2013 IEEE 37th Annual Computer Software and Applications Conference Workshops

Elevator acceleration sensing: Design and estimation recognition algorithm using crowdsourcing

Tianhui Yang, Katsuhiko Kaji and Nobuo Kawaguchi
Graduate School of Engineering
Nagoya University
Furo-cho, Chikusa-ku, Nagoya, Aichi, Japan

***Abstract*—In human activity sensing areas, we focus on elevator recognition using accelerometer. In this paper, our main challenge is to create an elevator acceleration recognition algorithm. In order to put this algorithm into real world use, This algorithm should be robust enough that can ignore the differences of the elevators, devices and subjects. To realize this aim, we build a crowdsourcing system which consists of an android application and a server. Android users can use this application to calculate elevator's moving distance. They can also upload elevators' accelerometer signals to the server, so we can get elevator acceleration data from all over the world. after publishing the system, we use the data we collected to build an elevator acceleration database. By using this database, we build a robust elevator recognition algorithm. we also use the features of elevators to distinguish them.**

***Index Terms*—accelerometer, smartphone, activity recognition, elevator, crowdsourcing.**

I. Introduction

Recently, with the development of MEMS technology, the smartphone which has many kinds of sensors are more and more popular. There are many kinds of sensors, such as microphone, camera, gyro, accelerometer and etc. we can get the sensor signals by using these sensors. The users always carry these sensors in their daily life, if we record signals from the sensors, we can get user's many information from the sensors. By using these information, we can build life management service, life log, navigation system, or context aware services.

Activity recognition is a main service within context aware services. Recently many researchers focus on the research of human activity recognition using Mobile device sensors[1][2][3][4].

Many researches [5][6] on elevator acceleration recognition are based on subjects less than 20 people. The subjects, elevators, and devices are specified by the researchers. As a result, they got a very high recognition rate. But in the real world, there are many kinds of devices, subjects and elevators. The algorithm based on specific devices, elevators, and subjects are not robust for real world use.

Our aim is to build a robust elevator recognition algorithm which is suitable for any elevator, device, subject in the world. One application of this algorithm is to embed it into location servive. The location service we use now only can get 2D location. By using this algorithm, we can recognize elevator movement and estimate the users' location in 3D.

In order to realize this aim, we need to collect many elevators' acceleration data using many kinds of devices by many people from all over the world. So we develop a crowdsourcing system to collect the data. This system consists of an android application and a server. Users who have an android cellphone can download this application from Google Play. This application can record elevator acceleration data and tell user the elevator moving distance. User can also upload the acceleration data to the server. By using this crowdsourcing system, we can collect elevator data from all over the world.

In the followings, in section 2, we introduce the related works based on mobile sensors. In section 3, we propose our recognition algorithm. In section 4, we discuss the design of the crowdsourcing system. In section 5,we will introduce the results of data collection by crowdsourcing. In section 6, we will test our algorithm to calcute recognition rate. Section 7 concludes the paper.

II. Related Work

The research on elevator recognition using mobile device is already researched by several researchers. In the following, we show 2 of them.

Watanabe [5] used the pressure sensor and accelerometer to estimate walk, stand, stairs up, stairs down and take elevators. The recognition method was to use the accelerometer to count walk step, record activity status changes, he used pressure sensor to estimate vertical movements. The sensor was put in the handbag. As a result, he got an average recall ratio of 95% and a precision ratio of 96%.

Koshimizu [6] used a single accelerometer, put the sensor device on to the subject's right trousers' pocket. In order to minimize processing time. He didn't use machine learning, he calculated variance and average of the acceleration data from each 3.2 seconds time window, and he used a threshold to recognize elevator's acceleration and deceleration. As a result, the recall ratio when elevator goes up was 96.7%, the recall ratio when elevator goes down was 100%.

III. Elevator Recognition Algorithm

A. Feature of the Elevator acceleration

In everyday life, most of the human activities are walking and staying. Acceleration data such as walk are short time repetitive movements. The acceleration data is vibrating periodically across the X-axis. When users take elevators,

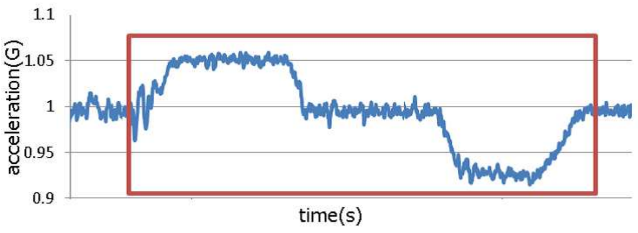


Fig. 1. The elevator acceleration data

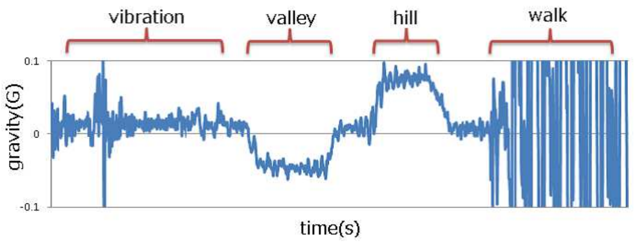


Fig. 2. A sequent acceleration data

the elevator acceleration data always has acceleration and deceleration that is longer than other human activities (Figure 1). In this section, we will use this feature to distinguish the elevator acceleration from human activity's acceleration.

B. *Delete direct current component*

The sensing data from accelerometer always has direct current component knows as the gravity. Gravity shifts due to time, place, and devices. So if we want to estimate elevator moving distance precisely, we need to delete gravity from acceleration data.

When a smartphone user is waiting for the elevator, the user is always standing still. If we record the acceleration data when user is standing still, we can get the gravity at that time. In order to get gravity, firstly, we calculate the variance of the accelerometer data and put a threshold on it. Secondly, we put a 1 second time window on the data, if the variance data within 1 second is lower than the threshold, we use this 1 second data's average as the gravity value. So we can get several gravity values from several time windows. Sometimes high speed elevators have a long-time acceleration which is longer than 1 second, so the elevator acceleration data maybe recognized as gravity value. In order to solve this problem, we calculate the variance of all gravity values and use the value which has the smallest variance as the gravity. Then we use this gravity value and cut the gravity from original data. We will get the elevator acceleration data without direct current component.

C. *Recognize Elevator acceleration*

The elevator's acceleration and decelerationfs time duration is much longer than other everyday activities such as walk, run, stairs up and down. From figure 3 we can see that the elevator acceleration data consists a ”hill” and a ”valley”. If

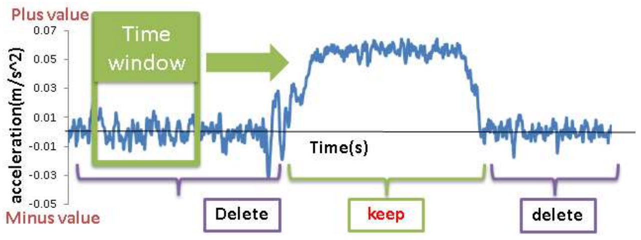


Fig. 3. Delete other activites

the ghillh is in front of the gvalleyh, the elevator goes up; if the gvalleyh is in front of the ghillh, the elevator goes down. So the feature of elevator acceleration data consists a continues ”plus” value and a continues ”minus” value. After investigated 80 different elevators, we find than the shortest elevator acceleration duration is longer than 1.2 seconds, so by using this feature, if we find vibrations across X-axis within 1.2 seconds, we consider the data is not elevator acceleration.

In order to recognize elevator acceleration automatically, firstly, we put a 1.2 seconds time window on the data. The time shift is 0.01 second. If within a 1 second time window there are ”minus” and ”plus” values, the acceleration data within this window is vibrating, so this window isn't elevator acceleration data, the data in this window will be deleted and set to 0. If in this window, all the acceleration data are ”plus” or ”minus” values, the accelerometer's data within this window means accelerating or decelerating, this time window's data is the elevator acceleration data. The data in this window will be reserved. By running this algorithm through all the acceleration data, at the end, only elevator acceleration data will be reserved.

D. *Distance Estimation*

We can integral the elevator acceleration data twice, so we can get the elevator moving distance.

IV. Build crowdsourcing system

The elevator recognition algorithm using accelerometer is researched by several researchers already. But most of the algorithms are based on several elevators specified by the researchers. So the recognition algorithms cannot be used for most of the elevators in the world. As a consequence, in order to recognize as many elevators as possible, we need to collect many elevator acceleration data and build an elevator acceleration database.

In order to collect data, we can ask volunteers and reaserch groups for help. But consequently, the subjects, devices and elevators are limited in numbers, less of diversity. In order to collect the data from all over the world. We build a crowdsourcing system.

Crowdsourcing is derivated from outsourcing. Outsourcing means contracting out of an internal business process to a third party organization. The crowdsourcing is also contracting out of an internal bussiness process, but it doesn't contract to third party organization, it uses the internet to find workers

Fig. 4. Crowdsourcing(Left) and outsourcing(Right)

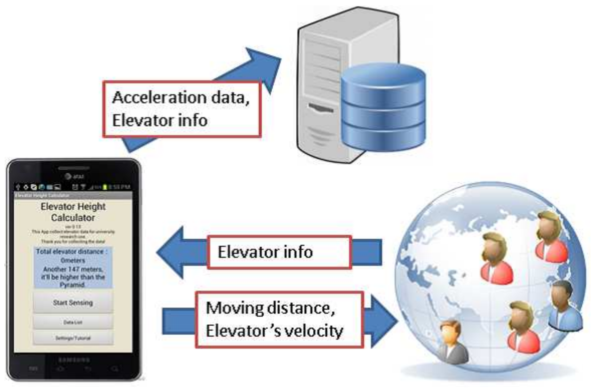
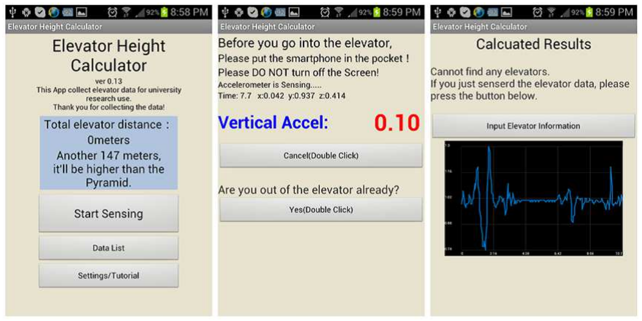


Fig. 5. Elevator.Locky service model


Elevator Height Calculator

Elevator Height Calculator
ver 0.13
This App collect elevator data for university research use.
Thank you for collecting the data!

Total elevator distance :
0meters
Another 147 meters, it'll be higher than the Pyramid.

Elevator Height Calculator

Before you go into the elevator,
Please put the smartphone in the pocket !
Please DO NOT turn off the Screen!
Accelerometer is Sensing.....
Time: 7.7 x:0.042 y:0.937 z:0.414

Vertical Accel: 0.10

Are you out of the elevator already?

Elevator Height Calculator

Calcuated Results

Cannot find any elevators.
If you just senserd the elevator data, please press the button below.

Fig. 6. The screenshoots of Elevator.Locky

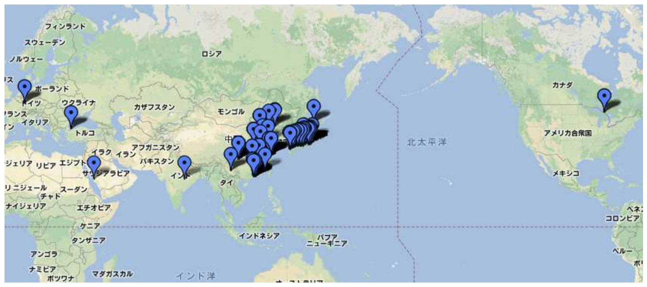


Fig. 7. Elevator's location

and organizations from all over the world. These people work together to finish the same project. By using the crowdsourcing, we can get many acceleration data from different devices, different subjects and different elevators.

A. System composition

This crowdsourcing system consists a client application and a server. The client application is android application named Elevator.Locky. This application uses smartphone's accelerometer to calculate user's elevator moving distance. For example, if a user is in a very high building, he maybe wants to know the height where he stands or the speed of the elevator he just took. So he can use this application to get the moving distance and the velocity of the elevator. After calculation, user can upload the acceleration data and elevator information to the server.

B. How to use this system

In this section, we will introduce how to use this system.

Firstly, before the user goes into the elevator, he starts the Elevator.Locky application. When the elevator comes, he presses Start-Sensing button, put the smartphone into a pocket, and goes into the elevator.

When elevator is moving, the application will record acceleration data from both the user and the elevator.

When the user arrives at the aim floor, he gets out of the elevator, takes the smartphone out of the pocket and press Finish-Sensing button. Then the application will automatically recognize elevator acceleration.

If elevator acceleration exists, the application will show the elevator moving distance and maximum velocity on the screen. So if the user starts from the first floor of the building, this application can tell user the height where he stands. If the user presses Input Elevator Information button, a simple survey will be shown on the screen. After finishing the survey, user can press upload button to upload the data to server. If the internet connection isn't avaliable, the data will be stored in the smartphone. The user can upload the data again when internet connection is avaliable.

V. Data collection results

Before Feb 2th, 2013, by running the crowdsourcing system for 53 days, the application has been installed at least for 230 android smartphones and we get 1056 files of acceleration data. Within these files, there are 660 files which have elevator accelerations. Within these 660 elevator acceleration files, we got at least 351 different elevators' acceleration data. We build an elevator database by using these 660 files of elevator acceleration data.

The users of this application are not limited to the researchers, if a person has an android smartphone with accelerometer, he can collect elevator data. This application is published worldwide, so we can get different locations, different makers and different devices' elevator data.

From this map (Figure 7), by advertising this application in Wide Project[8] mailing list, we got many uploads around Japan. We also got a lot of data from many cities around china. This is because we used the china's biggest crowdsourcing website[9] to propaganda.

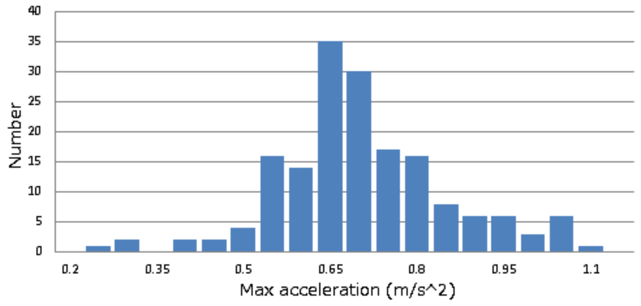


Fig. 8. The graph of max acceleration

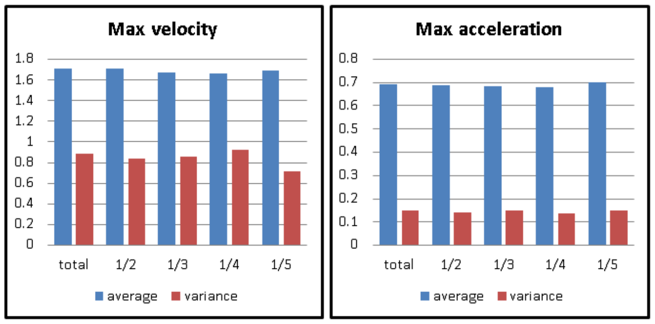
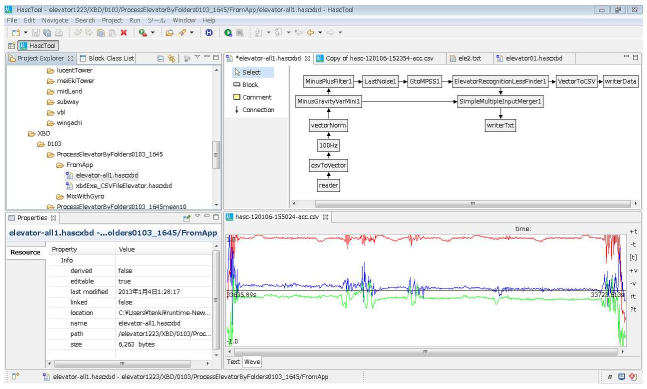


Fig. 9. The diversity of the database



HascTool - elevator1223/XBD/0103/ProcessElevatorByFolders0103_1645/FromApp/elevator-all1.hascxbd - HascTool
File Edit Navigate Search Project Run ツール Window Help
Project Explorer
Block Class List
lucentTower
meiEkiTower
midLand
subway
vbl
wingachi
XBD
0103
ProcessElevatorByFolders0103_1645
FromApp
elevator-all1.hascxbd
xbdExe_CSVFileElevator.hascxbd
MixWithGyro
ProcessElevatorByFolders0103_1645mean10
*elevator-all1.hascxbd
Copy of hasc-120106-152354-acc.csv
ele2.txt
elevator01.hascxbd
Select
Block
Comment
Connection
MinusPlusFilter1
LastNoise1
GtoMPSS1
ElevatorRecognitionLessFinder1
VectorToCSV
writerData
MinusGravityVarMini1
SimpleMultipleInputMerger1
vectorNorm
writerTxt
100Hz
csvToVector
reader
Properties
hasc-120106-155024-acc.csv
time:
1.0
33635.89s
33723.613s
-1.0
+t
-t
[t]
+v
-v
rt
?t
Text
Wave

elevator-all1.hascxbd -...olders0103_1645/FromApp

Property	Value
Info	
derived	false
editable	true
last modified	2013年1月4日1:28:17
linked	false
location	C:¥Users¥tenki¥runtime-New...
name	elevator-all1.hascxbd
path	/elevator1223/XBD/0103/Proc...
size	6,263 bytes

elevator-all1.hascxbd - elevator1223/XBD/0103/ProcessElevatorByFolders0103_1645/FromApp

Fig. 10. The screenshoot of HASC TooL

A. *Different devices*

The server not only receives elevator acceleration data, but also collects some of the smartphone information such as device brand and device maker. As a result, we get 74 different types of android devices.

B. *Smartphone sampling frequency*

All iPhones have the same sampling frequency of 100Hz, but the android devices vary a lot. The lowest average sampling frequency of the android device that we found is 36.0Hz. The highest sampling frequency is 200.5Hz, recorded by Nexsus 7.

C. *Maximum velocity*

By analyzing 351 different elevators, we calculate the elevators's maximum velocity. As a result, we find the largest velocity is 9.9m/s, the lowest velocity is 0.4m/s. Most of the elevators' max velocity ranges from 1.6m/s to 1.8m/s.

D. *Maximum acceleration*

By analyzing 351 different elevators, we find the largest elevator acceleration is 1.05 m/s^2, the smallest elevator acceleration is 0.24m/s^2.

E. *Elevator moving distance*

The longest elevator moving distance we find is 333m, it is from the elevator of Tokyo Sky Tree located in Tokyo, Japan. The shortest elevator moving distance we find is 3.5m. If we add all the elevator moving distance up, we get a total distance of 14449m.

F. *The diversity of the database*

After collecting many data from crowdsourcing, there is a question that whether these data are enough to cover most of the elevators around the world. So we randomly get 1/2, 1/3, 1/4, 1/5 of the data from database and calculate the maximum velocities and maximum accelerations, then, we plot the variances and averages of the maximum velocities and maximum accelerations.

From graph 9, both of the averages and variances of maximum velocities and maximum accelerations don't change a lot. So we can get a conclusion than the diversity of the data is enough to cover most of the elevators in the world. But there are still many special elevators we didn't know. so we will continue to collect the data by running the crowdsourcing system.

VI. Improve the Algorithm

By analyzing the elevator database, we find that our elevator recognition algorithm should be improved in 2 points below.

A. *Sampling frequency*

For all the android devices in the database, we find that the sampling frequency ranges from 36Hz to 200.5Hz. In order to solve this problem, we use linear interpolation to change all the sampling frequency to 100Hz.

B. *Elevator acceleration duration*

In section 3, we analysis 80 elevators and set the elevator recognition window size to 1.2 seconds. By analyzing the elevator database, we find the smallest elevator acceleration duration is 1.07 seconds, so we set the elevator recognition window to 1.0 second.

VII. Experiments

In this section, in order to evaluate the elevator recognition algorithm, we create 2 experiments, elevator recognition and elevator moving distance estimation. In order to process many data efficiently, we embed the algorithm into an open source signal processing tool named HASC TooL [7].

TABLE I
EXPRIMENT DATA

data	number of files	duration (m)	activites	elevator segments
collected by author	80	73	take elevators, walk	137
collceted by crowdsoucring	285	253	take elevators, walk	350
HASC2011corpus sequence data	166	349	walk, stay,jog, stairs up, etc.	0
HASC2011corpus realworld data	35	387	walk,stairs up stay, etc.	18
transportation data	37	68	take escalator, ride a bycicle, take a train, etc.	0
total	603	1130	15 kinds	505

A. Recognize the elevator acceleration

The elevator acceleration consists a ghillh and a gvalleyh, we call a set of ghillh and gvalleyh an elevator segment.

1) Experiment data: In order to test the robustness of the algorithm, the experiment data consists acceleration data files which have elevator segments and without elevator segments. The constitution of the experiment data will be shown in table I.

TABLE II
THE RESULTS OF THE ELEVATOR RECOGNITION

		real segment number	
		existence	non-existence
estimated segment number	existence	435	60
	non-existence	10	213

TABLE III
THE PRECISE AND RECALL RATIO OF SEGMENT RECOGNITION

	collected by author	collected by crowdscourcing	total
precise ratio(%)	99	95	92
recall ratio(%)	99	97	98

From table III, we get a precise ratio of 92%. By testing 1130 minutes of data, we get a recall ratio of 98%. As a result, we recognize 98% of elevators of the database. As a consequence, we confirmed the robustness of the algorithm.

B. Estimate the elevator moving distance

For many buildings, we don't know the height of each floor, so we only evaluate the buildings which have the height information. To evaluate the results, we divide the buildings into 3 categories, low-rise building(lower than 5 floor), middle-rise building(lower than 10 floor), high-rise building(higher than 11 floor). The results are shown in fig IV.

1) experiment results: As a result, for middle-rise building, the largest error is 1.41m, this is smaller than 1/2 of one floor(3.5m). In real world, this error is not big enough to influence floor estimation.

For high-rise building, the largest error is 2.41m. The reason is when the height of the building increases, accumulated error is also increasing.

TABLE IV
ELEVATOR MOVING DISTANCE ESTIMATION

	elevator segments	average error(m)	maxmium error(m)	error rate ()
high-rise	4	1.71	2.41	0.8
middle-rise	28	0.22	1.41	1.2
low-rise	11	0.21	0.77	1.9
total	43	0.36		1.4

C. Distinguish different elevators

After collecting many elevator acceleration data, we think if we utilize the features of the elevator acceleration, we maybe dishinguish different elevators. so we use 35 files elevator acceleration data from 9 different elevators as experiment data. we use the maximum velocity and maximum acceleration as features. we use J48 decision tree as learning machine. In order to evaluate the results, we use 10-cross validation. As a consequence, we get a recognition rate of 74.3%. Table V shows the confusion matrix of the 9 different elevators. Different alphabets in Table V represent different buildings' elevators.

TABLE V
CONFUSION MATRIX OF 9 ELEVATORS

	A	B left	B right	C	D	E	F left	F right	G
A	3	0	0	0	0	0	1	0	0
B left	0	4	0	0	0	0	0	0	0
B right	0	0	3	0	1	0	0	0	0
C	0	0	0	4	0	0	0	0	0
D	1	0	2	1	1	0	0	0	0
E	0	0	0	0	0	4	0	0	0
F left	0	1	0	0	0	0	1	0	1
F right	0	0	0	0	1	0	0	2	0
G	0	0	0	0	0	0	0	0	4

VIII. CONCLUSION

In this paper, in order to build the context aware system, we use the smartphone accelerometer to recognize elevator segments and estimate elevator moving distances.

In order to build a robust elevator recognition system, we build a crowdsourcing system to collect elevator acceleration data. As a result, within 53 days, we got 1056 uploads, including 660 elevator acceleration files. We find that the maximum accelerations of the elevators range from $0.24m/s^2$ to $1.05m/s^2$. By using these data, we build an elevator acceleration database.

By using this database, we improved our elevator recognition algorithm. As a consequence, we get a recall ratio of 98%, so we confirmed the robustness of the algorithm.

We also use the maximum acceleration and maximum velocity to distinguish different elevators. Consequently, we got a recognition rate of 74.3%.

In the future, we will run this crowdsourcing system continuesly to collect more elevator acceleraion data to improve the recognition algorithm.

References

[1] Ling Bao and Stephen S. Intille, Activity Recognition from User-Annotated Acceleration Data, *Proc of PERVASIVE 2004*, 2004.

[2] Stikic, M., Laerhoven, V. K. and Schiele, B., Exploring Semi-Supervised and Active Learning for Activity Recognition, *Proc of ISWC 2008*, 2008.

[3] Ravi, N., Dandekar, N., Mysore, P. and Littman, L. M., Activity Recognition from Accelerometer Data, *Proc of IAAI*, 2005.

[4] Berchtold, M., Budde, M., Gordon, D., Schmidtke, H. R. and Beigl, M., ActiServ: Activity Recognition Service for Mobile Phones, *Proc of ISWC 2010*, 2010.

[5] T. Watanabe, D. Kamisaka, S Muramatsu, A Kobayashi and H Yokoyama, Locomotion Estimation Method Using Pressure Sensor , *IEICE-MoMuC2011-30* , 2011.

[6] Kaoru Koshimizu and Eiji Kamioka Estimation of User's Behavior for Realizing Just-in-time Services, *IEICE-MoMuC2009-108*, 2009.

[7] HASC Tool, http://sourceforge.jp/projects/hasc/

[8] Wide Project, http://www.wide.ad.jp/

[9] zhubajie.com, http://www.zhubajie.com/