537–9552. https://doi.org/10.1007/s11042-018-6551-y

Inman, J. (1849). *Navigation and Nautical Astronomy, for the Use of British Seamen*. F. & J. Rivington.

Irawan, M. I., Mukhlash, I., Rizky, A., & Dewi, A. R. (2019). Application of Needleman-Wunch Algorithm to identify mutation in DNA sequences of Corona virus. *Journal of Physics: Conference Series*, *1218*(1), 12031.

Ju, S., Park, S., Lim, H., Yun, S. B., & Heo, J. (2018). Spatial-data-driven student characterization: Trajectory sequence alignment based on student smart card transactions. *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Prediction of Human Mobility, PredictGIS 2018*, 1–7. https://doi.org/10.1145/3283590.3283591

Magdy, N., Abdelkader, T., & El-Bahnasy, K. (2018). A comparative study of similarity evaluation methods among trajectories of moving objects. *Egyptian Informatics Journal*, *19*(3), 165–177. https://doi.org/10.1016/j.eij.2018.03.001

Mello, R. dos S., Bogorny, V., Alvares, L. O., Santana, L. H. Z., Ferrero, C. A., Frozza, A. A., Schreiner, G. A., & Renso, C. (2019). Master: A multiple aspect view on trajectories. *Transactions in GIS*, *23*(4), 805–822.

Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, *48*(3), 443–453. https://doi.org/10.1016/0022-2836(70)90057-4

Sabarish, B. A., Karthi, R., & Kumar, T. G. (2020). Graph Similarity-based Hierarchical Clustering of Trajectory Data. *Procedia Computer Science*, *171*(2019), 32–41. https://doi.org/10.1016/j.procs.2020.04.004

Sofwan, A., Soetrisno, Y. A. A., Ramadhani, N. P., Rahmayani, A., Handoyo, E., & Arfan, M. (2019). Vehicle Distance Measurement Tuning using Haversine and Micro-Segmentation. *2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 239–243.

Toohey, K., & Duckham, M. (2015). Trajectory similarity measures. *SIGSPATIAL Special*, *7*(1), 43–50. https://doi.org/10.1145/2782759.2782767

Wang, H., Su, H., Zheng, K., Sadiq, S., & Zhou, X. (2013). An effectiveness study on trajectory similarity measures. *Conferences in Research and Practice in Information Technology Series*, *137*(February), 13–22.