The SIGMA Framework for Managing Spatiotemporal Data Acquired by Various Cameras and Sensors

Kento Yamazaki
Graduate School of Information Science & Engineering Ritsumeikan University
Kusatsu, Japan
yamazaki@rm2c.ise.ritsumei.ac.jp

Tomonori Aritomi
Graduate School of Information Science & Engineering Ritsumeikan University
Kusatsu, Japan
aritomi@rm2c.ise.ritsumei.ac.jp

Guan Sikun
Graduate School of Information Science & Engineering Ritsumeikan University
Kusatsu, Japan
shikon@rm2c.ise.ritsumei.ac.jp

Asako Kimura
Graduate School of Information Science & Engineering Ritsumeikan University
Kusatsu, Japan
asa@rm2c.ise.ritsumei.ac.jp

Fumihisa Shibata
Graduate School of Information Science & Engineering Ritsumeikan University
Kusatsu, Japan
fshibata@is.ritsumei.ac.jp

***Abstract*—This paper describes the design of an application framework for advanced use of spatiotemporal imagery data, and the application developed with our framework. Our framework provides a mechanism to share the data captured from a wide variety of sensors in the real world and makes it easy for developers to implement applications which use those data. However, the captured data is scattered across time and space; thus, we designed the unified data management policy for handling a variety of sensor data as 4D data. We tried developing a few prototype applications with our framework and considered the prototype.**

Keywords*—*application framework, camera image, point cloud, diminished reality, tree structure

I. Introduction

There are a wide variety of sensors over the real world such as surveillance cameras, vehicle-mounted cameras, and so on. A society will soon be coming where these kinds of sensors will constantly obtain real world information. Additionally, these sensors are essential for autonomous mobility systems such as unmanned aerial vehicles and self-driving cars. Considering these backgrounds, the numbers of these sensors are increasing from here on out (Fig. 1).

We can provide various services which make use of the stored data including camera images and point cloud acquired by these sensors. For example, we can create a top-view image to synthesize multiple cameras mounted a car, and we can create a 3D model of a building using captured images from multiple cameras [1]. This is an example of using captured data spread spatially, however captured data spread not only in space but also in time. For example, if images from multiple cameras were accumulated in chronological order, we see the past image taken at about the same location on Mapillray [2].

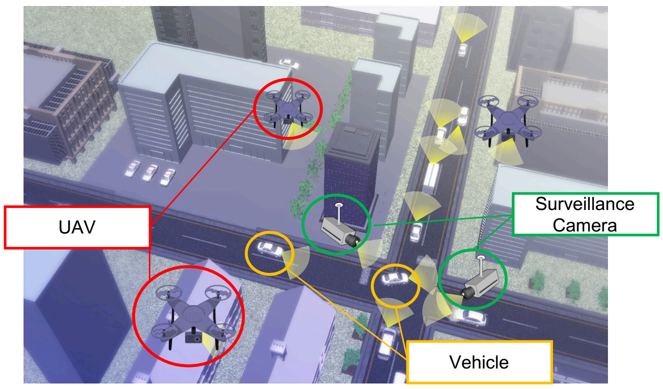

Fig. 1. Image of sensors everywhere

Furthermore, there is research that substitutes past images for some of the current images [3].

For example, self-driving cars use 3D maps created from various captured data. Therefore, a system that can collect and store data and use it easily is indispensable for making advanced use captured data. For the above reason, we propose an application framework named SIGMA (Spatiotemporal Images with Generalized Management Architecture), which can use data spread both temporally and spatially, so that collecting and storing a large amount of data scattered in time and space. Our framework has two required specifications below.

1) Handle uniformly on the time and space axes for sensor data
2) Has a mechanism to share data and easily acquire it

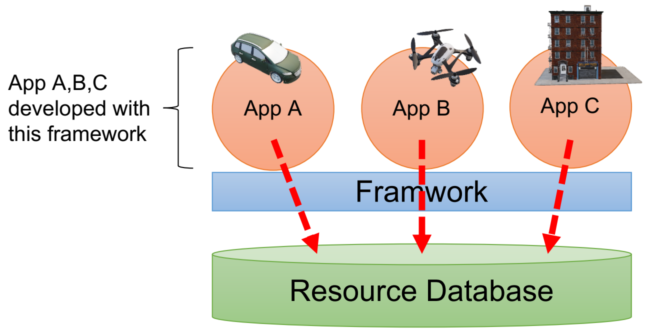

Fig. 2. Sharing the spatiotemporal data with inter-application

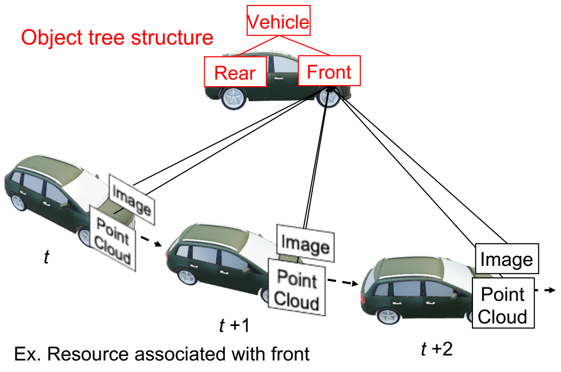

Fig. 3. "Object" and "Resource" on a tree structure

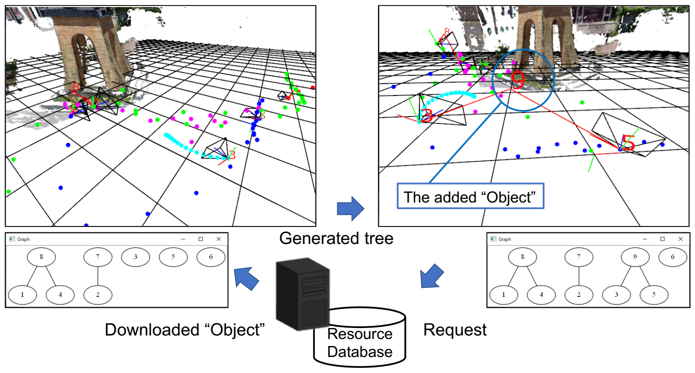

Fig. 4. Method of "Object" management

II. Related Work

AR cloud is one of the concepts close to our framework [4]. A lot of projects started realization of AR cloud. In order for many users to share the data, advanced handling of data is required. Sakurada *et al.* proposed a method to use point cloud data efficiently by regarding temporal data and spatial data as 4D data [5]. In our framework, we construct a system that can commonly use and share 4D information in order to enable advanced use of data obtained by observing the real world. Ogino *et al.* proposed the distributed framework which can share MR data [6]. The architecture of Ogino's framework is close to the architecture of our framework. Ogino *et al.* design the script language in Ogino's framework to make it easy for anyone to develop mobile MR systems. However, our framework does not target the novice developer because developed application is huge system.

III. Spatiotemporal Data

A. Design Concept

We made policies to store the data obtained from the sensor as it is and leaving the way to use it up to the application developer for using data obtained various cameras and sensors in the real world for multiple uses.

First, we focus on data is not obtained from a unique time and space but scattered across time and space. In this study, video and sensing data that have a spatial and temporal spread are shared with our framework. Spatiotemporal data is centrally managed by the server, so that applications developed using our framework can freely use spatiotemporal data by accessing the resource database on the server. Access to any resource database is not limited to the same application; however, permits access from various applications developed with this framework (Fig. 2).

Furthermore, it is necessary that an application permits to upload the data to resource database as an exchange condition for freely accessing the resource database. However, the upload mechanism does not have to be not implemented by an application developer but provided by our framework. In other words, the application developed with our framework uploads the captured data in the background.

When the images and point clouds obtained from scattered sensors are randomly stored in the server, the relationship in the real world is lost. Many sensors are not independent of each other and have a relationship with other sensors. For these reasons, the relationship between individual sensors can be made into a simple data structure. Therefore, we design to manage using a tree structure. It is difficult to manage the sensor data as it is. We design tree structure as an abstract "Object" that the application developer defines independently. We designed a structure that each "Object" node associate "Resourc" consisting of data such as images and meta information (Fig. 3).

"Object" is registered by requesting the tree structure assumed by the application developer to the resource database server. The tree structure that can be constructed is not limited to the "Object" handled by the application, As shown in Fig. 4, the application developer sets the tree structure, while checking the tree structure. Many application developers grow int a more advanced systems by increasing the tree structure.

It is difficult to search anything in the "Object" and "Resource" spread both temporally and spatially. We separate position and time data from metadata of "Object" and "Resource" and define those data as "Geometry". "Geometry" has 3 values on space axis and a value time axis without distinction, that is, "Geometry" is searched as 4D data.

B. Basic Architecture

Our framework is built a client-server system to share spatiotemporal data via a network, over and makes the de-

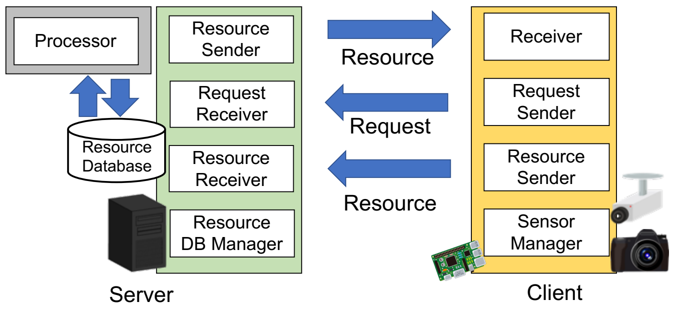

Fig. 5. System architecture

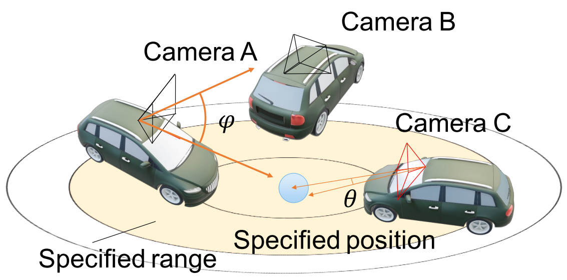

Fig. 6. Search using the specified position

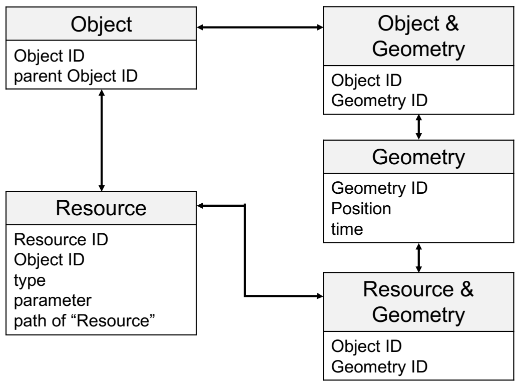

Fig. 7. Relation of the table

velopment of applications using these data (Fig. 5). Thus, it is necessary to split two layers, supplying system layer for share data and application layer for application developer. Also our framework provide the APIs of the necessary mechanisms for developing an application.

There are various mechanisms such as those required for frameworks such as network to the others required for advanced use such as pose estimation. We discussed the mechanisms for advanced use of spatiotemporal data. As a result of discussion, diminished reality (DR) technology that superimposes a hidden background in order to remove the object, image stitching technology, and 3D reconstruction were nominated [7], [8]. Therefore, the following five mechanisms are provided.

1) Pose Estimator: Estimation camera and sensor pose
2) 2D-3D Mapper: Corresponding 2D coordinate values to 3D coordinate values
3) 3D Reconstructor: Reconstruction 3D information from multi-view information
4) Object Detector: Object detection and tracking based on resources
5) Image Stitcher: Stitching multi-view images

Applications are developed using APIs of these mechanisms. Also, some of these mechanisms are not only required by the client for application processing, but also required by the server. As a reason, this framework not only centrally manages scattered resources and provides them to the application, but also permits the creation of data that does not change observation information in the real world like creating a 3D map from a point cloud.

C. Design of the Resource Database

It is necessary to store while retaining the tree structure of the object with which the resource is associated to store resources in the database and search by request. To meet these requirements, we decided to use a relational database (RDB) as the resource database. A general search uses Object ID or Resource ID. As a search unique to our framework, we provide a method to search 4D data by "range" or "distance" using "Geometry". However, the search method by "range" or "distance" depends on the observation position of the camera or sensor, thus; the observation position is not always the same as the position of the observed object. It is ideal to measure the position of the object of the observed a resource, but it is unrealistic to apply it to all resources.

Therefore, it is limited to those captured by the sensor facing the specified position. However, there are innumerable images and point groups in the search range just by facing them. If the difference in the resolution of each sensor is ignored, the sensor closer to the specified position has higher resolution. We added a search method under the following conditions to extract "Resource".

1) Any range from the specified position
2) Observed by the sensor facing the specified position

Whether the sensor is facing the specified position is judged by the angle between the vector from the sensor position to the specified position and the optical axis of the sensor being less than the threshold value. For example, as shown in Fig. 6, "Resource" by camera C is searched for. Since the condition 1 is satisfied for "Resource" by camera A and camera C. When the threshold th is $\theta < th < \phi$, the condition 2 is satisfied only in "Resource" by camera C.

We prepared five tables in the RDB to realize these search methods (Fig. 7). Table 1 shows an example of APIs that researches and loads "Resource" and "Object" from this RDB. We adopted C++ as the APIs, since the application developers targeted by our framework are assumed that they are familiar with programming languages.

TABLE I
SEARCH APIS

API name	Processing content
std::vector<Object>loadObject(Vector4 origin, Vector4 range, int mode)	Load fields in Object table by searching for "range" and "distance"
std::vector<ResourceMeta>loadResourceMeta (Vector4 origin, Vector4 range, int mode)	Load fields in Geometry and Resource table by searching for "range" and "distance"
std::vector<ResourceMeta>loadResourceMeta (Vector4 origin, Vector4 range, Vector3 target, double theta, int mode)	Load fields in Geometry and Resource table by searching using the specified position

IV. PROTOTYPE AND EVALUATION

A. *Evaluation of the Resource database*

We evaluate whether the required images can be loaded from the resource database; thus we developed a 3D reconstruction application and took pictures with two types of cameras while orbiting the clock tower at the square in front of Kamakura Station. Although the angle of view and focal length of each camera are difficult, both resolutions are FullHD. This evaluation is a functional evaluation using images taken by scattered sensors, not a functional evaluation of 3D reconstruction itself. Therefore, the accuracy of 3D reconstruction is not a target for confirmation.

We registered the 104 capture images in the resource database and searched "Resource" within specified area. The application downloaded the images and metadata and reconstructed clock tower model from http server. In this application can be more robustly estimated camera motion, since this application can arrange images in time series from "Object" data. The 3D model of the clock tower created using "Object" is shown on the left of Fig. 8.

Next, the 3D reconstruction method is the same, but the "Resource" search method is changed to the method using the specified position. It was specified as two points on the clock tower and downloaded from the HTTP server. The results of 3D reconstruction using the downloaded results are shown in the right side of Fig. 8. It can be seen that the 3D model in the red frame such as the map around is gone.

It was confirmed that the developed APIs work effectively. The angle threshold specified in this evaluation was $\pi/3$, which was close to the angle of view of the camera. The application downloaded 37 images under these conditions. The reason we specified two points in this evaluation is that the entire clock tower could not be reconstructed with one point because there was only images of a close subject.

B. *Evaluation of the Application Development*

We evaluate whether the our framework design is appropriate and practical by evaluating whether application development has become easier. The advantages of the application framework are as follows.

- Increased development speed
- Reusable features implemented
- The code writing method can be unified

In order to evaluate these, it is necessary to evaluate the development period, APIs reuse rate, and code readability,

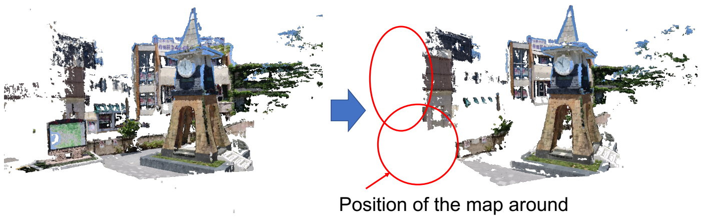

Fig. 8. Result of 3D reconstruction

but it is difficult to evaluate them accurately. Therefore, the number of steps of the two implemented applications is compared so that it can be evaluated easily and objectively. In this evaluation, we compare the number of steps between an application that uses our proposed framework and an application that is implemented without using it, assuming that it can be evaluated easily and objectively. If the number of steps decreases in this evaluation, it can be said that the design direction of this our framework is valid and practical.

The application is implemented using our framework. Therefore, we focused on DR technology as an advanced use. Recently, many practical examples using DR have been published. We chose the application that visualizes the blind area caused by the forward car described in Ref. [9] as a target for evaluation (Fig. 9). We compared the number of implemented steps between the application that uses our framework (App. A) and the application that is implemented without using our framework (App. B). However, App. B is used OSS such as OpenCV. The performance of both applications is almost the same. Both apps implemented on ROS (Robot Operating System). Therefore, App. A counts only the number of steps implemented by the application developer. App. A was implemented on the premise that it received a "Resource" from the resource database to make the performance the same as App. B.

The number of steps was 78 lines on App. A, and 425 lines on App. B, reducing the number of steps. Therefore, the validity and practicality of the design direction of the framework were demonstrated.

C. *Other Application Examples*

Various other applications can be easily implemented. Fig. 10 shows one example. In this example, a panoramic image is generated from three multi-view images.

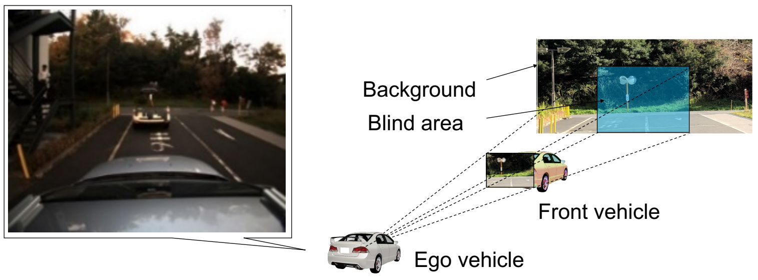

Fig. 9. Visualization to eliminate blind area

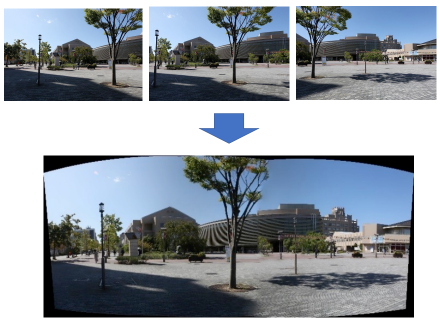

Fig. 10. Panoramic image

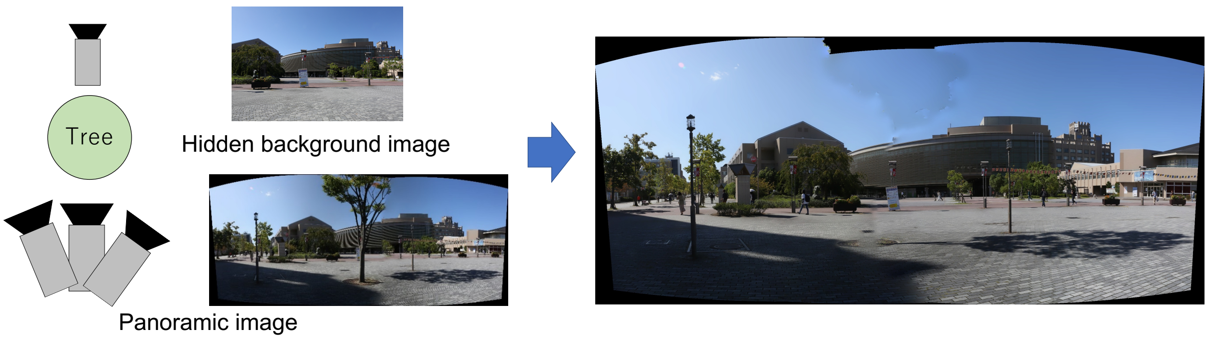

Fig. 11. Image with DR applied to panoramic image

Originally, in order to generate a panoramic image, it is necessary to estimate the pose of each camera. Pose estimation mechanism is necessary for the 3D reconstruction application too. In other words, there are many similar processes.

Therefore, by organizing the processing for advanced use with many applications, the value of our framework will be increased as it will be able to provide APIs with high reuse rates. For example, it is easy for the application developer to generate a panoramic image applied to DR by using the provided APIs. In other words, the range of advanced use of images can be expanded by combining various technologies (Fig. 11).

V. Conclusions

This paper describes a system construction for advanced utilization of spatiotemporal data, and proposed the basic design of the application framework for the application developer.

We prototyped an application using the prototyped framework, and compared the number of steps with the existing one. As a result, it was judged that the development cost of the application developer could be reduced due to the reduction in the number of steps, and the design direction of the proposed framework was confirmed to be appropriate and practical.

In the future, we plan to expand the functions of the proposed framework based on the knowledge obtained in this evaluation, and further improve the applications using the framework.

Acknowledgment

This work was supported in part by a Grant-in-Aid for Scientific Research (B), Grant Number 17H01747.

References

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, “Building Rome in a day,” *Communications of the ACM*, vol. 54, no. 10, pp. 105–112, 2011.

[2] L. Juhász and H. H. Hochmair, “Cross-linkage between Mapillary street level photos and OSM edits,” in *Geospatial Data in a Changing World*. Springer, 2016, pp. 141–156.

[3] K. Suzuki, S. Wakisaka, and N. Fujii, “Substitutional reality system: a novel experimental platform for experiencing alternative reality,” *Scientific reports*, vol. 2, p. 459, 2012.

[4] I. Ori, “Mirrorworld v. AR Cloud or: How I Learned to Stop Worrying and Love the Spatial Future,” accessed 2020-08-24. [Online]. Available: https://medium.com/super-ventures-blog/mirrorworld-v-ar-cloud-or-how-i-learned-to-stop-worrying-and-love-the-spatial-future-59de8fe8538f

[5] K. Sakurada, T. Okatani, and K. Deguchi, “Detecting changes in 3d structure of a scene from multi-view images captured by a vehicle-mounted camera,” in *Proceedings of 2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 137–144.

[6] T. Ogino, Y. Matsuda, Le Van Nghia, D. Kawabata, K. Yamazaki, A. Kimura, and F. Shibata, “A distributed framework for creating mobile mixed reality systems,” in *Proceedings of 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*, 2014, pp. 327–331.

[7] S. Mori, Y. Eguchi, S. Ikeda, F. Shibata, A. Kimura, and H. Tamura, “Design and construction of data acquisition facilities for diminished reality research,” *ITE Transactions on Media Technology and Applications*, vol. 4, no. 3, pp. 259–268, 2016.

[8] M. Brown and D. G. Lowe, “Automatic panoramic image stitching using invariant features,” *International journal of computer vision*, vol. 74, no. 1, pp. 59–73, 2007.

[9] S. Ikeda, I. Takemura, A. Kimura, and F. Shibata, “Diminished reality system based on open-source software for self-driving mobility,” in *Proceedings of 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 2018, pp. 354–357.