# Stream Geo-location Data Processing for Detecting Rescuers in a Large-scale Disaster

Hideki Hayashi
Center for Technology Innovation - System Engineering, Research & Development Group, Hitachi, Ltd.
Kokubunji-shi, Tokyo
hideki.hayashi.xu@hitachi.com

Yuichi Ogawa
IoT & Cloud Services Business Div., Service Platform Business Division Group, Hitachi, Ltd.
Yokohama, Kanagawa
yuichi.ogawa.nf@hitachi.com

Natsuko Sugaya
IoT & Cloud Services Business Div., Service Platform Business Division Group, Hitachi, Ltd.
Yokohama, Kanagawa
natsuko.sugaya.tr@hitachi.com

Akinori Asahara
Center for Technology Innovation - System Engineering, Research & Development Group, Hitachi, Ltd.
Kokubunji-shi, Tokyo
akinori.asahara.bq@hitachi.com

Hitoshi Tomita
Social Innovation Business Division, Hitachi, Ltd.
Minato-ku, Tokyo
hitoshi.tomita.jg@hitachi.com

## ABSTRACT

In a large-scale disaster, it is important for rescue workers to quickly grasp the disaster situation that changes from moment to moment on rescue activities in the field. In this study, we focused on large-scale stream geo-location data sequentially collected by moving sensors such as smartphones. Then we developed a technology which can send a notice to external systems in real-time when the distance between the two streams data becomes closer. In a scenario of Tokyo Inland Earthquake, we assumed one million stream data items must be processed in 5 minutes and our evaluation showed that our proposed method can achieve it.

## CCS CONCEPTS

• **Information systems** → **Location based services**; **Data streaming**;

## KEYWORDS

disaster management, mobile, stream data, location data, real-time

**ACM Reference Format:**
Hideki Hayashi, Yuichi Ogawa, Natsuko Sugaya, Akinori Asahara, and Hitoshi Tomita. 2017. Stream Geo-location Data Processing for Detecting Rescuers in a Large-scale Disaster. In *EM-GIS'17: EM-GIS'17:3rd ACM SIGSPATIAL Workshop on Emergency Management using , November 7–10, 2017, Redondo Beach, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3152465.3154400


*EM-GIS'17, November 7–10, 2017, Redondo Beach, CA, USA*

ACM ISBN 978-1-4503-5493-6/17/11. . . $15.00
https://doi.org/10.1145/3152465.3154400

## 1 INTRODUCTION

Japan is located in the "Circum-Pacific Mobile Belt," where seismic and volcanic activities occur constantly [1]. Although the country covers only 0.25% of the land area on the planet, it experiences a high number of earthquakes and has many active volcanoes (From the ratio of natural disasters in Japan to those in of the world described by [1], the ratio of earthquakes with magnitude of 6.0 or greater from 2004 to 2013 is 18.5% and the ratio of active volcanoes in 2014 is 7.1%). Due to its geographical, topographical, and meteorological conditions, Japan is subject to frequent natural disasters such as typhoons, torrential rainfall, and heavy snowfall. Consequently, such natural disasters can result in a large loss of life and significant damage to property. Examples of recent large-scale natural disasters that caused immense damage are the Great East Japan Earthquake (March 2011), landslides in Hiroshima Prefecture (August 2014), the volcanic eruption of Mt. Ontake (September 2014), and the Kumamoto earthquakes (April 2016). The probability of an M7-class earthquake occurring in the South Kanto area (around Tokyo) within 30 years is estimated to be 70%. If an earthquake directly hits the Tokyo metropolitan area, the human and material damage caused by collapsing buildings and the spread of fires will be extremely serious. Accordingly, it is a national priority to protect citizens' lives, livelihoods, and property from large-scale natural disasters.

In the event of a large-scale disaster, it is important for on-site rescue workers to quickly grasp the constantly changing disaster situation. For example, if rescue workers can grasp the disaster situation near the site of a large-scale natural disaster (i.e., when, where, and what kind of disaster occurred) and the distribution and movement of people present near the site in real time, decision making could be quicker

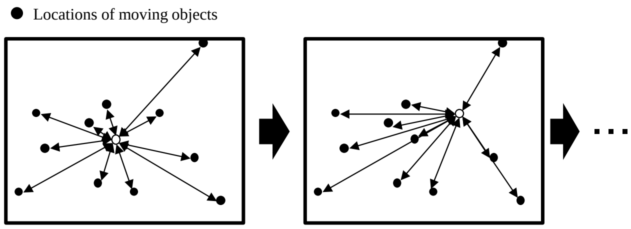


**Figure 1: Processing of stream geo-location data (conventional method).**

and more efficient. Specifically, at the disaster site, it is important to detect people in dangerous situations and rescue them.

To detect a person in a dangerous situation and rescue them, it is assumed that the person (called a "moving object" hereafter) has a mobile terminal, such as a smartphone, capable of location measurement and periodically transmitting the location data to a server. In this case, the server must continually receive the location data from the moving object and determine the distance between any two moving objects. Hereafter, the location data that the server continuously receives from the moving object is referred to as "stream geo-location data."

When the amount of stream geo-location data is large, it is a problem that the amount of processing (such as for the distance calculation) abruptly increases. In this process, when the server receives location data from a moving object, it focuses on the location data, calculates the distance to surrounding moving objects, and detects the surrounding moving object with the shortest distance from the target moving object. A solution to the abrupt increase in the amount of processing is to prepare a sufficient number of computers on the server side and share the processing between them. However, it is not realistic to prepare a large number of computers to handle this enormous amount of processing.

Given the issues described above, in this study, we propose a method for processing stream geo-location data in real time with the aim of enabling rescue workers to quickly grasp those who are in a dangerous situation and require immediate rescue during a large-scale disaster. This method efficiently calculates the distance between moving objects from the stream geo-location data every time the location of a target moving object changes and outputs the calculation results in real time. An inland earthquake in the Tokyo area was assumed as the use case for this technique. Specifically, an area with a high risk of fire and building collapse due to such a Tokyo inland earthquake was assumed. This area has a population of one million and a population density of 10,000 people/km2. Under this assumption, one million moving objects transmit location-data once every five minutes (3,334 location-data transmissions/second). Processing in real time with a single server and making maximum use of given hardware resources were set as performance requirements that must be satisfied by the proposed method.

The remainder of this paper is organized as follows. Section II describes related works and clarifies the position of this research. Section III explains the proposed method and a system based on the proposed method that detects people in an isolated state in real time. Section IV presents the results of an evaluation of the performance of the system. Finally, Section V concludes this paper and describes directions for future work.

## 2 RELATED WORKS

Several efforts have been made to grasp a disaster situation by utilizing location data transmitted by sensors and data associated with that location data for assisting disaster-prevention-related organizations during large-scale disasters. For example, in Japan, an earthquake disaster-prevention information system (DIS: Disaster Information System) has been constructed [1]. As for this system, on the basis of seismic-intensity information collected at fixed observation sensors (about 4,200) installed by the Japan Meteorological Agency and other organizations, it automatically starts when an intensity of four or more on the Japanese seven-stage seismic scale is observed, and estimates the seismic intensity distribution within approximately 10 minutes after the earthquake occurs. The proposed method differs from this system because it deals with data whose location changes every moment, like the case of a moving object.

Disaster situations have been estimated by utilizing location data collected from probe cars during a large-scale disasters being conducted. This case estimates the damage by identifying the road with the car's track record from the location data sent from the probe car and visualizing a road map. In other researches, the damage situation is being determined by taking into consideration the shortage of bandwidth caused by a communication network damaged during a disaster. For example, on the basis of "crowdsourcing," visual data such as movies taken by a mobile terminal at the disaster site are being efficiently collected and transmitted [2]. These researches are similar to our research in terms of utilizing location data of moving objects. However, our research differs in that it takes into consideration the positional relationship with surrounding moving objects and real-time processing.

In the field of data-stream processing, various data-stream-processing systems with the aim of real-time processing of continuously generated stream data have been developed[3–5]. In recent years, several platforms connecting a large number of computers in parallel have been developed, and they can process large-scale stream data by distributed processing in parallel[6, 7]. Since our proposed method (targeting location data of moving objects) is independent of its implementation, it is thought to be applicable to these platforms too.

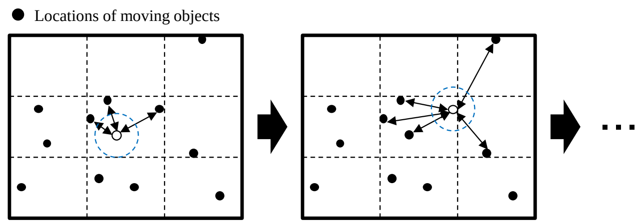


**Figure 2: Processing of stream geo-location data (prposed method).**

Real-time processing of location data has been widely researched. As representative results of such research, spatio-temporal data stream technology has been proposed. With such technology, location data is continuously generated, and in an environment where users' queries are also continuously registered, the answers to users' inquiries are returned in real time[8]. This technology returns a "range query" (e.g., an object within a radius of 10 meters around a certain point) and a "k-nearest-neighbor search query" (e.g., high-order k moving objects within a distance from a certain point). Our research is similar to this research in the sense that it deals with real-time processing of location data of moving objects. It differs in the sense that it detects when the distance between two or more moving objects is less than a threshold value.

# 3 PROPOSED METHOD

## 3.1 Basic Ideas

As for the basic approach taken by the proposed method, a geographical space is divided into lattices and stream geo-location data in each lattice is managed (see Fig. 2). In particular, space is divided in advance into lattice-shaped regions and identification numbers are allocated to each region. Upon receiving a location data of a moving object, a server determines the region in which the location is included and gives an identification number to that region. When the distance between moving objects is calculated, instead of calculating the distance by using all the moving objects, as in the conventional method, only moving objects with identification numbers of regions within a certain short distance (i.e., regions overlapping a circle marked by a broken line) are used. This approach greatly reduces the amount of stream geo-location data that requires distance calculation and reduces the processing cost of the calculation. With this approach, the number of distance calculations can be greatly reduced, and the processing cost of distance calculation can be reduced.

## 3.2 Implementation

The attributes included in the stream geo-location data handled by the proposed method are listed in Table 1. It is assumed that this data is collected from mobile terminals such as smartphones.

**Table 1: Attributes of stream geo-location data.**

| Attribute | Description |
|---|---|
| Creation time | Time when location data is generated |
| ID | Identifier associated with a sensor that generates location data |
| Longtitude | Longitude of moving object |
| Latitude | Latitude of moving object |
| X coordinate | X coordinate converted into a projected coordinate system |
| Y coordinate | Y coordinate converted into a projected coordinate system |

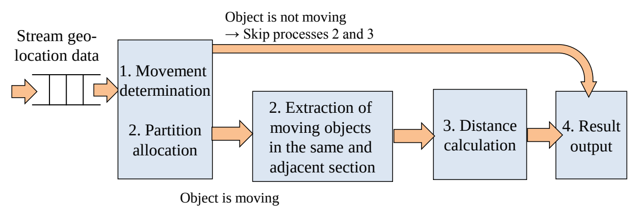


**Figure 3: Processing procedure of proposed method.**

The processing procedure used by the proposed method is explained schematically in Fig. 3. It consists of the following four steps: 1) movement determination, 2) partition allocation and extraction of moving objects in the same and adjacent sections, 3) distance calculation, and 4) result output. Each step is executed one by one each time location data (received data) of a moving object is received. Each step is described in detail below.

*3.2.1 Movement Determination.* The location (coordinates) included in the received data and the data with the same ID received previously are compared, and whether or not the moving object corresponding to the ID of the received data is moving is determined. If the moving object has not moved since the previous data reception, steps 2 and 3 are skipped (because if the object is not moving, distances to the surrounding moving objects do not change and it is not necessary to calculate the distances). On the other hand, when the moving object has moved since the previous data reception, it is necessary to perform the steps 2 and 3 (because the distances to the surrounding moving objects have change).

*3.2.2 Partition allocation and extraction of moving objects in the same and adjacent sections.* The region of the lattice shape (on the projected coordinate system) to which the received data belongs is determined from the location of the received data (i.e., the coordinates of the projected coordinate system), and the region ID is given to the received data. In this step, to reduce the computational cost of determining which region the moving object is included in, the region is given a fixed size and divided into regular spaces. By setting minimum coordinates, maximum coordinates, and division width of each divided space and giving the position of the moving object, it is possible to calculate the region in which the moving object is included. For example, in an $XY$ plane, the minimum coordinates are $(X_{min}, Y_{min})$, the maximum coordinate is $(X_{max}, Y_{max})$, and the division width between the $X$ and $Y$ axes is $\Delta W$. As the definition of region ID, the minimum value of region ID is zero, region ID takes a small value for a region with small $Y$ coordinate, and the region with the same $Y$ coordinate is small as the $X$ coordinate is small. Given a location $(x, y)$ of a moving object, the region ID can be obtained from $\lfloor (y - Y_{min})/\Delta W \rfloor \times (X_{max} - X_{min})/\Delta W + \lfloor (x - X_{min})/\Delta W \rfloor$ (where $\lfloor\rfloor$ is the largest integer value not exceeding the numerical value and $(X_{max} - X_{min})$ is divisible by $\Delta W$). As described later, with the proposed method, moving objects in the same and adjacent regions as the target moving object are set as objects of the distance calculation. Therefore, the width of a region is calculated within a given range of distances(surrounding distance). For example, in the case that the surrounding distance is r, to ensure that the moving object to be subjected to the distance calculation is included in the same or adjacent regions with the moving object, the conditional expression of the region width $\Delta W$ is $\Delta W \geq r$. If r is set to 200 meters, $\Delta W$ is set to 200 meters or more. When the width of the region is increased, the number of moving objects included in the region increases, and the number of moving objects subject to distance calculation tends to increase. It is therefore desirable to set $\Delta W$ to the smallest-possible value (namely, set $\Delta W$ to $r$). Accordingly, the stream geo-location data from the same region as the received data and the region ID of the adjacent region are extracted as the geo-location data for distance calculation.

*3.2.3 Distance calculation.* The distance to the received data is calculated with respect to the stream geo-location data to be subjected to the distance calculation extracted in step 2. Here, as a representative distance, the distance is the squared distance based on the Euclidean distance. In this implementation of the proposed method, identifying "a person isolated from surrounding people" is focused on from the viewpoint of detection of the requisite rescuer at the time of a large-scale disaster. It is assumed that every time a location data of the moving object is received, a person isolated from the surroundings is detected by counting the number of persons around that person. Such isolated-person detection using the proposed method is shown in Fig. 4. In this figure, the distance at which the number of people around a certain person is counted is set to 200 meters. This distance is hereafter referred to as the "surrounding distance." When no single person is within the surrounding distance of the target person, that person is considered isolated. When the location of moving object A is received, A is counted as having six people within 200 meters around it; in other words, A is not detected as isolated. On the other hand, when the location of moving object B is received, since no person is within 200 meters of B, it is considered that B is most probably isolated.

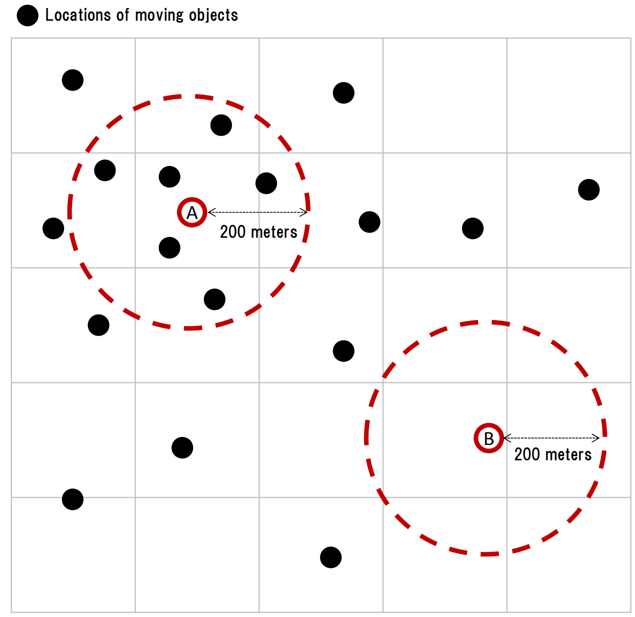


**Figure 4: Detection of isolated person by proposed method.**

**Table 2: Prerequisite of perfomance evaluation.**

| Conditions | Values |
|---|---|
| Area to be evaluated | Northen part of the 23 wards of Tokyo |
| Population | About one million people |
| Population density | Approximately 10,000 people/km$^2$ |
| Distance within which to count people ("surrounding disntace") | 200 meters |

*3.2.4 Result Output.* In this process, the result of step 3 is held in the main memory of the server. This result is consecutively referred to by requests from applications such as geographic-information systems.

# 4 PERFORMANCE EVALUATION

## 4.1 Evaluation Objective

The effectiveness of the proposed method in terms of performance was evaluated. Moreover, the evaluation results confirm that the proposed method satisfies the performance requirement defined in the use case listed in Table 2.

## 4.2 Evaluation environment

As stream geo-location data, pseudo data generated from "People flow data" provided by the Center for Spatial Information Science, the University of Tokyo[9] was used. "People flow data" is movement data of persons created from data obtained by personal investigation by the national government, local governments, and other organizations and used

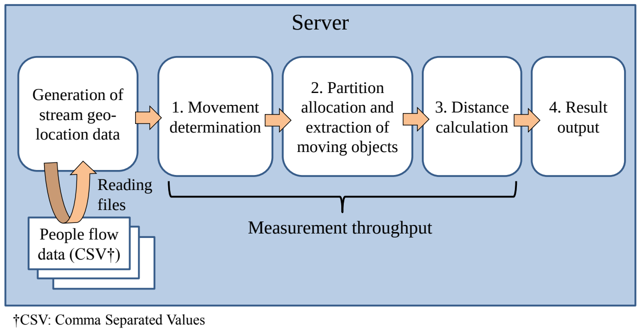


**Figure 5: Measurement environment of performance evaluation.**

in research fields such as disaster prevention and transportation.

*4.2.1 Measurement Environment.* The measurement environment used in the performance evaluation of the proposed method is shown in Fig. 5. A server with a 1.80-GHz × 4-core CPU, a 16-GB memory, and a 600-GB hard disk (2.5 inch, SAS 2.0, and 10,000 rpm) were used. Since the proposed method does not involve access to the hard disk, in terms of hardware, the performance of the proposed method is affected by the performance of the CPU of the measuring server. CentOS 6.4 was installed on the server, and a commercial stream data-processing platform was used. The proposed method and a conventional method were implemented on this stream-data processing platform, and their performances were evaluated.

*4.2.2 Evaluation Method.* As an evaluation criteria, throughput (i.e., the number of processes that can be processed per second in steps 2 and 3 in Fig. 5) was used. As a method for calculating the throughput, "transfer rate" is gradually reduced by thinning out the transfer interval from one minute to n minutes sequentially by using a pseudo-data-generation program, and the queue is almost not accumulated. The transfer rate was taken as the throughput that can be processed by the proposed method. Here, the transfer rates for the one-minute interval and a two-minute interval differ. In other words, the actual maximum throughput that can be processed differs according to the interval. In this case, the queue accumulation was confirmed, and the throughput was estimated.

*4.2.3 Generation of Pseudo Data.* A program to continuously transmit pseudo location data was created. Utilizing "People flow data: 2008 Tokyo Metropolitan area," this program provided stream geo-location data of areas that are considered to be at high risk of fire and building collapse during an inland earthquake in the Tokyo. This area, which is shown in a gray rectangle of Fig. 6, has a population of about one million at a population density of about 10,000 people / km$^2$.


**Figure 6: Evaluation target area.**

**Table 3: Input rate of location data.**

| Time | Number of people | Interval for reading a file including people flow data [/sec] 1-min interval | 5-min interval | 15-min interval |
|---|---|---|---|---|
| 8:00 | 1,395,382 (15,461) | 23,256 | 4,651 | 1,550 |
| 14:00 | 1,559,764 (17,283) | 25,996 | 5,199 | 1,733 |
| 18:00 | 1,471,472 (16,320) | 24,525 | 4,905 | 1,635 |
| 21:00 | 1,228,724 (13,615) | 20,479 | 4,096 | 1,365 |

Input rates of location data used in this evaluation are listed in Table 3. For example, if 1,395,382 people are present in a certain area at 8 am and all their location data are transmitted at one-minute intervals, 23,256 locations will be transferred to the server per second.

## 4.3 Evaluation Results

*4.3.1 Comparative Evaluation of Conventional Method and Proposed Method.* Fig. 7 shows the throughputs of the proposed method and the conventional method when the time is from 8:00 to 9:00. The vertical axis of this graph shows throughput in logarithmic scale display. The reason for choosing the result of this time is because it is the time when the person moves the most, which is the most severe condition for the proposed method.

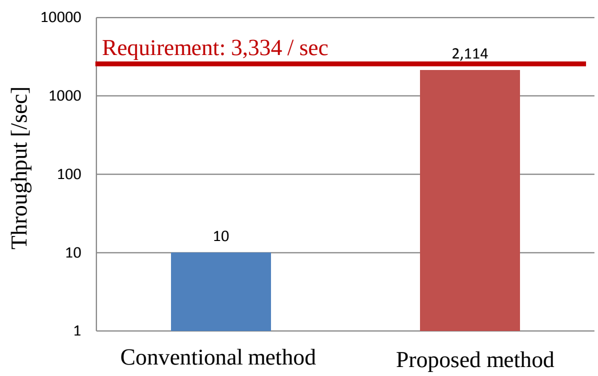


**Figure 7: Throuput of conventional and proposed methods.**

Throughputs of the proposed method and the conventional method when the data is transmitted from 8 am to 9 am are shown in Fig. 7. The vertical axis of this graph shows throughput on a logarithmic scale. The result (throughput) for this time period was chosen because it is the period when people move the most, which is the most-severe condition for evaluating the proposed method.

The figure shows that the proposed method achieves a higher throughput than that of the conventional method. This is because the proposed method makes it possible to significantly reduce the amount of processing required for the distance calculation, since only a moving object within the distance criterion of a region is to be subjected to the distance calculation. Moreover, the proportion of people not moving at 8 am is 43%, so the proposed method can skip the distance calculation for that time accordingly. On the other hand, the throughput of the proposed method did not satisfy the performance requirement (3,334 locations/second).

*4.3.2 Process Parallelization of Proposed Method.* Since the proposed method could not achieve the performance requirement (3,334 locations/second) as shown in Fig. 7, the utilization states of the CPU and memory of the server was investigated. The result of that investigation showed that both CPU utilization rate and memory utilization rate of the server indicate reserve capacity. Especially in the state that the memory is empty, CPU utilization rate was about 25% on average. Accordingly, the performance of the proposed method can be improved by using the CPU more efficiently. As a way to achieve more efficiency, parallelizing the proposed method with a single server was considered. As a basic idea, dividing the target area into multiple areas and processing each area in parallel was considered. Hereafter, as an implementation for parallelizing the proposed method, a method for dividing areas and a system architecture for parallelization were devised as explained below.

- Area division method for parallelization
  As for the method for dividing areas for parallelization, it is important to consider a moving object near the boundary of the area. Hereafter, an example in which the area is divided into four sections is shown in Fig. 8.
  As shown on the left side of Fig. 8, when an area is simply divided, a moving object (indicated by "x") in section 1 and two moving objects (indicated by black circles) in section 2 are not subject to the distance calculation. Therefore, as shown on the right side of the figure, this parallelization method divides areas by overlapping divided sections, and aggregates the process results in the overlapping sections. As a result, it is possible to eliminate omission of the distance calculation near the boundary of those sections. The result of applying the area-division method to the target area in this use case is shown in Fig. 9.
- System architecture for parallelization of proposed method

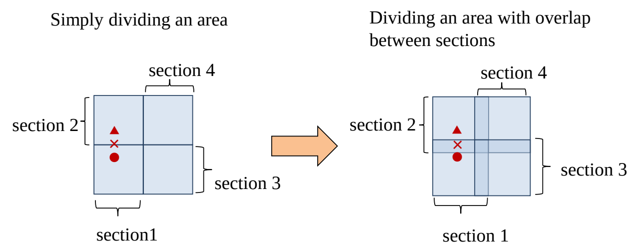


**Figure 8: Area-division method for parallelization of proposed method.**

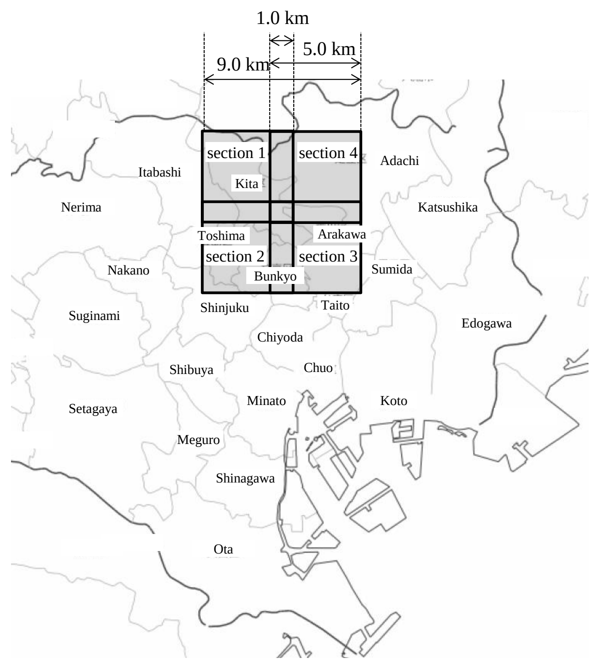


**Figure 9: Area-division for target area.**

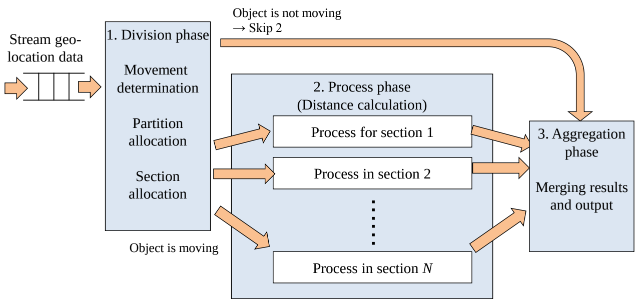


**Figure 10: System architecutre for parallelization of the proposed method.**

The system architecture for parallelization of the proposed method and its processing flow are shown in Fig. 10. Each parallelization process is described below.

- Division phase
  In the division phase, the section of the divided area to which the received location data belongs is determined, and an section ID is attached to the received location data. A plurality of section IDs are assigned to location data near the boundary of the divided sections.
- Process phase
  In the process phase, the distance between any two moving objects within each section is calculated on the basis of the section IDs assigned in the division phase. Then, the number of people within a given surrounding distance from the received location data is determined for each section. Location data near the boundary of sections are counted in a plurality of sections.
- Aggregation phase
  In the process aggregation phase, location data in near section boundaries is counted in duplicate in a plurality of sections The aggregation process eliminates the duplication of these count results and merges them into one result. For example, let us consider a situation in which three people are in section 1 and five people are in section 2 when the server receives location data. In this case, the number of people in the surroundings is counted as eight. On the basis of this aggregation result, an isolated person is detected.

*4.3.3 Effect of Processing Parallelization.* The effectiveness of parallelizing the proposed method was confirmed as follows. The results of single processing and four, six, and eight parallel processing under the conditions listed in Table [**?** ] from 8:00 to 9:00 are shown in Fig. 11.

This result indicates that parallel processing achieved higher throughput than single processing, and the parallelization method worked effectively. Additionally, it confirms that the parallel processing satisfied the performance requirement (3,334 locations/second). It is thought that the reason that the performance improvement in the case of four-parallel processing was slightly less than three times as compared with the single processing is due to the bias in the number of people in each section (as shown in Table 4). It was found that data was not evenly distributed among the four divided sections, and it was greatly affected by section 3, in which the most data must be processed. It is therefore considered possible to further improve performance by dividing the target area so that the numbers of people in each divided section are even.

In the case of four parallel processing, average throughput per thread was 1,454 locations/second. For six parallel processing, average throughput per thread was 1,478 locations/second. These results indicate that six parallel processing had the same effect of parallelization as four parallel processing. However, in the case of eight parallel processing, average throughput per thread was 1,214 locations/second, and the effect of parallelization was lower than in the case of four parallels and six parallel processing. This is because CPU utilization of the server was 65% during four parallel processing and 79% during six parallel processing, whereas it reached close to 90% during eight parallel processing. In the case of using a server in the measurement environment, it is thought that up to eight parallel processing can effectively improve throughput. If the number of CPU cores of the server is larger than that of this server, throughput can

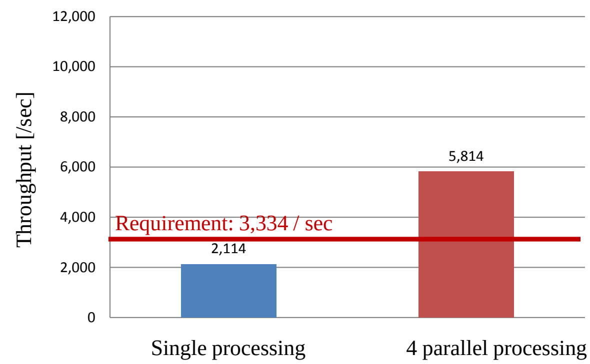


**Figure 11: Results of single and parallel processing.**

**Table 4: Number of people and rate of processing for each section at 8:00.**

| Section | Main districts | Number of people | Rate of processing[%] |
|---|---|---|---|
| Section 1 | Kita ward<br>Itabashi ward | 577,816 | 30 |
| Section 2 | Bunkyo ward<br>Toshima ward | 306,149 | 20 |
| Section 3 | Arakawa ward<br>Taito ward | 641,841 | 38 |
| Section 4 | Adachi ward | 179,928 | 12 |

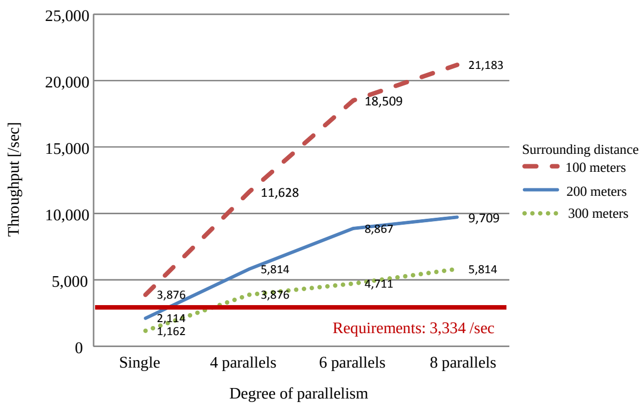


**Figure 12: Effects of varing surrounding distance.**

be expected to be improved even if parallelization by eight or more parallel processing is performed.

*4.3.4 Effect of Surrounding Distance.* To investigate the influence of the performance of the proposed method on the change of the surrounding distance, the performance under various surrounding distances was evaluated. The evaluation results are shown in Fig. 12.

This result shows that throughput decreases as surrounding distance increases. This is because as the surrounding distance increases, the number of mobile objects to be subjected to the distance calculation increases, since the server receives the location data of the moving object, and the processing amount of the distance calculation increases. In addition, it shows that the effect of performance improvement by parallelization is smaller as the surrounding distance increases. This is because the bias of the number of people present in the section becomes more conspicuous when the surrounding distance becomes longer.

*4.3.5 Summary of Evaluation results.* The evaluation results are summarized as follows.

- Compared with the conventional method, the proposed method greatly improves throughput. However, in the case of single processing, the proposed method could not achieve the target throughput of 3,334 locations per second.
- With the proposed method, the target area is divided into multiple areas, and processing is parallelized. As a result, in the case of four parallel processing, throughput was improved about three times compared to that achieved with single processing, and the target throughput of 3,334 locations per second could be achieved. In this measurement environment, it was confirmed that parallelization by up to eight parallel processing is effective.
- In the parallelization of the proposed method, when the number of people in each section is biased, the processing concentrates on the section with the largest number of people, so the effect of parallelization decreases. This trend becomes more prominent when the surrounding distance is enlarged. To solve this problem, an approach to dynamically divide an area based on the bias of the number of people is effective and is considered as a future task.

# 5 CONCLUSIONS

Large-scale stream geo-location data sequentially collected by moving sensors such as smartphones was focused on. A technology that sends a notice to external systems in real-time when the distance between the two streams data becomes closer was developed. In a scenario of inland earthquake in the vicinity of Tokyo, it was assumed that one million streaming data must be processed in five minutes, and our evaluation showed that the proposed method can satisfy that requirement. Specifically, it was confirmed that process parallelization of the proposed method can be performed by a single server, hardware resources can be utilized to the fullest, and it can be processed in real time.

In the future, it is necessary to set a more concrete scenario of large-scale disasters and to verify the practicality of this technology.

# ACKNOWLEDGMENTS

This work is supported by consignment research and development of techniques about use and application of real-time information in G-space platform from the Ministry of Internal Affairs and Communications, Japan.

# REFERENCES

[1] Cabinet Office, Government of Japan: Disaster management in Japan, http://www.bousai.go.jp/1info/pdf/saigaipamphlet_je.pdf
[2] To, H., Kim, S.H, and Shahabi, C., "Effectively Crowdsourcing the Acquisition and Analysis of Visual Data for Disaster Response," *Proc. of IEEE BigData'15* (2015).
[3] Babu, S. and Jennifer, W., "Continuous Queries over Data Streams," SIGMOD Record, vol. 30, issue. 3, pp.109-120 (2001).
[4] Abadi, D.J., Carney, D., Cetintemel, U., Cherniack, M., Convey, C.; Lee, S., Stonebraker, M., Tatbul, N., and Zdonik, S., "Aurora: A New Model and Architecture for Data Stream Management," *The VLDB Journal*, vol.12, no.2, pp.120-139 (2003).
[5] Hitachi Data Systems, Hitachi Streaming Data Platform, https://www.hds.com/en-us/products-solutions/big-data-analytics/streaming-data-platform.html
[6] Apache Storm, http://storm.apache.org/
[7] Apache Spark, http://spark.apache.org/
[8] Mokbel, M.F. and Aref, W.G., "SOLE: Scalable On-line Execution of Continuous Queries on Spatio-temporal Data Streams," *VLDB J.*, vol.17, no.5, pp.971-995 (2008).
[9] Center for Spatial Information Science, The University of Tokyo, Collaborative research, Urban space information platform able to assimilate dynamic data, People Flow Project, http://pflow.csis.u-tokyo.ac.jp/?page_id=943