Construction of a World Modeler for Smart Floor*

Jung-Hwa Choi, Young-Tack Park, Hee-Dong Ko+
School of Computing, Soongsil University, Seoul, Korea
+IMRC, Korea Institute of Science and Technology, Seoul, South Korea
cjh79@ailab.ssu.ac.kr, park@computing.ssu.ac.kr, +ko@kist.re.kr

Abstract

In this paper, we propose a generic location-based world modeler for smart floor in ubiquitous systems. Our World Modeler for Smart Floor (WMSF) algorithm can be applied to generate location-based context in any smart floor in ubiquitous environment. Smart floor is a major input device of ubiquitous environment. The input data is an array of weight, each array element is the weight information of objects in the position. We use a graphical representation to describe weight information and each object is modeled according to physical dimensions. We propose a generic world model algorithm based on WMSF architecture and have been able to recognize an area of objects using this constraints-based approach. We have also shown that the effect of object's relations is on recognition accuracy.

Key words: Ubiquitous Environment, Smart Floor, World Model, Context Awareness, Location-based Context

1. Introduction

At present, sensor network is major part in ubiquitous environment. Because sensor is main element that it transits the alteration of real world to electronic space. Therefore a user has to posses the sensor for the sensor networking, and it is maintained by connection of sensor nodes of all objects. If the user do not posses the sensor, however, it is hard to maintain. Also the connection of network is done by wireless, and there must be a problem in device's and sensor's power. Anyway smart floor does not make a problem that was explained above [1, 2].

Smart floor consist of tiles that have sensor in its bottom, so it can sense the object. The information of object's weight, that is in current space, is given to agent, e.g., smart offices, smart meeting rooms. By this information, it is very important to construct the world model. Because context awareness [3] have to be possible, and world model information is applied by ubiquitous agent. The world modeler that suggested in this research is designed generally at smart floor.

We are developing a generic world model algorithm for location-based world model on smart floor called ***World Modeler for Smart Floor (WMSF)*** algorithm. In this paper, we study on WMSF based on WMSF architecture. In order to propose the generic world model algorithm, WMSF architecture defines contexts by Web Ontology Language OWL. Contexts (i.e., at (object_id, tile's x-coordinates, tile's y-coordinates)) are generated based on smart floor sensor inputs. By context, we refer object information based on context associated ontologies. These contexts are very useful for ubiquitous agents. Ubiquitous systems need to maintain a world model of the current context that can be shared by all the ubiquitous agents.

WMSF architecture is proposed about a collection of automatic contexts for locations of moving objects. We propose a five-layered WMSF architecture based on objects in the position from smart floor sensors. *Situation-aware layer* is defined specification of smart floor. And then, *object modeling layer* is built based on the *situation-aware layer* to specify the constraint of objects. Next, *color map generation layer* is defined of color value about weight of tiles. Then the *WMSF process layer* is represented in color value based on the lower layer. Finally, the *contexts awareness layer* is inferred from output of WMSF algorithm using context knowledge base. The proposed system is based on ontologies.

A WMSF ontology can help independently developed agents to share context knowledge and thus minimizes the cost of context sensing. It is defined using the Web Ontology Language OWL, the latest Semantic Web language standard recommended by W3C [4]. Our Ontology is defined about smart floor's area for generic world model about same spaces, such as physical spaces and objects. The *triple converter* is used so that agent can process this ontology. Via the converted triple (i.e., (predicate subject object)) the agent catches which place of the smart floor the objects exist in.

The basic idea of WMSF algorithm is followed: First, the input of smart floor's weight is represented in graph. Second, each object has the model of weight and satisfies model constraint, so we can get the information of object's location in environment by weight graph.

* This research is supported in part by the Ubiquitous Autonomic Computing and Network Project, the Ministry of Science and Technology (MOST) 21st Century Frontier R & D Program in Korea.

This paper describes a layered approach to construct world modeler from smart floor sensors in ubiquitous environment. This approach can be applied to any smart floors. It generates model constraints via *situation-aware layer*, *object modeling layer* and *color map generation layer*. If this constraint model applies this to WMSF algorithm in *WMSF process layer* the world model will be possible in any smart floors.

The rest of this document is organized as the following: In section 2, we review other research that is designed for smart floor. In section 3, we present the design of WMSF architecture. Section 4 describes our WMSF ontology and a context maintenance model for agent. Section 5 proposes a generic world model algorithm based on WMSF architecture. In section 6, we discuss construction of a world modeler with that recognizes world model context. Lastly, some concluding comments and future works are given in Section 7.

2. Related Works

Our work is closely related to other smart floor and active floor research. In comparison to the smart floor system, our design of WMSF architecture takes a context awareness approach to build ontologies of contexts and attempts to use generic algorithm for modeling the location context of agents.

Robert, et al. [1] created user footstep models based on footstep profile features. They have outfitted a floor tile with force measuring instruments and are using the data gathered as users walk over the tile to identify them. This research clusters a footstep force profile of a user and then identifies the user. In our research, we make a space model by using ontology. As using ontology, also, it enables the agent to do model share and context reasoning for spaces.

Addlesee, et al. [5] used load tiles to measure force on a floor tile and used footstep data to perform user identification. They created user footstep models with hidden Markov models (HMMs) and compared unknown identity footsteps against the stored HMMs. Comparing to our research, this research recognizes an object with weight on floor tile but has not dealt with multiple objects. But we made it possible to recognize multiple objects with weight on floor tile. We, however, have not dealt with recognition on a person so it recognizes only things. Each of these systems was designed for awareness moving object within a small space. We propose a static object's location cognizing approach for precise modeling on moving objects.

3. World Modeler for Smart Floor Architecture

Fig. 1 shows the architecture of WMSF. World Modeler for Smart Floor (WMSF) is a generic location-based world modeler for smart floor in ubiquitous environments. Ubiquitous environments are physical spaces (e.g., offices, meeting rooms, lounges, bedrooms and vehicles) that provide ubiquitous computing services to users [6]. This system proposes WMSF algorithm to model objects that exist in physical space.

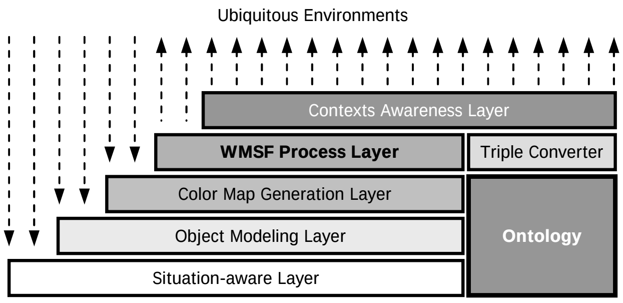


Fig. 1 WMSF architecture

It defines lower layer (i.e., *situation-aware layer, object modeling layer* and *color map generation layer*) in order to perform WMSF algorithm of four layers (i.e., *WMSF process layer*). Via lower layer, it generates constraint model of objects that exist in smart floor. As we generate object context model, the WMSF algorithm can be generically applied to any smart floors.

WMSF Architecture has developed the following five layers:

1. **Situation-aware Layer**: a collection of situational information for situation-context knowledge in a ubiquitous environment. It defines situational model using acquiring contexts from information (i.e., room's name, total area, each tile size and total number of tiles) of current space.

2. **Object Modeling Layer**: a creation of object's information for location-based context generation of existing objects in a current space. It defines object's model constraint (i.e., object name, weight of a leg, width, length, number of legs).

3. **Color Map Generation Layer**: a definition of color value about weight of tiles. It created this research by theorem. It can catch which object exists in a tile via color value.

4. **WMSF Process Layer**: a awareness of context using WMSF algorithm. It analyzes input data (i.e., set of weight on smart floor) in the basis of model constraint of lower layer. The analyzed data appears as set of object's color.

5. **Contexts Awareness Layer**: conversion of sensing information into context is available by an agent. It is based on the output of *WMSF process layer*. It creates context which an agent can process and stored useful context in ubiquitous environment to knowledge-base.

4. Modeling for Smart Floor

This system has modeled a space where smart floor had been set up. Smart floor is composed of sensor array of 22x20. A 2x2 sensor is placed on one floor tile, and it reads values of the sensor through scanning.

With a fixed specification, modeling for smart floor composes the space model and object constraint model via lower layer and it reads sensor value from sensor array. The modeling in lower layer will be discussed in this chapter.

4.1 WMSF Ontologies

We have developed a set of ontologies for model context on smart floor. In this paper, we define typical concepts and relations for describing physical spaces and objects. It is very important in world modeler to define context which can be existed on Smart floor. We use OWL/XML syntax to represent context ontologies, and we exploit an agent to support context reasoning. Ontologies of WMSF include the descriptions of spaces with identifiable geographic boundaries of floor (e.g., does area of meeting room's floor and property of tile?) and object constraint (e.g., how is the area of floor an object?).

The Space ontology includes concepts of space displays (e.g., space width and length, number of tiles), information of tiles in the space (e.g., existing space of tiles, tile size). Properties of this class include `area_x`, `area_y`, `numOfTilesWidth` and `numOfTiles Height`, which define the width and length size of smart floor and the number of width and height tiles. Subclass of `Space` is `Tiles`, which is concept of the tile size (i.e., `axis_x`, `axis_y`).

It is defined to create the model constraint of the Object ontology. In this ontology the class `Object` is the upper class, and properties include `width`, `length`, `numOfLegs` and `weightOfaLeg`. The `width` and `length` property expresses the tile size, the `numOfLegs` property represents number of legs of an object and the `weightOfaLeg` property represents the weight of one leg of object.

We apply ontology when we define space and the constraint of object. It enables us to get benefit from reuse and share.

4.2 Situation-aware Modeling

We re-define the smart floor as follows in order to help your understanding on WMSF algorithm proposed in this paper. Total area is 4800cm × 3600 cm. Such smart floor consists of 60cm × 60 cm tiles that contain pressure sensor, so total number of tiles are 8(width) × 6(height).

We define its instance as follows in the basis of WMSF ontology, so that the agent can make such situation information as context.

OWL instance (see Fig. 2) is used as the constant of WMSF algorithm. Instance is changed to predicate (i.e., triple) forms by *Triple Converter*. The followings are predicate forms generated by Jena (after removing OWL namespaces).

```
<Tiles rdf:ID="MeetingRoomTiles">
  <OfSpace>
    <Space rdf:ID="MeetingRoom"/>
  </OfSpace>
  <spc_name>SmartMeetingRoom</spc_name>
  <area_x>4800</area_x>
   <area_y>3600</area_y>
  <numOfTilesWidth>8</numOfTiles_width>
  <numOfTilesHeight>6</numOfTiles_height>
  <axis_x>60</axis_x>
   <axis_y>60</axis_y>
</Tiles>
```

Fig. 2 An example of OWL instance of a Space ontology

```
(type MeetingRoomTiles Tiles)
(OfSpace MeetingRoomTiles MeetingRoom)
(area_x MeetingRoomTiles 4800)
(area_y MeetingRoomTiles 3600)
(spc_name MeetingRoomTiles "SmartMeetingRoom")
(numOfTilesWidth MeetingRoomTiles 8)
(numOfTilesHeight MeetingRoomTiles 6)
(axis_x MeetingRoomTiles 60)
(axis_y MeetingRoomTiles 60)
```

This context can be processed by the agent. It grasps the meaning of context and applies to WMSF algorithm.

4.3 Object Modeling

Constraints of all objects that can exist in current space are defined in *object modeling layer*. Defined object and constraint for this algorithm are represented in Table 1.

Table 1. Constraint of Objects Model

Object	Weight of a leg	Width	Length	Number of legs
Chair	20	50	50	4
Table	100	200	100	4
Sofa	300	260	80	4

```
<Object rdf:ID="Table">
   <weightOfaLeg>100</weightOfaLeg>
   <width>200</width>
   <length>100</length>
   <numOfLegs>4</numOfLegs>
 </Object>
 <Object rdf:ID="Chair">
   <weightOfaLeg>20</weightOfaLeg>
   <width>50</width>
   <length>50</length>
   <numOfLegs>4</numOfLegs>
 </Object>
 <Object rdf:ID="Sofa">
   <weightOfaLeg>300</weightOfaLeg>
   <width>260</width>
   <length>80</length>
   <numOfLegs>4</numOfLegs>
</Object>
```

Fig. 3 An example of OWL instance of Tiles ontology

Fig. 3 describes the constraint of object modeling in OWL agent can process. It is changed to following predicate logic. This formation is easy to change the data, even if the object is not changed. It is stored to context knowledge base to share and reuse between agents.

```
(type Table Object)   (weightOfaLeg Table 100)
(width Table 200)     (length Table 100)
(numOfLegs Table 4)
(type Chair Object)   (weightOfaLeg Chair 20)
(width Chair 50)      (length Chair 50
(numOfLegs Chair 4)
(type Sofa Object)    (weightOfaLeg Sofa 20)
(width Sofa 50)       (length Sofa 80)
(numOfLegs Sofa 4)
```

4.4 Color Map Generation

Smart floor tile is scanned in order from left top. Tile initial value is 0. And if there is an object on the tile, weight value is given to the tile. Figure 4 is the input and output example of WMSF algorithm.

(a) is the weight value of tile which is scanned from smart floor sensor. (b) is color value of each object areas after processing WMSF algorithm. This value is defined in *color map generation layer* (Fig. 4).

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 140 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 100 & 0 & 0 \\ 0 & 300 & 0 & 0 & 0 & 300 & 0 & 0 \\ 0 & 300 & 0 & 0 & 0 & 300 & 0 & 0 \end{bmatrix}$$

M : Set of weight values from smart floor sensors, **M** is 8 X 6

(a) Input

$$\mathbf{N} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C10 & C2 & C2 & C2 & 0 & 0 \\ 0 & 0 & C2 & C2 & C2 & C2 & 0 & 0 \\ 0 & C3 & C3 & C3 & C3 & C3 & 0 & 0 \\ 0 & C3 & C3 & C3 & C3 & C3 & 0 & 0 \end{bmatrix}$$

N : Set of object's color by weight, **N** is 8 X 6

(b) Output

Fig. 4 Example of input and output

Object modeling layer create color value within number, weight of object. Follow theorem is to create color value when there are three objects.

Theorem: Creation of color's value

If N is 3,
  $ColorVal_{100}$ = C1,
  $ColorVal_{010}$ = C2,
  $ColorVal_{001}$ = C3, then
  $ColorVal_{ijk} = (ColorVal_{i-1\ j\ k} + ColorVal_{i\ j-1\ k} + ColorVal_{i\ j\ k-1} + C1 + C2 + C3) / N$.

The index of variable `ColorVal` and color's value increase by the number of objects. Table2 is color's value made of theorem (Table 2 shows minimum data by size of tile.).

Table 2 Definition of Color's Value

Value of Color	Object (num)[a]	W.[b]	Value of Color	Object (num)[a]	W.[b]
C1	Chair(1)	20	C7	Chair(1) Table(1)	120
C2	Table(1)	100	C8	Chair(1) Sofa(1)	320
C3	Sofa(1)	300	C9	Table(1) Sofa(1)	400
C4	Chair(2)	40	C10	Chair(2) Table(1)	140
C5	Table(2)	200	C11	Chair(2) Sofa(1)	340
C6	Sofa(2)	600	C12	Chair(4)	80

[a]number in () is the number of object's legs,
[b]W. is weight

5. WMSF Algorithm

Problem: Constructing world modeler about smart floor
Inputs: An array **M** of tile weights of a smart floor
Outputs: Object's color set **N** by weights

Assumption:
$\mathbf{W}_{atomic} = [w_{a1}, w_{a2}, \ldots, w_{an}]$
w_{ai} = measured weight when one leg of each objects resides in a tile
$\mathbf{W}_{double} = [w_{d1}, w_{d2}, \ldots, w_{dn}]$
w_{di} = measured weight of two legs of each objects reside in a tile

```
void wmsf (int M[])
{
    if ($2w_{ai} \neq w_{aj}$) {
        if (tile's weight is atomic) {                ………①
            while ($\mathbf{M}_{ij} \notin \mathbf{W}_{atomic}$) {
                find $\mathbf{M}_{ij} \in \max(\mathbf{W}_{atomic})$
                find_atomicObjLookup($\mathbf{M}_{ij}$)
                insert_color($\mathbf{M}_{ij}$)
            }
        }

        if (tile's weight is multiple) {              ………②
            while ($\mathbf{M}_{ij} \notin \mathbf{W}_{double}$) {
                find $\mathbf{M}_{ij} \in \mathbf{W}_{double}$
                find_multipleObjLookup($\mathbf{M}_{ij}$)
                insert_color($\mathbf{M}_{ij}$)
            }
        }
    }
}
```

Fig. 5 WMSF Algorithm

WMSF algorithm gets smart floor's weight as input data (e.g., `[0,0,0, …, 40, 0, … 0, 140, 0, …0, 300, 0, 0]`) and the information of object's location is represented in color's value based on Table 1's object constraint that is defined in chapter 4.

WMSF algorithm that suggested in Fig. 5 is pseudo code for constructing world modeler about smart floor. We will research it in detail in this chapter. First of all, configure M vector by input data as smart floor's weight.

5.1 Assumption

WMSF algorithm is divided to W_{atomic} (i.e., one leg per one tile; chair, table, sofa) and W_{double} (i.e., legs per one tile; 2, 4 chair legs). And it can reduce the possibility of multiple objects happening by setting object's area in case of W_{atomic}.

However, in case one object's weight is the same with the two atomic object's weight, the algorithm is not applied to. For example, if a table's leg weight is 100s, and a chair leg weight is 50, if there are two chairs for satisfying table's constraint, two chairs must be recognized as the table.

5.2 Tile's Weight is Atomic

If there is a leg in a tile, WMSF algorithm recognizes heaviest weight object first. As recognize heaviest weight object first, it suggests technique that reducing the number of color values in Table 2.

In WMSF algorithm ① (Fig. 5), finding tiles that sensing a leg's weight. If so we can expect which object to be. And find a tile that have equal or more weight tile among the same row. Next, compare expected object's width or height with the distance between two tiles. If it equal to width, compare it with row as origin. Find a tile far from the expected object's length. And then compare its weight exceeds expected object's weight or not. Finally, if a tile that exist in diagonal direction from origin tile has over weight than expected object's weight, object is decided and get information of location.

5.2.1 Insert Color Value into Output Vector

In this stage, put color value into output vector N corresponds to the tile's area that have the information of location. First of all, recognized four tiles become static points. And then these points can be recognized as a object. Finally, put the difference value between recognized tile's value and recognized object's weight into input vector M. Therefore, it can recursively search atomic weight on this area and reduce multiple weight witch we have to extract.

Fig. 6 is the result of WMSF algorithm at one time from Fig. 4 (a). If repeat WMSF algorithm's ① (Fig. 5),, output vector will fill out color value. And the weight value in input vector remains only multiple weight case.

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 300 & 0 & 0 & 0 & 300 & 0 & 0 \\ 0 & 300 & 0 & 0 & 0 & 300 & 0 & 0 \end{bmatrix}$$

M vector

$$\mathbf{N} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C10 & C2 & C2 & C2 & 0 & 0 \\ 0 & 0 & C2 & C2 & C2 & C2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

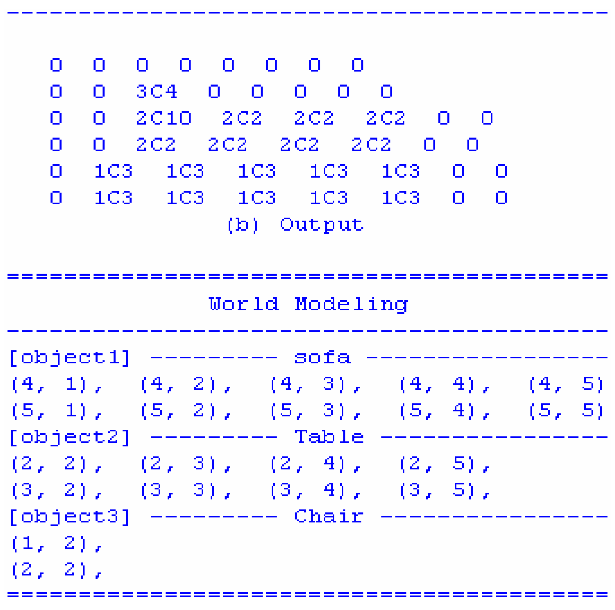
$\mathbf{N}$ vector

Fig. 6 Example of WMSF Algorithm

5.3 Tile's Weight is Multiple

Eliminate multiple weight case from input vector M, if still there exist a value, it is data of chair. There exist two cases. One is that a chair is in a tile. The other is that two legs are in a tile. Therefore, multiple weight case is processed after processing atomic weight case in WMSF algorithm. In case of first, apply color value to output vector N. In case of second, subtract from input vector M to weight value. And in case of two legs in a tile, investigate 8 direction tiles. Then perform 5.2.1 part. If it repeats until input vector M's value does not appear in W_{double}, result will be Fig. 4 (b).

6. World Modeling for Agents



```
-----------------------------------------

  0   0   0   0   0   0   0   0
  0   0   3C4   0   0   0   0   0
  0   0   2C10   2C2   2C2   2C2   0   0
  0   0   2C2   2C2   2C2   2C2   0   0
  0   1C3   1C3   1C3   1C3   1C3   0   0
  0   1C3   1C3   1C3   1C3   1C3   0   0
                 (b) Output

=========================================
              World Modeling
-----------------------------------------
[object1] --------- sofa ----------------
(4, 1),  (4, 2),  (4, 3),  (4, 4),  (4, 5)
(5, 1),  (5, 2),  (5, 3),  (5, 4),  (5, 5)
[object2] --------- Table ---------------
(2, 2),  (2, 3),  (2, 4),  (2, 5),
(3, 2),  (3, 3),  (3, 4),  (3, 5),
[object3] --------- Chair ---------------
(1, 2),
(2, 2),
=========================================
```

Fig. 7 an output of WMSF algorithm

Agent is modeling ubiquitous environment to web within recognizing world modeling context. It is useful data when agents make an inference from services witch agent s will give to person in current space. Fig. 7 is the result of WMSF algorithm using input of this thesis.

The number in front of color value of (b) means index if objects (Fig. 7). Looking at the result of world modeling, we can know location information about three objects. It means coordinates value of tile existing object. We complete context recognition (Section 6.1) and world model construction (Section 6.2) within this output data.

6.1 Context Awareness

Context awareness needs to identify relations between objects. Relations mean proximity relations between objects, such as at, left-sided and inside. *Context awareness layer* deduces the mean of output which is made by WMSF algorithm. We use a knowledge based approach to model semantic relations between objects. Follow figure is contexts which the agent analyzes and creates the output of *WMSF process layer*. Agent is able to grasp the position of object based on contexts, so it is able to understand each area of objects on smart floor.

```
at(object1,4,1) at(object1,4,2) at(object1,4,3)
at(object1,4,4) at(object1,4,5) at(object1,5,1)
at(object1,5,2) at(object1,5,3) at(object1,5,4)
at(object1,5,5) at(object2,2,2) at(object2,2,3)
at(object2,2,4) at(object2,2,5) at(object2,3,2)
at(object2,3,3) at(object2,3,4) at(object2,3,5)
at(object3,1,2) at(object3,2,2)
inside(object3, object2)
```

We are able to know the coordinates value of object2 and object3 by above context. Therefore, the agent deduces these objects are same tile position. Also it is able to know the object3 is inside of object2 (i.e., inside (object3, object2)).

6.2 World Model

In order to generate contexts of objects, a world model has been created. Each object in ubiquitous systems has been modeled based on WMSF algorithm. Our system is supposed to be used in smart room. The floor sensor recognizes which objects exist in the space. The scanned sensor date have input to WMSF algorithm by one dimension array. The input data is divided to atomic weight and multiple weights. It is able to increase the performance of algorithm through extracting atomic weight and removing it.

Looking at the world modeling result of the examples of this paper, we are able to see that there are three objects in current space. Also we are able to see that they are sofa, table and chair. The sofa is located at (4,1) and on tile which is located at (5,5) from tile. The table is located crosswise from (2,2) tile to (3,5) tile. The chair locate at (1,2) tile and (2,2) tile is inside of the table.

As this, world model is used for agent in ubiquitous environment. We are able to see what object the person is located in when the person enters. Therefore, we are able to deduce why the person is in object area. World model is the first work to recognize in ubiquitous environment. It could be useful data to give service reasoning to the person in current space.

7. Conclusion and future work

Most important element to configure world model for smart floor is reducing the multiple weight case after recognizes atomic object's weight. Heavy weight value among atomic objects is processed firstly based on each object's constraint. Therefore it needs that getting information of location from object that has special weight.

In this research, context awareness could be possible, consider world model information applied by ubiquitous agent. Therefore, we suggest WMSF architecture. We divide lower layer and upper layer, the lower layer creates constraint model referred to the input WMSF process and the upper layer creates contexts by the output of WMSF process. It creates world model by agent. Our WMSF algorithm which is the core of this architecture is suggested to be important to make world model from weight. However, this paper does not consider disorder objects. Also we do not consider backtracking when it misrecognizes object caused by wrong clustering. The further study will consider this problems and the assumption of this study.

References

1. Robert J. Orr, Gregory D. Abowd. (2000). "The Smart Floor: A Mechanism for Natural User Identification and Tracking." CHI: Conference on Human Factors and Computing Systems. Vol. 00, No. 00, pp. 0275-0276.
2. J. Hightower,, G. Borrillo. (2001). "Location system for ubiquitous computing." IEEE Computer. Vol. 34, No. 08, pp. 0057-0066.
3. Harry Chen AND Tim Finin AND Anupam Joshi. "An Ontology for Context-Aware Pervasive Computing Environments," In Workshop on Ontologies and Distributed Systems, August 2003.
4. Michael K. Smith, Chris Welty, and Deborah McGuinness. "Owl web ontology language guide," http://www.w3.org/TR/owl-guide/, 2003.
5. Addlesee, M., Jones, A., Livesey, F., and Samaria, F. "The ORL Active Floor," IEEE Personal Communications, October 1997, 35-41.
6. L.Kagal, V. Korolev, H. Chen, A. Joshi, and T. Finin. "Centaurus : A framework for intelligent services in a mobile environment," In Proceedings of the International Workshop on Smart Appliances and Wearable Computing (IWSAWC), 2001.