

A Micro-Climate Viewer with Combination of Heterogeneous Sensor Sources

Keisuke Kanai

Tokyo Denki University
ksk@u-netlab.jp

Atsuhiko Takagi

Tokyo Denki University
atsu@u-netlab.jp

Shin'ichi Konomi

Tokyo Denki University
JST CREST
konomi@osoite.jp

Hiroki Ishizuka

The University of Tokyo
JST CREST
isi@mcl.iis.u-tokyo.ac.jp

Masayuki Iwai

The University of Tokyo
masa@iis.u-tokyo.ac.jp

Kaoru Sezaki

The University of Tokyo
JST CREST
sezaki@iis.u-tokyo.ac.jp

Yasuyuki Ishida

Tokyo Denki University
yasu@u-netlab.jp

Niwat Thepvilojanapong

Tokyo Denki University
JST CREST
wat@osoite.jp

Yoshito Tobe

Tokyo Denki University
JST CREST
Yoshito_tobe@osoite.jp

ABSTRACT

Recently city residents have been faced with climate disasters including heavy rain concentrated to a local area. To avoid such disasters, the residents as well as a local government need to identify correct micro climate. Conventional climate stations only observe coarse-grained climate. However, the advent of wireless sensor networks has enabled measuring the micro climate. In addition, even a person can announce the status quo by uploading a captured photo to show the current climate. Based on the background, we propose a viewing system with combined heterogeneous sensor sources, Micro Climate Viewer (MCV). MCV provides photo-image-based visualization and introduces the definition of Spatio-Temporal Correctness (STC) for a result of a user's query. In this paper, we describe the system architecture of MCV and how STC values are calculated.

Keywords

Query Processing, Visualization Techniques, Sensor Data Managements.

INTRODUCTION

Understanding climate in our living environment has been an important activity since early times. In urban life, we struggle with climatic disasters [1] such as a concentrated heavy rain and a heat island phenomenon. In order to avoid and predict the disasters, a city government needs to figure out correct micro climate information and provide useful climate information for city residents. In recent years, energy consumption of a city has been increasing, because of the growth of urban activities due to the concentration of a population and limitless desires for economic performance. Furthermore, high-rise and dense office buildings complicate environmental conditions in the city. We cannot understand correct urban climate by conventional observation methods because of such complex conditions.

Recent technological advances in electronics, wireless communications and networking have accelerated the development of sensor networks technology. Sensor networks enable us to observe detailed urban climate easily. We developed a sensor network called UScan [2] by which temperature in fine resolution was measured in downtown Tokyo. The work in [3] describes a system called "Parasitic Ambient Logger" which is an attachable to a mobile device in order to sense ambient air environment. On the web, there are much sensed data which have been uploaded from meteorological agencies or companies related to climate. Moreover, we can also access the number of tagged photos of a city through Flickr or Picasa services. Therefore we can utilize heterogeneous sensor sources to observe a city climate and city view. In this paper, we propose a Micro Climate Viewer (MCV) with combination of heterogeneous sensor sources for a city government or city residents. An MCV regards a heterogeneous sensor source as a database. The databases create an overlay network among each other. A user sends a visualization query, including the user's specified view, time and sensor types, to an anonymous node in the overlay network. Then, the MCV returns an extended photo image which matches the user's specified view and time from the overlay network. Sensed data which match user's specified view, time and sensor type are also included on the photo images as a text. One of the features in MCVs is a visualization method based on a photo image which is a user's specified view. A user using our visualization method can instinctively recognize a micro climate around the user rather than map based methods.

When there are no sensed data and photo image which match the user's specified view and time, an MCV provides a user the nearest photo image and sensed data as a result. However, the returned result has spatio-temporal difference between practical user's specified view and time in that case. The spatio-temporal difference strongly affects the accuracy of visualization in MCVs. Therefore, we provide the spatio-temporal difference as Spatio-Temporal Correctness (STC)

to a result image as text. A STC consists of difference of photo images and difference of sensed data.

In this paper, Section 2 describes a system architecture and detailed design of an MCV. Section 3 shows a definition and a calculation algorithm of Spatio-Temporal Correctness. Section 4 explains a prototype implementation and shows simple experiments and its results. Finally, section 5 and 6 describe some related works and conclusion of this work.

SYSTEM ARCHITECTURE

An MCV system consists of MCV clients and MCV servers. An MCV client just creates and sends a user's query and visualizes results from an MCV server. MCV servers work as heterogeneous sensor sources and process the query among MCV servers. Figure 1 shows architecture of MCV system. There are two types of sensed data that MCV servers have archived. The first type called environmental sensor type, including temperature data has been observed just at a point. The second type, called multimedia sensor type, like a photo data has been observed according to a sensing range and angle. In this work, we define a sensing range and angle in Figure 2. Let $S(n)$ and (a_n, b_n) represent the sensing range of a sensor n as the multimedia sensor type and the center of node n , respectively. In addition, let r_n , θ_n , α_n be the sensing radius, the angular offset, and the sensing angle width for $S(n)$. Table 1 shows two type of sensed data table schema.

We assume that MCV servers have connectivity to the internet and create an overlay network among each other. Each server manages its own property including the unique MCV_{id} , sensor type MCV_{Type} that it manages, a covered arean $MCV_{area}\{(x_{min}, y_{min}), (x_{max}, y_{max})\}$ and a covered time-span $MCV_{time}\{T_s, T_e\}$. MCV servers broadcast periodically the property to whole server in the overlay network. Then, an MCV server manages current properties of all MCV servers as QF_{table} . The following sections explain a query and visualization processing method in detail.

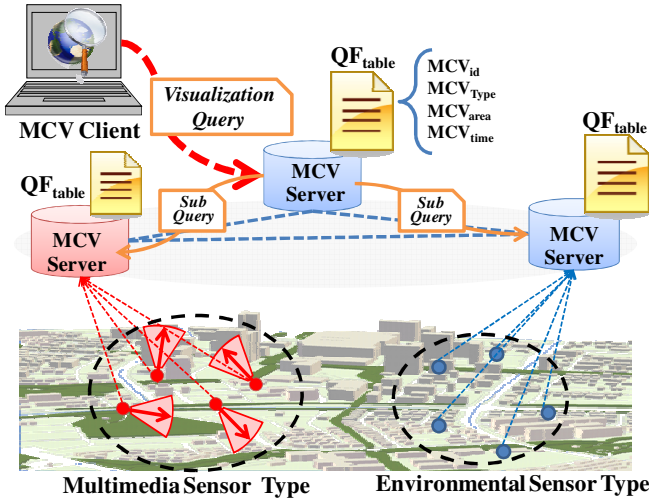


Fig 1. The system architecture of the MCV

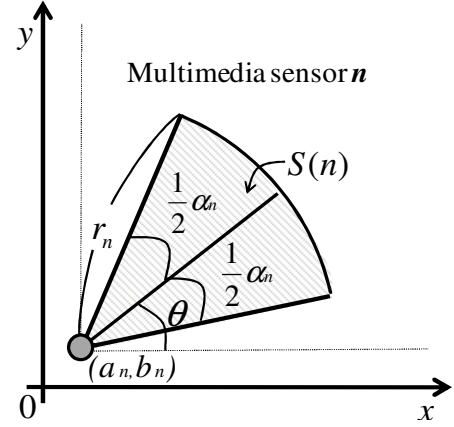


Fig 2. Definition of a sensing area and angle

Query Processing

An MCV client sends a query to an MCV server when the client wants to know temperature information at a view and a time. There are seven parameters in a query created by an MCV client. Let five parameters $(a_n, b_n, r_n, \theta_n, \alpha_n)$ in Figure 2, MCV_{Type} , and MCV_{time} represent a user's specified view, sensor type, and time-span, respectively. The MCV_{svr_origin} (an MCV server that received a query) divides the query into a IMG_{ret} operation (A retrieving operation of a result image) and a $DATA_{ret}$ operation (A retrieving operation of sensed data of specified sensor type). In the IMG_{ret} operation, MCV_{svr_origin} searches all MCV servers managing image data that match the view and time of the query from own QF_{table} and forwards the query to the MCV servers. In the $DATA_{ret}$ operation, the MCV_{svr_origin} searches against all MCV servers managing MCV_{type} in the query as well as the IMG_{ret} operation and forwards the query to matched servers. Forwarded queries are stored in the query stack of the MCV_{svr_origin} , until results of queries return. The MCV_{svr_db} (an MCV server that received a forwarded query) retrieves data matched the query from its own database. If the matched data obtained from the MCV_{svr_db} , a result returns to the MCV_{svr_origin} . When results of two operations return, the MCV_{svr_origin} adds a Spatio-Temporal Correctness (STC) which means the error between a practical user's query and the matched data to the returned results and removes query entries of the operations from the query stack. Finally, the MCV_{svr_origin} transmits the results to the visualization processing. Additionally, all MCV servers can process a user's query since they support same functions.

Table 1. Table schemas of two sensed data types

Table Schema of Environmental Sensor Type

Type	Value	x	y	Time
Text	Double precision	Double precision	Double precision	Timestamp

Table Schema of Multimedia Sensor Type

Type	Value	x	y	r	θ	α	Time
Text	Double precision	Double precision	Double precision	Double precision	Real	Real	Timestamp

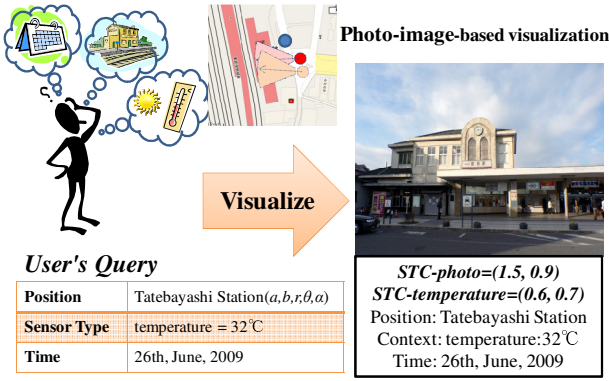


Fig 3. A sample query and its visualization result.

Visualization

Conventional visualization methods of sensed data have plots the data on a map. In contrast, our approach visualizes sensed data on a photo image which is a user's specified view. A user using our method can instinctively recognize a micro climate with the user's surrounding environment rather than map based approaches. In our approach, photo images for visualizing results are also sensed data which managed in a part of MCV servers. Such MCV servers have collected city photos from photo archives like Flickr on the web. Therefore, photo images for visualizing results will be changed according to user's specified sensor type and time-span. For example, let us assume that a user wants to know a photo view in front of a train station when the temperature at the location is 32 degrees during 26th of June, 2009. In a map based approach, some colored sensed data just plotted on the map around the train station. On the other hand, sensed photo based approach plots temperature data on the photo but also changes the visualizing photo to the nearest photo that shot in front of the train station when the temperature is 32 degrees during the day. Accordingly, the user is enabled to feel the circumstance in front of the train station at the condition of user's query. To avoid climatic disasters, a city government has to understand atmospheres of city residents with a particular climate state at a landmark in the city. Therefore, our approach is a valid solution as micro climate viewer for a city government. In Figure 3, we express one of the scenarios and an example of visualization results.

SPATIO-TEMPORAL CORRECTNESS

Sometimes a result from MCV servers is not completely consistent with the conditions specified by a user. There are a little spatio-temporal error between the result and conditions. The spatio-temporal error highly affects an accuracy of visualization in the MCV system. Therefore, we provide Spatio-Temporal Correctness (STC), instead of the spatio-temporal error, as an additional information to each result image. The text shown in Figure 3 is an example of the STC value. The STC consists of spatial and temporal correctnesses. Equation 1 expresses the definition of STC.

$$STC = (Cor_s, Cor_t) \quad (1)$$

Sensed data that MCV servers have archived are classified into two types: environmental and multimedia data. In this section, we present algorithms of spatial and temporal correctnesses for two types of sensed data.

Spatio Correctness

Spatial correctness indicates spatial error of returned sensor data in comparison with the view specified by a user. We explain algorithms of spatial correctness for both types of sensor data as follows.

Environmental sensor type

Cor_s is estimated from dist, the distance between the location of sensed data and the gravity point of the view specified by a user. In preprocessing process, we derive a reference function of spatial error ($f_s(dist)$) from the variations of sensed values against variations of distances among all sensed data at a given time. Figure 4 shows the deriving algorithm of $f_s(dist)$. However, the reference function cannot share among MCV servers that manages the same MCV_{type} if the spatial densities of managed sensors in MCV servers are difference.

Hence, we calculate a spatial density by dividing the sensors managed by a MCV server into a minimum bounding rectangle of the MCV server. If a view in a query intersects MCV_{area} s of some MCV servers, $f_s(dist)$ that is derived from the MCV server that has the highest spatial density is applied.

Multimedia sensor type

Since multimedia data consists of sensing area and angle, we estimate Cor_s from two components: (i) *Dir*, an angle difference between an angle of a result photo and the angle of the view specified by a user; (ii) *Over*, an area difference between a sensing area of a result photo and the sensing area of the view specified by a user. The Cor_s of the multimedia sensor data is defined in Equation 2.

$$Cor_s = Dir \cdot Over \quad (2)$$

Algorithm 1 : The Deriving Algorithm of $f_s(dist)$

```

1. sensor_data_list x;
2. position_difference_array hp;
3. value_difference_array v;
4.
5. x=<Select data from all-sensor where time=one-time>
6.
7. for(i=0; i<n; i++){
8.     for(j=i+1; j<n; j++){
9.         hp[i][j]=sqrt((x[j].longitude-x[i].longitude)^2
10.                    +(x[j].latitude-x[i].latitude)^2);
11.         v[i][j]=absolute(x[j].value-x[i].value);
12.     }
13. }
14.
15. fs=least-squares-method(h,v);

```

Fig 4. The Deriving Algorithm of $f_s(dist)$

Algorithm 2 : The Deriving Algorithm of $f_t(time\ lag)$

```

1. sensor_data_list x;
2. time_difference_array ht;
3. value_difference_array v;
4.
5. x=<Select data from one-sensor where time=all-time>
6.
7. for(i=0; i<n; i++){
8.     for(j=i+1; j<n; j++){
9.         ht[i][j]=absolute(x[j].time-x[i].time);
10.        v[i][j]=absolute(x[j].value-x[i].value);
11.     }
12. }
13.
14. ft=least-squares-method(h,v);

```

Fig 5. The Deriving Algorithm of $f_t(time\ lag)$

Temporal Correctness

Cor_t for both sensor types represents an error of a sensed value according to temporal difference between the time that returned sensed data were observed and the time specified by a user. To estimate Cor_t , we derive a reference function for temporal error ($f_t(timelag)$), where $timelag$ is the time difference based on the variations of sensed values against the variations of all sensed data at a given location. Figure 5 shows the algorithm for deriving $f_t(timelag)$. The Cor_t is the ratio of the time difference between the time that a returned sensed data were observed and the time specified by a user to $timelag$ in $f_t(timelag)$. If the time in a query intersects MCV_{time} s of some MCV servers, $f_t(timelag)$ is selected according to the election rule of $f_s(dist)$.

EXPERIMENTS

We implemented a prototype of the MCV system and conducted an experiment to evaluate the accuracy of STC. There are three MCV servers in our experiment. The first MCV server manages the photos of Tatebayashi city which locates in the southeastern part of Gunma Prefecture. The second stores temperature data sensed by TScan project on July 26, 2009 in Tatebayashi city. TScan project installs 43 temperature sensors in real world, and acquires fine-grained temperature data. The third stores temperature data open to the public on web site. This data is roughly measured by 13 observing stations on July 26, 2009 in Gunma Prefecture. The observing stations are deployed by Japan Meteorological Agency. In this experiment, we inject a query assumed in chapter of Visualization and evaluate STCs calculated by using each data of TScan and web. Table 2 shows query and STC results.

Results

$f_s(dist)$ and $f_t(timelag)$ in Figure 6 were derived by two MCV servers that managed temperature data. The STC results were obtained by assigning the spatio-temporal differences between each returned sensed data and the query. The STCs of each source is shown in Table 2. Since the result, we can see the returned TScan data is spatially more accurate but temporally less accurate than the returned web data in this condition. Although the STC is valid as one of error metrics for the results of queries against heterogeneous sensor sources, the accuracy of STC was not sufficient because the algorithms $f_s(dist)$ and $f_t(timelag)$ were too simple.

RELATED WORKS

SenseCam [4] is the project which can extend memory of humans using camera data and many type of sensors. Their approach can combine image data and sensor data, however SenseCam do not have facilities to share image data between many users. On the other hand MCV have a query schema to search the sensors or sensor data.

SensEye [5] is a multi-tier camera sensor network over a single-tier network. SensEye can localize object position in three dimensions using two cameras. As the point of using cameras, this research is similar to our approach. However, they did not consider the relationship between other types of sensors, like temperature, acceleration sensors.

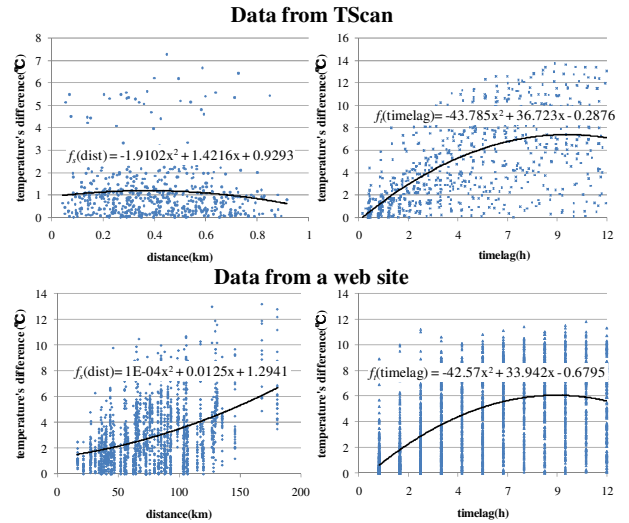


Fig 6. $f_s(dist)$ and $f_t(timelag)$ from web data and Tscan

Table 2. Query and its results in experiments.

Experiment Query	a (longitude)		b (latitude)	r (km)
		139.528671205043	36.2458350080381	
Experiment Query	θ (°)	α (°)	type	time
	180	60	temperature=32	2009/07/26 12:30:00
TScan Data	Cor_s (°)		Cor_t (°)	
STC	0.97415917047633		1.8311256986095	
Web Data	Cor_s (°)		Cor_t (°)	
STC	1.3276823983868		0.0091484375780182	

CONCLUSION

In order to avoid climatic disasters, a city government needs to identify correct micro climate. Therefore, we propose an MCV with combination of heterogeneous sensor sources. Features in MCVs are the visualization method based on a photo image and the definition of STC. In this paper, we described the system architecture of an MCV and show calculation results of STC.

REFERENCES

- Hunt, A. and Watkiss, P. Literature Review on Climate Change Impacts on Urban City Centres: Initial Findings. Environment Directorate. OECD. December 2007.
- Ono, T., Ishizuka, H., Ito, I., Ishida, Y., Miyazaki, S., Mihirogi, O. and Tobe, Y. UScan: Towards Fine-Grained Urban Sensing. International Workshop on Real Field Identification(RFId2007), 2007.
- Miyaki, M. and Rekimoto, J. Sensoromy: Envisioning Folksonomic Urban Sensing. Ubicomp 2008 Workshop Programs, 2008.
- Hodges, S., Williams, L., Berry, E., Izadi, S., Srinivasan, J., Butler, A., Smyth, G., Kapur, N. and Wood, K. SenseCam: a retrospective memory aid. Ubicomp 2006, 2006.
- Kulkarni, P., Ganesan, D., Shenoy, P. and Lu, Q. SensEye: a multi-tier camera sensor network. Proceedings of the 13th annual ACM international conference on Multimedia table of contents, 2005.