Sensors & Transducers, Vol. 265, Issue 2, May 2024, pp. 17-26

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2024
http://www.sensorsportal.com

Network-wide Localization Algorithm Based on MEMS Sensor Data and Its Evaluation

Takayoshi YOKOTA
Faculty of Information Design, Tokyo Information Design Professional University,
2-7-1 Komatsugawa, Edogawa-ku, Tokyo, 132–0034, Japan
Tel.:+81-35-875-3117, fax: +81-35-875-3178
E-mail: yokota@tid.ac.jp

Received: 18 April 2024 / Accepted: 20 May 2024 / Published: 30 May 2024

Abstract: The Global Navigation Satellite System (GNSS) has been widely utilized as a method to accurately determine the position of a vehicle, but at present, the accuracy may be degraded depending on the radio wave reception conditions from the satellite. There have also been growing concerns about cyber-attacks on GNSS. We have been developing a method for estimating the position of a traveling vehicle using MEMS sensor data acquired using acceleration, gyro, and geomagnetic sensors. This algorithm has been evaluated in several fixed courses and obtained good results with a position error of less than 1 m. In the current study, we present a solution to extend this vehicle localization algorithm with MEMS sensor data by combining it with a map-matching algorithm in order to associate reference sensor data with road links. This will greatly reduce the computational time by avoiding exhaustive cross-correlation calculation to vast amount of sensor data.

Keywords: Localization, Acceleration, Gyroscope, Geomagnetic, MEMS sensor, Map-matching.

1. Introduction

As part of our ongoing research, we have developed a vehicle localization algorithm based on micro-electromechanical system (MEMS) sensors and demonstrated the error of 0.37 m [3] in a smooth traffic region and about 1.6 m [3] in fluctuating traffic through evaluation tests on a flat road along a river in Tottori, Japan. However, these results were for evaluations on the same fixed road sections. In order to generalize our vehicle localization algorithm, it must be capable of handling arbitrary road network reference data. This article extends the algorithm by including road link information on which the reference vehicle is running within the reference sensor data in order to associate sensor data with road link data. This will greatly reduce the cross-correlation calculation time because exhaustive comparison to all of the sensor data can be avoided by limiting the cross-correlation calculation to only those of adjacent links. By applying a map-matching algorithm, we can acquire not only the location (latitude and longitude) by RTK-GNSS but also the road link on which the vehicle is running. This article is an extension of an earlier conference paper [1] with new results concerning the multiple road link processing based on newly developed road link management table based on Digital RoadMap data base. In terrain-based vehicle localization research, there were no approach in which general road network is targeted. [3-8, 10-14] like this article.

2. Target Area of the Evaluation

Fig. 1(a) shows the target area of about 3×7 km located in Tottori, Japan. It covers an almost completely flat area facing the Japan Sea and running alongside a river. Fig. 1(b) shows the corresponding satellite image. Fig. 2 shows the reference vehicle

trajectory around node 42 together with its map-matched results (indicated by red dots).

Fig. 1. Target area of about 3×7 km and reference vehicle trajectory. Reference sensor data of total milage of 31.8 km was acquired.

Fig. 2. Satellite image of 3×7 km target area in Tottori, Japan.

This area was chosen because it is fairly flat and because it is difficult and challenging to localize a vehicle from the features of this road that are obtained by MEMS sensors.

3. Vehicle Localization in a Series of Road Links

3.1. Map-matching Algorithm for Preprocessing the Reference Data

Map-matching algorithms are well-known in the field of vehicle navigation systems or vehicle route analysis in the logistic research field. Since the GNSS data of the reference vehicle is high-precision thanks to using RTK-GNSS, most of the anxiety about its accuracy is needless. We developed an in-house map-matching algorithm [9] that is highly precise even though the GNSS data interval is quite long (e.g., 10 seconds). For our application, we utilize the Japan Digital Roadmap as road links. For RTK-GNSS data that has a cm-order accuracy and a 5-Hz sampling interval of the reference vehicle, the error rate of the map-matching algorithm is usually 0 %.

Fig. 3 shows examples of the reference vehicle trajectories by RTK-GNSS and the results of map-matching, where red and green respectively indicate slow and moderate speeds. With this map-matching, we can determine which road link the reference vehicle was running on and at what time. This means that each time series of the MEMS sensor data of the reference vehicle can be associated with each road link. Colored rectangles indicate the positions of the reference vehicle along with the direction in which it is heading. Small red dots are map-matched points that are projected to appropriate positions on road links. The map-matching algorithm [9] has a grid resolution of one meter, so the red points are plotted on 1-m grid points.

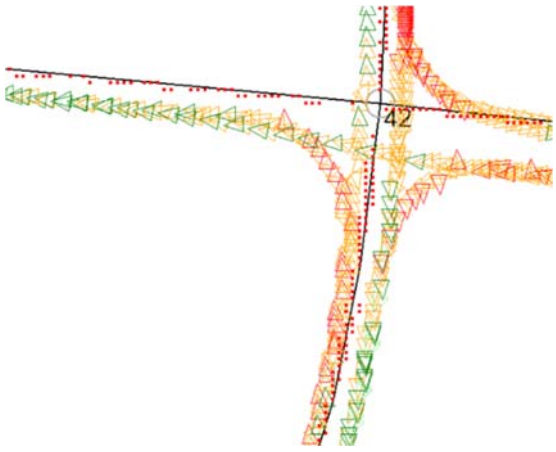

Fig. 3. Example of reference vehicle trajectories and its map-matched result. Reference vehicle trajectories around node 42 are shown. Map-matched results are indicated by red dots.

Fig. 4 shows the flow of constructing the road link management table referencing Digital Road Map. Database by a map-matching algorithm

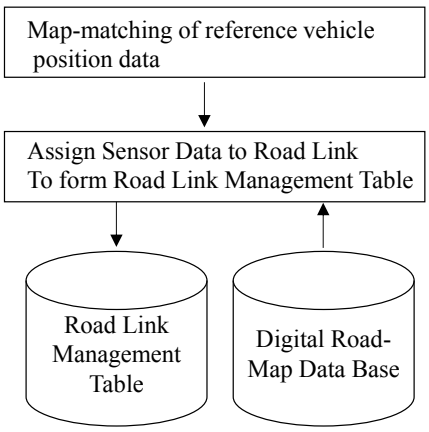

Fig. 4. Flow of constructing Road network management table.

3.2. Vehicle Localization Traversing a Road Link

The sensor data is a time series and the amount of it will continue to grow, so it needs to be organized in a sophisticated way. The sensor data are assigned to road link data by a map-matching algorithm. Fig. 5 shows the ground-truth trajectory of an evaluation vehicle by RTK-GNSS. As an RTK-GNSS U-box F9P [17] was used at operating frequency at $5H_z$.

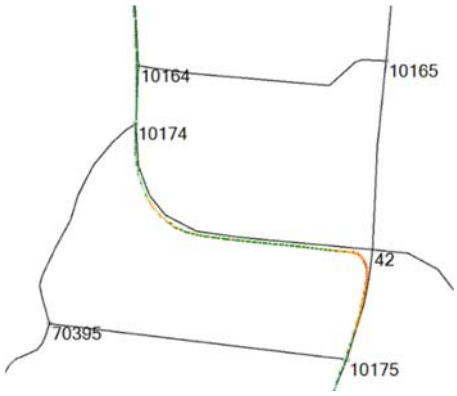

Fig. 5. Example of reference vehicle's trajectory. The vehicle is running in the north-west direction.

Numbers indicated on the map are node numbers of the digital road map. In this case, each road link is denoted by a number (e.g., 42_10174), which is a non-directed link. The reference vehicle proceeded in a north-west direction via nodes 10175, 42, 10174, and 10164. When traversing node 42, the vehicle localization algorithm [1-3] must compare the sensor data time series of the evaluation vehicle with those of associated neighbor links. For example, sensor data associated with links 42-10174, 42-10165, and 42-10175 are candidates that must be evaluated. Thanks to the digital road map data, there is no need to perform an exhaustive comparison to all sensor data stored in the database.

3.3. Associating Reference Sensor Data with Road Links

Fig. 6 shows the data structure for associating reference sensor data with road links. The system always has a current link id, so by following the current link_id table, it can tell where associated sensor data exists, and which are next candidate links. By defining this data structure, the cross-correlation calculation and vehicle localization [1], [2, 3] can be applied to a general road network as long as the reference sensor data exists. In this way, we can effectively focus on several candidate sensor data items at the same time – usually three per intersection, if we omit the rare case of a U-turn maneuver.

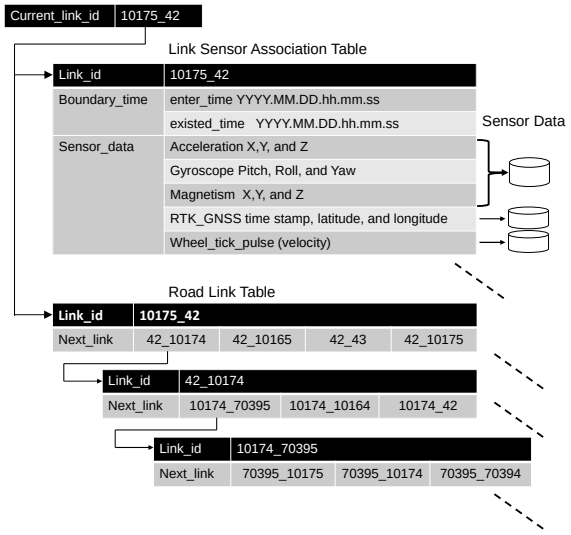

Fig. 6. Data structure for associating sensor data with road links.

Fig. 7 shows a screen shot of the Visual C++ 2022 software debugger, where six trips that passed through link 42_10175 are extracted: 2:13:24–2:13:33.6, 2:30:23.8–2:20:35.6, 2:45:41.6-2–45:48.2, 2:57:26.8–2:57:35.0, 3:20:18.4–3:20:19.2, and 3:47:48.8–3:47:55.4 (see the purple part). These are in Greenwich Mean Time (GMT), which is nine hours behind Japan Standard Time (JST). The travel time of each trip is 9.6 s, 11.6 s, 6.6 s, 8.2 s, 0.8 s, and 7.1 s. Among these, the travel time of 3:20:18.4–3:20:19.2, 0.8 s, is too short because of a miss-matching of the map-matching algorithm. Note that every trip except this trip took the same node 42 as the enter node and exit node (see the red part in Fig. 5). Therefore, this trip is ignored. Using the same procedure, we examined all links to determine how many trips

existed on each link and ignored any that were too short. Other links of the target area are processed in the same manner. This process extracts the information of what time the reference vehicle entered and exited each link. The time resolution is that of the RTK-GNSS equipped on the reference vehicle and the time resolution is 0.2 s. This resolution could possibly be improved by interpolation, but since map-matching processing is time-consuming for higher resolution position data, it is kept to 0.2 s here. Using the same procedure, we examined all links to determine how many trips existed on each link. Other links of the target area are processed in the same manner. This process extracts the information of what time the reference vehicle entered and exited each link. The time resolution is that of the RTK-GNSS equipped on the reference vehicle and the time resolution is 0.2 s. This resolution could possibly be improved by interpolation, but since map-matching processing is time-consuming for higher resolution position data, it is kept to 0.2 s here. The total milage of the reference data is 31.8 km.

[91]	{link_id=91 node1=42 node2=10175 ...}
link_id	91
node1	42
node2	10175
meshcode	533410
link_length	86
enter_time	0x00007ff612cdca8c {21324.0000, 23023.8008, 24541.5996, 25726.8008, 32018.4004, 34748.8008, -1.
existed_time	0x00007ff612cdcadc {21333.5996, 23035.5996, 24548.1992, 25735.0000, 32019.1992, 34755.3984, -1
enter_node	0x00007ff612cdcb2c {42, 10175, 42, 10175, 42, 42, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, ...}
exit_node	0x00007ff612cdcb7c {10175, 42, 10175, 42, 42, 10175, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, ...}
ref_MEMS_sensordata_start_index	0x00007ff612cdcbcc {34229, 85219, 131109, 166369, 234949, 317469, -1, -1, -1, -1, -1, -1, -1, -1, -1, ..
ref_MEMS_sensordata_end_index	0x00007ff612cdcc1c {34709, 85809, 131439, 166779, 234989, 317799, -1, -1, -1, -1, -1, -1, -1, -1, -1, ..
ref_pulse_data_start_index	0x00007ff612cdcc6c {922, 1941, 2858, 3563, 4933, 6583, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
ref_pulse_data_end_index	0x00007ff612cdccbc {932, 1953, 2864, 3571, 4934, 6589, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
previous_link_id	0x00007ff612cdcd0c {88, 610, 88, 610, 88, 88, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1}
traverse_count	5
next	0x0000000000000000 <NULL>
checked	1
node1_next	0x00007ff612cdcd6c {43, 10165, 10174, -1}
node2_next	0x00007ff612cdcd7c {10038, 70395, -1, -1}
node1_next_link_index	0x00007ff612cdcd8c {88, 89, 90, -1}
node2_next_link_index	0x00007ff612cdcd9c {405, 610, -1, -1}

Fig. 7. Example of link management table. This is a screen shot of the Visual C++2020 debugger showing the trip information of the reference vehicle along link 42_1015. This is the implementation of the data structure shown in Fig. 5.

3.4 Assigning MEMS Sensor and Wheel Pulse Data to Each Trip Traversing a Link

The sensor data is a time series of nine axis data corresponding to acceleration, angular velocity, and geomagnetism for three axes of x, y, and z. As the MEMS sensor, InvenSense MPU-9250 was used at 50 H_z sampling frequency [16]. As in the processing in the previous procedure, the beginning and the end of the record number of the MEMS sensor data are extracted by comparing the time stamp of the enter time and exit time of the link data in Fig. 7 (see the purple part) to those of the MEMS sensor data. For example, the second and fourth trips, which entered node 10175 and exited node 42 of the reference vehicle, started at 2:30:23.8 and ended at 2:30:35.6 and started at 2:57:26.8 and ended at 2:57:35.0 These trips are shown in Fig. 8(a) and (b).

These trips correspond to the MEMS sensor data that starts at index 85,219 and ends at 85,809 and starts at 166,369 and ends at 166,779 (see the green part of Fig. 7). The profiles of MEMS sensor data are shown Fig. 9(a) and (b), which correspond to the trips in Fig. 8(a) and (b). "Time of day" is depicted in seconds, as calculated by $hour \times 3{,}600 + minute \times 60 + sec$. From these results, the sensor data shows rather different aspects depending on whether the vehicle is running straight or turning when the link is an intersection link. Another important sensor is the wheel tick pulse, which serves as a velocity sensor when GNSS is not available. It might be possible to integrate the acceleration sensor to estimate the velocity, but such integration is difficult because eliminating the effect of gravity is extremely difficult. The corresponding wheel pulse sensor data indices are extracted from 1941 to 1953 and from 3563 to 3571 (see yellow part in Fig. 7). The profiles of the velocity obtained by the wheel pulse data are shown in Fig. 10(a) and (b).

(a) 2:30:23.8–2:20:35.6 (b) 2:57:26.8–2:57:35.0

Fig. 8. Examples of reference vehicle trips over a link.

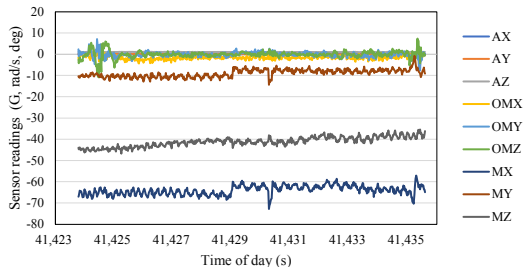

Fig. 9. (a) MEMS sensor data of reference vehicle.

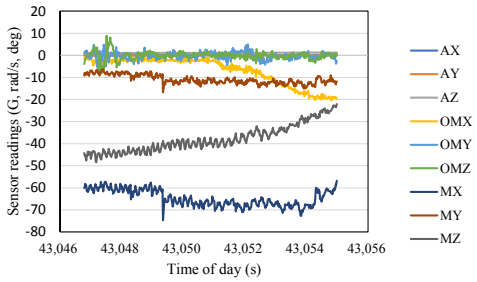

Fig. 9. (b) MEMS sensor data of reference vehicle.

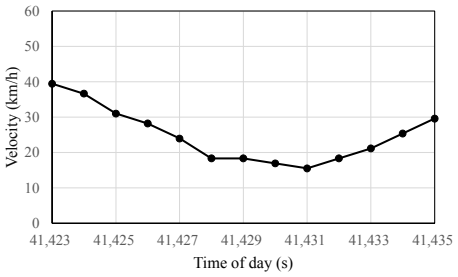

Fig. 10. (a) Velocity obtained by wheel pulse data (index 1941 to 1953).

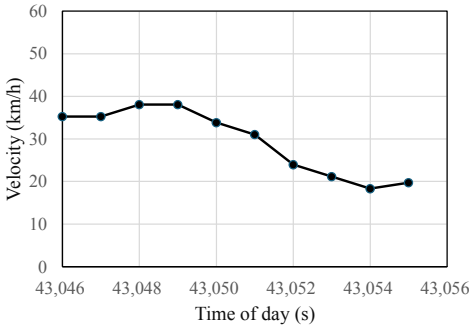

Fig. 10. (b) Velocity obtained by wheel pulse data (index 3563 to 3571).

3.5. Velocity Compensation of Vehicle Sensor Data and its Cross-Correlation

We obtained almost the same amount of evaluation vehicle data as reference vehicle data, but in this article, only a portion of the evaluation data is evaluated for simplicity (the overall evaluation will be reported in the near future). The trajectory of a sample evaluation vehicle is shown in Fig. 11. The satellite picture of the evaluation road network is also shown in Fig. 12. A portion of the evaluation vehicles MEMS sensor data is shown in Fig. 13 and its velocity profile in Fig. 14. The reference MEMS sensor data of the same portion of the trajectory is shown in Fig. 15 with velocity profile in Fig. 16. Comparing these Fig. 13-16, it can be seen that although the sensor data of the evaluation vehicle and those of reference vehicle show some similarities, with some variation in the time axis, there is a big difference in the speed data.

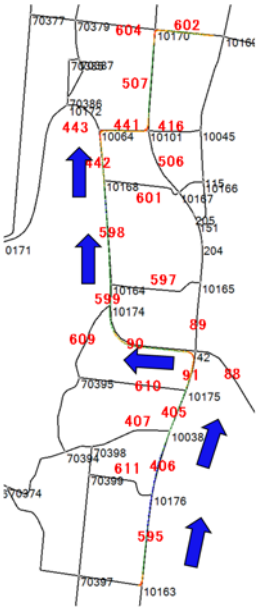

Fig. 11. Trajectory of the evaluation vehicle. Blue arrows indicate the direction of the evaluation vehicle. Red numbers indicate road link indices. Black numbers indicate node "numbers".

Fig. 12. Satellite picture of evaluation road network.

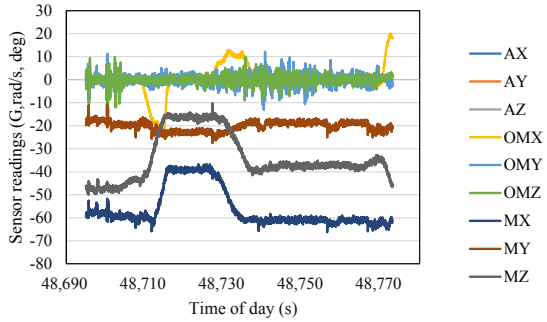

Fig. 13. MEMS sensor data of evaluation vehicle.

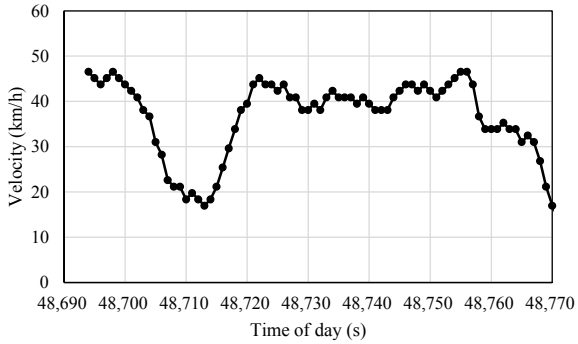

Fig. 14. Velocity profile of evaluation vehicle.

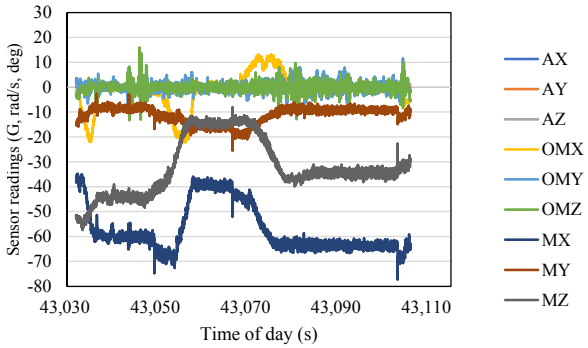

Fig. 15. Best matched reference MEMS sensor data.

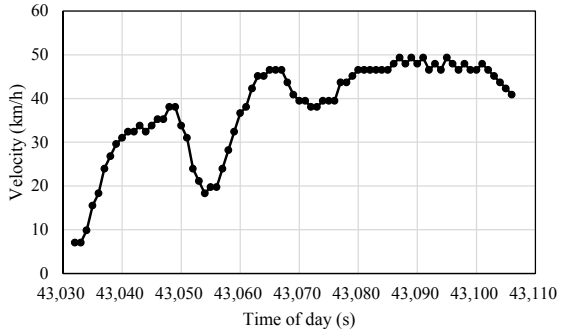

Fig. 16. Velocity profile corresponding to reference MEMS sensor in Fig. 11.

As shown in Fig. 11, the evaluation vehicle ran through six links via nodes 10163,10176,10038, 10175, 42, 10174, 10164, 10168, and 10064. The travel time was 107 s, starting at 4:31:00 and reaching node 10064 at 4:32:53.0.

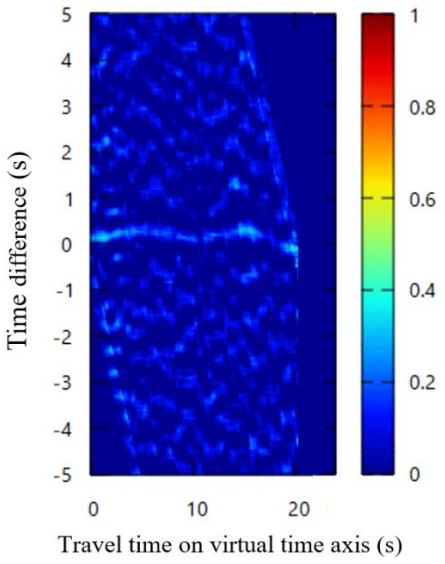

Fig. 17. Averaged cross-correlation function on link 10164_10168. A clear bright line can be seen near time

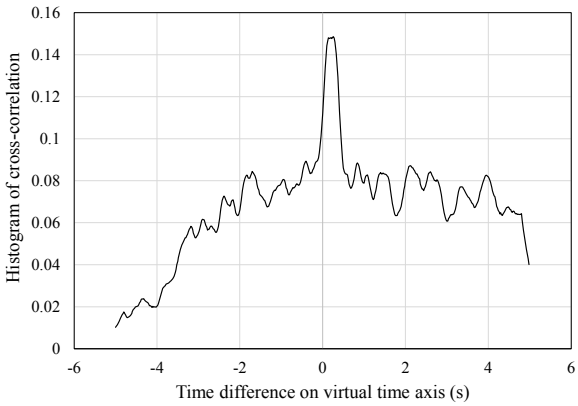

Fig. 18. Histogram of time difference. The histogram is calculated from the cross-correlation function of Fig. 15.

Vehicle velocity is usually changing all the time and it makes it difficult to calculate the cross-correlation function to find the best matched sensor data. To overcome this problem, The author developed velocity compensation algorithm

$$\varphi(t) = \int_{\tau=0}^{t} \frac{v(\tau)}{v_s} d\tau, \quad (1)$$

where $v(\tau)$ is the velocity of vehicle at time τ. τ is the time in real time axis. The vehicle velocity v can be easily obtained by multiplying wheel tick pulse by a scale factor. v_s is the standard velocity which is taken as 50 km/h in this article. $\varphi(t)$ is the transformed time axis assuming that the vehicle runs at the standard velocity. Of course, $\varphi(t)$ becomes linear to t if the

velocity profile $v(\tau)$ is a constant value. The virtual time axis φ(t) transforms the time series of the sensor data as if the vehicle was running at the standard velocity v_s. After transforming the raw sensor data into that of the virtual time axis. On the virtual time axis, both evaluation vehicle and reference vehicle should run at the same velocity v_s and the difference of the time between the two vehicles must be constant on the virtual time axis. If they start at the same point at the same time, then the time difference between them should be zero.

The normalized cross correlation function is given by

$$R(i,j) = = \frac{\sum_{k=0}^{N-1}(f_{i+j+k}-\bar{f}_{i+j})(g_{i+k}-\bar{g}_i)}{\sqrt{\sum_{k=0}^{N-1}(f_{i+j+k}-\bar{f}_{i+j})^2}\sqrt{\sum_{k=0}^{N-1}(g_{i+k}-\bar{g}_i)^2}}, \quad (2)$$

$$\bar{f}_{i+j} = \frac{1}{N}\sum_{k=0}^{N-1} f(i+j+k), \quad (3)$$

$$\bar{g}_i = \frac{1}{N}\sum_{k=0}^{N-1} g(i+k), \quad (4)$$

where f and g are MEMS sensor data of the evaluation vehicle and reference vehicle, respectively, both of them are transformed from actual time axis to virtual time axis and then resampled on the virtual time axis mentioned above. Resampling was done based on nearest neighborhood method. Index i is the time index of evaluation vehicle sensor data on virtual time axis and index j is that of time difference between the evaluation vehicle and reference vehicle on the virtual time axis. Index k is the time index for average window to calculate the cross-correlation of f and g. We then calculate $R(i,j)$ for each sensor data type, and average them using an optimum weight [2] to obtain a weighted cross-correlation function. The example of the weighted cross-correlation function is shown in Fig. 17 which corresponds to the cross correlation on link 10164_10168. The 'jet' colormap was used in gnuplot software. A clear bright line can be seen near time difference zero. The reason why the bright line appeared in the cross-correlation function is not strictly a horizontal line is due to the fact that the accuracy of velocity compensation is not enough. Time resolution of the wheel pulse is one second and the pulse itself is digitized as several fixed number pulses per a wheel round in the experiment setup and the resolution is limited. If we adopt more accurate and high resolution velocity data the bright line in Fig. 17 may have been more strictly horizontal. To estimate the time difference from the cross-correlation function such as shown in Fig. 17, the histogram was calculated and is shown in Fig. 18. A clear peak at near time difference zero is recognizable. This implies that the velocity compensation by Eq. (1) and the succeeding resampling procedure worked effectively.

4. Algorithm: for Searching of Best Reference Data Based on Road Link Management Table and Evaluation Results

As the evaluation vehicle proceeds, the system searches for candidates of MEMS sensor data based on the road link management table and find a best matched reference data. From the best matched reference data, the system estimates the location of the evaluation vehicle by substituting reference vehicle position with estimated time difference. The system assumes that the initial position of the evaluation vehicle is known, that is, the initial road link is assumed to be known. In this case the vehicle starts from the node 10163 in Fig. 11, and proceed to road link index 595. The position is estimated by correlating the sensor data of the evaluation vehicle with the reference sensor data which has been assigned to this road link. The window width of calculating the cross-correlation is set to 1 s. The schematic view of the algorithm is shown in Fig. 19. The sensor data was sampled at 50 H_z and the number of samples for calculating the cross-correlation is 50. Each time the system finds the end of the link of the best matched link it notices the end node, and then, the system searches for road links that connects to the node. The candidate road link may have reference data, and the system tries evaluate each reference data, and in a short time (1 s), the system decide which reference data is the best by examining the histogram and continues calculating the cross-correlation until it hits the end of the best matched road-link. This process is repeated until the evaluation data exhausts. All of the important information of reference data. has been stored in the road link management table shown in Figs. 20 and 21 for examples. The system starts by looking at the road link management table of Fig. 20. It first notices that the best matched reference data is the second one that that begins at 2;29:58.8. After continuing the cross-correlation calculation along virtual time axis, evaluation vehicle reached the end of the road link 595, the algorithm tries to pick up candidate road links according to the road link management table shown in Fig. 20. As the exit node is 10176, the algorithm searches for road links which connect to this node and finds link index 406 as a candidate. The system calculates cross-correlation between the evaluation vehicle and that of the sensor data which was assigned to this road link and finds that the second data marked red in Fig. 21 which begins at 2:30:07.2 gives the best result. The time difference is calculated by using the histogram shown in Fig. 18 as an example. Repeating this procedure for entire evaluation vehicle sensor data gives the estimate of the evaluation vehicle.

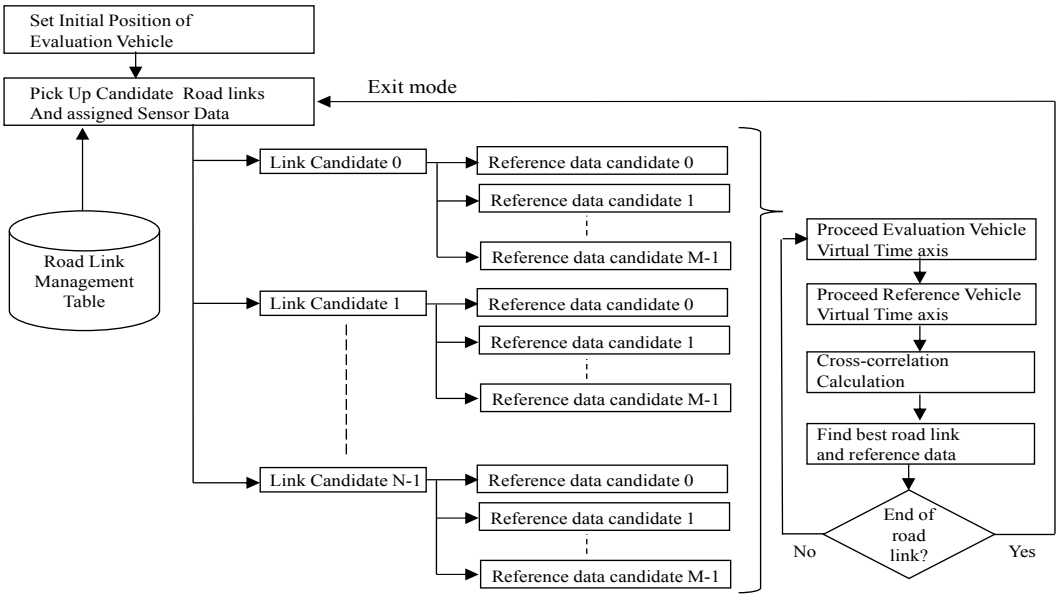

Fig. 19. Algorithm of Searching for best match reference data.

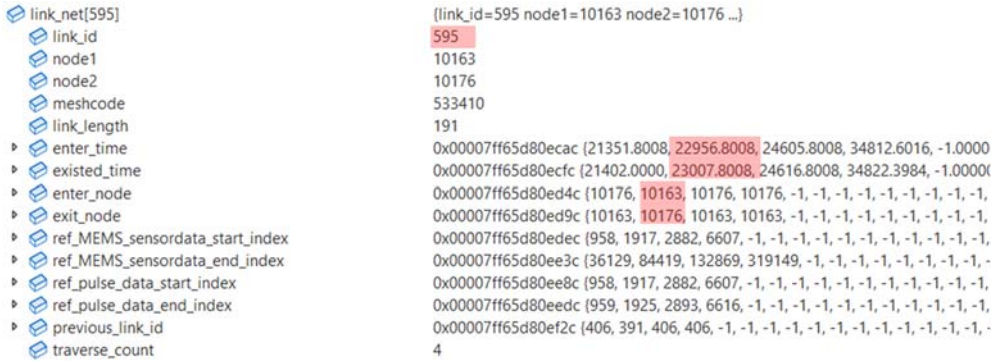

Fig. 20. Example of road link management table. This is a screen shot of the Visual C++2020 debugger showing the trip information of the reference vehicle along link 595 (10163-10176). This is the implementation of the data structure shown in Fig. 5.

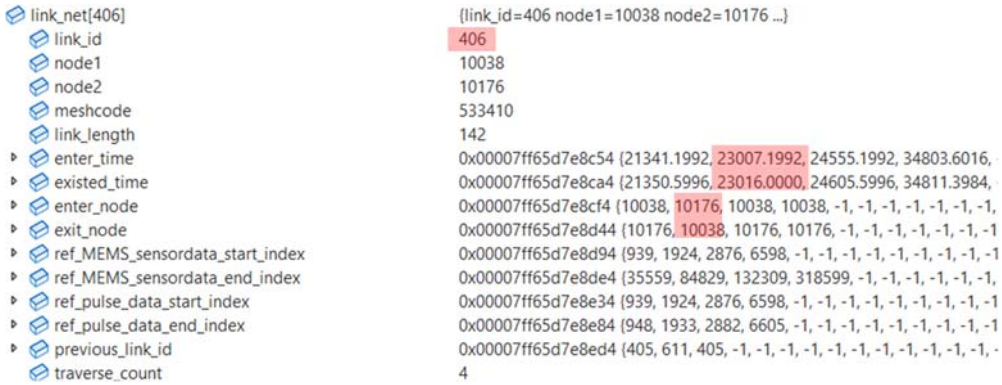

Fig. 21. Example of link management table. This is a screen shot of the Visual C++2020 debugger showing the trip information of the reference vehicle along link 405 10176-10038. This is the implementation of the data structure shown in Fig. 5.

This algorithm is currently somewhat time consuming because it tries candidates of road links and reference data.

Fig. 22 shows the position estimation error along actual time axis of evaluation vehicle. The time difference was estimated by the maximum value of

each histogram for each rad link. The RMS error along this evaluation course is 3.33 m, the maximum error is 13.8 m. In a region where the position estimation is stable and good for instance from 20 to 40 s, the RMS error is 1.11 m. The beginning of the trip of the evaluation vehicle is entering the road link index 595 by left turn. Because, there is no reference which runs on link index 595 by entering by left turn and all of the reference data was straight run, Hence, the position error occurs at near the intersection of this node of 10163. Other portions whose position error are significant (more than $3m$ or so are due to the algorithm of difference time estimation. This time the difference time was estimated by adopting the maximum value of the histogram. However, since the bright lines of the cross-correlation function is not strictly horizontal, the time difference error becomes an order of 0,1 s which can be a position error of several meters.

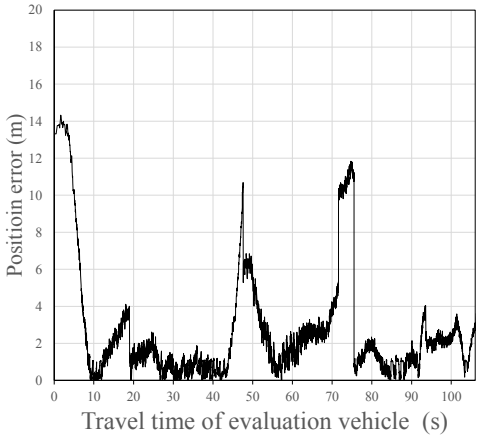

Fig. 22. Evaluation result of position estimation error. RMSE 3.33 m, maximum error 13.8 m.

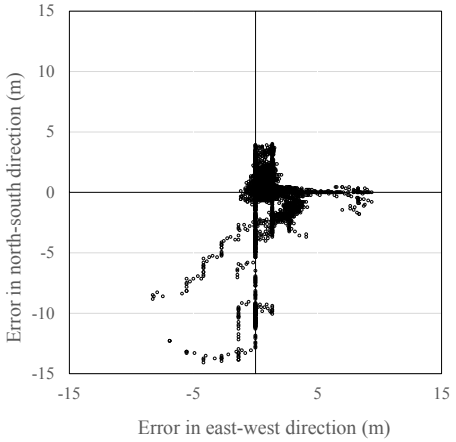

Fig. 23. Directional position error distribution.

The directional position error is shown in Fig. 23. The direct m position error depends on the trajectory of the reference vehicle. The main position error is the deviation in the direction of travel.

5. Conclusion

We have presented a solution to extend our vehicle localization algorithm using MEMS sensor data by combining it with a map-matching algorithm to associate reference sensor data to road links.

The following summarizes this research and our findings:

1) We devised a data structure that makes it possible to apply an algorithm that extracts road characteristics from acceleration, angular velocity, geomagnetism, and other sensors and estimates the position of a vehicle on a general road network. To this end, we developed a method that uses a map matching algorithm and digital road map information.

2) We found that MEMS sensor data, which is managed separately for each road link, needs to be kept for several cases, as the sensor data show different characteristics when entering or leaving that link by turning left or right.

3) Although it is necessary to connect several MEMS sensor data in a time series on time axis that is managed separately for each road link for candidate paths, the number of combinations is not considered too high. The candidate reference data are picked up according to the road link management table.

4) In estimating optimum time difference between evaluation vehicle and the reference vehicle, the maximum value of the histogram of the cross-correlation function was adopted. In this method, the position error increases if the bright line of the cross-correlation function diverts from strictly horizontal line.

5) In this work, we only evaluated a sample evaluation vehicle run for about 1.2 km, for 106 s. The overall evaluation results will be reported in the near future.

Acknowledgements

This work was supported in part by JSPS KAKENHI and the Japan Road Map Association.

References

[1]. T. Yokota, Network-wide vehicle localization algorithm based on MEMS sensor data, in *Proceedings of the 4th IFSA Winter Conference on Automation, Robotics and Communications for Industry 4.0/5.0 (ARCI'24)*, Innsbruck, Austria, 7-9 February 2024, pp. 145-150.

[2]. T. Yokota, Vehicle localization by correlated MEMS sensor data with velocity compensation, in *Proceedings of the IEEE 26th International Conference on Intelligent Transportation Systems (ITSC'23)*, September 2023.

[3]. T. Yokota, T. Yamagiwa, Vehicle localization by optimally weighted use of MEMS sensor data, in *Proceedings of the 3rd IFSA Winter Conference on Automation, Robotics and Communications for Industry 4.0/5.0 (ARCI' 2023)*, February 2023, pp. 79-84.

[4]. T. Yamagiwa, T. Yokota, Vehicle localization method based on MEMS sensor data comprising pressure, acceleration and angular velocity, in *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA'22)*, Aug. 2022, pp. 786-791.
[5]. T. Yokota, Vehicle localization by altitude data matching in spatial domain and its fusion with dead reckoning, *International Journal of Mechatronics and Automation*, Vol. 8, Issue 4, Jan. 2021, pp. 208-216.
[6]. T. Yokota, Vehicle localization by dynamic programming from altitude and yaw rate time series acquired by MEMS sensor, *SICE Journal of Control, Measurement, and System Integration*, Vol. 14. Issue 1, April 2021, pp. 78-88.
[7]. T. Yokota, Vehicle localization based on MEMS sensor data, in *Proceedings of the 60th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE'21)*, Sept. 2021, pp. 1094-1100.
[8]. T. Yokota, Localization algorithm based on altitude time series in GNSS-denied environment, in *Proceedings of the 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE'20)*, September 23-26, 2020, pp. 952-957.
[9]. T. Yokota, M. Okude, T. Sakamoto, R. Kitahara, Fast and robust map-matching algorithm based on a global measure and dynamic programming for sparse probe data, *IET Intell. Transp. Syst.*, Vol. 13, Issue 11, 2019, pp. 1613-1623.
[10]. J. Tsurushiro, T. Nagaosa, Vehicle localization using its vibration caused by road surface roughness, in *Proceedings of the IEEE International Conference on Vehicular Electronics and Safety*, Yokohama, Japan, Nov. 2015, pp. 164-169.
[11]. A. J. Dean, R. D. Martini, S. N. Brennan, Terrain-based road vehicle localization using particle filters, *International Journal of Vehicle Mechanics and Mobility*, Vol. 49, Issue 8, 2011, pp. 1209-1223.
[12]. E. Laftchiev, C. Lagoa, S. Brennan, Terrain-based vehicle localization from real-time data using dynamical models, in *Proceedings of the IEEE 51st IEEE Conference on Decision and Control (CDC'12)*, Maui, HI, USA, 2012, pp. 3366-3371.
[13]. J. Gim, C. Ahn, Ground feature-based vehicle positioning, in *Proceedings of the 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE'20)*, 2020, pp. 983-984.
[14]. J. Gim, C. Ahn, IMU-based virtual road profile sensor for vehicle localization, *Sensors*, Vol. 18, Issue 10, 2018, pp. 33-44.
[15]. X. Qu, B. Soheilian, N. Paparoditis, Landmark based localization in urban environment, *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 140, June 2019, pp. 90-103.
[16]. MPU-9250, Nine-Axis (Gyro + Accelerometer + Compass) MEMS MotionTracking™ Device, https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/
[17]. ZED-F9P module, u-blox F9 high precision GNSS module, https://www.u-blox.com/en/product/zed-f9p-module