Industrial Paper

A Data Synchronization Method for a Vehicle Using Multimodal Data Features

Yosuke Ishiwatari[*] [**], Takahiro Otsuka[*], Masahiro Abukawa[*], and Hiroshi Mineno[**]

[*] Information Technology R&D Center, Mitsubishi Electric Corporation, Japan
[**] Graduate School of Science and Technology, Shizuoka University, Japan
{Ishiwatari.Yosuke@dr, Otsuka.Takahiro@dw, Abukawa.Masahiro@bx}.MitsubishiElectric.co.jp,
mineno@inf.shizuoka.ac.jp

Abstract - The installation of multiple sensors in vehicles for acquisition and analysis of data is gaining popularity, owing to the increasing diversity, miniaturization and inexpensiveness of sensors. However, these sensors are not necessarily integrated into the same overall system. For instance, an owner-installed dashboard camera may not be connected to the factory-installed network of the vehicle. Therefore, it is important to synchronize data from multiple sensor systems to analyze the relation between the time series datasets of multiple sensors. A time-synchronization method is popular for this purpose, but this approach is not suitable for synchronizing offline sensor data. In this study, we propose a method for synchronizing video data with acceleration data from a moving vehicle's onboard sensors that use image features to detect synchronization points, which are then matched to corresponding points in the acceleration data. We evaluated the performance of our method by comparing video data with acceleration data (both collected via smartphone) when the vehicle turns right or left. Using this approach, we found the error to be 39.103 milliseconds. We intend to expand and further optimize our methodology by extracting and comparing data from different driving scenarios.

Keywords: autonomous driving, multimodal, data synchronization, motion estimation of a vehicle.

1 INTRODUCTION

The installation of multiple sensors in vehicles for acquisition and analysis of data is gaining popularity, owing to the increasing diversity, miniaturization, and inexpensiveness of sensors. Autonomous driving is one of the applications using this approach, as autonomous vehicles have multiple sensors, such as global positioning system (GPS) receivers, cameras, and acceleration, laser, and radar sensors [1][2]. In addition, many consumer-grade vehicles use dashboard cameras (dashcams), which include a GPS receiver and/or some acceleration sensors. However, these sensors are not necessarily connected to the same central system, for instance, a dashcam is usually separated from other sensors within a vehicle (Fig.1). In this situation, multiple systems are therefore used for analyzing data acquired from multiple sensors. Vehicles that do not possess any sensors are rare. For some events such as a traffic accident in which the car is involved, if someone who was not involved in the accident (e.g., policeman or insurance representative) is investigating the driver's role in the accident, even if not at fault, the driver has to provide some form of proof of innocence. If the driver uses unsynchronized dashcam video and other sensing data, the proof depended on using the data separately. So, even if the driver could prove innocence by using synchronized data (for example, proving that the driver "was correctly braking when the accident occurred" by using integrated camera video and acceleration data), that driver may not be able to prove it by using the data separately. Many consumer-grade vehicles do not come with a camera installed, so it is common for drivers to equip a car with an after-market dashcam. For commercial-transportation vehicles [3], synchronization among multiple sensors (such as a dashcam and other sensors that are original equipment in a vehicle) is important.

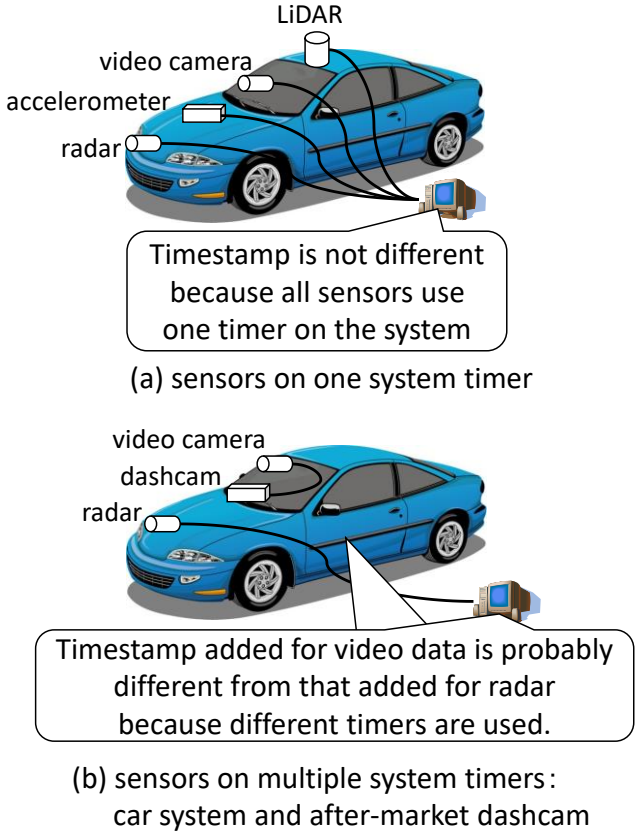

Figure 1: System timer and timestamps of sensor

Synchronization of acquisition times of different sensor data is very important for analyzing and enabling correlation between these data. If sensor data are acquired in a single integrated system, acquisition times are synchronized seamlessly; however, data obtained by multiple systems may not be synchronized because of different times recorded by the separate systems. Because this commonly occurs when a vehicle owner installs a dashcam or some other after-market device, as vehicles generally do not have a synchronization method for such situations, an extra system must be added.

Precise time differences between systems are needed to correct the discrepancies and enable data synchronization. Typical correction methods involve synchronizing system times or inserting timestamps in the sensor data [4][5], but because separate system clocks are different, system times become incorrect after multiple synchronizations over a long period of time. Too, when data from multiple systems are analyzed after all the data are recorded, differences between data-acquisition times cannot be obtained because the unsynchronized system times are not known when the data are recorded. Thus, if time synchronization is not in effect during data acquisition, it is impossible to synchronize data at a specific time to enable full use of all data.

Furthermore, if external time, such as GPS time, is used for synchronization, GPS receivers are needed for all systems, but if this external time is in error, such as when vehicles are passing through a tunnel or an urban area containing many buildings, this error is not detected during synchronization. Also, the cumulative cost of a GPS receiver for each sensor can be considerable, so it would be advantageous for synchronization to be possible without having to purchase multiple GPS receivers.

We have developed a method for synchronizing sensor data by extracting the data ranges of different vehicle motions through analysis of the characteristics of sensor data, without needing additional devices to record time data, thereby providing a more cost-effective method for synchronizing data from multiple sensors in a vehicle.

2 OUR PROPOSED METHOD AND RELATED STUDIES

2.1 Proposed Method

In this study, we aimed to synchronize video and acceleration data recorded by sensors in a vehicle, where each sensor was connected to a different system (Fig. 2). We intended to perform synchronization after rather than simultaneously with data acquisition.

Although the data-acquisition times for the various devices were approximately the same, the exact differences in times between the systems were unknown because the system clocks differed. Thus, for instance, because the differences in time between the camera and acceleration data were not always constant, if the synchronization was performed at one data point, it did not necessarily mean that all data points could be synchronized in a similar way. This was resolved by synchronizing some points of the data and correcting the data between these synchronizing points using the differences in time at the said points. Video and acceleration data also have many different characteristics, therefore reference points related to characteristics common to both data types are required for synchronization to be possible. Therefore, we propose a synchronization method that is based on the detection of vehicle-motion behavior from sensor data and matches ranges of data. The fact that we did not use time data to detect vehicle motion meant that the inconsistency in the differences in time (resulting from different clocks in the systems for each sensor) was negligible.

2.2 Proposed Vehicle-Motion Events

Because our method requires the use of multiple synchronization points, vehicle-motion events must be easily detectable and occur frequently while driving. However, easily detectable events such as passing over a bump (video result: vertical displacement; acceleration result: vertical vibration) and heavy braking (video result: variation in moving vector of objects; acceleration result: variation of acceleration in the direction the vehicle is moving) do not necessarily occur frequently in a given journey. Accordingly, we focused on extracting the required data from very common events such as when a vehicle is turned right or left, with the beginning and end of these turning events treated as synchronization points. It rarely occurs that a car is driving without turning right/left (i.e., the car is moving only straight ahead or back). In car-related studies, such as those concerning automonous driving, those algorithms/methods are evaluated by driving a variety of car motions, such as changing lanes, turning right/left at an intersection, or passing [6]-[12]. That means a car is not just moving straight ahead. Therefore, we chose to use the time when the car is turning right/left as a synchronizing point.

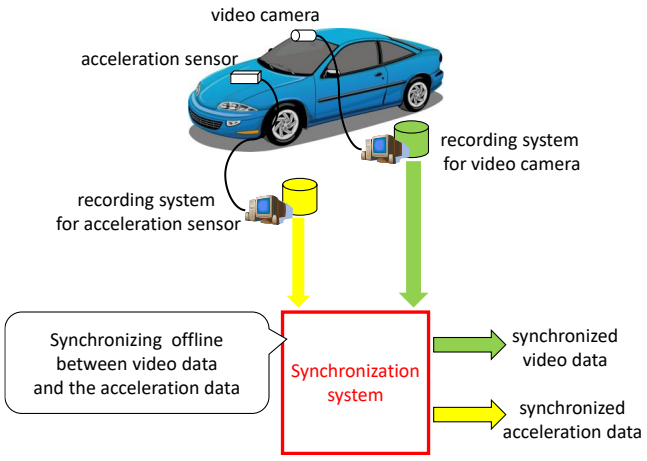

Figure 2: Proposed system

2.3 Proposed Method

For synchronization between data that have different characteristics, it is necessary to convert the data and/or isolate the characteristics that make the data mutually comparable. For example, in order to match with acceleration data, car speed can be calculated by a bird's-eye view created by camera images [13]. If the speed is calculated correctly, synchronization of video data (each frame being the converted speed of the car) and acceleration data is possible, but according to Morimoto et al. [13], calculation accuracy is not good when the car is moving slowly. For example, when the speed is 20 km/h, the error is >10%, which means the synchronization error is >10%. Sometimes synchronization is impossible because the pattern of the speed differences and the values of the acceleration data are very different. Because such low-speed driving sometimes occurs in urban areas, their study [13] is not suitable for our research.

Another method is to convert optical flows to be comparable to events characterized by other sensor data. Fridman et al. [14] devised a method to synchronize sensor data using optical flow by detecting vibrating events or steering events; this method is good for relative synchronization but cannot enable absolute synchronization. Giachetti et al. [15] developed a method for estimation of egomotion using optical flow; however, it is obviously not useful for vehicles traveling on a flat road, where vertical displacement is negligible.

There are a lot of synchronization methods that focus on acceleration and camera imagery. For example, Tanaka et al. [16] described a method using a correlation value between sensor data without consideration of time, but it is applicable to acceleration data only and thus is not suitable for our system.

3 DETERMINING SYNCHRONIZAION POINTS USING IMAGE FEATURES AND CHARACTERISTICS OF ACCELERATION DATA

3.1 Overview

Our method consists of three functions, 1) detecting "turning right" and "turning left" events using image features, 2) detecting "turning right" and "turning left" events using acceleration data, 3) detecting synchronization points using detected events.

We describe the functions in the following sections.

3.2 Detecting "Turning Right" and "Turning Left" Events Using Image Features

We used the optical flows of image features for detecting behaviors of the vehicle as "turning right" or "turning left" from camera images. Because the optical flows of image features of stationary objects have vectors that are opposite in direction to those of a moving vehicle, we can acquire vectors from stationary objects that correspond to those of a moving vehicle. We can calculate the tendencies of the vectors from the optical flows of image features using whole frames. These image features are not solely on stationary objects; however, there are not many objects that move around the vehicle, so the tendencies of the vectors that can be regarded as a vector are the same as those of a vector that shows the movement of the vehicle. Fig. 3 depicts the optical flow when a vehicle turns right, showing how at this point a vector that is opposite to that of the moving vehicle can be acquired (i.e., a vector in which the direction is from the left to the right of the image as drawn).

When a vehicle is turning right or left, the optical flows from the image features on stationary objects are opposite to the direction in which the vehicle is moving. Thus, if a camera is recording in front of a vehicle, when the vehicle is turning right, optical flows turn left. Conversely, when a vehicle is turning left, optical flows turn right. Moreover, if a vehicle is not moving right or left, optical flows do not change, meaning that the direction of optical flow is very useful for detection of a vehicle-motion event.

Meanwhile, the vertical direction of the optical flow varies with the position of each image feature within the camera image. Fig. 4 is an example of "turning left," which shows that the vertical direction of the optical flow for each image feature varies from one to another.

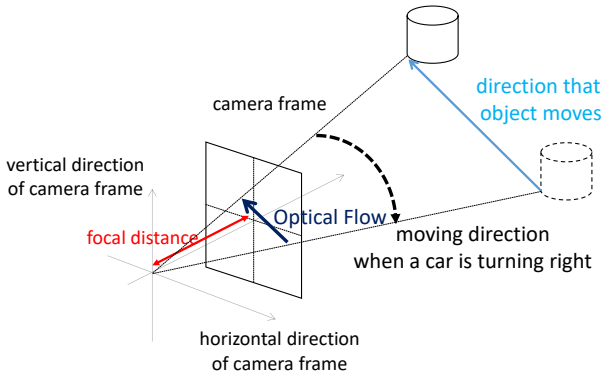

Figure 3: Optical flow when car is turning right

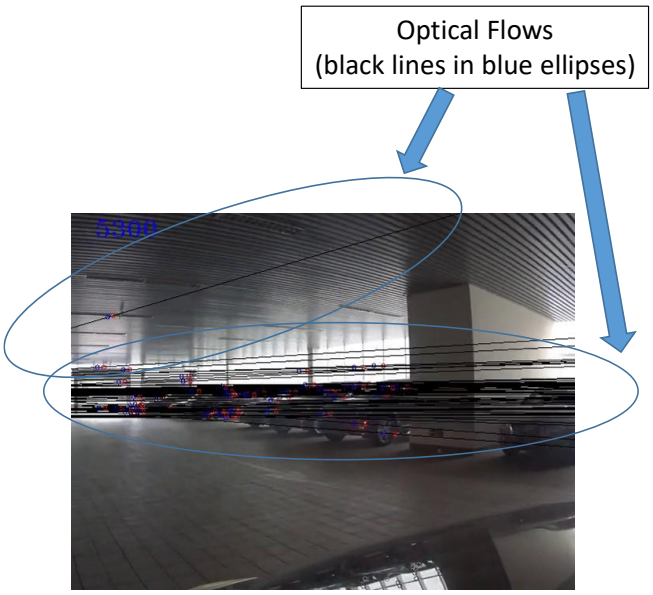

Figure 4: Optical flows of "turning left" event

3.3 Detecting "Turning Right" and "Turning Left" Events Using Acceleration Data

In this study, we hypothesized that we could acquire acceleration data in the horizontal direction. If a vehicle is moving and the velocity in the horizontal direction is always zero unless a vehicle does not turn right or left, we can detect the event of right/left turning by detecting whether the velocity in the horizontal direction is zero. This can be achieved using sensor data by calculating integration of the acceleration value. However, a vehicle is moving in the horizontal direction even if it is not turning right/left; therefore, we cannot detect vehicle-motion events by confirming that the velocity of the vehicle is zero.

When a vehicle turns right or left, the driver operates the steering wheel to move in the horizontal direction. Thus, this operation is equal to accelerating the vehicle in the horizontal direction, so the start of this operation causes a significant change in acceleration (Figs. 5 and 6). This means that this operation can be detected from the change in acceleration. The start and the end of this turning operation can be detected as a peak or inflection in the acceleration

In general, raw acceleration data from acceleration sensors includes some noise and bias that must be removed before calculating the peak or the inflection.

3.4 Detecting Synchronization Points

Based on the above method for detecting right/left turning events from video or acceleration data, we propose a method to synchronize video and acceleration data.

Our method comprises two functions. The first is a function that detects ranges of the frame that indicate the vehicle is turning right or left. The second is a function that calculates the difference between these ranges and the range of acceleration data by searching points that correspond with the desired data. To reduce the searching range, the times of the systems are approximately similar and the difference in time is not known. However, the start of the searching point (the point that would match the point of the other sensor if the difference were zero) can be determined.

In videos, "turning right/left" events have characteristics such that the optical flows tend to turn left/right. We therefore use these characteristics for detecting the ranges of the frames (Fig. 7), as follows:

1) Obtain the optical flows of the image features between successive frames;
2) Calculate the tendency of the vectors of the optical flows by classifying the vectors into 16 bins based on the direction of the vector, and select the bin that includes the majority of the vectors;
3) Calculate the range by counting the frames in which the bin of the start frame is the left bin (bins 7 and 8 in Fig. 7) or the right bin (bins 1 and 16 in Fig. 7) if the bin is near the former frame (within two consecutive bins).

The ranges calculated by the method are regarded as "turning left (continuously classifying left bin)" or "turning right (continuously classifying right bin)." Fig. 7 shows an example of classifying a "turning right" event and the event preceding it.

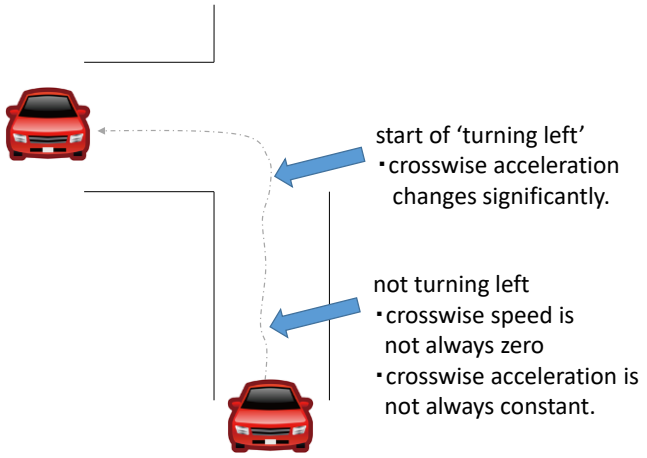

Figure 5: Acceleration change in "turning left" event

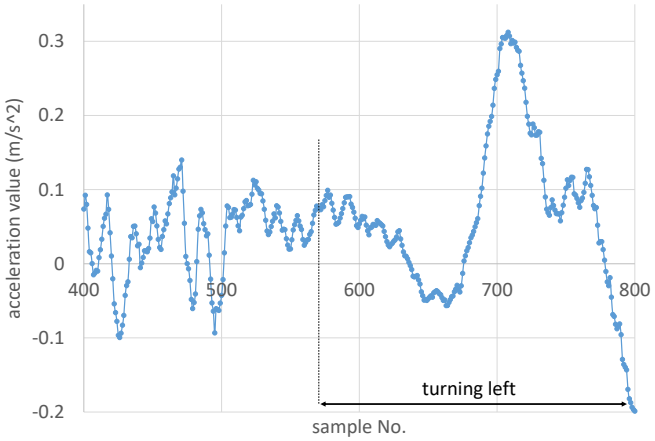

Figure 6: Change of acceleration in "turning left" event

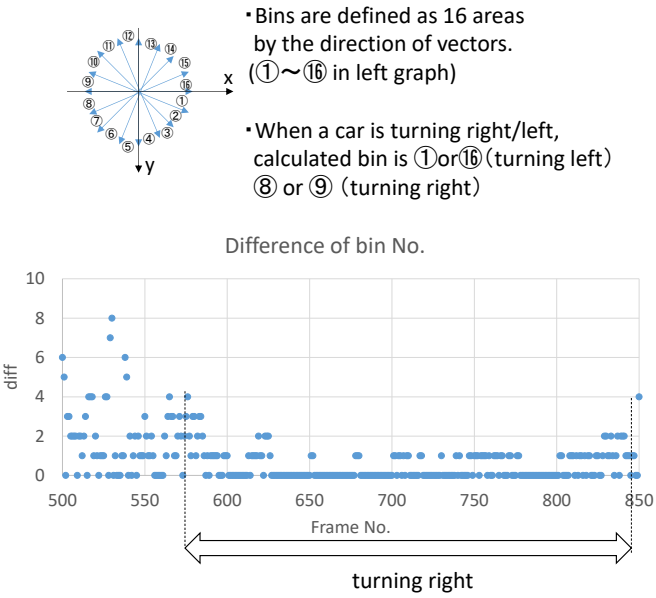

Figure 7: Bin differences in "turning right" event

In our earlier research [17], we discussed how the possibility of incorrect detection of a turning right/left event was controlled for checking whether such an event was a combination of more than one event. This may arise when another action occurs simultaneously with the vehicle turning right or left, such as riding on a curb or heavy braking. Thus, if the number of frames between two of the "turning right/left" events is very small (a few frames) and if these situations are the same (i.e., these situations are "turning right/right" and vice versa), these situations should be regarded as one event (Fig. 8).

After calculating the range of the frames, the matching points between the start/end frame of the desired range and the acceleration data are calculated. The data is examined for the presence of peak or an inflection point of the acceleration. A peak or an inflection point of the acceleration data closest to the start point is regarded as the corresponding point of the start/end frame.

After calculating the range of the frames, the matching points between the start/end frame of the desired range and the acceleration data are determined. The data is examined for the presence of acceleration peaks or inflection points. Either one closest to the start point is regarded as corresponding to the start/end frame.

After calculating the corresponding point of the start frame (point C_1) in the acceleration data (point A_1) and the corresponding point of the end frame (point C_2) in the same data (point A_2), A_1 and A_2 are corrected to have the same range of time between the range from C_1 to C_2 and the range from A_1 to A_2. In detail, point A_1 is moving to A_1' and point A_2 is moving to A_2'. Therefore, "(time of A_2')−(time of A_1') = (time of C_2)–(time of C_1)" and "(time of A_1)−(time of A_1') = −[(time of A_2)−(time of A_2')]." Accordingly, A_1' and A_2' are calculated as follows:

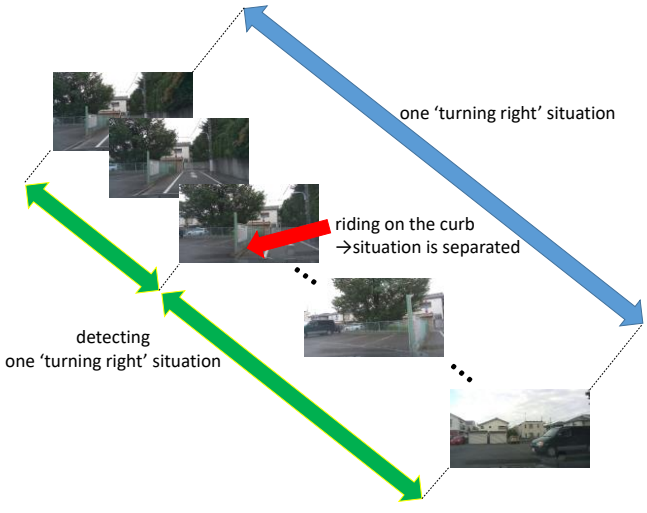

Figure 8: Separating one "turning right" situation into two situations

$$\Delta t_c = (\text{time of } C_2) - (\text{time of } C_1)$$
$$\Delta t_a = (\text{time of } A_2) - (\text{time of } A_1)$$
$$\text{diff} = (\Delta t_c - \Delta t_a)/2$$
$$\text{time of } A_1' = (\text{time of } A_1) - \text{diff}$$
$$\text{time of } A_2' = (\text{time of } A_2) + \text{diff}$$

Hence, D_1 = (time of C_1)−(time of A_1'), which is the difference in time between the start frame of the video and the start of the acceleration data of that range, and D_2 = (time of C_2)−(time of A_2'), which is the difference in time between the end frame of the video and the end of the acceleration data of that range. D_1 and D_2 are not always identical. Therefore, the difference value (ΔE) at time E (between A_1' and A_2') is calculated as follows:

$$\Delta E = \frac{D_1 * ((time\ of\ A_2') - (time\ of\ E)) + D_2 * ((time\ of\ E) - (time\ of\ A_1'))}{(time\ of\ A_2') - (time\ of\ A_1')}$$

For synchronizing point X in the range other than at a right/left turning event, the difference (Δd_1) between the point X and point X_1 (the end frame of the range of the right/left turning that occurs immediately before the point X), and the difference (Δd_2) between point X and point X2 (the start frame of the range of the right/left turning event that occurs immediately after the correcting point X) is used. The time difference ΔX at point X is calculated as follows:

$$\Delta X = \frac{\Delta d_2 * ((time\ of\ X) - (time\ of\ X_1)) + \Delta d_1 * ((timeof\ X_2) - (time\ of\ X))}{(time\ of\ X_2) - (time\ of\ X_1)}$$

4 FUNDAMENTAL EVALUATION OF OUR METHOD

We evaluated the fundamental accuracy of our method. The data and our evaluation method are described as follows.

4.1 Data Setting

We used video data and acceleration data acquired by a smartphone in a vehicle. The camera recorded the front view of the vehicle. The acceleration sensor recorded accelerations in three dimensions: the direction in which the vehicle moved, the horizontal direction of the vehicle, and the vertical direction of the vehicle. Fig. 9 shows the course we traversed with the smartphone-equipped vehicle.

The recorded time was also acquired for these data (camera frames and acceleration data).

For an evaluation, we decided on evaluation events (synchronization timing) to check a video and extract frames that correspond to the start or end point of turning right/left. We also calculated the acceleration data at each frame by linear interpolation. If those data are to be used in applying our method, the difference must be zero at all evaluation points, but some points had nonzero difference values because of noise. Accordingly, in this evaluation, we investigated robustness of our method.

In this evaluation, the number of right/left turning events was 26, and these were all used for evaluation of the method.

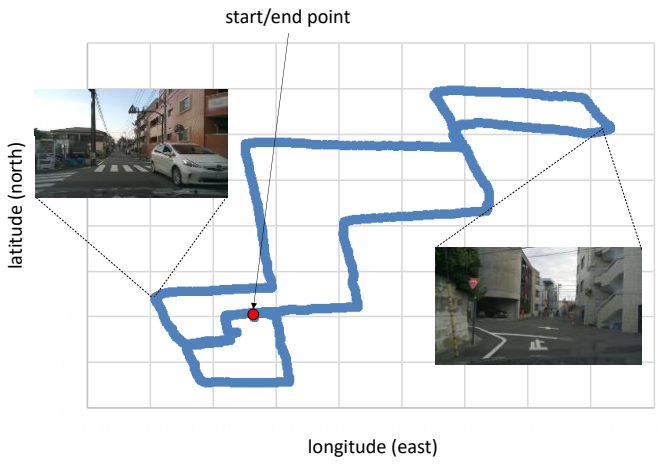

Figure 9: Course used in driving experiment

To reduce noise in the acceleration data, we used a smoothing process, and we applied an interpolation process to adjust the sampling rate of the acceleration data to the interval between the frames of the video (30 fps).

The ranges calculated by the method in some cases had overlapping subranges, which showed ranges that correlated with the same right/left turning event. Accordingly, we added a merge process within our method so as to match one range to one event. In that process, some ranges that have the same frames got merged into one range.

In addition, a range of one right/left turning event was sometimes split, so one event was sometimes described by more than one range. Accordingly, we added a concatenate process within our method so as to ascribe one range to one event. In that process, two ranges having an interval of one or two frames got merged into one range.

4.2 Parameter Setting

We searched for peaks or inflection points from the start point in the acceleration data. During evaluation, the search range was defined as 30 samples (corresponding to 1 sec.).

From the ranges calculated by the method, we extracted those spanning ≥120 frames (i.e., >4 sec.). This parameter is based on the result of an examination described by Fukuda et al. [18], using elapsed time of a turning right event as being 3–6 sec.

4.3 Experimental Results

We evaluated the 26 situations by calculating the difference between the start time of camera frames and the synchronized time of the acceleration data during right/left turning events, after applying our method (Fig. 10).

In Fig. 10, the calculated value (difference between camera frame and acceleration data) for each event is shown, together with the absolute value of each calculated data point. The average value is 39.103 milliseconds (= 1.173 frames), and the standard deviation is 46.026 milliseconds (= 0.824 frames). Event 14 has a large error compared to that of other events. That is because acceleration data are not varying at that point, so the shift in synchronization points is larger. We consider the cause to be data-acquisition error or bad noise filtering, and we intend to further investigate this kind of error.

In our experiment, we obtained a timing error of ~40 milliseconds between the video data (from an added dashcam) and the acceleration data measured by the factory-installed sensor in the vehicle. We propose a method for synchronization of the data from each of these sensors. If another sensor is added to the vehicle (e.g., a radar sensor at the front of the vehicle) is synchronized with acceleration data, this sensor will also be synchronized to video data.

For evaluating the accuracy of our method, we examined the accuracy of object detection by multiple sensors. When multiple sensors detect the same object, the detection results should theoretically be the same if sensor data are correctly synchronized. However, if synchronization is not correct, there is an error that, in the case of a moving vehicle, equates to a measurable distance. For example, in detecting pedestrians moving at 80 m/min, an error of 40 milliseconds equates to a 5.3 cm difference in distance, whereas in detecting cars moving at 60 km/h, this error is 66.7 cm. Accordingly, these distances can be considered negligibly small.

For calculating optical flows, we have to select two frames. If the interval between those two frames is short, the length of the optical flow tends to shorten. This means that the optical flow is significantly affected by the error resulting from the matching. By enlarging the interval between these two frames, the impact of the matching error can be smaller. However, enlargement of the interval means that the accuracy of the detecting ranges of the right/left turning event decreases, because the accuracy depends on the intervals. This may result in the process failing to detect some ranges. To avoid this problem, we evaluated the accuracy by alternating the intervals. Fig. 11 shows the average and the standard deviation of the errors at some intervals from no frame (optical flow is calculated using consecutive frames) to nine frames.

In Fig. 11, the average of errors is greatest at the interval “0frame” (more than 120% of the average at other intervals), and the average of errors and sum of average and standard deviations is smallest at the interval “6frames”. Although some intervals have greater averages or standard deviations, Fig. 11 seems to show that the magnitude of the average is inversely proportional to the magnitude of the interval. Fig. 12 shows the ratio of a number of corrected events to a number of detected events, which shows that the smaller the magnitude of an interval, the larger the ratio.

Figs. 11 and 12 show that both increasing and decreasing intervals have disadvantages, which suggest that use of the same optical flow for both detecting ranges of data and synchronizing data is not optimal. However, in our study, the use of the interval “0frame” for detecting range and the interval “6frames” for synchronizing data does appear to be acceptable.

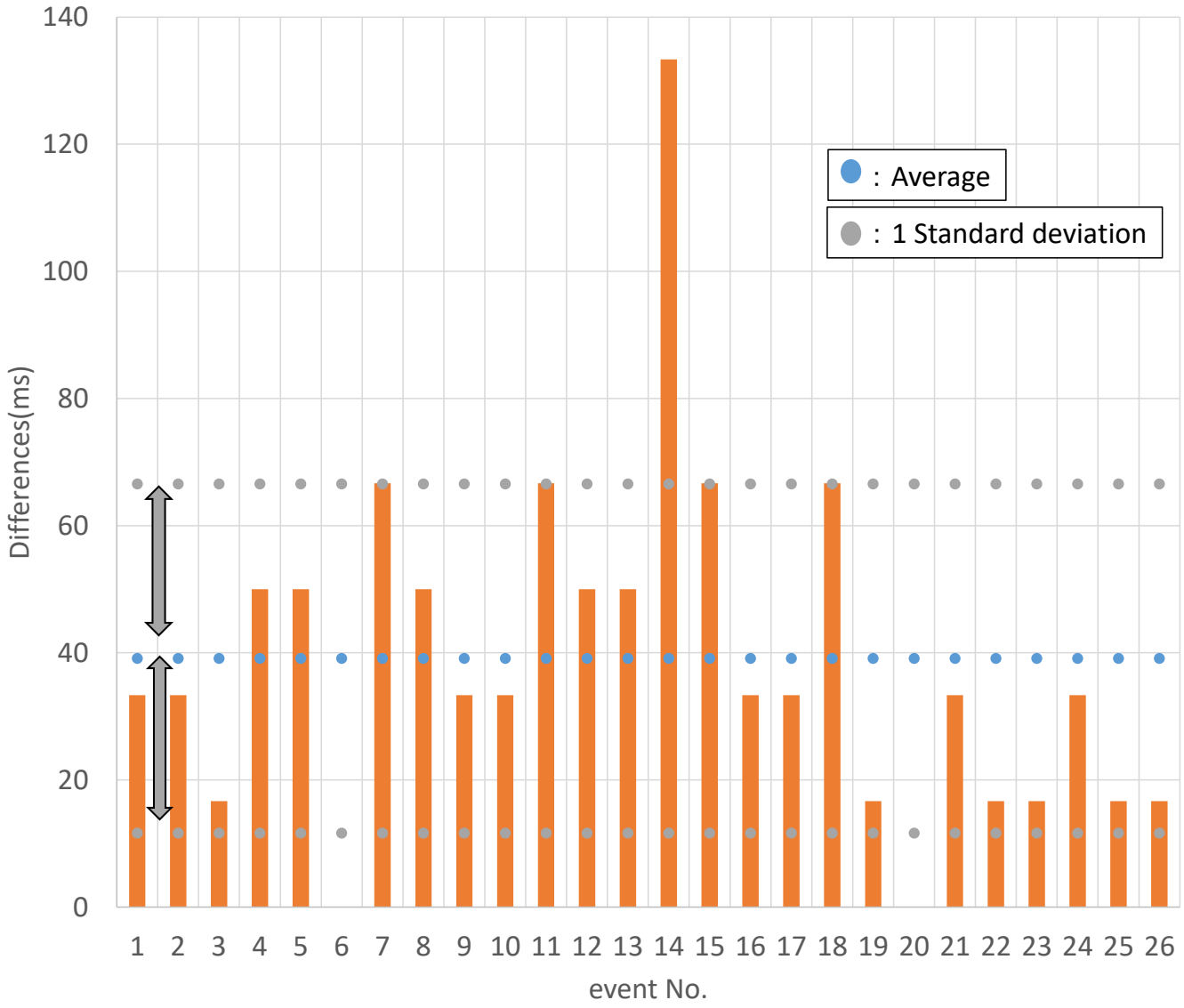

Figure. 10: Evaluation results

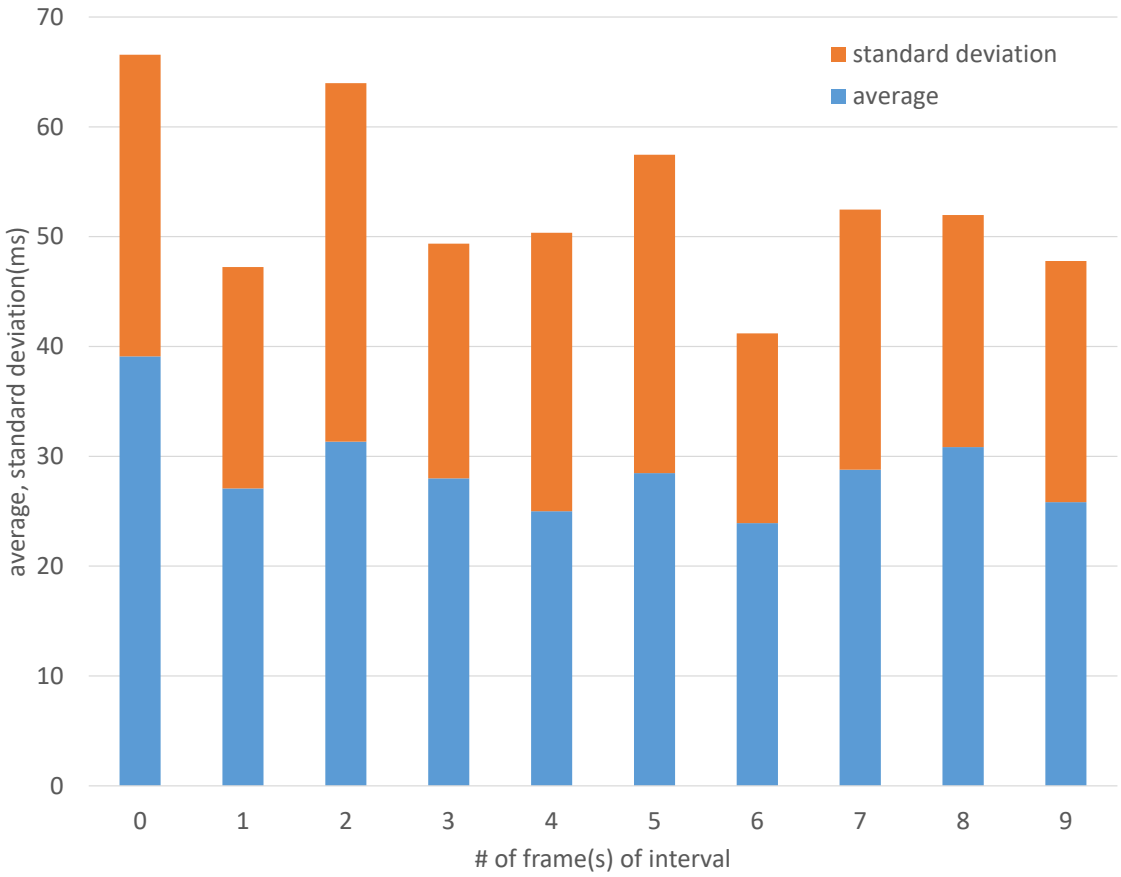

Figure. 11: average/standard deviation of errors and frame intervals

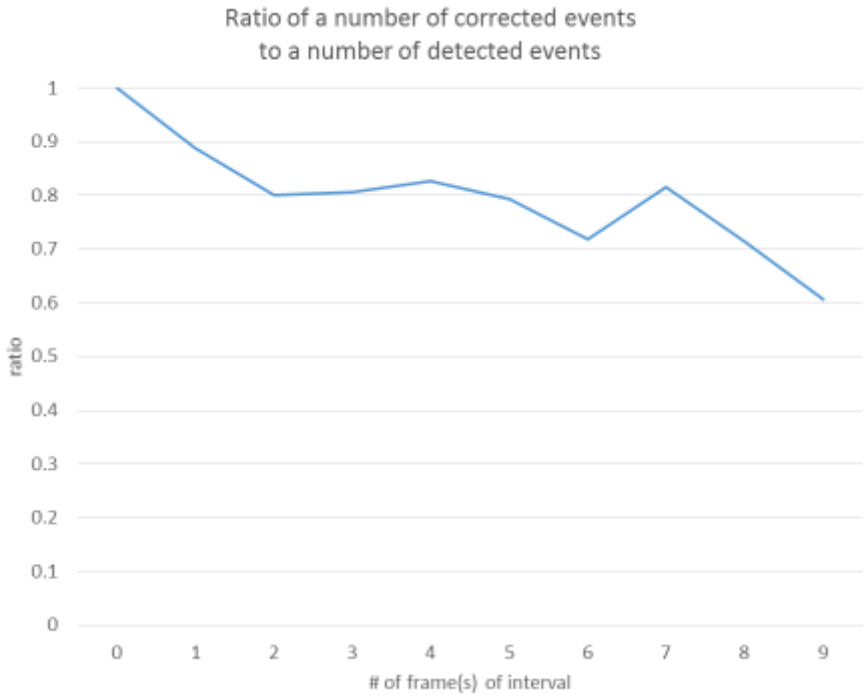

Figure. 12: Ratio of corrected events and frame intervals

In right/left turning events, moving radius and velocity of a car varies by situation. When the car is slowly turning, noise in the acceleration data may affect the result. Table 1 shows the moving radius (calculated by using the positions and algorithm (Fig. 13), velocity calculated approximately by integrating acceleration data and subtracting the integrated data at the point that the car is stopped nearest to but before the point because of drifts of the acceleration, and the difference value (Fig. 10)). In this evaluation, we applied noise reduction to the acceleration data, so it is assumed that the effect of the noise is small. Too, the correlation between errors and velocities is −0.151. That value means velocities are affected but only slightly. However, the correlation between errors and moving radius is 0.554, meaning that large radii tend to produce larger errors. We will investigate this more precisely in the future.

Table. 1: radius, velocity and error

event No.	radius [m]	approx. velocity [m/s]	error [# of frames]
1	5.32	1.96	1
2	5.25	2.22	1
4	8.69	0	1.5
5	2.04	0	1.5
6	37.37	0.70	0
7	18.52	0	2
8	31.20	0.91	1.5
9	31.54	1.74	1
12	45.94	1.00	1.5
14	355.41	0.80	4
15	232.86	2.36	2
16	143.80	0.50	1
23	135.38	0.60	0.5
24	210.75	1.73	1
25	98.92	1.76	0.5

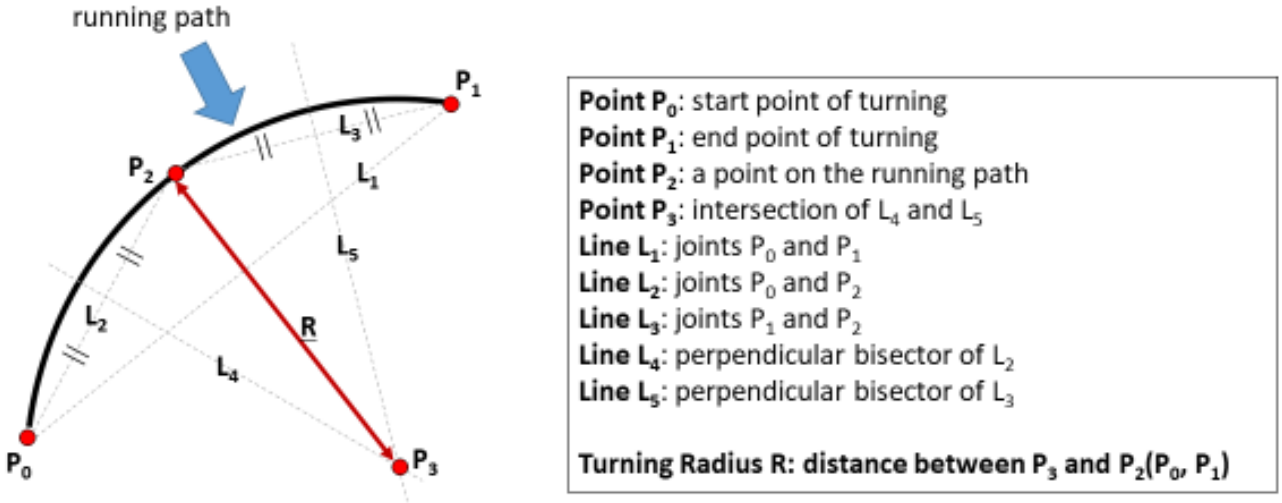

Figure. 13: Calculating radius using points

4.4 Future Work And Discussion

In the 26 events, the average of the error frames is short, so it can be concluded that our methodology adequately accounts for the data. In future work, we will consider aspects such as the following:

1) Detecting the range of the frames more precisely
   Our method detects the range of frames based on the tendency of the optical flow direction. If the range is short, the event corresponding to the range is not right/left turning but is similar in action to a part of the right/left turning event, e.g., an S-shaped curve.
2) Handling the difference between Δt_a and Δt_c (see section 3.4)
   In our method, the difference between Δt_a and Δt_c is divided equally and used for A_1 and A_2. This is not always true, as the difference between A_1' and A_1 is not always the same as that between A_2' and A_2. We can potentially resolve this issue by matching correlation values.
3) Special situations: a lot of objects move in the same direction but other than the car's direction, etc.
   In our method, we assumed that a lot of characteristic points have characteristics that are the same as for the motion of the car, and extracted frames as the right/left turning frames by checking that the frame had optical flows of mainly left/right direction. If the car is not turning left/right but other objects are moving mainly left/right, and the characteristic points are mainly on the objects, our method may detect that frame as right/left turning. However, that means other objects are moving across the car because a camera is recording in front of the car, thus the situation could occur that when the car is stopping, it crashes into those objects. We expect to be able to detect that situation and remove it from among the right/left turning situations.
   In our evaluation, the video data has a situation where the car is stopping and waiting at the railroad crossing, and the situation occupies 1.4% of the frames, and our method was not detect that situation as right/left turning.
4) Problems in obtaining image features from objects
   Our method assumes that the tendency of the optical flow direction is almost the same as the direction of the optical flows, based on the image features of the stationary object, so we assumed that multiple image features are acquired from an object that does not move in any frame. We currently use all the optical flows, but selecting the appropriate optical flows should enable improved detection and analysis. However, because our method applies all the frames, extensive calculation processes using optical flows will be required. We do not intend to use a heavily calculated method for, say, object detection, so this aspect will be carefully explored.
5) Problems associated with insufficient number of image features
   Some frames do not possess multiple image features, for example when data is acquired at night. A frame may also be occupied by the sky or the ground with no lines, signs, and other objects.
6) Utilization of camera images from other than the front
   As shown in Chapter 4, the installed camera is taking pictures of the front of the vehicle. Since the front is photographed, it is assumed that the optical flow swings from side to side when turning left to right. However, when using a camera image taken in a direction other than the front, this premise changes. In this situation, it would not be possible to use the optical flow obtained from the captured image, so it would be necessary to convert these optical flows to match those in the front direction.

   The following is a matching example. It is assumed that a feature point (u_1, v_1) in frame F_i is corresponding to a 3D relative position $\overrightarrow{x_1} = (x_1, y_1, z_1)$ (where the X axis represents the right direction of the camera, the Y axis represents the optical axis of the camera, the Z axis represents the upward direction of the camera, and the origin is the camera sensor), and at the time in frame F_j, the same point as $\overrightarrow{x_1}$ is represented the 3D relative position $\overrightarrow{x_2} = (x_2, y_2, z_2)$, and (u_2, v_2) in frame F_j, and the vehicle advances $\triangle\vec{X}$ (described using vehicle coordinate system, X axis represents the right direction of the car, Y axis represents the front direction of the car, Z axis represents the upward direction of the car, and the origin is the position of the vehicle) from frame F_i to frame F_j (Fig.14). In that case, the following equations hold, where f is the focal length of the camera, p is the pixel interval of the camera image, and (C_X, C_Y) is the central position of the camera image.

$$u_1 = \frac{f}{p \cdot y_1} x_1 + C_X, v_1 = \frac{-f}{p \cdot y_1} z_1 + C_Y$$
$$u_2 = \frac{f}{p \cdot y_2} x_2 + C_X, v_2 = \frac{-f}{p \cdot y_2} z_2 + C_Y$$

   And, an equation $\overrightarrow{x_2} = \overrightarrow{x_1} - R^{-1}\Delta\vec{X}$ holds, where is R represents the transformation matrix, which transforms the vehicle coordinate system to the camera coordinate system. So, if R and $\Delta\vec{X}$ is acquired, $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ can be calculated from (u_1, v_1) and (u_2, v_2). When $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ are calculated, the feature points (u_1', v_1') and (u_2', v_2') in the front images are calculated, where $R^{-1}\overrightarrow{x_1} = (x_1', y_1', z_1')$ and $R^{-1}\overrightarrow{x_2} = (x_2', y_2', z_2')$.

$$u_1' = \frac{f}{p \cdot y_1'} x_1' + C_X, v_1' = \frac{-f}{p \cdot y_1'} x_1' + C_Y$$
$$u_2' = \frac{f}{p \cdot y_2'} x_2' + C_X, v_2' = \frac{-f}{p \cdot y_2'} x_2' + C_Y$$

   Figs.15-17 are examples of this calculation. In the illustrated range, the vehicle is going straight after a right turn. Fig. 15 shows the most frequent bins calculated by the algorithm presented in this paper by

using the optical flows of the same range obtained from the front camera. Fig. 16 shows the most frequent bins by using the optical flows obtained from camera images taken from the left side of the vehicle without matching. In Fig. 15, the optical flows after the right turn is mainly 'right to left', that means the vehicle moves the left side, but it is not shown at all in Fig. 16. Meanwhile, Fig. 17 shows the most frequent bins by using the same optical flows but matched using $R = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$(, where R means rotation by an angle 90° around the Z axis, as well as$\triangle\vec{X} = (v_x\Delta t, v_y\Delta t, v_z\Delta t)$, where (v_x, v_y, v_z) is the velocity of the vehicle and Δt is the interval of timestamp acquisition of the optical flows. Fig.17 has a similar tendency as Fig. 15. However, in this example, the velocity of the vehicle must correspond with the frame. In the assumptions of this paper, there is a gap between the timestamps of the velocity of the vehicle and those of the image frames, so the deviation will be included by the conversion method described above. In this paper, the gap is small, so we consider that the impact is minor.

7) Expanding our method to other situations

Our method uses all the optical flows in the frame to calculate the tendency of their direction. A right/left turning event is an appropriate situation to be detected using the method. However, our method is not suitable for some situations, such as moving straight ahead. In that situation, not all the optical flows turn in the same direction. The direction is determined according to the point of the image feature within the frame.

To expand our methodology, we can split a frame into subframes and calculate the tendencies within the subframes, followed by detection of the range based on the characteristics of those tendencies. This should enable data to be obtained when a vehicle is moving in a straight line, and thus enable sensors to be synchronized at any time during the journey of a vehicle.

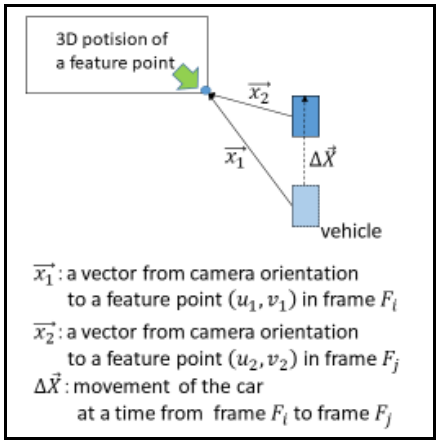

Figure. 14: The case of similar feature points at different frames

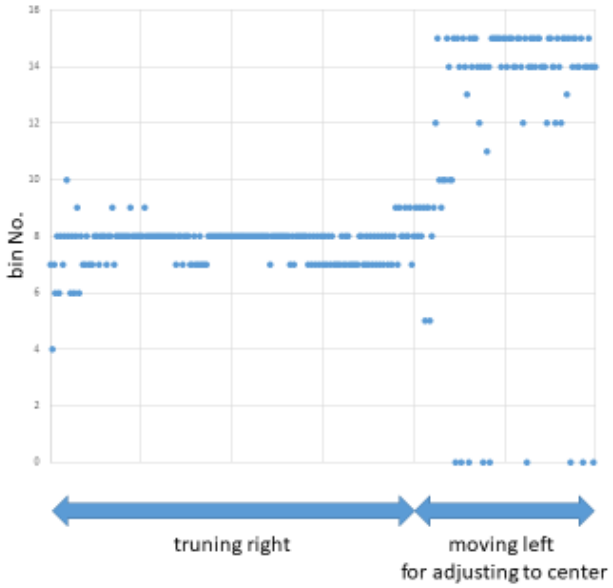

Figure. 15: Changes of bins of the case from images of the front of the car

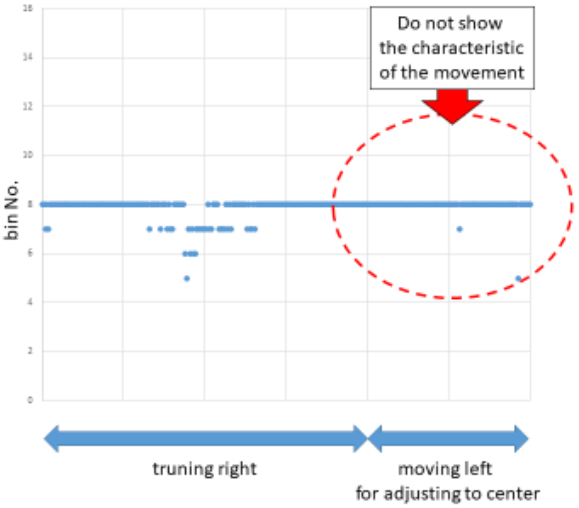

Figure. 16: Changes of bins from images taken from the left side of the car

5 CONCLUSION

In this study, we have proposed and evaluated a method to synchronize video and acceleration data from different sensors, connected to different systems, on a moving vehicle. In our method, we calculated the synchronization points by determining a right/left turning event from camera image data and acceleration data. From the camera image, we used the tendency of optical flows of the camera frame to detect the range of the event by continuously detecting the specific tendency of the corresponding vectors.

From the acceleration data, we detected the situation by identifying the peak or inflection point of the acceleration.

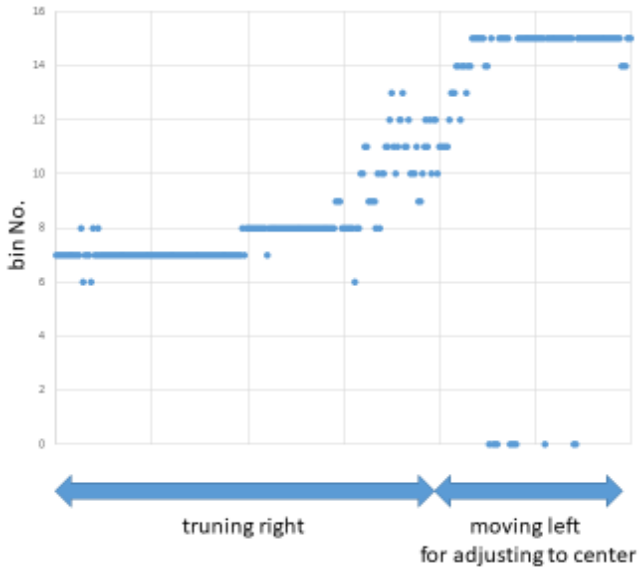

Figure. 17: Changes of bins from images
from images of the left side of the car and matched

We evaluated the fundamental performance of our method using the camera image and the acceleration data acquired from a smartphone in a vehicle, and the error-frame average was 39.103 ms. However, some problems need to be addressed, such as improving the precision of detecting the range of an event from the camera image.

In addition, even though we determined that the difference between the acquisition time of video data and that of acceleration data is small, we will investigate a means of entirely removing this time difference, so as to further enhance the overall accuracy and utility of our method.

REFERENCES

[1] "Autonomous long-distance drive", Mercedes-Benz, https://www.mercedes-benz.com/en/mercedes-benz/innovation/autonomous-long-distance-drive/
[2] Google Self-Driving Vehicle Project, https://www.google.com/selfdrivingvehicle/
[3] https://www.mlit.go.jp/common/001217658.pdf
[4] T. Kantonen, "Sensor Synchronization for AR Applications", the 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp.245-246 (2010).
[5] S. Schneider, *et al.*, "Fusing Vision and LIDAR – Synchronization, Correction and Occlusion Reasoning", 2010 IEEE Intelligent Vehicles Symposium, pp.388-393 (2010).
[6] S. Kitazawa, *et al.*, "Study of Update Timing of Control Target Calculation for an Autonomous Vehicle using Risk Potential", 2018 JSAE Annual Congress (Spring), No. 54 (2018).
[7] R. Yanase, *et al.*, "Self-Localization based on Elevation Map for Autonomous Driving", 2018 JSAE Annual Congress (Spring), No. 55 (2018).
[8] K. Sato, *et al.*, "Changes in Drivers' Levels of Wakefulness during Automatic Driving in an Actual Driving Environment", 2018 JSAE Annual Congress (Spring), No. 139 (2018).
[9] M. Obayashi, *et al.*, "Fast Optimization for Motion Planning Using MPC based on Approximated Problem", 2018 JSAE Annual Congress (Spring), No. 238 (2018).
[10] S. Kitajima, *et al.*, "Development of Proving Ground Testing of Safety Related Performance Evaluation for an Automated Driving Prototype Car before Field Operation Tests", 2018 JSAE Annual Congress (Spring), No. 340 (2018).
[11] K. Okamura, *et al.*, "Comparison of Trajectory Accuracy between RTK GPS and QZSS for the Autonomous Driving", 2018 JSAE Annual Congress (Spring), No. 373 (2018).
[12] T. D. Son, *et al.*, "Autonomous Valet Parking Planning and Control Developments", 2018 JSAE Annual Congress (Spring), No. 435 (2018).
[13] M. Morimoto, *et.al.*, "Estimating Vehicle Speed from In-Vehicle Monocular Camera Footage", The Journal of The Institute of Image Information and Television Engineers, Vol. 68, No. 1, pp.J47-J54 (2014).
[14] L. Fridman, *et al.*, "Automated Synchronization of Driving Data Using Vibration and Steering Events", Pattern Recognition Letters, Vol. 75, pp.9-15 (2016).
[15] A. Giachetti, *et al.*, "The Use of Optical Flow for Road Navigation", IEEE Transactions on Robotics And Automation, Vol. 14, No. 1, pp.34-48 (1998).
[16] S. Tanaka, *et al.*, "Study of Time Synchronization Method between Multiple Wearable Sensors", the 13th of the SOFT Kyushu Chapter Annual Conference. (2011).
[17] Y. Ishiwatari, *et al.*, "A Synchronization Method Between Movie and Sensor Data Using Directions of Vectors Determined by Corresponding Points Between Video Frames", IPSJ SIG Technical Reports, 2018-ITS-73 (2018).
[18] K. Fukuda, *et al.*, "Effect of Experience on Estimate of Time for Turning across Opposite Traffic", Transactions of Japan Society of Kansei Engineering, Vol. 11, No. 1, pp.97-102 (2012).

(Received January 16, 2019)
(Revised September 2, 2019)

Yosuke Ishiwatari received the B.E. and M.E. degrees from the University of Tokyo, Japan, in 1997 and 1999, respectively. In 2002, he completed coursework in Ph.D. program in information science at the Graduate School of Science, University of Tokyo. In 2003, he joined the Research and Development Center of Mitsubishi Electric Corporation as a researcher. He is mainly engaging in risk estimation for autonomous driving. He is also a member of IEEE, ACM, IEICE and IPSJ.

Takahiro Ohtsuka received his B.E., M.E., and Ph.D. degrees from the Utsunomiya University, Japan, in 1997, 1999, and 2002, respectively. Since 2002 he has been primarily engaged in signal processing, machine learning, and artificial intelligence at the Research and Development Center of Mitsubishi Electric Co., Japan.

Masahiro Abukawa received his B.E. degree from Yokohama National University in 1990. He joined Mitsubishi Electric Corp. Currently he is a general manager of Human Intelligent Technology Dept. at Information Technology R&D Center. He is also a member of IPSJ.

Hiroshi Mineno received his B.E. and M.E. degrees from the Shizuoka University, Japan in 1997 and 1999, respectively. In 2006, he received his Ph.D. degree in information science and electrical engineering from the Kyushu University, Japan. Between 1999 and 2002, he was a researcher at the NTT Service Integration Laboratories. In 2002, he joined the Department of Computer Science of Shizuoka University as an Assistant Professor. He is currently a Professor. His research interests include intelligent IoT systems as well as heterogeneous network convergence. He is also a member of ACM, IEICE, IPSJ, and the Informatics Society.