Situation Awareness and Conflict Resolution in Smart Home with Multiple Users

Ya-Hua Lee and Fuchun Joseph Lin
Department of Computer Science
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
{yahua.cs05g, fjlin}@g2.nctu.edu.tw

***Abstract* — There are many research efforts on situation awareness in smart home with a single user. However, a harder problem is situation awareness when multiple users are present. We investigate the problem of recognizing multiuser activities using wearable devices in a home environment. The objective is to provide situation awareness so that a smart home can respond to the needs of its residents based on their activities. An important part of the research is to detect whether there is any service conflict resulting from the ongoing activities. If so, a conflict resolution algorithm needs to be applied so that an acceptable service by multiple users can be achieved. To verify both the activity detection and the conflict resolution algorithms, two experimental systems are constructed with different architecture alternatives to evaluate the algorithms and decide the most efficient implementation design.**

Keywords—Wearable Device; Internet of Things; Decision Tree; Multiple Users; Conflict Resolution Algorithm

I. INTRODUCTION

There are many research efforts on situation awareness in smart home with a single user [2-3]. However, a home normally accommodates multiple users who perform various activities simultaneously [1, 4-7]. Consequently, the problem arises on how to provide the home service if these multiple activities have conflicting demands such as the situation where one user intends to take a nap in a quiet setting while the other desires to exercise under loud music [8-10].

Our approach makes use of both accelerometer and gyroscope on a wearable device to recognize the activities of multiple residents at home. These activities will then be analyzed for conflict detection and resolution in order to decide what home environment to provide to these users as a group. The unique advantage of our proposed method lies in its low cost as only one wearable device per user is required for raw data collection. In our experiment systems, we use BLE-equipped smartphones as the gateways for data collection. But they can be easily replaced by any other form of home gateways.

Our research assumes a living room environment at home where each user may perform a different activity such as TV watching, newspaper reading, nap taking and workout exercising. To recognize which activity is being performed by each user, the models of activities need to be constructed first by learning the features of activity data generated from the wearable device. Then, when users perform their activities, the newly generated data can be compared with the established models to identify the types of activities.

Our last paper on multiple user activities recognition [1] compared applying artificial neural network, decision tree and simple logistic regression for model construction and activity detection. Moreover, we also evaluated several different alternatives of system architecture design to discover the best one. In this paper, we address the issue of how smart home should respond to potentially conflicting demands from multiple user activities with a conflict resolution algorithm.

The rest of this paper is organized as follows: Section II surveys related work for activity learning and recognition algorithms. Section III introduces our system including both its hardware and software components. Section IV explains the conflict resolution algorithm. Section V presents our experimental results of different model constructions and compares system architectural alternatives. Finally, in Section VI we provide our conclusion and future work.

II. RELATED WORK

Lee et al. [2] did the single user activity recognition based on the wearable device. The algorithms applied include Decision Tree, Hidden Markov Model (HMM) and Viterbi algorithm. The method requires only one wearable device and a couple of location beacons for achieving the desired situation awareness. However, for the recognition of anomalous activities Bakar et al. [3] explained that the method in smart home is still immature when compared with other domains such as computer security, manufacturing defect detection, medical image processing, etc. They reviewed dense sensing approaches of smart home, and provided an extensive review from sensors, data, analysis, algorithms, prompting reminder system, to the recent development of anomaly activity detection.

In order to realize the smart home environment, the single user activity recognition is not enough; therefore, research effort has been extended to cover multiple user activity recognition at home. Chen et al. [4] explained a two-stage method for solving multiple user activity recognition. Their method defined combined label states at the model building phase with the help of data association, then at the new activity recognition phase the method could learn combined label states without the help of data association. Wanga et al. [5] investigated the problem of recognizing multiuser activities using wearable sensors in a home. Their method developed a multi-modal, wearable sensor platform to collect sensor data from multiple users, then used Coupled Hidden Markov Model (CHMM) and Factorial Conditional Random Field (FCRF) to model interacting processes in a multiuser scenario. Prossegger et al. [6] used the

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

ARAS dataset which is a real-world multiple user dataset stemming from two houses in order to evaluate their proposed algorithm. Unlike our approach, most of these efforts are limited to two or fixed number of users. Neither do they have solutions to decide the environment output whenever there is a conflict.

To identify multiple user activities, some research would use multiple sensors. To cope with the large dimension of activity data thus generated, some would use mapping methods to reduce data or employ Hidden Markov Model (HMM) and Viterbi algorithm to find the hidden states of multiple users. For example, Chiang et al. [7] proposed dynamic Bayesian networks which extended Coupled Hidden Markov Models (CHMMs) by adding some vertices to model both individual and cooperative activities. In order to improve the model performance, they categorized sensor observations based on data association and domain knowledge in order to model multiple user activity patterns.

In the area of conflict resolution, Wang et al. [8] declared that the conflict is a natural disagreement between different attitudes, beliefs, values, or needs. Shin et al. [9] proposed a Context Manager to resolve conflicts that arise when multiple users accessed various context-aware applications while the applications were trying to share resources in ubiquitous computing environments. In order to resolve conflicts among users, the proposed Context Manager maintained the conflict history of users, calculated the weight of context with Bayes theory, and then selected one having the highest priority among users. In addition, Resendes et al. [10] did a systematic review of existing literature concerning conflict detection and resolution in these systems.

III. SYSTEM COMPONENTS

We explain sensors, processing hardware and mining tools required by our system in the following.

1) *Wearable Device*

Koala is developed by National Chiao Tung University. It has two sensors: a 3-axis accelerometer and a 3-axis gyroscope. It uses Bluetooth Low Energy (BLE) as the communication protocol. We show its specification in Fig. 1.

The smart phone can establish connection with Koala and receive approximately 30 sets of acceleration and gyroscope data per second. Koala is powered by a CR2032 coin battery. Under this experiment, it is used as a waistband wearable device on the user's right waist as shown in Fig. 2.

2) *Smartphone*

The smartphone acts as a gateway because Koala cannot directly communicate with the server. In addition, we use the smartphone to identify each user's identity and priority and to detect the environment situation such as brightness and temperature. In our implementation, a Samsung Note 3 with Android 5.0 is used. The Samsung Note 3 can communicate with Koala using BLE and with the Python or oneM2M server using WiFi. The sensors on the phone include Compass, Magnetometer, Proximity sensor, Accelerometer, Ambient light sensor, Gyroscope and Barometer.

3) *Server*

Both Python and oneM2M servers reside in a Raspberry Pi3 with 1.2GHz 64-bit quad-core ARMv8 CPU and 1 GB RAM. The operating system running on Pi3 is Raspbian.

4) *Philip Hue*

To provide a comfortable environment for the users in a home, Philips Hue [11] smart light is used to control the illumination at home. Our system can send signals to a Hue bridge through RESTful (REST denotes Representational State Transfer) communications to control its color and brightness.

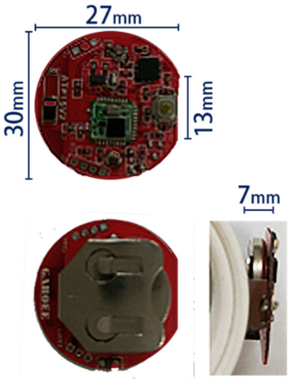
5) *Mining Tool*

Waikato Environment for Knowledge Analysis (Weka) is a popular suite of machine learning algorithms written in Java [12]. The algorithms under support include artificial neural network, decision tree and simple logistic regression. We used Weka to generate each model in our experiment.

6) *PyQt*

PyQt is the Qt tool for python language. It is a set of Python v2 and v3 bindings for The Qt Company's Qt application framework and runs on all platforms supported by Qt including Windows, OS X, Linux, iOS and Android.

The activity recognition system for multiple users we developed provides a PyQt5 user interface as shown in Fig. 3. The explanation of this interface is expressed in the annotations shown in Fig. 4. This user interface helps users understand the analysis results with a clear display. With a large variety of potential situations in the home environment, such an interface allows us to easily evaluate and modify various options of home parameters in order to meet the potentially conflicting needs of



MCU	**nRF51822 (Cortex-M0) 16MHz**
RAM	**16KB**
ROM	**256KB**
Bluetooth	**Bluetooth Low Energy (Bluetooth Smart)**
Sensor	**MPU6050** **- 3 axis Accelerometer** **- 3 axis Gyroscope**
LED	**Red (Status Indicator)**

Fig. 1. Koala hardware and specification

Fig. 2. Koala worn on the waist

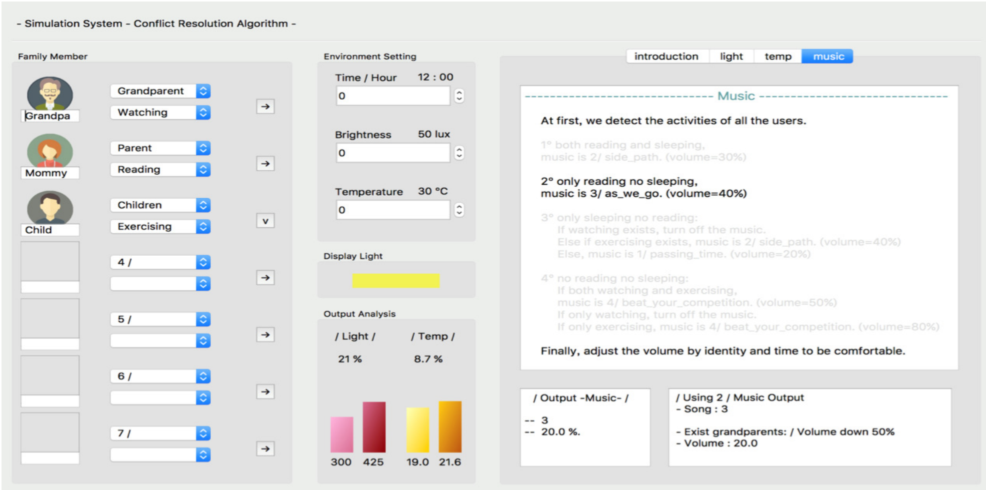


Fig. 3. Graphical user interface for conflict resolution algorithm

multiple uses.

IV. Conflict Resolution Algorithm

The Conflict Resolution Algorithm is required when the environments desired by multiple user activities are in conflict. In this section, the variables and algorithms designed for conflict resolution are introduced and explained. First, we introduce "variables" that play a key role in conflict resolution.

1) Introduction of Variables

Before we describe the conflict resolution algorithm, we need to set the default outcomes when there is only one user. These default outcome settings are defined in Table 1. For example, the default settings for lighting will be at 500 lux and 0 lux, respectively, for reading and for taking a nap and denoted as Lighting(Reading) = 500 and Lighting(Taking a nap) = 0. Here the home service can be lighting, temperature, music volume or any other.

Now when there are multiple activities by more than one family member at home, the outcomes defaulted by these activities may be in conflict. To resolve such a conflict, a variable "identity" is introduced to assign different precedence

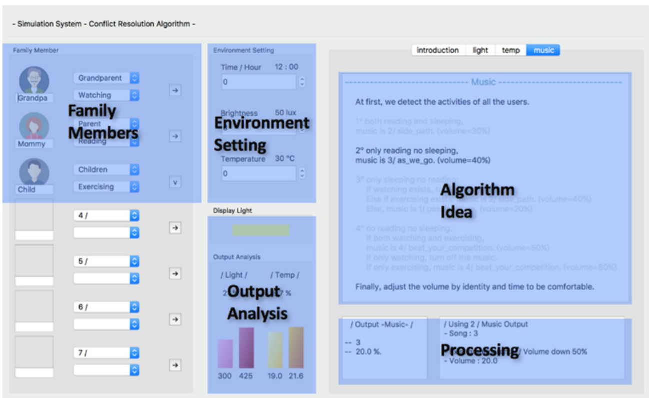


Fig. 4. Explanation of graphical user interface

to each family member. As shown in Table 2, we define such precedence as a weighted value based on a member's identity in the family – the larger the value is, at the higher precedence our resolution algorithm will consider the need of the person.

In addition, another variable "time" is also defined for conflict resolution (see Table 3). For example, it is suitable to take a nap around 1 p.m., so we would set higher precedence to the activity "Taking a nap" at 1 p.m. By adjusting the value of "time", we can easily customize the preference of an activity at a particular time.

Two more variables we defined for conflict resolution are to reflect the values of environment brightness and temperature as detected by the smart phone. In order to adjust the resolution result with the real situation in a smart home, we would modify the output of our resolution algorithm according to these two variables.

Table 1. Default outcome of single activity description

Activity	Lighting	Temp.	Music Volume
Reading	500 lux	25.0°C	40 %
Taking a Nap	0 lux	26.5°C	20 %
Watching TV	200 lux	23.5°C	0 %
Exercising	300 lux	19.0°C	80 %

Table 2. Values of Identity Variable

Identity	Weighted Value
Grandparent	8
Parent	4
Children	1

Table 3. Values of Time(Hour) Variable

Time (Hour)	Time Priority Value		
13 or 14	Taking a Nap: 3		
15 or 16	Exercising: 3	Reading: 2	
19 or 20	Watching TV: 3	Exercising: 1	Reading: 1

2) Conflict Resolution Algorithm

As discussed before, a conflict resolution algorithm is required to decide a final setting for the home environment when the environments desired by multiple user activities are in conflict.

To describe our conflict resolution algorithm, we first identify the most conflicting pair of activities for each of lighting, temperature and music volume control settings as shown in Table 4.

- Lighting control: between reading (brightest) and taking a nap (dimmest).
- Temperature control: between taking a nap (warmest) and exercising (coolest).
- Music volume control: between exercising (loudest) and watching TV (Quietest).

For each conflicting activity, we assume their maximum tolerance values are as specified in Table 5. Taking the lighting service as an example: For "Reading", since its default value is 500 lux and maximum tolerance assumption is –100 lux, the endurable output is between 500 lux and 400 lux. On the other hand, for "Taking a nap", because its default value is 0 lux and maximum tolerance is +100 lux, the endurable output falls between 0 lux and 100 lux. These maximum tolerance values can be adjusted easily according to users' needs. We define Compromised_Service (Activity_c) as follows:

$$\textit{Compromised\_Service}\ (\textit{Activity}_c) = \textit{Service}(\textit{Activity}_c) + \hat{T}(\textit{Service}\ (\textit{Activity}_c))$$

where $\hat{T}$ denotes the maximum tolerance assumption in Table 5 and Compromised_Service = { Lighting, Temperature, Volume }.

We name each of the above most conflicting activities Activity_c in our algorithm. Below we provide the details of our algorithms step by step for each service including lighting, temperature and music volume control. The algorithm has been defined in such a way that it can be applied to any service beyond lighting, temperature and music volume control. Moreover, the algorithm can be adjusted easily according to the unique properties of an IoT service. Note that all home services are set to a series of values whether they are binary (e.g. on/off) or have many different settings (e.g. levels of illumination).

Algorithm for Conflict Resolution

1) Identify whether there exists any Activity_c for the offered service (The offered service can be lighting, temperature, music volume or any other one defined for smart home control).
2) If only one exists, assuming there are n other non-Activity_c activities, then Output = $\varepsilon \cdot \textit{Service}\ (\text{Activity}_c) + (1 - \varepsilon) \times (\sum_{x=1}^{n} \textit{Service}(\text{Activity}_x) / n)$, where Activity_x denotes a non-Activity_c activity and ε is a pre-computed value that will be explained below.
3) Else if two exist, name them Activity_{c1} and Activity_{c2}, assuming there are n_1 users with Activity_{c1} and n_2 users with Activity_{c2}, we compare the summed weighted values for all n_1 users and n_2 users:
   $\sum_{x=1}^{n1} \textit{Weighted}(\text{user}_x)$, $\text{user}_x \in$ Users with Activity_{c1}
   $\sum_{x=1}^{n2} \textit{Weighted}(\text{user}_x)$, $\text{user}_x \in$ Users with Activity_{c2}

   Where $\textit{Weighted}(\text{user}_x) = \textit{Identity}(\text{user}_x) + \textit{Time}(\text{user}_x)$. Let the one with the highest summed weighted value be Activity_c, then Output = Compromised_Service (Activity_c).
4) Else, we calculate the weighted value of all n users: Output $= \sum_{x=1}^{n} (\textit{Weighted}(\text{user}_x) \times \textit{Service}(\text{Activity}_x)) / \sum_{x=1}^{n} \textit{Weighted}(\text{user}_x)$
5) Finally, make output adjustment according to the adjustment algorithm for each service.

The value of ε in Step 2 varies according to unique properties of each service. Basically, it is derived from the assumed maximum tolerance for each service. For example, if we assume the maximum tolerance of lighting is 100 lux, ε for lighting control would be derived as 0.67. Likewise, we can derive ε = 0.83 and ε = 0.75 for temperature and volume control by assuming their maximum tolerance at 1°C and 15%, respectively.

Algorithm for deriving the ε

The ε is only applied in computing Output = $\varepsilon \cdot \text{Service}(\text{Activity}_c) + (1 - \varepsilon) \times \text{Mean}\ (\text{Service}\ (\neg\text{Activity}_c))$ when there exists only one Activity_c. It is used to ensure that the service output would always lie within the maximum tolerance for Activity_c regardless the value of non-Activity_c ($\neg\text{Activity}_c$).

For example, when considering the lighting service in Table 6, we should consider the two situations below to compute the ε: "Exercising" against "Taking a Nap" and "Reading" against "Watching TV" because they have the largest difference in the lighting demand.

Table 4. Conflicting Activities for each service

Pair / Services	Most Conflicting Activities	
	Activity with lowest value	Activity with highest value
Lighting	Taking a nap (0 lux)	Reading (500 lux)
Temperature	Exercising (19.0°C)	Taking a nap (26.5°C)
Music Volume	Watching TV (0%)	Exercising (80%)

. *Table 5. Maximum tolerance assumption of each service*

Services / Activities	Lighting	Temp.	Volume
Reading	−100 lux		
Taking a Nap	+100 lux	−1.0°C	
Watching TV			+15%
Exercising		+1.0°C	−15%

Table 6. Computation of ε for lighting service

Variables / Activities	Default output for lighting	$Activity_c$ and its $\widehat{T}$ (tolerance assumption)	Non-$Activity_c$ with the most difference with $Activity_c$
Taking a Nap	0 lux	+100 lux	
Exercising	300 lux		Difference with Taking a Nap (300 lux)
Reading	500 lux	−100 lux	
Watching TV	200 lux		Difference with Reading (300 lux)

Basically, two situations need to be taken into consideration:

Situation 1. For Activityc which has the lowest value:

$$Service(Activity_c)+\widehat{T}(Activity_c)$$
$$= \varepsilon_1 (Service(Activity_c)) + (1-\varepsilon_1) \max(Service(\neg Activity_c))$$

For example, for lighting service, "Taking a Nap" is $Activity_c$ which has the lowest value and the equation will become the ratio problem (ε_1 and (1- ε_1)) for "Taking a Nap" and "Exercising" since they have the largest difference.

Situation 2. For Activityc which has the highest value:

$$Service(Activity_c)+\widehat{T}(Activity_c)$$
$$= \varepsilon_2 (Service(Activity_c)) + (1- \varepsilon_2) \min(Service(\neg Activity_c))$$

For example, for lighting service, "Reading" is $Activity_c$ which has the highest value and the equation will become the ratio problem (ε_2 and (1- ε_2)) for "Reading" and "Watching TV" since they have the largest difference.

Finally, the ε for lighting service should set as maximum of ε_1 and ε_2 to ensure the output will lie within the tolerance value of each $activity_c$ ("Taking a nap" or "Reading"). That is, **ε** = max (ε_1 , ε_2).

Below ε value deviation for lighting service is illustrated.

For $Activity_c$ which has lowest value: (Taking a nap):

$$Lighting(Taking\ a\ nap) + \widehat{T}(Taking\ a\ nap) = 0 + 100$$
$$= \varepsilon_1 (Lighting(Taking\ a\ nap))+(1- \varepsilon_1) \max(Lighting(\neg Taking\ a\ nap))$$
$$= \varepsilon_1 (0) + (1- \varepsilon_1)(300) \quad \text{Hence } \varepsilon_1 = 0.67$$

For $Activity_c$ which has highest value: (Reading):

$$Lighting(Reading) + \widehat{T}(Reading) = 500 + (-100)$$
$$= \varepsilon_2 (Lighting(Reading))+(1- \varepsilon_2) \min(Lighting(\neg Reading))$$
$$= \varepsilon_2 (500) + (1- \varepsilon_2)(200). \text{ Hence } \varepsilon_2 = 0.67.$$

Thus ε for lighting service should take maximum value of ε_1 and ε_2 : $\varepsilon = \max(\varepsilon_1, \varepsilon_2) = \max(0.67, 0.67) = 0.67$.

Table 7. Adjustment algorithm examples

variables	Condition	Adjustment of output
Environment Brightness (E.B.)	E.B. + 50 lux > output brightness	turn off the light
Environment Temperature (E.T.)	E.T. < output temperature	turn off the air conditioner
	E.T. > 32°C	output temperature - 0.5°C
	26 °C > E.T. > 22 °C	output temperature + 0.5°C
Identity	If a grandparent presents	output volume * 0.5
	Else if a parent presents	output volume * 0.8
Time(Hour)	Hour > 22 or Hour < 8	output volume max is 20

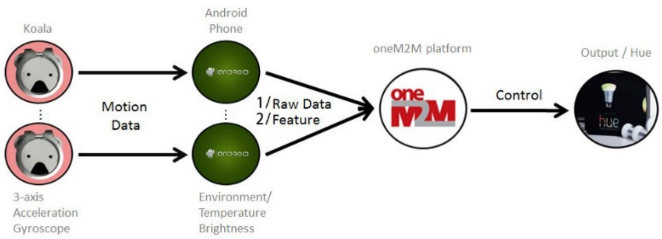


Fig. 5. System architectural alternatives with oneM2M Server

For the adjustment in the last step, we consider the environment variables in order to save energy and improve comfort level. The details of these adjustment procedures are provided in Table 7. For example, if the brightness output is within 50 lux difference of the environment brightness, the light will be turned off. Also, if the environment temperature is already lower than the temperature output, the air conditioner won't be turned on. Moreover, the output volume will be adjusted according to the identities of family members and the current time.

V. Experimental Results

We enhance our previously built activity recognition system for multiple users with the conflict resolution algorithm discussed in Section IV. Moreover, different architectural alternatives of the system design are evaluated again in order to find out the most efficient system configuration that includes wearable devices, smartphone gateway and backend server. This section presents our experimental results of such comparison and evluation.

We evaluate two system architectural alternatives as depicted in Fig. 5. Their differences lie in where the feature processing and activity detection procedures are carried out. The backend is a oneM2M server in both alternatives. In Alternative 1, the smart phone won't do feature processing but just relay the raw data to the server; on the other hand, in Alternative 2 feature processing is done on the smart phone.

Table 8. Evaluation of Memory, Processing and CPU at Server

Result / System	Memory Usage	Processing Time	CPU Usage
1/ Raw Data	2.80%	33.32000017s	12.833333s
2/ Feature	1.40%	0.309285726s	6.973333s

Note that Figs. 5 also indicates that multiple smart phones and multiple wearable devices are employed for the data collection. Essentially, there will be one wearable device and one smart phone for each user. The communications between the smart phone and the oneM2M server will follow the oneM2M RESTful standards to post and get the data to and from the server.

1) No Processing by Smart Phone

Feature processing, activity detection and conflict resolution will all be performed on the oneM2M server in the first architecture. After the algorithms are finished, the outputs will be stored in oneM2M platform.

2) Feature Processing by Smart Phone

In the second architecture, feature processing will be done in the smart phone. The oneM2M platform only needs to store the features from the RESTful posts and carry out both activity detection and conflict resolution algorithms based on the received features.

To discover which architecture can perform better, these two architectural alternatives are evaluated based on a set of server metrics including memory usage, processing time and CPU utilization. The results are shown in Table 8.

- Memory Usage - The first architecture demands much more memory resource, and it has 2.8% of memory usage as compared to 1%. In order to post and get the data to and from the server, there is more memory demand on the server when smart phone is not doing feature processing.
- Processing Time - Table 8 shows that if the server processing starts from raw data, it will take 107 times more processing time than the other starting from feature data.
- CPU Usage - With the help from smartphones, the server can reduce its CPU usage about 1.84 times in average.

Readers are referred to our first paper for more detailed evaluation results of different architectural alternatives [1].

VI. CONCLUSIONS AND FUTURE WORK

In this research, we have developed a situation awareness system for smart home based on multiple user activities recognition with two different architectures. Our activity recognition is based on Decision Tree.

The smart home situations under our consideration include brightness, temperature and music volume. Among different architectural alternatives, the system of feature processing done on the smartphone will provide a better performance.

Our unique contribution lies in designing several conflict resolution algorithms that are dynamic and flexible for multiple user activities at home. Our design not only can meet the priority of activity, identity and time, but also can be easily extended to cover any other situations. Furthermore, our design just relies on a simple wearable device Koala and a smart phone to identify each user's activity and to get the values of each user's priority and environment variables.

Via becoming situation aware based on the detection of multiple user activities at home, we can create a more convenient and comfortable environment for users through the control of lighting, temperature, and music. The future research under our consideration includes (1) detection of more activities and (2) collection of more user experience data in order to better statistically define the maximum tolerance value for each service.

ACKNOWLEDGMENT

The project reported in this paper is sponsored by Ministry of Science and Technology (MOST) of Taiwan Government under Project Number MOST 105-2218-E-009-004.

REFERENCES

[1] Yahua Lee, Fuchun Joseph Lin and Wei-Han Chen, "Multiple User Activities Recognition in Smart Home," The 3rd EAI International Conference on IoT as a Service, September 20–22, 2017, Taichung, Taiwan.

[2] Shu-Yun Lee and Fuchun Joseph Lin, "Situation Awareness in a Smart Home Environment", IEEE WF-IoT 2016, Reston, Virginia, USA, 12/12-14/2016.

[3] U.A.B.U.A. Bakar, Hemant Ghayvat, S.F. Hasanm and S.C. Mukhopadhyay, "Activity and Anomaly Detection in Smart Home: A Survey," Springer International Publishing Switzerland 2016 S.C. Mukhopadhyay (ed.), Next Generation Sensors and Systems, Smart Sensors, Measurement and Instrumentation 16, 2016.

[4] Rong Chen and Yu Tong, "A Two-stage Method for Solving Multi-resident Activity Recognition in Smart Environments," College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China, 2014.

[5] Liang Wanga, Tao Gu, Xianping Tao, Hanhua Chen, Jian Lu, "Recognizing multi-user activities using wearable sensors in a smart home," Pervasive and Mobile Computing 7, pp.287–298, 2011.

[6] Markus Prossegger and Abdelhamid Bouchachia, "Multi-resident Activity Recognition Using Incremental Decision Trees," ICAIS 2014, LNAI 8779, pp.182–191, 2014.

[7] Yi-Ting Chiang, Kuo-Chung Hsu, Ching-Hu Lu, Li-Chen Fu, and Jane Yung-Jen Hsu, "Interaction Models for Multiple-Resident Activity Recognition in a Smart Home," The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.

[8] W. Wang W, S. Ting, "Development of a computational simulation model for conflict management in team building," Int J Eng Business Manag 3(2): 9–15, 2011.

[9] Choonsung Shin, Yoosoo Oh and Woontack Woo, "History-based Conflict Management for Multi-users and Multi-services," Context Modeling and Decision Support 2005, CEUR-WS Proceedings, Vol.144, 2005.

[10] Sílvia Resendes, Paulo Carreira and André C. Santos, "Conflict detection and resolution in home and building automation systems: a literature review," Journal of Ambient Intelligence and Humanized Computing, Volume 5, Issue 5, pp 699–715, 2014.

[11] "Philip Hue," Philips, [Online]. Available: http://www2.meethue.com/. [Accessed 02 2017].

[12] "Weka (machine learning)," 01 2017. [Online]. Available: https://en.wikipedia.org/wiki/Weka_(machine_learning). [Accessed 02 2017].