DEA Training Report

THEORY OF UNIFIED CONTEXT

WenTao SHI

Internship done at the I3S laboratory
Sophia Antipolis (France)
May – September 2004

Supervisors

Stéphane Lavirotte
Diane Lingrand
Jean-Yves Tigli

Résumé

L’informatique contextuelle est un facteur clé des nouvelles applications en ubiquité numérique, et a pour but d’améliorer les interactions homme-machine par la découverte des informations sur l’utilisateur et son environnement (par exemple la position de l’utilisateur, la température, le temps, les gens ou le matériel proche, l’activité de l’utilisateur, etc.)

Dans ce rapport de stage, nous commençons par une étude des différents types de contextes dans les applications existantes afin de parvenir à la définition plus globale d’un contexte unifié. Nous proposons ensuite de traiter les informations contextuelles, en représentant et gérant les entités dans une structure commune. Ceci permet la réutilisabilité de toutes les informations contextuelles dans les différentes applications. Notre théorie de contexte unifié est basée sur la comparaison entre une distance correspondant à une caractéristique du contexte unifié et une zone limite, ce qui permet de déterminer à quel moment un contexte est actif dans l’application. Nous construisons deux méthodes majeures (S-endo et S-exo) pour sélectionner ces moments. Par composition des distances et des méthodes, nous pouvons adapter l’application aux nouveaux besoins.

Cette étude constitue le point de départ de travaux de recherches pour la création d’une plate-forme de développement d’applications utilisant l’approche contextuelle.

Abstract

The context-aware is a key factor for new applications in the area of ubiquitous computing, and it is used to improve the human-computer interaction by discovering more information of the user and his environment (such as user's location, temperature, time, nearby people and devices, and user's activity).

In this training course report, we first review the different types of contexts in the existing applications in order to obtain the most global definition of unified context. We propose then to process the contextual information by representing and treating the entities in a common structure. This allows all contextual information to be reused in the different applications. Our unified context theory is based on comparing the distance of a characteristic of unified context with a limit zone, which determines the moment when the context has an action to application. We build two major methods (S-endo and S-exo) to select the moments. By composition of distances and methods, we can adapt applications to new requirements.

This study is a starting point of research works for the creation of applications development platform using context-awareness.

Acknowledgements

I would like to thank my supervisors Stéphane Lavirotte, Diane Lingrand and Jean-Yves Tigli for their guidance and for their support. I thank Daniel Cheung for his advices.

Special thanks go to the Mainline and Rainbow teams of the I3S laboratory , Michel Riveill, Samuel Weibel and Thierry Viéville.

Table of Contents

Introduction

Since a decade, with the growth of mobile technologies, context-aware applications appeared which can take advantage of their surrounding environment, i.e. using contextual information such as the location of the user, current time, nearby people, nearby devices… This sort of applications aims at enhancing the interaction between the user and the machine by letting the machine more aware about its environment so that it can adapt its behaviour and respond in the best appropriate way to the user. This can be done by giving the user useful information about his personal data, his environment, and scheduling things to do according to the user needs and demands.

Context-aware applications are intended to simplify the interface between the user and the machine; they give the most relevant information to the user with the least demands from him. The knowledge of the application can be divided among several computers, communicating with the most recent technologies such as wireless networks, while all of this is hidden to the user who does not need to be aware of the way the application works. It is to the application to be aware of the user and the information to give him or her the best interaction.

One of the first report of context-aware application is the Active Badge [WHF+92] [SCH+02]. It is a location-based application and tracks people wearing it inside a building equipped with sensors that can receive badge signals. A monitoring application is used to display names of the wearers, tell in which room they are within the building, and to send alarm signals to target badges.

Since then, many other applications have been developed and more and more context-aware computing technologies are tried to be used in media product in everyday life, such like Cyberguide[LKA+96, AAH+97].

The goal of this training course is to go through the different existing context-aware applications to find all contextual information that are used. Then we create a general model for context which can be applied for any context-aware application.

This report is made up of three main parts. In the first part we review many of the existing context-aware applications, in order to see the use of the various contextual information that we can find. In the second part, we will define a common framework for using any context in applications, i.e. using a general model of theory, which is called the theory of unified context. And last, in a third part, we will present the application prototype we have designed to test this theory.

1. State of the Art

1.1 Definitions of Context

Before anything, we might define what is “context” to understand well what are context-aware applications. We have found two definitions for the word “context” in the English online dictionary of Cambridge (Cambridge Advanced Learner’s Dictionary)[1]:

Definition 1:
the situation within which something exists or happens, and that can help explain it.

Definition 2 :
the text or speech that comes immediately before and after a particular phrase or piece of text and helps to explain its meaning.

Some researchers try to formally define context. Schmidt et al. define context as “knowledge about the user’s and IT device’s state, including surroundings, situation, and to a less extent, location” [SAT+99].

Chen and Kotz define context is “the set of environmental states and settings that either determines an application’s behaviour or in which an application event occurs and is interesting to the user.”

However these definitions are not enough precise nor clear to be understood in the computer science domain. That is why there are other definitions of context proposed by people in the context-aware domain, but one is specifically well accepted and very often quoted as a reference. This definition comes from Dey and Abowd in [DA00]:

Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.

We can note that they define too the **entity**, which is a very important notion we will use later.

1.2 Categories of Context

Researchers have also tried to enumerate the different contexts and classify them into different categories. Schilit [SAW94] has made three categories for context, environment, computing, user, and other searchers like Chen and Kotz [CK00] have found very relevant to add a fourth category related to the time:

[1] http://dictionary.cambridge.org

- Environment context: all the information which we can observe around user, just like temperature, location, nearby person of user,etc…

- Computing context: the resources for calculus and memory or imprint etc.

- User context: the state of user, such as the emotion, pressure of blood, location of user, profile of user, etc.

- Time context: a service for history recorder and schedule plannifier.

We will use the same category of context to classify the contexts after.

1.3 Applications of Context

In order to study what are the contexts used in the previous existing applications, we made a list of the most known context-aware applications and describe them a little. Some searchers already done similar work as Chen and Kotz [CK00] and Dey and Abowd [DA00]. We have also classify the contexts of each of these applications according to the four categories of context. The result can be seen in the context table further in this report (see table 1).

1.3.1 Table of the Context awareness applications

In table 1, the context awareness applications are ordered according to the year they were developed. For each application, the contextual information it uses has been classified under the four categories enumerated previously: environment context, user context, computing context and time context. When looking at this table, it is quite clear that any context-aware applications are very focused on the user information (but not all). They are mainly focused on the location and the activities of the user. Also, it is often that applications look at the nearby people and/or material from the user and application that trace history or plan a schedule for the user.

Technology	Laboratory	Year	Application	Environment Context	User Context	Computing Context	Time Context
The Active Badge	ORL	1992	Telephone Receptionist Aid	people nearby	location		history of locations
Teleporting	ORL	1992	X session Teleporting System			nearby X servers / displays	
	University of Washington	1994	Mobisaic Web Browser		user state	URL, Active Documents	automatic updates
	AT&T Bell Laboratories	1994	Shopping Assistant	items in the shop, their location, type, price, …	location, selected items, preferences		
	Future Computing Environments	1996	Cyberguide	locations of interest in the area to visit	location, personal interest	GPS and IR for location	History of visits
	Future Computing Environments	1999	Conference Assistant	conference topics	research interest, location	resources in presentation rooms	conference schedule
	University of Kent	1997	People and Object Pager	nearby people and objects	location	Active Badge	
	University of Kent	1998	Fieldwork	everything	location		current time
	Technology for Enabling Awareness, Starlab	1999	Adaptive GSM phone and PDA	nearby people, light level, pressure	activity		
	MIT Media Laboratory	2000	Office Assistant		purpose of the visit	pressure mats	Schedule of the office owner
ComMotion	MIT Media Laboratory	2000	Location-aware Information Delivery		location		current time
Rome Project	Stanford	2000	Location-aware Information Delivery		location		current time
CybreMinder	Georgia Tech	2000	Location-aware Information Delivery	Weather, people nearby	location		current time
MemoClip	TecO	2000	Location-aware Information Delivery		location		current time
Cyberguide		1997	Virtual Guide	Exposition objects location	location		
Cyberguide		2002	Virtual Guide	Exposition objects location, other visitors	location	Central server	
EasyLiving	Microsoft	2000	Indoors Teleporting		location	Cameras, computers and display devices	
Modular approach		2002	Colaborating work		activity	Three cameras	
		2002	E-Supermarket	Items in the supermarket	Current selected items, items bought	RFID labels	

Table 1 – Applications and their Contexts

1.3.2 Hierarchical Tree of Contexts

In another point of view, we can classify these contexts we have found in a hierarchical tree according to the four categories. Some contextual information could be detected or identified by vision detection, with appropriate camera and algorithms; they are noted with a little green disc at their side.

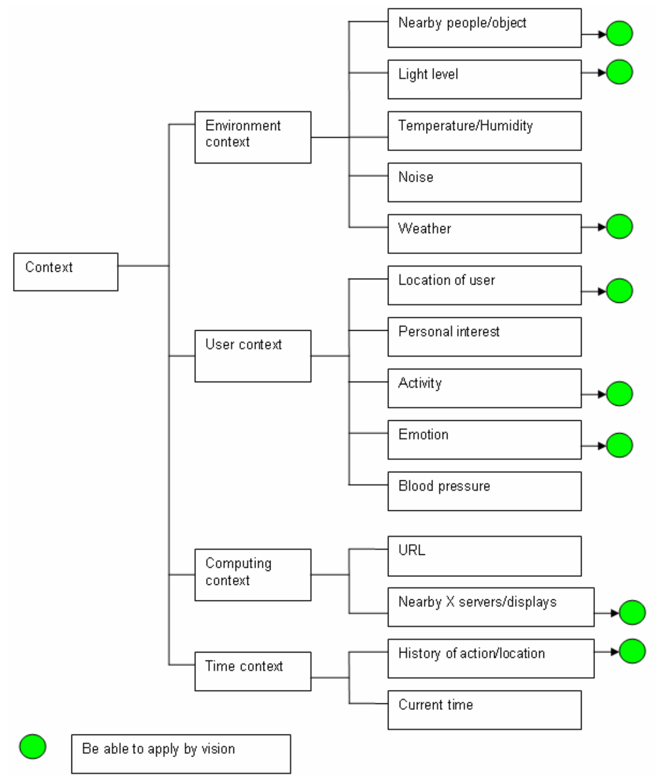


Figure 1 – Hierarchy Tree of Contexts

1.4 Context Awareness

1.4.1 Category of Application of Schilit[SAW94]

Context-awareness [SAW 94] is the ability of a program or device to sense various states of its environment and itself. But how to effectively use that information is still a challenging problem for application programmers. Schilit defines context-aware computing by categorizing context-aware applications as follows [CK00]:

1 . *Proximate selection*, a user-interface technique where the objects located nearby are emphasized or otherwise made easier to choose.

2. *Automatic contextual reconfiguration*, a process of adding new components, removing existing components, or altering the connections between components due to context changes.

3. *Contextual information and commands*, which can produce different results according to the context in which they are issued.

4. *Context-triggered actions*, simple IF-THEN rules used to specify how context-aware systems should adapt.

1.4.2 Our Category of Application

From our definition of unified context in the previous section, we give two definitions of context awareness application:

The first category, *automatic context adaptation*, an application automatically reacts depend on discovered context. Usually, this kind of application can use context to propose appropriate selections of actions to the user or reconfigure the system on behalf of the user according to context changes. Examples include: a camera that captures an image when the user is startled as sensed by biometric sensors(Healey & Picard, 1998); the Teleport system in which a user's desktop environment follows her as she moves from workstation to workstation (Want *et al.*, 1992) and the same work which has been done in Olivetti Research Laboratory [BRH94]; car navigation systems that recompute driving directions when the user misses a turn (Hertz, 1999), also in Active Map, see Annexes (A.2), the system updates the location of people; a recording whiteboard that senses when an informal and unscheduled encounter of individuals occurs and automatically starts recording the ensuing meeting (Brotherton, Abowd & Truong, 1999), the likely work is Conference Assistant, which is done by Future Computing Environments (FCE) at the Georgia Institute of Technology [DFS+99] (A.4); mobile devices enhanced with sensors that determine their context of use to change their settings and actions (Harrison *et al.*, 1998, Hinckley *et al.*, 2000, Schmidt *et al.*, 1999); and devices that deliver reminders when users are at a specified location (Beigl, 2000, Marmasse & Schmandt, 2000).

The second category, *context attachment*, an application capture data with relevant context information. For example, a museum guide which send the introduction of the picture that the visitor looks at [PCB 04]; a zoology application tags notes taken by the user with the location and the time of the observation (Pascoe, Ryan & Morse, 1998). The informal meeting capture system mentioned above provides an interface to access informal meeting notes based on who was there, when the meeting occurred and where the meeting was located. In a similar vein are Time-Machine Computing (Rekimoto, 1999) and Placeless Documents (Dourish *et al.*, 2000), two systems that attach context to desktop or networked files to enable easier retrieval. Some of the more complex examples in this category are memory augmentation applications such as Forget-Me-Not (Lamming & Flynn, 1994) and the Remembrance Agent (Rhodes, 1997).

Now, we survey previous research work about context awareness, focusing on applications. We give a few example in details, and we can study many other applications in the Annexes to see what context they use, and how contextual information is leveraged.

1.4.3 Examples of applications

Teleporting System [BRH94]

The Teleporting System permits 'mobile application', where the "user interface is able to be dynamically mapped onto the resources of the surrounding computer and communications facilities". The computers "follow" the user. The system introduces a level of indirection between the processing of an application and its interface, so that the interface can be created on any display. The system works with X window and the concerned applications are clients of X servers. Instead of using a real X server, the teleporting system uses a special X server known as a proxy server. This server allows only one command, teleport, along a user name and password, which transport session to a X workstation. Even several users can teleports their resources on the same display, allow easy sharing.

The Teleporting System can be used with the Active Badge, making it automated. As the badge provides the location of its owner, the X session of this user can them be teleported to a X server where s/he currently is. The badge provides unique user information identification, and prevents the user to do it manually. Two buttons on the badge are used for selecting a X server within the room and for initiate teleporting. The both systems can allows user to teleport their sessions at home, as they left them at the office.

Mobisaic [VB94]

"Mobisaic is a World Wide Web information system designed to serve users in a mobile wireless computing environment". It allows dynamic contextual information within web documents. Documents refer and react to change of this contextual information. Active Documents automatically updates.

Mobile computing contexts are represented by dynamic environment variables. These variables are associated with users and locations. The user can update these variables with application that can access the environment. The client cannot.

Dynamic URLs (Uniform Resources Locators) allow the client to enter one URL that actually depends on the user state.

Shopping Assistant [AGK94]

A device that helps the customer in the shop: it can guide the customer, detail for him items of the shop, help him locating items. It can also compare prices, do some price analysing. The customer can use the device in two modes: a signed up mode, where his profile is stored and actions (or selected items) are tracked, but who can get discounts; the anonymous mode in which the customer keeps his privacy.

1.5 Result and Conclusion

From the definition of Dey and Abowd [DA 00], we can see the context like all the information which can be used to characterize the situation of an entity.

Even, there is numerous applications use the geographical location as the only representation of context to represent physical information such like the position of users or objects. Context is not only location. But how can we use a uniform definition for all type of context, and how can we reuse it in different applications are becoming the challenges for us today.

Most of the existing applications actually use only few context values. The most used ones are location, identity and time. The reason for this probably lies in the difficulty for computer systems to obtain context information and to process it. Most of the applications are prototypes developed in research labs and academic world. There do not yet exist many commercial solutions. The main area where such exists is different kind of location-based services and guides.

Context-awareness means that one is able to use context information. A system is context-aware if it can extract, interpret and use context information and adapt its functionality to the current context of use. The challenge for such systems lies in the complexity of capturing, representing and processing contextual data. To capture context information generally some additional sensors and/or programs are required. To transfer the context information to applications and for different applications to be able to use the same context information a common representation format for such information should exist. In addition to being able to obtain the context-information, applications must include some "intelligence" to process the information and to deduce the meaning. This is probably the most challenging issue, since context is often indirect or deducible by combining different pieces of context information.

2. Theory of unified context

As context-aware information can be very difficult to obtain and process. It is desirable that applications that use context information and context-sensors should be interoperable, which means there is a need for a standardized infrastructure for the exchange of context information. Several researchers have proposed and prototyped general frameworks to support context-aware applications. Schilit's infrastructure for ubiquitous computing [Sch95] is probably the earliest attempt in this direction. In his Ph.D. work he proposed a general architecture, concentrating mostly on location and an active badge infrastructure [SAW 94]. We propose a more general framework where contextual information are contained by entities, structures that are all managed the same way, but that can hold any possible context.

2.1 Models of Contextual Information

There are many types of contextual information used in the various applications and they have too many different representations. Generally, each application has its own way to model the information, so the same information having several representations. This manner prevents the contextual information to be exchanged between the different applications and thus make systems incompatible and information not reusable.

That is why we think it is better to model the contextual information with a unique model that will allow the design of various context-aware applications using this theory. This is the theory of unified context.

2.2 Unified Context

We present here a general model of contextual information representation and processing. Firstly, we want to rewrite the definition of context to our unified context.

The definition from Dey and Abowd [DA 00] clearly states that any information that can be used to describe the situation of an entity is a part of its context.

Pauly, Couderc and Banaitre of IRISA (France) [PCB 04] state that the context of an entity is what surrounds it and then restricts the context to the close entities. Their definition seems wrong to us. But it is a good manner of selection, for example the selection of the entity.

2.2.1 Definition of Unified Context

We define that *unified context is a set of information, which can influence the situation of the entity and is used to calculate the unified contextual zone.*

For example, in a conference system which can inform whether a person attends a meeting which he wants. The system compares the position of the person, which is got from GPS, and the calendar of this person to obtain a result of him. So, in this application, the unified context of this person includes the calendar and the location, but not the health of him, although it will influence his attendance in reality.

2.2.2 Representation of Entity

We give entity a representation by the unified context and it is unified for all kinds of applications. An entity is represented by a set of contextual information and their associated limit functions.

$$e = \{ \{c_1, \dots , c_n\} \in C , \{l_1, \dots , l_n\} \in L\}$$

C is the set of type of context. c_i represents a type of context, such as temperature, 3D position, height, emotion, time, etc… L is the set of function. l_i represents the limit which defines the moment the influence of the context to an entity is active. This limit can be a constant or any function, for example an ellipse. In fact, it's just the edge of the unified contextual zone.

2.3 Definition of Unified Contextual Zone

Secondly, we will define the unified contextual zone. The notations used here come from Pauly et al. [PCB 04]. From them, a context of an entity is defined as a set of elements contained in a spherical zone centered on the entity. This definition is neither like our definition nor like Dey's, it's not complete because of the context can be any kind of information not only a set of elements. But we use his definition of contextual zone.

2.3.1 Definition of Pauly and al. [PCB 04]

The spherical zone is defined in the sense of mathematical ball. Let the space *A* populated of elements of the same type. Let *E* an entity within this space. The context of the entity *E* is defined by:

$$C(E) = \{e_1, \dots, e_n\} \ / \ \forall\ i,\ d\ (E, e_i) \leq L \text{ and } e_i \in A_C$$

C(E) is a subset of elements of *A* and represents the context of *E*. The e_i elements are called *contextual elements*, and are all the elements of *A* that could belong to the context of the entity *E*. A_C is the set of all contextual elements in *A*. Then $A_C \subset A$, and $C(E) \subset A_C$. The context of *E* is represented by a set of elements of A_C which are all at a distance from *E* less than a maximum constant distance *L*.

2.3.2 Balls Definition

In a metric space, a *ball* is a set containing all points within a specified distance of a given point. With the Euclidean metric, if the space is the line, the ball is an interval, if the space is the plane, the ball is a disk. The *open ball* of radius $r > 0$ centred at point *p* is often written as *Br(p)*. This is defined as:

$$Br(p) = \{ x \mid d(x,p) < r \}$$

where *d* is the distance function or metric. The *closed ball* is the same definition including all points of distance equal to r.

$$Br(p) = \{ x \mid d(x,p) \leq r \}$$

2.3.3 Our definition of Unified Contextual Zone

There is no reason to restrict the contextual zone to a ball, and we want to be able to define any possible form. For example, if the space is the plane, the unified contextual zone can be not only a disk, but also a square or any other form. That is why we define the contextual zone limit L depending on E and e_i:

$$Z(E) = \{e_1, \ldots, e_n\} \mathbin{/} \forall\ i,\ d(E, e_i) \leq L(E, e_i) \text{ and } e_i \in A$$

The unified contextual zone Z(E) is defined by all the possible entities of the space A which are at a distance from E inferior to the limit function L. The function L describes the form of the contextual zone.

Unified contextual zone is composed by all the contexts, which is the information useful to calculate the state of entities, to indicate the moment of the influence of each context. So we separate the unified contextual zone into many contextual zones, each contextual zone corresponds a type of context just like time, position, speed, etc…We call them time-zone, position-zone, speed-zone, etc… Each contextual zone can also be separated into several sub contextual zones. For example, we can separate position context into 2D or 3D coordinates subcontext.

2.4 Entities Selection

2.4.1 Two Major Selections

There are two major ways of selecting the entities relevant to an entity E. The endo-selection and the exo-selection.

The endo-selection is defined by:

$$S\text{-}endo(E) = \{e_1, ..., e_n\} \,/\, \forall i,\ e_i \in Z(E) \text{ and } e_i \in A_C$$

The elements e_i belong to the selection of E if they are in the contextual zone of E and that they are contextual elements, i.e $e_i \in A_C$. Ac is the set of all contextual element of the environment.

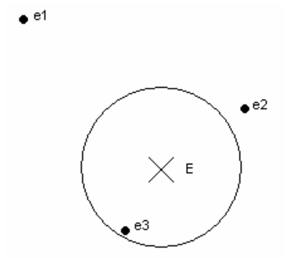


Figure 2 – Endo-selection : entities that belong to the selection of entity E are in its contextual zone.

The exo-selection is defined by:

$$S\text{-}exo(E) = \{e_1, ..., e_n\} \,/\, \forall i,\ E \in Z(e_i) \text{ and } e_i \in A_C$$

An element belongs to the context of E if its contextual zone contains E. As a consequence, E is found in the partition of A defined by:

$$E \in Z(e1) \cap Z(e2) \cap \ldots \cap Z(en)$$

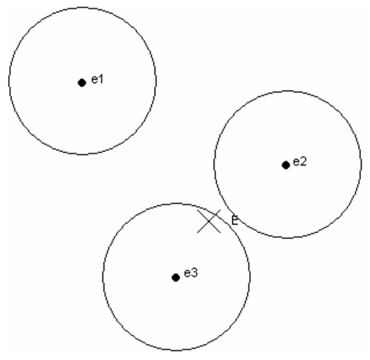


Figure 3 – Exo-selection : entities that belong to the selection of entity E contain E in their contextual zone.

We can use these two definitions to make combinations and create other way of selecting entities. For example:

S-endo (E) or *S-exo (E),*
S-endo (E) and *S-exo (E)*,
S-endo (E) and not *S-exo (E)*, etc…

2.4.2 Bilateral Selection

Among the most interesting combinations, we would point the bilateral selection, that we define for an entity E as:

$$S\text{-}bilateral(E) = S\text{-}endo\ (E) \cap S\text{-}exo\ (E).$$

2.4.3 Example of the Museum: Visitor and Painting

We illustrate the interest of the bilateral selection with a simple example of the guide of a museum. In this application, we will send the interpretation of the picture which the visitor is looking at. We have naturally a definition of what is the context of the visitor: his context is the distance and the angle between him and the paintings. Then we must define how we get this context. First, we can state that the contextual zone of the visitor is his sight field: he can potentially look at any painting within his sight. This particularly defines a part of a disc with a certain opening angle φ and a certain *radius*. This sight unified contextual zone obviously depends on the direction and the distance limit of visitor's sight. Then we might want to define a similar contextual zone for the painting, where the signification is the zone from where a visitor can "face" the painting.

With these conditions, it appears that a visitor can actually "see" a painting if the painting is in his context and if himself is in the painting's contextual zone. Let's see some examples to that demonstrate the need of the bilateral context:

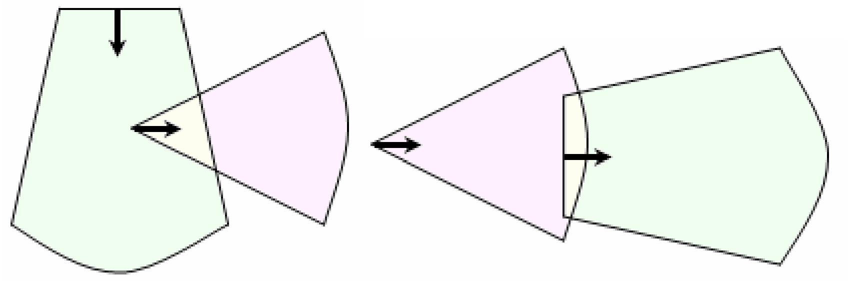


Figure 4a, b – Two examples of non-bilateral context in sight

The visitor sight of context is coloured in pink while the painting sight of context is coloured in green. In the first example (figure 4a), the visitor is indeed within the painting sight, but the painting is not in the sight of the visitor, since he does not look in its direction. On the second example (figure 4b), that is the visitor who has the painting in his sight, but the face of the painting does not show up in the direction of the visitor, as if the visitor sees the back of the painting. In both cases, we can say the painting cannot be elicited as a possible context choice for the visitor since the conditions for it are not satisfied.

A solution to that problem is to consider *bilateral selection*. That means that both elements must be in the possible context of the other. In the case of the user and the painting, it gives the following configuration:

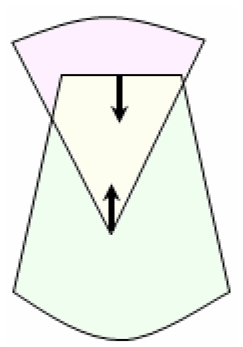


Figure 5 – Bilateral contexts between visitor and painting

Now the user can see the painting and the painting faces the visitor: this is the condition we want to say that the visitor is looking at a painting.

2.5 Calculus Model

Now, we discuss how we can calculate the unified contextual zone. After the development of electronic technology, from hundreds and thousands kind of sensors which can observe the information and transfer them into numerical data, we can integrate numerical information to get the exact context (meta-data) that we want. Here we call the primitive data, obtained by the sensor or to integrate to get the meta-data which is the exact context of application, subcontext. Thus, all kinds of context, we can get it by analyse the numerical context or subcontext that we can get. Normally, we compare these subcontexts with a model, which we have defined depending on every application. So the difference of the value will decide which context it is, that is the technology of recognize. For all these reason, we can find that to define a context is to calculate the distance between the entities or the distance between the subcontext and the defined mode.

So, first of all, let’s look at the definition of a distance.

2.5.1 Definition of Distance

Let a space *E*, a distance *d* defined on *E* is an application of $E \times E$ within R^+ as followed:

$$d : E \times E \to R^+$$
$$(x,y) \to d(x,y)$$

The distance function must satisfy the following conditions:
$\forall$ x,y,z $\in$ to E,

(1) d(x,y) $\geq$ 0 (positivity)
(2) d(x,x) = 0 (reflexivity)
(3) d(x,y) = d(y,x) (symmetry)
(4) d(x,z) $\leq$ d(x,y) + d(y,z) (triangle inequality)

2.5.2 Functions of Distance

There are many functions of distance. Here, we give the most common distance functions:

Euclidean $d(x,y) = \sqrt{|x_1 - y_1|^2 + ... + |x_p - y_p|^2}$

Where x = $\begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$ and y = $\begin{pmatrix} y_1 \\ \vdots \\ y_p \end{pmatrix}$; x and y are vectors of dimension p.

Manhattan $d(x,y) = |x_1 - y_1| + ... + |x_p - y_p|$

Minkowski $d(x,y) = \left(|x_1 - y_1|^q + ... + |x_p - y_p|^q\right)^{\frac{1}{q}}$

From the function like above, Minkowski is often used to calculate the total distance of several dimensions between the entities. Here each dimension represents a field of context, and a field of context could have perhaps many dimensions.

2.5.3 Separation of Space

Since the unified context is composed by several contexts, we separate the unified contextual zone into many contextual zones, each contextual zone corresponds to a type of context. These separated zones can be time-zone, position-zone, speed-zone, etc… Each contextual zone can also be separated into several sub contextual zones.

As we separate the unified contextual zone, we need to separate the spaces in order to measure whether each type of context is active[2] in applications.

For example, split the space A in two subspace A_1 and A_2. Let the unified context C of entity E be split up into $\{ c_1 \ \ c_2\}$ where c_1 is the part of C in A_1 and c_2 the part of C in A_2. The same for an entity E' = $\{ c_1' \ c_2'\}$. Then give distance functions d_1 (.) and d_2(.) to these subspaces. We can then write:

$$d_1^2 (c_1,c_1') = (c_1' - c_1)^T M_1 (c_1' - c_1) \leq L_1^2$$

$$d_2^2 (c_2,c_2') = (c_2' - c_2)^T M_2 (c_2' - c_2) \leq L_2^2$$

Where M are matrices of weights. Then the resulting distance in A is:

$$d^2 (C,C') = (C' - C)^T M (C' - C) = \alpha d_1^2 + \beta d_2^2$$

$$M = \begin{pmatrix} \alpha M_1 & 0 \\ 0 & \beta M_2 \end{pmatrix}, \alpha + \beta = 1$$

The coefficients αM_1 *and* βM_2 represent the importance of each type of context in the unified context of every application.

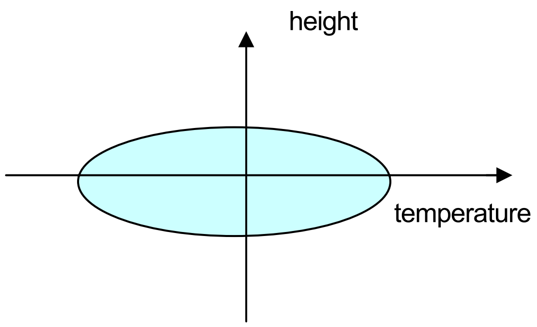


Figure 6 –Context zone composed by temperature and height

In Figure 6, we give an example of a space which is separate in two dimensions, one is temperature, the other is height, so by using the function Minkowski with coefficients we can calculate the distance between two objects by calculating the difference of temperature and the difference of height.

[2] A context is active means the context is influencing the behaviour of the context-aware application.

3. Application description and simulation

3.1 Presentation

We propose to verify the theory of unified context with a prototype application. We have designed an application where the user chooses objects in his environment by selecting them with his finger. The application locates and simulates the finger and the objects into its system and display them on a feedback screen. The environment in our case is supposed to be a two dimensional plane, like a table or a desk where objects are placed on it. We represent the finger in the system by two points, allowing it to have a direction, useful for pointing at objects. This application runs in three different scenes: *virtual simulation scene*, *static image environment scene* and *video camera scene*. Each mode brings its own advantages and drawbacks, but all are complementary to verify the theory of unified context we have integrated in the system. Here is a description of each of these modes:

Virtual simulation scene: in this scene, all is virtual and the objects are represented by geometrical forms, possibly randomly placed on the environment. The objects can have different size and contextual zones that depend on their form. The finger is also virtual and is represented by a red spot and a green spot. The green spot correspond to the location of the finger and the red spot is used to compute the direction angle of the finger. It is controlled by the mouse in the application. The user can move the finger everywhere and turn it in any directions. This is simplest mode to test our theory.

Static image environment scene: this scene is used to set the objects environment from a photograph; thus the photograph must represent a scene with objects over a plane, like stuff on a desk for example. The system will extract the object forms from the image by doing a sequence of image processing. Basically we do an edge detection to find the contours of the objects, and then we do a morphological dilatation to have more connected forms and possibly to fill up some holes in the forms. Then each form is labelled and an object is assigned to each label. As this is an automatic process we can’t guarantee that the extracted objects are totally exact, and we also may find little objects due to the noise in the image. Such objects can be simply rejected. The finger is still virtual and controlled by the mouse in this scene, and it can now point and selects objects from a real image.

Video camera scene: this is the most complete scene as it uses the features of the two previous scenes while adding real finger detection with a video camera. Finger in the system is now controlled by the user finger, with the help of two bright spots on his finger in order to do the detection. We will detail the material used and the processing later.

Moreover, the application contains three rules of entities selection method to determinate how the finger can select the objects in the environment. The *endo-selection* is the most evident rule where an entity E belongs to the selection of the finger if the entity E is within the contextual zone of the finger. The second method is the *exo-selection* where an entity A belongs to the entity E if the E is within the contextual zone of A. This definition is less natural. At last we propose the *bilateral selection*, where the two entities A and E must be both in the zone of the other.

3.2 Hardware and Software

We have built the application on a Intel Pentium laptop equipped with a processor 1300 MHz and 256 Mb of RAM. It runs under the Microsoft Windows 2000 exploitation system. We use the QuickCam® video camera for Notebooks Pro.

The application has been programmed with language C++ using Microsoft Visual Studio. For image processing, we have used the Intel Image Library OpenCV[3].

3.3 Architecture

Our application is based on a simple simulator running the information processing and other modules to provide the input and the output of the simulator. The inputs come from either still images or video images (live camera or recorded video). There is a special Image Processing module integrated in the system to convert the input images into information that the simulator system can use, i.e. locations and forms of the objects and position of the captured finger if any.

At another side, there is a strong link with the interface (done in a dialog box using the MFC), where interaction between system and user can be done. In the virtual scene, the user can then control the finger with the mouse; he can also set some parameters of the contextual zone of his finger (angle of vision, maximum distance of sight).

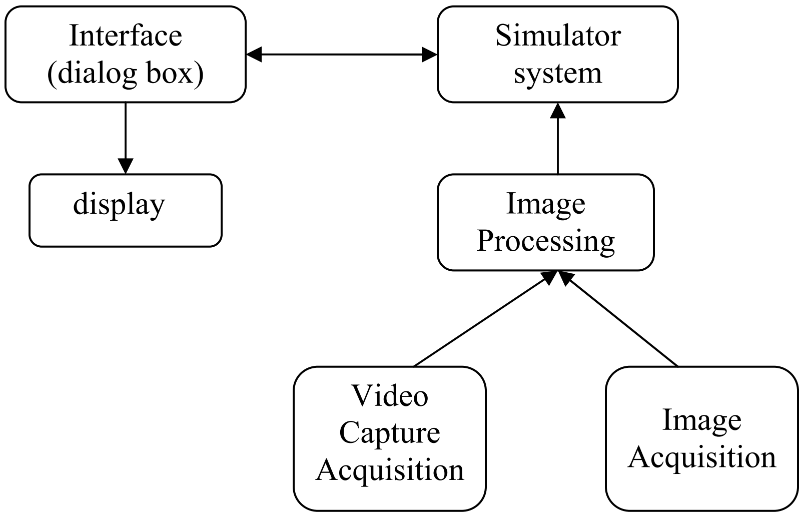
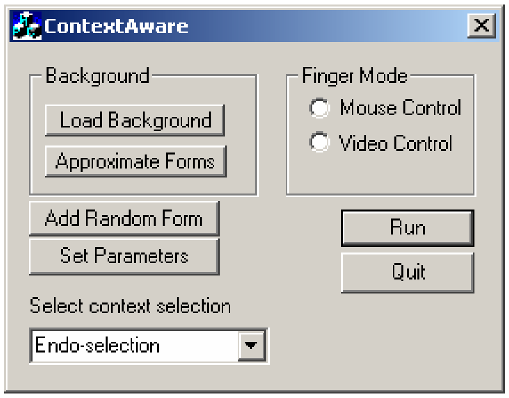


Figure 7 – Architecture of the application

[2] http://www.intel.com/research/mrl/research/opencv/

3.4 Interface

The interface is based on a dialog box that display the main possibilities of the application. It provides a way to choose the selection rule for the entities. The three major rules as described in the theory are selectable: endo selection, exo selection and bilateral selection. It allows to load an image for extracting objects from it, and approximate ellipse from the found forms.



Select context selection

Endo-selection

Figure 8 – Interface of the application

The interface also shows a screen containing the objects of the simulation system. These objects can be provided either from real image or random generation. Whatever, they are displayed the same way on the feedback screen. The background is set to black and the objects are represented in grey. The contextual zone of each object is delimited with a dark green line. In the case of virtual objects, the form of the zone depend on the type of the object form. If objects are coming from an image, the button approximation can be used to convert any objects found in the image into an ellipse with its contextual zone associated the form of the objects.

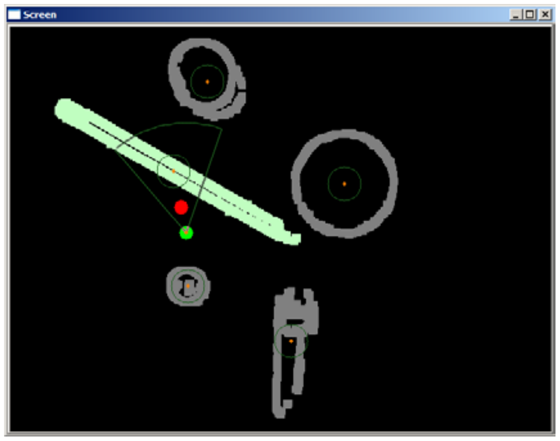


Figure 9 – Screen of the feedback display of the system

3.5 Representation

The object representation is based on the entity representation. As said in the theory of unified context, an entity contains both *properties* and *functions* (zone properties) to compute its contextual zone.

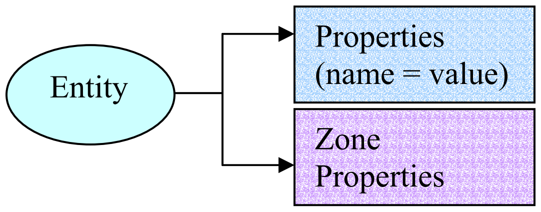


Figure 10 – Entity representation

3.5.1 Properties

The properties are sets of associated names and values. Names are used to call the contexts and the values correspond to these. For example, we can have the context “temperature” with the value “18”, this correspond to one property. A context is not necessarily restricted to one value. The context “2d location” can have the two values x and y.

3.5.2 Zone Properties

The zone properties are sets of functions used to compare elements of a same context. It generally consists of a limit function L associated for each context. As it is a limit function, we also need to know the function to compute distance between two elements of the same context. This function is a distance function D, and there is one associated for each context. But, contrarily to the limit function, there is one function D per context, whereas the functions L may differ for each entity. We can group the functions D of each context in a Metric System.

For example the entities have the two contexts “temperature” and “2d location”. A conceptual representation of the entity seems like this drawing:

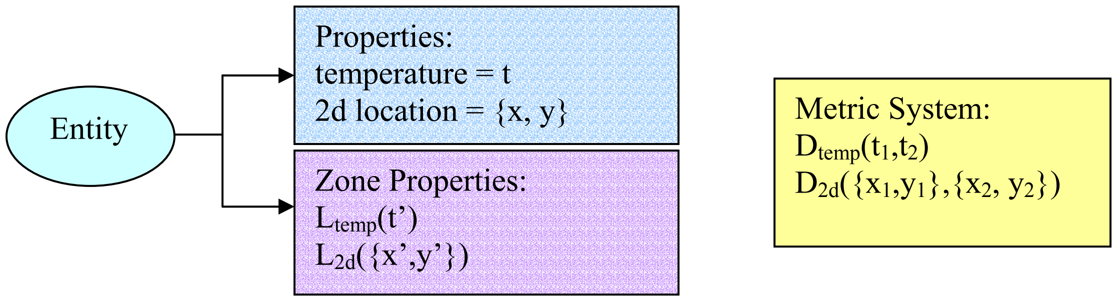


Figure 11 – Example of entity representation

If we consider another entity, it will have different properties (different values for t, x and y), different limit functions (not necessarily but possible) but will use the same distance functions, those defined in the metric system. We can also note that for each context within an entity, there are value(s) and a limit function. We can now move into a better programmable object, using the Context Field.

3.5.3 Representation of Entity

Because of the unified context and the generality of the model we use, we can define that a context field is a set of values, a limit function and a name for a context. Of course between several entities, the name of a context never change. We can see this representation in the following figure.

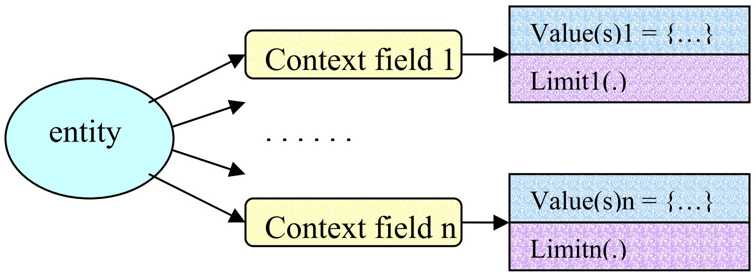


Figure 12 – application of entity representation

In this manner, we can well separate the context information between the different parameters. So the different contextual information will be hold in the context field structure, but data will differ for each context. In the program, it is represented by a class ContextField that sets the interface between the entity and the further data. Actually, we use polymorphed classes of ContextField to add specific data such as the context properties and obect limit function. For example the data structure for an entity using a temperature and a two dimensional location contexts are represented in the program following this diagram:

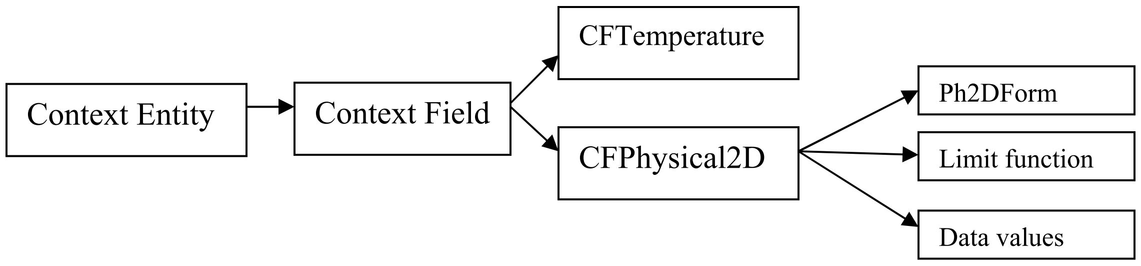


Figure 13 – Architecture of context entity

Here is the corresponding class representing the entity. An entity is first set with no context field; they must be added to the entity by the programmer, giving a name for the context. In our model, the entity has an unlimited number of context fields. The programmer of the future application can add as much contexts as he or she wants. Context fields are then identified by their name, so there is no particular order to enter the context fields within an object.

```
class ContextEntity
{
private:
  unsigned int n_ContextFields;

public:
  ContextField** fields;
  ContextEntity();
  virtual ~ContextEntity();
  bool AddContextField(ContextField* cf);
  void Display(IplImage* img, int flag);
  bool IsInZone(ContextEntity* e);
};
```

3.5.4 Contextual Zone

The contextual zone is defined by the limit function. The simplest case is the case of a spherical zone. The limit is a constant, thus the function will be of this sort:

```
double Ph2DForm::Limit(Ph2DForm* form)
{
  return constant_limit;
}
```

The Ph2DForm form contains the specific data of the entity we compare to.

Another limit function example can be the function that defines the contextual zone of the finger. It is based on the distance and the angle between the finger and the entity to compare to.

```
double Ph2DFormSelectionFinger::Limit(Ph2DForm* form)
{
  double anglediff = mAngle(form->Center, this->Center);
  if (anglediff > (this->apertureAngle + this->directionAngle) ||
   anglediff < (this->directionAngle - this->apertureAngle))
   return -1;
  return this->sight;
}
```

3.5.5 Selection Mode

We discussed previously that there are three modes implemented for the selection method.

There is one important function within the entity class `IsInZone` which tests if an entity is in the contextual zone of another entity. The result is a Boolean. The function is declared as:

```
bool ContextEntity::IsInZone(ContextEntity* c)
```

The use is simple and for example if one want to test if the entity e is in the contextual zone of the entity finger, it must be called like this :

```
e->IsInZone(finger))
```

Here is now the sample code that go through all entities of the system and checks if they are selected by the finger, according to the three different selection methods:

```
unsigned int n = ceEntitiesListGetNumber();
unsigned int i;
int fflag = 0; // fingerflag
for (i = 0; i < n; i++)
{
 int flag = DISPFLAG_CONTEXTUALZONE;
 ContextEntity* e;
 e = ceEntitiesListGet(i);
 // our selection method is between finger and other entities.

 if (finger != NULL && can_redraw)
 {
  if (selectionMethod == SELECTION_METHOD_ENDO)
  {
   if (e->IsInZone(finger))
    flag += DISPFLAG_SELECTED;
  }
  else if (selectionMethod == SELECTION_METHOD_EXO)
  {
   if (finger-> IsInZone (e))
   {
    flag += DISPFLAG_SELECTED;
    fflag += DISPFLAG_SELECTED;
   }
  }
  else if (selectionMethod == SELECTION_METHOD_BILATERAL)
  {
   if (finger-> IsInZone (e) && e-> IsInZone (finger))
   {
    flag += (DISPFLAG_SELECTED + DISPFLAG_BILATERAL);
    fflag += (DISPFLAG_SELECTED + DISPFLAG_BILATERAL);
   }
  }
 }
e->Display(imgSynthesis, flag);
}
```

The result of this function is to display the entities with different display parameters (the flags). In this sample, two flags can be set, “selected” and “bilateral”. The result of the “selected” flag is to draw the object with a light green color so that they are well distinguished among the other entities. The “bilateral” flag will make the selection color light purple, just to distinguish that is the bilateral selection method.

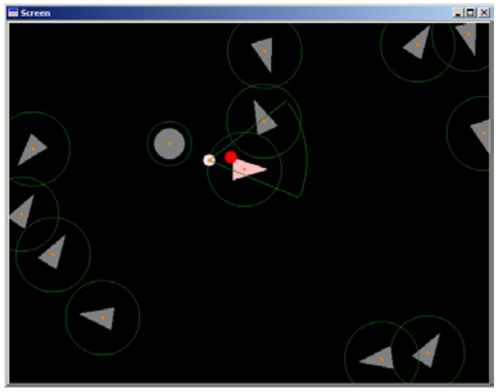


Figure 14 – Example of bilateral selection. Both selected entity is marked with pink color.

3.6 Image Processing

By using the openCV library, we could access many popular image processing functions that we could reuse within our application. Among those functions are edges detection, morphological operations. We added some parts we needed.

3.6.1 Object Extraction from Image

One interesting point was to find objects in the image. First we suppose that the objects won't move in the image, so we want to process only one image to initialize the position of the objects. The image can come from a user image, or from the camera. For our tests, we choose a reference image that was in the samples image provided by the openCV library. This image is called "stuff" and depicts several items on a desk, including a pencil, a coin, a lighter and two round shaped objects. The objects are enough far each from the others so that they don't touch together. This is a very good image for our application since it corresponds to the sort of situation we are in, and it provides a good image for testing our vision algorithm.

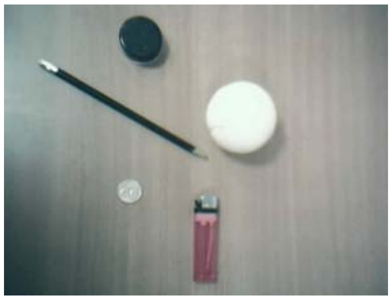


Figure 15 – Stuff, an openCV sample image that we used for extracting objects.

Here is how we process the image to extract the forms: we first compute the edges of the image using a Canny algorithm (openCV function), then we dilate the image using morphological operation with a square element 3*3. This operation is done to close more forms and connect contours that were lost. Of course we cannot assure that the form will be closed, but a connected form is enough for us since we can then label the connected forms in order to differentiate them. By labeling the forms, we can know how many forms there are in the image, and what are these forms. Some forms could be the result of noise on the image. Such forms will be little compared to the "true" forms. Our algorithm treats all forms smaller than a threshold as noisy forms and reject them.

3.6.2 Entity Creation from Contours

We then have to create an entity from the extracted forms of the image. We create an entity with a 2d object context field. We initialize this context field with a "pixels blob" form. This form corresponds exactly to the pixel belonging to a same label. The centroid of this form is computed then the distance from all points to the center is computed to find the "size" of the object.

There is later a function that the user can call to transform all available "pixels blob" forms into ellipses. This function will give best result on simple geometrical forms such as triangles, rectangles, ellipses, …

3.7 Video Camera

Another image processing is the video processing used for detecting the position and the direction of the finger. In order to help the detection, we use two rings with LEDs each place on the finger we'll use for pointing at objects. The LEDs (green and red) are bright enough to mark the captured image with small white spots. We aim to find two bright spots on the images to deduce the position of the two LEDs.

We can decompose an RGB image into three subchannels, any containing a color channel of the original image. Because of the choice of the green and red LEDs, we have a great a priori that tells "if there is a small white spot in the red channel, it must be black for both green and blue channels, corresponding to this place" for detecting the red LED, and the same thing for the green LED. We may have several spots in the detection. In that case, we must choose a spot that corresponds best to the form we want, and which is closest to the previous position of the spot. This manner, we can reduce wrong estimation.

The last operation is to compute the angle between the two LEDs to have the direction of the finger. Then selected entities are marked with red circles over the original image when they are selected.

3.8 Results and Conclusion

The application allows to show the use of the unified context theory. The system has been designed to integrate any possible context field and compute an overall distance between entities of several contexts. We have especially tested it with a location context using a two-dimensional coordinates and a direction angle for objects (when possible). The simulation showed that the theory was good to allow the selection of nearby objects depending on the form of the contextual zones of each objects, including the finger.

We also showed how we can create the contextual information from the image or from the video, by extracting the contexts of the objects (location, form) from image processing.

Conclusion

In this report, we showed a general method for selecting the context information which will influence a context-aware system. This method is context-independent and any contextual information is stored within the entities. The selection of information is then done by setting contextual zones for the entities and selection rules that can be combined. The great advantage of our method is that any sort of context can be represented and used in one model, allowing a framework for any possible context-aware application.

In order to demonstrate our theory, we built fast a simple context-aware application prototype simulating a painting gallery where the user selects painting by his location, his finger direction and personal interest on paintings.

Our theory offers the possibilities to be extended and provides more complex intelligent selection methods in the future while keeping the generality of the theory of unified context.

References

[AAH+97] Abowd, G. D., Atkeson, C. G., Hong, J., Long, S., Kooper, R., and Pinkerton, M. Cyberguide: A mobile context-aware tour guide. Wireless Networks, 3(5):421-433, October 1997.

[AGK94] Asthana, A., Cravatts, M., Krzyzanowski, P. An indoor wireless system for personalized shopping assisstance. In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications, pages 69-74, Santa Cruz, California, December 1994. IEEE Computer Society Press.

[BBC97] Brown, P. J., Bovey, J. D., and Chen, X. Context-aware applications: from the laboratory to the marketplace. IEEE Personal Communications, 4(5):58-64, October 1997.

[BRH94] Bennett, F., Richardson T., Harter, A. Teleporting – Making Applications Mobile. 1994.

[Bro98] Brown, P. J. Triggering information by context. Personal Technologies, 2(1), March 1998.

[CK00] Chen, G., Kotz, David. A Survey of Context-Aware Mobile Computing Research, Technical Report, Dept. of Computer Science, Dartmouth, November 2000.

[DA00] Dey, A. K. and Abowd, G. D. Towards a better understanding of context and context-awareness. Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness, affiliated with the CHI 2000 Conference on Human Factors in Computer Systems, New York, NY: ACM Press.

[DFS+99] Dey, A. K., Futakawa, M., Salber, D., and Abowd, G. D. The Conference Assistant: Combining Context-Awareness with Wearable Computing. In Proceedings of the 3rd Internatinal Symposium on Wearable Computers (ISWC '99), pages 21-28, San Francisco, CA, October 1999. IEEE Computer Society Press.

[MS00] Marmasse, N. and Schmandt,C., Location-aware information delivery with ComMotion. In Proceedings of Second International Symposium on Handheld and Ubiquitous Computin, HUC 2000, pages 157-171, Bristol, UK, September 2000. Springer Verlag.

[LKA+96] Long, S., Kooper, R., Abowd, G. D., and Atkeson, C. G. Rapid prototyping of mobile context-aware applications: the Cyberguide case study. In Proceedings of the Second Annual International Conference on Mobile Computing and Networking, pages 97-107, White Plains, NY, November 1996. ACM Press.

[Pas98] Pascoe, J., Adding generic contextual capabilities to wearable computers. In Proceedings of the Second International Symposium on Wearable Computers, Pittsburgh, Pennsylvania, October 1998. IEEE Computer Society Press.

[PCB 04] Pauly, J., Couderc, P., Banâtre, M. Synthèse des méthodes de programmation en informatique contextuelle. Publication interne n° 1595, IRISA, janvier 2004.

[PRM98] Pascoe, J., Morse, D., and Ryan, Nick. Developing personal technology for the Field. Personal Technologies, 2(1), March 1998.

[SAT+99] Schmidt, A., Aidoo, K. A., Takaluoma, A., Tuomela, U., Van Laerhoven, K., and Van de Velde W. Advanced interaction in context. In Proceedings of First International Symposium on Handheld and Ubiquitous Computing, HUC'99, pages 89-101, Karlsruhe, Germany, September 1999. Springer Verlag.

[SAW94] Schilit, B., Adams, N., Want, R. Context-aware computing applications. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, pages 85-90, Santa Cruz, California, December 1994. IEEE Computer Society Press.

[Sch95] Schilit, W.N. System Architecture for Context-Aware Mobile Computing, Ph.D. Thesis, Columbia University. 1995.

[Sch+02] Schilit, B. N., et al. 'Context-Aware Communication' Context-Aware Computing. IEEE Wireless Communication Magazine, Vol. 9, No. 5., October 15, 2002

[VB94] Voelker, G. M., Bershad, B. N. Mobisaic: An Information System for a Mobile Wireless Computing Environment. 1994.

[Wei93] Weiser, M. Some computer science issues in ubiquitous computing. Communications of the ACM, 36(7):75-84, July 1993.

[WHF+92] Want, R., Hopper, A., Falcão, V., Gibbons, J. *The Active Badge Location System*. ACM Transactions on Information Systems, January 1992.

[WSA+95] Want, R., Schilit, B. N., Adams, N. I., Gold, R., Petersen, K., Goldberg, D., Ellis, J. R., and Weiser, M. An overview of the PARCTAB ubiquitous computing experiment. IEEE Personal Communications, 2(6):28-43, December 1995.

[WSA+96] Want, R., Schilit, B. N., Adams, N. I., Gold, R., Petersen, K., Goldberg, D., Ellis, J. R., and Weiser, M. The ParcTab Ubiquitous Computing Experiment. In Tomasz Imielinski and Henry F. Korth, editors, Mobile Computing, chapter 2, Kluwer Academic Publishers, 1996.

[YS00] Yan, H., Selker, T. Context-aware office assistant. In Proceedings of the 2000 International Conference on Intelligent User Interfaces, pages 276-279, New Orleans, LA, January 2000. ACM Press.

Annexes

A.1 The Active Badge Location System [WHF+92]

The Active Badge allows to locate its owner within a building equipped with special sensors for this badge. Its main application (a demonstration application) is to help a telephone receptionist in transmitting calls within the building. The receptionist looks for the closest telephone from the person who must receive the call. The application can display the list of people using active badges and their current location. It comes with five principal commands: FIND to locate the current location of a badge, WITH to locate surrounding badges from a target badge, LOOK to find badges near a certain location, NOTIFY to signal a badge, and HISTORY to generate a report of location history of a certain badge.

A.2 Active Map [Wei93, WSA+95, WSA+96]

A system that locates individuals within a building, where each room is equiped with a wireless base station. The system updates the location of each people every few seconds.

A.3 Cyberguide [LKA+96, AAH+97]

A system used to track and helps visits of a tourist, indoors (by infrared) and outdoors (GPS). The tourist can use an interactive map, leave comments on his map, get suggestions of next visits according to his previous visits.

A.4 Conference Assistant [DFS+99]

The assistant examines a conference schedule, the topics discussed and to be presented, so that it can tell to its user-researcher to which presentation he must assist, regarding his personal research interests. The assistant can display the presenter, the title of the presentation and other related information whenever the user enters in a presentation room.

A.5 People and Object Page [BBC97, Bro98]

The pager is based on an active badge system to locate the users within the building. There are two examples of use for this system: when there is a visitor with no pager, a message can be sent to him by transmitting it to the closest person to the visitor who has a pager and then deliver him the message. The other example is to broadcast a request for a certain need which other users could resolve, and who notify the requester when they can help him.

A.6 Adaptive GSM phone and PDA [SAT+99]

Use of an adaptative system for PDA and for a phone. For the PDA, the systems adapt the text font size according to the current user activity and environment: if the user is stopped, the text is small (normal), if the user is walking, the text is big. Also if there is no much light, the text will be bigger than normal. For the phone, the systems can change parameters (such as ringing mode and its volume) according to the environment of the user. It is specified it can adapts whether the phone is outside or inside, in the hand, in the pocket, on the table, …

A.7 Office Assistant [YS00]

The assistant detects when a visitor approaches an office. It interacts with him, and can decide to let the visitor enter or not depending on his identity, the purpose of his visit and the schedule of the owner of the office (owner status). The detection is done by pressure sensitive mats placed on sides of the office door.

A.8 Location-aware Information Delivery [MS00] [HLP+00] [DA00] [Bei00]

Reminders are created with associated location. When the user, recipient of the reminder message arrives at this location, a speech synthesizer delivers the message to him. Such applications are supported by the ComMotion project at MIT [MS00], the Rome project at Stanford [HLP+00], CybreMinder at Georgia Tech [DA00] and MemoClip at TecO [Bei00].

A.9 Cyberguide [AAH+97]

A virtual guide for expositions, which give informations for the users / visitors about the objects of the exposition. The guide is on a PDA possessed by a visitor, and is localised by infrared for indoor location and GPS for outdoor location. All information about the exposition and the guide system are directly stored within the PDA, and as each visitor gets a PDA for their virtual guide, it implies a problem for any update since all guides must be updated each.

A.10 Cyberguide at Equator [MMR+02]

A Cyberguide for physical and virtual visitors. Virtual visitors can do the visit with a Web browser or a 3D display, while real visitors get a PDA which give them the 3D view of the exposition room with contextual information and characters representing the other visitors. The PDA retrieve the information from a central server. The location is done with ultrasound sensors.

A.11 Modular Approach [CCR+02]

Model based on image recognition and designed as modules so that applications are built from assembling of elementary modules. Such modules are physic features detectors, and relation modules. An application using these technics is an application of collaboration work where the users see each other via monitors. Each user is filmed by three cameras: one for his eyes, one for his face and the last for the workspace, view from above. The other users can view him depending on his context, i. e. the system choose the camera view if the user looks at his monitor, or looks aside, or is drawing on his workspace.

A.12 EasyLiving [BMK+00]

This is an intelligent environment where the user environment is displayed on a appropriate display, that can be a video projector, or a keyboard/screen. The user display is moved on the place he is. The user context is captured by several cams in the rooms. The environment and the places of the computing services are modelled in a geometric space.