Activity Recognition in Ubiquitous Learning Environment

Tao Lu, Shaokun Zhang, Qian Hao
System Engineering Institute, Dalian University of Technology, Dalian, China
lutao@dlut.edu.cn, {zhangshk, haoqian}@mail.dlut.edu.cn

***Abstract*—With the advance and the development of sensors, computing devices and wireless communication networks, ubiquitous computing has been an active and fast growing research area. The technology is applied in educational domain, resulting in ubiquitous learning, in which the system can detect the students' behavior and provide personalized support to guide the students to learn in real world. This paper focuses on activity recognition in ubiquitous learning environment which assists the novice user to conduct a complex experiment. Here the activity is a kind of complex activity which is an independent task to achieve a certain goal and is composed of simple actions. The approach we propose is knowledge-driven technique. We first present concepts of context pattern and context evolving pattern, and based on these concepts define the activity model. Then we analyze the condition of distinguishable activity and propose the algorithm of activity recognition. Activity model in single-crystal X-ray diffraction experiment support system is designed and the method is further interpreted through this case.**

***Index Terms*—ubiquitous learning, context awareness, context reasoning, activity recognition**

I. INTRODUCTION

In the early 90s of 20 century, Mark Weiser proposed the concept of ubiquitous computing as a new paradigm for next-generation systems in which computers disappear in the background of the user's everyday activities[1]. One feature of such ubiquitous computing environment is context awareness, i.e. the systems sense the user's context and provide adapted service accordingly.

The technology is applied in educational domain, resulting in ubiquitous learning, in which the system can detect the students' behavior and provide personalized support to guide the students to learn in real world[2][3][4]. Many studies concerning this issue have been conducted on language training courses[5][6][7], natural science courses[8][9], and science experiment[4].

In this paper, we focus on the activity recognition in ubiquitous learning environment which assists the novice user to conduct a complex experiment, such as single-crystal X-ray diffraction experiment. The environment will provide real time guidance and remind information when needed. This should be performed on the basis of correctly judging the situation, i.e. recognizing the activity being conducted and determining whether it is conducted normally.

Activity recognition is one of issues in ubiquitous computing research. From the perspective of data source, activity recognition can be divided into wearable sensor-based and machine vision-based[10]. From the methodology view, there are data-driven approaches and knowledge-driven approaches[11]. Most previous researches focus on the recognition of single action, while there have been efforts on the recognition of complex activities like social ones[12].

Data-driven techniques are based on the machine learning methods and are well suited for recognizing simple activities and gestures from raw sensor data or video data[13]. A wide range of algorithms and models for activity recognition include Hidden Markov Models [14], dynamic and naïve Bayes networks[15], decision trees[16] and so on.

Knowledge-driven techniques, concerning representation of action and situation as well as reasoning with them, is closely related with classical topics in artificial intelligence[11]. Situation calculus, event calculus and their variants are adopted in some research to model the temporal and causality relation between activities and events[17][18][19]. Ontology-based method is another frequently used technique. For example, Chen has proposed a technique to recognize activities through ontological reasoning[20]. Besides dealing with low-level sensor data, knowledge-driven techniques can also be used in higher level, i.e. recognizing complex activities based on the recognized simple action[21].

In the process of conducting science experiment, the learner should conduct every activity as standard workflow. Here the activity refers to independent task which is meaningful in workflow perspective, such as preparing a fiber in single-crystal x-ray experiment. The simple action like entering Room 101 is not an activity to be recognized in this paper, but is regarded as a kind of high-level context which is useful in recognizing the activity. Therefore, the activity to be recognized in our work is complex activity and the approach we propose is knowledge-driven technique.

Compared with other knowledge-driven techniques, the approach we propose considers the temporal relation between activities and contexts. That is, activity is seemed as process with duration, and context may change in the process. This kind of knowledge is integrated into

doi:10.4304/jait.3.1.29-35

activity models. In addition, other types of constraints like activity sequence are also taken into account.

For example, in ref.[4], there is a rule:

If Location(*student*)=**Location**(*R204*)

Then Phase(*student*)=*select a crystal*

which can be used to determine the activity *select a crystal*. However, the knowledge about the relations of activity and context is incomplete and may lead to errors. In this paper, we focus on modeling the context and context change in the process when the activity is conducted. This will enhance the ability of activity recognition.

The rest of the paper is organized as follows. Section 2 models the activity with context pattern and context evolving pattern. Activity recognition algorithm is presented in section 3. Section 4 reports our case study and section 5 concludes the paper

II. MODELING ACTIVITY WITH CONTEXT PATTERN AND CONTEXT EVOLVING PATTERN

A. The Architecture of Experiment Support System

In this paper, we focus on the experiment support system, a kind of ubiquitous learning system which provides guidance for novice learner to conduct science experiment. The architecture of the system is shown in Fig.1.

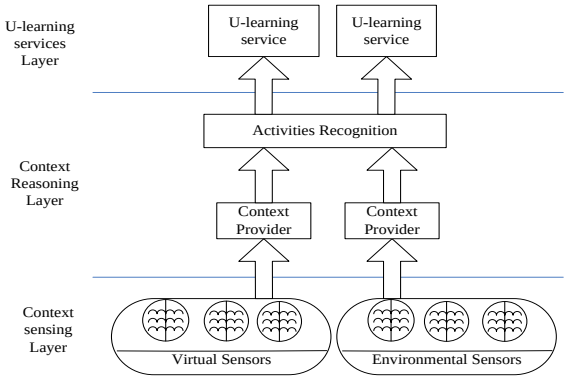

Figure 1. Architecture of experiment support system.

In context sensing layer, sensors (either virtual sensors or environmental sensors) are used to acquire the raw context. The acquired low-level context data is transmitted into context reasoning layer where higher-level context information, e.g. recognized activity is derived by context reasoning. In context reasoning layer, context provider module deals with the raw context data, abstracting and representing context information in formal format. This function can be provided by many context-aware middleware or infrastructure. On the basis of the context provided by context provider module, activity recognition, which is studied in this paper, is conducted. According to the recognized activity, adapted services are provided to the learner in the u-learning services layer.

B. Context pattern and context evolving pattern

Suppose the application is related to a set of contexts which are denoted as $c_1, c_2, .. c_n$, and the domain of c_i is D_i. At given time t, the value of context c_i is denoted as $c_i(t)(c_i(t) \in D_i), i = 1,2,..n$. For instance, the user is located in the kitchen at the time t can be represented as **Location**(t) =*kitchen*. The values of context $c_1, c_2, .. c_n$ at time t is denoted as $C(t)$, $C(t)=(c_1(t),c_2(t),\ldots c_n(t))$.

Suppose $d_j^{(i)} \subseteq D_i$, $(c_i, d_j^{(i)})$ is called atomic context pattern which represents a type of context. Composite context pattern can be derived from one or more atomic context pattern using conjunction, disjunction and negative operation on atomic context pattern. Here conjunction, disjunction and negative operation have their usual meanings.

Definition 1: Context pattern is defined recursively as follow:

(1) If α is an atomic context pattern, then α is a context pattern.

(2) If α, β are context patterns, then $\alpha \wedge \beta$, $\alpha \vee \beta$, $\neg\alpha$ are context patterns.

The context pattern $(c_i, d_j^{(i)})$ holds at time t, if and only if $c_i(t) \in d_j^{(i)}$, denoted as $hold\_at((c_i, d_j^{(i)}), t)$.

$$hold\_at((c_i, d_j^{(i)}), t) \Leftrightarrow c_i(t) \in d_j^{(i)}$$

Apparently we have following rules:

$$hold\_at(\alpha \wedge \beta, t) \Leftrightarrow hold\_at(\alpha, t) \wedge hold\_at(\beta, t)$$
$$hold\_at(\alpha \vee \beta, t) \Leftrightarrow hold\_at(\alpha, t) \vee hold\_at(\beta, t)$$
$$hold\_at(\neg\alpha, t) \Leftrightarrow \neg hold\_at(\alpha, t)$$

Definition 2: Let $v = (v_1, v_2, .. v_n), v_i \in D_i, i = 1,2,..n$, p be an context pattern. v is called an instance of context pattern p if and only if $\forall t((C(t) = v) \rightarrow hold\_at(p, t))$.

We use *Instance*(p) to denote the instance set of context pattern p.

The context may change during a period $(t_1, t_2), t_1 < t_2$. We use operators "#", "*" to represent the context state change during a certain period.

If p is a context pattern, t_1, t_2 are two time points, $t_1 < t_2$, the meanings of the operators are defined as follows:

$$(\# p)_{(t_1,t_2)} \Leftrightarrow \forall t((t_1 \le t \le t_2) \rightarrow hold\_at(p, t))$$
$$(* p)_{(t_1,t_2)} \Leftrightarrow \exists t((t_1 \le t \le t_2) \wedge hold\_at(p, t))$$

Definition 3: If p are context patterns, #p and *p are called atomic context evolving patterns which represent context change types.

Definition 4: Context evolving pattern is defined recursively as follow:

(1) If S_1 is an atomic context evolving pattern, then S_1 is a context evolving pattern.

(2) If S_1, S_2 are atomic context evolving patterns, then $S_1 \wedge S_2$ is a context evolving pattern.

Context pattern describes context feature in time point, while context evolving patterns describes context feature in a time period.

Definition 5: Let $v=(v_1,v_2,..v_n), v_i \in D_i, i=1,2,..n$, S be a context evolving pattern. If for any (t_1,t_2) and t, $t_1 \le t \le t_2$, $(S)_{(t_1,t_2)} \to (C(t) \ne v)$, then v is incompatible with S, denoted as $v \downarrow S$, else v is compatible with S, denoted as $v \uparrow S$.

Definition 6: Let p be a context pattern, S be a context evolving pattern. If for any (t_1,t_2), $(S)_{(t_1,t_2)} \to (\#\neg p)_{(t_1,t_2)}$, then p is incompatible with S, denoted as $p \downarrow S$, else p is compatible with S, denoted as $p \uparrow S$.

Theorem 1: Let p be a context pattern, S be a context evolving pattern. If $p \downarrow S$, then for any $v \in Instance(p)$, $v \downarrow S$.

Proof: If $p \downarrow S$, then as definition 6, for any time period (t_1,t_2), we have:

$$(S)_{(t_1,t_2)} \to (\#\neg p)_{(t_1,t_2)}. \quad (2\text{-}1)$$

Since $v \in Instance(p)$, as definition 2, for any time t, $(C(t)=v) \to hold\_at(p,t)$, that is:

$$\neg hold\_at(p,t) \to (C(t) \ne v) \quad (2\text{-}2)$$

From the meaning of operator "#", we have:

$$(\#\neg p)_{(t_1,t_2)} \Leftrightarrow \forall t((t_1 \le t \le t_2) \to \neg hold\_at(p,t)) \quad (2\text{-}3)$$

According to (2-1), (2-2), (2-3), we can deduce that for any $t, t_1 \le t \le t_2$, $(S)_{(t_1,t_2)} \to (C(t) \ne v)$. Therefore $v \downarrow S$.

C. Activity model

(i). Integrating context pattern and context evolving pattern into activity model

In experiment support system, an activity is a task with duration for a certain goal. With the aim of providing adaptive assistance, the system should recognize the activity being carried out, and determine if it is performed normally.

The underlying assumption of activity recognition is that different activities may result in different context states and context change. So the relationship between context and activity must be integrated into activity model.

Definition 7: Let a be an activity, and S be a context evolving pattern, $Interval(a)$ be the time period in which the activity a is being conducted normally. If $S_{Interval(a)}$ and $\neg\exists S'((S' \to S) \wedge (S'_{Interval(a)}))$, then S is running pattern of a, denoted as $\mathcal{P}_{\text{on}}$.

Definition 8: Let a be an activity, and q be a context pattern. If q holding at a time point after the predecessor activities of a has ended marks a has started, q is called start flag of a, denoted as $\mathcal{P}_{\text{start}}$.

Similarly, we can define the end flag of an activity.

Definition 9: Let a be an activity, and q be a context pattern. If q holding at a time point when activity a is performing marks the end of a, q is called end flag of a, denoted as $\mathcal{P}_{\text{end}}$.

When one activity has been ended, and a new activity has been started, it generally takes time to change the context. Therefore, even the activity has been recognized to be started according to its start flag, the context value may be incompatible with its running pattern. When this situation occurs, we say that the activity is starting. So starting can be regarded as a state of activity.

When the activity is suspended by some reason, the context may change resulting in the context value incompatible with running pattern. Therefore, activity being suspended can be detected if the end flag does not hold and the context value is incompatible with running pattern.

Activity is suspended by many reasons, some of which should be recognized for providing adaptive support.

Definition 10: Let $H(a)$ be a set of suspending type of activity a which the system should recognize. For $h \in H(a)$, if context pattern q holding at a time point when activity a is being conducted marks that a is suspended by h, q is called suspend flag of h.

The suspend flag set of an activity a is denoted as $\mathcal{P}_{\text{suspend}}$. Let $sus\_flag : H(a) \to \mathcal{P}_{\text{suspend}}$ be a function which mapping a suspend to its suspend flag. Apparently, $\mathcal{P}_{\text{suspend}} = \{sus\_flag(h) \mid h \in H(a)\}$.

Definition 11: Let A be an activity set. For each $a \in A$, the activity model of a, denoted as $\mathcal{A}$(a), is a tuple $\mathcal{A}$(a)=<, $\mathcal{P}_{\text{on}}$, $\mathcal{P}_{\text{start}}$, $\mathcal{P}_{\text{end}}$, $\mathcal{P}_{\text{suspend}}$, $H(a)$, sus_flag>, where $sus\_flag : H(a) \to \mathcal{P}_{\text{suspend}}$ is a function which mapping a suspend to its suspend flag.

(ii). Activity flow

The activity should be carried out in predefined sequence. Fig. 2 shows an example of activity process, where split and join structure are same with general workflow definition.

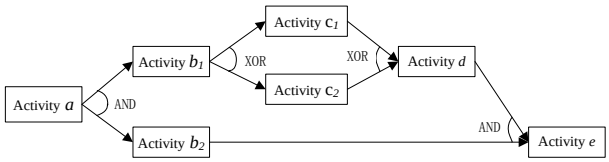

Figure 2. The sequence of the activities process

Definition 12: Activity process is 4-tuple <*A*, *E*, *join*, *split*>, where *A* an activity set, $E \subseteq A \times A$,representing the sequence of activities, *join*: $A \to \{AND, XOR, NULL\}$, *split*: $A \to \{AND, XOR, NULL\}$.

<*A*, *E*> is a directed graph which represents the dependency between activities. *AND*, *XOR* and *NULL* are *join* or *split* types of nodes. *AND-join*: Multiple parallel executed activities join into a single activity. *AND-split*: An activity splits into multiple parallel activities that are all executed. *XOR-join*: Multiple mutually exclusive alternative activities join into a single activity. *XOR-split*: An activity splits into multiple mutually exclusive alternative activities, only one of which is followed. *NULL* means the corresponding activity has a single incoming or outgoing activity.

Let $a \in A$, if *split*(*a*)=*XOR*, then *a* is called *XOR-split* activity. *AND-split*, *XOR-join*, *AND-join* can be defined similarly.

Let $a,b,c \in A$, if $(a,b) \in E, (a,c) \in E, split(a) = XOR$, then b and c are called *XOR-Sibling* activities of each other.

Definition 13: Let A be an activity set, the activity model of A is a tuple $<AP,MA>$, where AP is an activity process model, $AP = <A, E, join, split>$, $MA = \bigcup_{a \in A} \mathcal{P}(a)$.

Ⅲ. ACTIVITY RECOGNITION ALGORITHM

A. Condition of distinguishing activity

When one activity has ended, the activities whose predecessor activities have been executed are the candidates to be selected to execute next. Among them, whether activities can be distinguishable depends on the differences of starting flags of them.

Suppose $a \in A$, let *neighbor*(a) to denote the set of activities which have possibility in selecting-to-execute-next candidate set including a. Obviously the predecessor activities and postdecessor activities can't be included in *neighbor*(a).

Besides, the activities which has common *XOR-split* predecessor with a but in other branches, should be excluded in *neighbor*(a) if *XOR-split* predecessor is not direct predecessor of a. Similarly, the postdecessor activities of *XOR-sibling* of a should be also excluded.

Let A be an activity set, $<AP,MA>$ is the activity model of A. For each activity $a \in A$, we have following rules.

Rule 1: for each $a' \in neighbor(a)$, any time point t,

$hold(a.\mathcal{P}_{start}, t) \Rightarrow \neg hold(a'.\mathcal{P}_{start}, t)$

Rule 2: $a.\mathcal{P}_{end} \downarrow a.\mathcal{P}_{on}$

Rule 3: for each $h \in H(a)$, $sus\_flag(h) \downarrow a.\mathcal{P}_{on}$

Rule 4: for $h_i, h_j \in H(a)$, any time point t,

$hold(sus\_flag(h_i),t) \Rightarrow \neg hold(sus\_flag(h_j),t)$

If the activity model of A satisfies above rules, the activities in A are distinguishable to each other. Therefore, the activity can be recognized.

B. Activity recognition algorithm

According to activity model which satisfies the rules in 3.1, activity and its state in any time can be recognized. Suppose the system can detect every changes of context. Algorithm **ActReg** is used to determine current activity and state. The algorithm is as follow.

Algorithm 3.2.1 ActReg
Input: $<AP,MA>$, LastActivity, LastState, v
Output: CurActivity, CurState
BEGIN
CASE LastState **OF**
starting:
IF $v \uparrow$ LastActivity.$\mathcal{P}_{on}$
THEN CurActivity=LastActivity, CurState=*on*
on:
IF $v \in Instance$(LastActivity.$\mathcal{P}_{end}$)
THEN CurActivity=LastActivity, CurState=*end*
ELSE IF $v \uparrow$ LastActivity.$\mathcal{P}_{on}$
THEN CurActivity=LastActivity, CurState=*on*
ELSE CurActivity=LastActivity,
CurState=*suspend*
suspend:
IF $v \uparrow$ LastActivity.$\mathcal{P}_{on}$
THEN CurActivity=LastActivity, CurState=*on*
ELSE CurActivity=LastActivity,
CurState=*suspend*
end: **BEGIN**
CurActivity=LastActivitivy, CurState=*waiting*
FOR each $a \in next$(LastActivity)
IF $v \in Instance(a.\mathcal{P}_{start})$
THEN CurActivity=*a*,CurState=*starting*
END
END

The input of above algorithm are the activity model, determined activity and state last time i.e. LastActtivity and LastState, current context values v which is different from last time based on which to determine LastActivity and LastState. The outputs are current activity and state corresponding to v. In algorithm *next*() is a function which mapping an activity to its next possible activity set.

In order to use the algorithm to determine the first activity, the initial value of LastAcitivity and LastState is defined as *NULL* and *end*. Thus when the first activity has been started, the system can detect the context change and the activity can be recognized by the algorithm.

When the output of the system is that CurState=*suspend*, the reason should be determined in order to provide proper support. This can be accomplished by the following algorithm **JudgeSuspendType**.

Algorithm 3.2.2 JudgeSuspendType
Input: $<AP,MA>$, CurActivity, v
Output: SuspendType
BEGIN
SuspendType=*general*
FOR each sustype $\in H$(CurActivity)
IF $v \uparrow sus\_flag$(sustype)
THEN SuspendType=sustype
END

If the output of last algorithm is *general*, it means that no support is needed and it is not necessary to determine further reason.

Ⅳ. CASE STUDY

According to the approaches presented above, we design an activity model which can be used in single-crystal X-ray diffraction experiment support. Huang et al has designed same kind of system and analyze its effect in education[4]. However technology is not an important issue in that paper and activity recognition is simply realized by rule-based reason. Our work focuses on the technical aspects.

The single-crystal X-ray diffraction procedure consists of three phases, in which activities are conducted in three places, such as Room 101, Room 102 and Room 103 in our example. The three phases are: selecting a crystal which is carried out in Room 101, analyzing the crystal which are conducted in Room 103 and structural determination which is accomplished in Room 103.

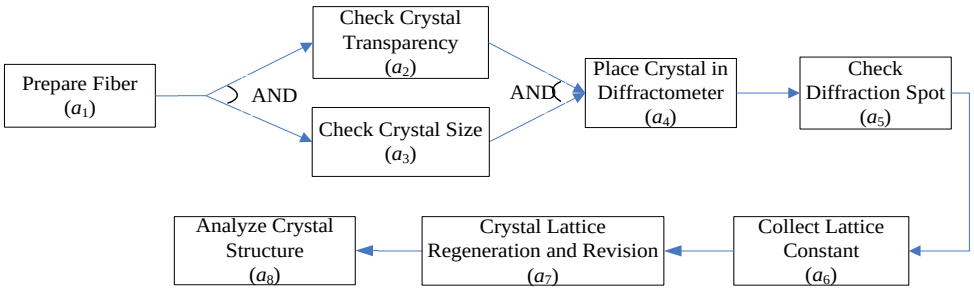

Figure 3.The activity sequence of single-crystal X-ray diffraction

There are eight activities in these phases and they should be performed as a certain sequence. The process model is shown in Fig. 3.

To perform every activity, different tools are needed. The location of user, location of tools as well as their states can be sensed and detected, which are used to determine the activity. For example, in selecting crystal phase, in order to make sure that the crystal is of good quality and suitable size, an optical microscope is used. The crystal is mounted on the top of the glass fiber fixed in the brass pin, and ocular lens of two types are needed to check crystal transparency and size. Therefore the state of microscope and location of ocular lens are contexts related to the activity model.

There are 19 types of contexts related to the activities in experiment. In experiment process not all the contexts are related with current activities. Their value remains static until it is involved. Table 1 shows default value of each context. For simplicity, if the context value is default value in atomic context pattern, it is ignored in context pattern and context evolving pattern expression.

Table 2 is the activity model of each activity in experiment process.

The activity models shown in Table 2 satisfy the rules in 3.1, therefore each activity and its state can be distinguished to each other. For instance, *Check Crystal Transparency* and *Check Crystal Size* are two activities which have possibility to be selected to execute after *Prepare Fiber* have conducted. The start flag of *Check Crystal Transparency* is (**OcularlensI_Loc**, *Microscope*) $\wedge$ (**OcularlensII_Loc**, *Ocularlensbox*), where the context **OcularlensII_Loc** is *Ocularlensbox* is ignored because of default value. On the contrary, the start flag of activity *Check Crystal Size* is (**OcularlensI_Loc**, *Ocularlensbox*) $\wedge$ (**OcularlensII_Loc**, *Microscope*). This satisfies Rule 1 and ensures that the two activities can be distinguished.

TABLE I.
THE DEFAULT VALUE OF EACH CONTEXT

No	Context	Default value
1	**User_Loc**	*Room 101*
2	**Copperbar_Loc**	*Lab Cabinet*
3	**Slicer**	*Off*
4	**Microscope**	*Off*
5	**OcularlensI_Loc**	*Ocularlensbox*
6	**OcularlensII_Loc**	*Ocularlensbox*
7	**Adjustmentscrew**	*Off*
8	**Temperature**	*20℃*
9	**Voltage**	*0KV*
10	**Controller**	*Off*
11	**Crystalcenter_Loc**	*Screenleft*
12	**Diffractometer**	*Off*
13	**SpotResult_Bool**	*False*
14	**Matrix**	*Off*
15	**Saint**	*Off*
16	**Shelxt**	*Off*
17	**ConstantResult_Bool**	*False*
18	**RevisedResult_Bool**	*False*
19	**AnalyzeResult_Bool**	*False*

Ⅴ. CONCLUSION AND FUTURE WORK

Activity recognition is one of issues in ubiquitous computing research. In this paper, we propose an approach to accomplish activity recognition in ubiquitous learning environment. We first present the concepts of context pattern and context evolving pattern. The activity model is defined by its running pattern, start flag, end flag and suspend flag set which are in forms of context pattern or context involving pattern. Considering the sequence constraint by the activity flow at the same time, we analyze the condition the model should satisfy to distinguish the activity and its different states. The condition can be expressed by four rules. We develop an activity recognition algorithm based on this model. According to the approach, we design activity models in single-crystal X-ray experiment support system and further interpret our method through this example.

The limitation of our approach is that the activity to be recognized should have apparent starting and ending signs. This is very strict condition to many applications. Another limitation is uncertainty has not been considered which exists in many real situations. These problems should be taken into account in future research work.

TABLE II.
THE CONTEXT PATTERN OR CONTEXT EVOLVING PATTERN

Activities	The state of activity	Context pattern or the context evolving pattern
Prepare Fiber	$\mathcal{P}_{start}$	(**User_Loc**,*Room 101*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room 101*)∧#(**Copperbar_Loc**,*Cuttingmat*) ∧*(**Slicer**,*On*)
	User is interrupted	(**User_Loc**,¬*Room101*)
	$\mathcal{P}_{end}$	(**Copperbar_Loc**,*Styrofoam*)
Check Crystal Transparency	$\mathcal{P}_{start}$	(**OcularlensI_Loc**,*Microscope*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room 101*)∧#(**Microscope**,*On*) ∧#(**OcularlensI_Loc**,*Microscope*) ∧*(**Adjustmentscrew**,*On*)
	Microscope is off	(**Microscope**, ¬*On*)
	$\mathcal{P}_{end}$	(**OcularlensI_Loc**,*Ocularlensbox*)
Check Crystal Size	$\mathcal{P}_{start}$	(**OcularlensII_Loc**,*Microscope*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room 101*)∧#(**Microscope**,*On*) ∧#(**OcularlensI_Loc**,*Microscope*) ∧*(**Adjustmentscrew**,*On*)
	Microscope is off	(**Microscope**, ¬*On*)
	$\mathcal{P}_{end}$	(**OcularlensII_Loc,** *Ocularlensbox*)
Place Crystal in Diffractometer	$\mathcal{P}_{start}$	(**User_Loc**,*Room 102*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room102*)∧#(**PCI**,*On*)∧#(**Temperature**,[*10*,*25*) ∧ #(**Voltage**,*50KV*) ∧ #(**Copperbar_Loc**,*Diffractometer*) ∧ *(**Controller**,*On*)
	Temperature is out of range	(**Temperature**, ¬ [10,25])
	$\mathcal{P}_{end}$	(**Crystalcenter**,*Screencenter*)
Diffractometer Spot Check	$\mathcal{P}_{start}$	(**Diffractometer**,*On*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room102*) ∧ #(**Voltage**,*50KV*) ∧ #(**Diffractometer**,*On*) ∧ #(**Copperbar_Loc**,*Diffactometer*)
	Voltage is wrong	(**Voltage**, ¬ *50KV*)
	$\mathcal{P}_{end}$	(**SpotResult_Bool**,*True*)
Lattice Constant Collection	$\mathcal{P}_{start}$	(**Matrix**,*On*)
	$\mathcal{P}_{on}$	# (**Matrix**,*On*) ∧ #(**User_Loc**,*Room102*) ∧ #(***Voltage***,*20KV*)
	Matrix is off	(**Matrix**, ¬*On*)
	$\mathcal{P}_{end}$	(**ConstantResult_Bool**,*True*)
Crystal Lattice Regeneration and Revise	$\mathcal{P}_{start}$	(**Saint**,*On*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room 102*) ∧ #(**Saint**,*On*)
	Saint is off	(**Saint**, ¬*On*)
	$\mathcal{P}_{end}$	(**RevisedResult_Bool**,*True*)
Crystal Structure Analyze	$\mathcal{P}_{start}$	(**Shelxt**,*On*)
	$\mathcal{P}_{on}$	#(**User_Loc**,*Room 103*) ∧ #(**Shelxt**,*On*)
	Shelxt is not in operation	(**Shelxt**, ¬*On*)
	$\mathcal{P}_{end}$	(**AnalyzeResult_Bool**,*True*)

.

ACKNOWLEDGMENT

This research work was supported by the National Natural Science Foundation of China (Grant No. 70771017)

REFERENCES

[1] M. Weiser, "The computer for 21st century," *Scientific American*, vol.261, no.30, pp.94-104, 1991.

[2] H. Ogata and Y. Yano, "How Ubiquitous Computing can support language learning," In *Proceedings of KEST*, pp.1-6, 2003.
[3] G.J. Hwang, C.C. Tsai, and S.J.H. Yang, "Criteria, strategies and research issues of context-aware ubiquitous learning," *Educational Technology and Society*, vol.11, no.1, pp. 81-91, 2008.
[4] G.J. Hwang, T.C. Yang, and C.C. Tsa, "A context-aware ubiquitous learning environment for conducting complex science experiments," *Computers and Education*, vol.53, no.2 , pp.402-413, 2009.
[5] H. Ogata, C. Yin, and Y. Yano, "JAMIOLAS: Supporting Japanese Mimicry and Onomatopoeia Learning with Sensors," *the Wireless, Mobile and Ubiquitous Technology in education*, pp.111-115, 2006.
[6] R. Joiner, J. Nethercott, R, and Hull, J Reid, "Designing educational experiences using ubiquitous technology," *Computers in Human Behaviors*, vol.22, no.1, pp.67–76,2006.
[7] H. Ogata, and Y. Yano, "Context-aware support for computer-supported ubiquitous learning," *the 2nd IEEE International Workshop on Wireless and Mobile Technologies in Education, JhongLi, Taiwan*, pp.27-34, March, 2004.
[8] Y. Rogers, S. Price, C. Randell, and D.S. Fraser, "Ubi-learning integrating indoor and outdoor learning experiences," *Communications of the ACM*, vol.48, no.1, pp.55–59, 2005.
[9] H.C. Chu, G.J. Hwang, S.X. Huang, and T.T. Wu, "A knowledge engineering approach to developing e-libraries for mobile learning," *Electronic Library*, vol.26, no.3,pp.303-317, 2008.
[10] C. Zhu, and W.H. Sheng, "Motion- and location-based online human daily activity recognition," *Pervasive and Mobile Computing*, vol.7, no.2, pp.256-269, 2011.
[11] D. Riboni, and C. Bettini, "OWL 2 modeling and reasoning with complex human activities," *Pervasive and Mobile Computing*, vol.7, no.3, pp.379-395, 2011.
[12] M.S. Ryoo, and J.K. Aggarwal, "Recognition of composite human activities through context-free grammar based representation," *IEEE Conference on Computer Vision and Pattern Recognition, New York, USA*, pp.1709-1718, June 2006.
[13] J. Lester, T. Choudhury, and N. Kern, "A hybrid discriminative/generative approach for modeling human activities," *In Proceedings of the 19th International Joint Conference on Artificial Intelligence, Professional Book Center, Acapulco, Mexico*, pp. 766-772, 2005.
[14] L. Liao, D. Fox, and H. Kautz, "Location-based activity recognition using relational Markov networks," In *Proceedings of the 19th International Joint Conference on Artificial Intelligence, Acapulco, Mexico*, pp.773-778, 2005.
[15] D.W Albrecht, and I. Zukerman, "Bayesian Models for Keyhole Plan Recognition in an Adventure Game," *User Modeling and User-Adapted Interaction*, vol.8, pp.5–47,1998.
[16] E.M Tapia, and S. Instille, "Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor," *the 11th IEEE International Symposium on Wearable Computers, Boston, USA*, pp. 37 – 40, October 2007.
[17] H. Kautz, "A Formal Theory of Plan Recognition and its Implementation," *Reasoning About Plans*, San Francisco: Morgan Kaufmann, 1991, pp.69-125.
[18] R. Kowalski, and M. Sergot, "A logic-based calculus of events," *New Generation Computing*, vol.4, no.1, pp.67-95, 1986.
[19] O. Brdiczka, J.L. Crowley, and P. Reignier, "Learning situation models for providing context-aware services," *Ambient Interaction 4th International Conference on Universal Access in Human-Computer Interaction, Beijing, China*, pp.23-32, July 2007.
[20] L. Chen, and C.D. Nugent, "Ontology-based activity recognition in intelligent pervasive environments," *International Journal of Web Information Systems*, vol.5, no.4, pp.410-430, 2009.
[21] L. Chen, C. Nugent, M. Mulvenna, D. Finaly, X. Hong, and M. Poland, "Using event calculus for behaviour reasoning and assistance in a smart home," *in Proceedings of the 6th International Conference on Smart Homes and Health Telematics, IA, USA*, vol. 5120, pp. 81–89, June 2008.