

# A Scalable and Efficient Design of WebSignSys

UCLA CSD TR # 020048

Jun-Hong Cui

Computer Science Department, UCLA, Los Angeles, CA90095

***Abstract** WebSignSys is an infrastructure which bridges the physical world and the virtual world (World Wide Web). It provides a form of augmented reality by creating, mapping, delivering, and detecting websigns (introduced in [7]), which are defined as hyperlinks from physical locations to web resources. In WebSignSys design, there are many challenges, such as scalability, efficiency, etc. In this report, we will present a scalable and efficient design for WebSignSys.*

**Keywords** Augmented reality, location aware, enhanced environment, mobile computing

## 1. Introduction

Today, the explosion of wireless technologies including mobile computing, small and easily deployable sensors, PDA and smart phones, GPS and location trackers, to name a few, has made augmented reality and enhanced environment very hot areas of research, development, and testing. Most current mobile computing facilities provide mobile users with wireless connection to the Internet, freeing users from their desktops. However, besides this freedom support, it is even more desirable and challenging to provide a smart environment and an augmented reality of the physical world. This issue has been explored by the research program, CoolTown [2].

CoolTown has proposed web presence [3], which aims to connect web resources (represented by URLs) to the real world. It extends the scope of web browsers, enabling them to sense physical entities in the environment and map them onto web resources. The connection between a physical place and a web resource can be created by attaching beacons, RFID tags or barcodes to the physical place and storing and associating an appropriate URL in it. CoolTown demos show many promising applications, such as smart museums, smart conference rooms, smart buses, etc., which will make our lives in 'CoolTown' exciting.

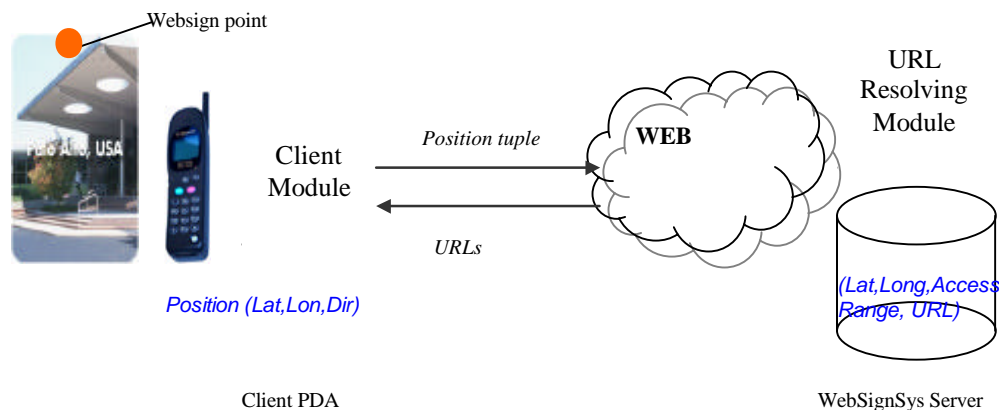
Though CoolTown has achieved good results in initial trials, it faces the critical scalability problem when deployed in wide areas. The current CoolTown beacons use infrared technology to broadcast URLs. However, the beacons are directional and have an access range of about one meter. To receive a URL, users need to point their mobile device in the direction of a beacon and receive a periodic broadcast. If the same beacons are used in a wide-area deployment, for example, to associate a URL with a big building, just imagine how many beacons would be needed so that any user near the building can sense a beacon signal, and how expensive this kind of deployment would be. Besides the hardware cost of beacons, the management cost is also very critical: beacons may run out of batteries and need periodic maintenance; when web resources are updated, associated beacons have to be reconfigured. Solving the scalability problems of physical beacons requires a new solution. [7] introduces websigns, which are defined as hyperlinks from physical locations to web resources, to provide users augmented reality over a wide area. However, a basic client/server model based websign system (a prototype is presented in [7]) will face other kinds of problems, such as efficiency, scalability (in terms of number of websigns), etc. In

other words, a websign system needs scalable and efficient design. For the convenience of discussion, we refer the infrastructure bridging the physical world and the virtual world (World Wide Web) based on websigns as WebSignSys. In this report, we will present a scalable and efficient design for WebSignSys.

The rest of this report is organized as follows. Section 2 reviews the concept of websigns, and briefly describes the basic model of WebSignSys. In Section 3, we state the design challenges. In Section 4, we provide a detailed description of a scalable and efficient design for WebSignSys. Then we discuss applications and related work in Section 5. Finally, we give a short conclusion in Section 6.

## 2. Concept of Websigns

A **websign is a personalized virtual beacon deployed in space and time, which directs intended visitor(s) to an associated web resource [7].** To be concrete, a websign is a three-dimensional point in space that links any person who selects it with a web resource. While it functions like a beacon in CoolTown, it is not a physical beacon; it is a virtual one, represented by software. Unlike the CoolTown beacons, the URL information associated with websigns is not point-broadcast; instead it is resolved based on the users' position and orientation. Websigns have an arbitrary access range, and are activated only when the intended visitor enters their access range. The activated websigns are usually presented to the nomadic user when he/she points his/her mobile device in their general direction. Since websigns are virtual entities, they can be created and deployed directly from a computing infrastructure. Though there are no physical entities to deploy and maintain, websigns still preserve one of the most useful features of beacons: users can point to websigns and click a button to indicate their intention to select a resource.



*Fig. 1 WebSignSys client/server model*

WebSignSys is an infrastructure to create, map, deliver, and detect websigns. To deploy websigns, WebSignSys utilizes a client/server model (see Fig. 1). In this basic model, a websign is represented by a tuple (latitude, longitude, AccessRange, URL). Places, such as restaurants, theaters, shopping malls, etc, can subscribe to WebSignSys servers with their own websign tuples and WebSignSys servers maintain websign databases. When a client PDA is interested in a given place, it sends its tuple position (latitude, longitude, direction), which is obtained by positioning devices such as GPS and compasses, to a proper WebSignSys server. On receiving the client's position tuple, the URL resolving module on

the server side will compute the URLs of the websigns pointed at by the client PDA, and then the server sends the corresponding URLs back to the client. Though a websign can be three-dimensional, to simplify our discussion, we use a two-dimensional space position (latitude, longitude) in this report.

### 3 Challenges in Designing WebSignSys

The basic client/server model described in section 2 is very simple, but there are many issues to consider in the system design.

**Scalability:** WebSignSys is designed to solve the deployment scalability and flexibility issues entailed in physical beacons, using 'websigns'. A websign is a virtual beacon, not limited by the access range and hardware cost of physical beacons. On the other hand, as many websigns as possible should be supported in the system. However, in the real world, there are so many places that need to be associated with websigns and thus subscribe to WebSignSys servers. This would mean that the websign databases in the servers would have to store billions, even trillions of websigns. Therefore, how to organize all the websigns efficiently is a challenge. The access range that was mentioned in section 2 is one feature to help this organization. In addition, to achieve our goal, we will use a websign hierarchy, which will be discussed in the design of WebSignSys servers.

**Privacy:** As in most popular mobile service models, a mobile user's physical location is one criterion by which personalized service can be provided in WebSignSys. However, unscrupulous services may sell not only the user's online profile but also his/her physical whereabouts tracked through the day, which raises serious privacy or even safety concerns. To solve this problem, we propose two approaches. One approach is to send a wider area to the WebSignSys server instead of the current physical location. For example, if the user is in Palo Alto, the request to the server can contain only 'Palo Alto', not the physical location of the user. On the server side, more URLs will be sent back because of the wider-area request, which might cost more bandwidth and storage space on the client side. However, at the same time, by getting more URLs, pre-caching can be achieved. Another approach is to add a proxy in the system between the client and the server. Generally, the client has a tight relationship with its proxy and can trust its proxy and send its physical location there. The proxy will act as its client, sending a wider area to the server. After receiving related URLs, the proxy can conduct the first level filtering, and send back all the URLs of the places activated by the client.

**Efficiency:** Because of low bandwidth in wireless networks, and low power and computability of client PDAs, PDAs and wireless links should not be overloaded. In addition, the server should not be turned into a bottleneck. To solve all these problems, it is very natural to add a proxy in the system. As we mentioned above, the proxy can achieve pre-caching by sending wider areas and caching more URLs. Since it is common that users stay in an area for a period of time, the proxy will efficiently obtain user-interested URLs without consulting the server again. Moreover, part of the URL resolving module's task (i.e., finding URLs the client points to) can be moved to the proxy, reducing the task of the server side to that of searching the related URLs according to an area in the websign database. In this way, the server is relieved of much computing load.

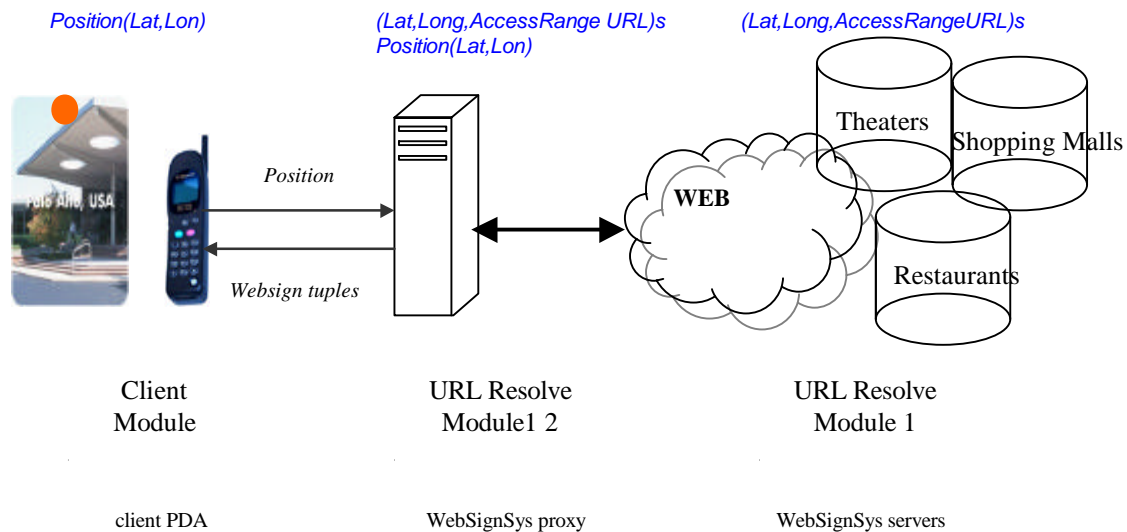
**Positioning technology:** Like CoolTown and many other location-aware service models, WebSignSys requires efficient positioning technologies. For outdoor applications, a possible positioning technology is a GPS plus compass module, which can give us longitude, latitude, and direction. For indoor applications, some potential techniques are Nibble [1] and Bluetooth, which have not been widely investigated in this area yet. One critical issue with positioning is accuracy control. The more accurate the positioning, the

better vision the websigns achieve. Thus, it is necessary to design a good accuracy control algorithm in WebSignSys.

**Other social issues:** WebSignSys consists of clients, proxies, servers, and customers (or websign owners). How to charge customers and where to place websigns---all these are important business issues to consider. As we know, a websign has an access range, which will help to build a business model: if a websign has a bigger access range, its owner---the customer--should be charged more. It makes sense that better advertisement is worth more money. As to the websign placement, it is similar to domain names, where the one who owns the trademark owns the domain.

Based on all the above considerations, we propose a scalable and efficient design for WebSignSys. In the following section, we describe the design in detail.

#### 4 A Scalable and Efficient Design of WebSignSys



*Fig. 2 A Scalable and Efficient Design of WebSignSys*

Fig. 2 illustrates a scalable and efficient design of WebSignSys. In this design, proxies are employed. A client PDA sends requests to its proxy instead of WebSignSys servers. Note that, there may be many servers designed for different categories, such as restaurants, shopping malls, theaters, etc. After getting requests from its client PDA, the proxy will consult the servers just like an agent. Let 's examine how the whole system works.

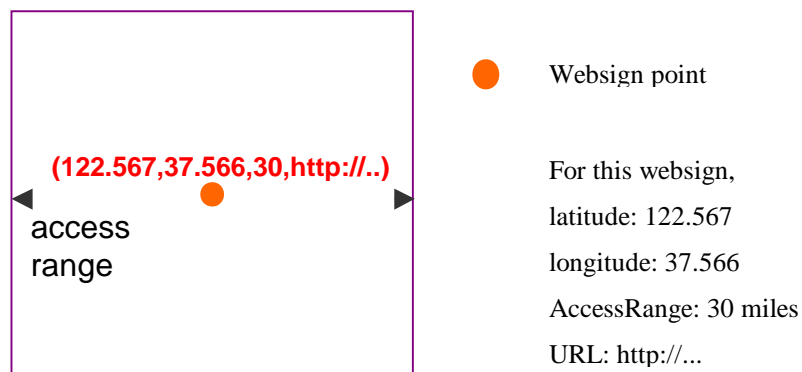
First, the client PDA sends its position tuple to its proxy (notice that, only (latitude, longitude) is sent out in this design, and direction is not used in the proxy). To provide privacy, the proxy does not send this position to the servers directly, but it sends a wider area, where this ' ' position ' ' resides, to the servers instead. For example, when a user points its PDA to a movie theater in Palo Alto, its proxy will send area ' ' Palo Alto ' ' to the theater server. And the theater server will search all the websign tuples (latitude, longitude, AccessRange, URL) of movie theaters in Palo Alto and send them back to the proxy. According to the received position tuple, the proxy can conduct an efficient searching algorithm to get the URLs of the movie theater websigns activated by the client PDA. After receiving activated websigns, the client module can compute the URL list of the places pointed at by the PDA based on the information about direction. From the above description, it is clear that there is a task distribution among clients, proxies, and servers in

this design in order to achieve good performance. We will look at each part 's task in the following subsections.

#### 4.1 WebSignSys Server Design

In WebSignSys, a server receives an HTTP request (containing an area) from a proxy, and searches for the relevant websigns in the database through the URL revolving module. An XML file containing the returned websigns can be generated and sent back to the proxy through an HTTP response. Thus, the communication between the proxy and the server can be easily realized by standard HTTP protocol. However, as we mentioned before, organizing virtual beacons (that is, websigns) efficiently is a critical issue.

As we know, each websign has an arbitrary access range, which is claimed by its owner. The basic structure of a websign is illustrated in Fig. 3. A websign is represented by a tuple (latitude, longitude, AccessRange, URL), where AccessRange is a measure of the maximum distance required for users to access the websign. In other words, only users in the access range of the websign have the right to obtain its information. For example, there is a virtual beacon for Jade Fountain Restaurant in Palo Alto, and its access range is 10 miles. Then only when the PDA holder is within 10 miles of the restaurant can he get the websign 's URL automatically, which makes sense---if the user is too far away from the restaurant, it is not useful to provide the URL automatically for him/her even though he/she points in the right direction, because, generally, there is a nearer restaurant for the user to go to, or the user has no interest at all since this restaurant is too far.



*Fig. 3 Websign structure*

Sensitive readers may be wondering why square-shaped instead of circle-shaped access range is used for a websign. In fact, this is designed to facilitate resolving websigns in the proxy. Circle-shaped access range can also be employed, but the URL resolving algorithm may not be so efficient as when square-shape is used.

To organize the tremendous number of websigns in the real world efficiently, we employ a hierarchy. On the one hand, websigns can be divided into different categories, such as movie theaters, restaurants, shopping malls, amusement parks, etc. Different WebSignSys servers may be designed for websigns in different categories with one server holding one websign database or several websign databases, depending on the implementation. On the other hand, websigns can also be classified at different levels, such as world, nation, state/province, city, etc. Generally, higher-level websigns have bigger access range. If the user only shows interest in nation-size places, then only websigns at

nation level will be sent back. For instance, if the user wants to know the information about those states northern to the California, he just points his PDA to the north, and selects ' 'state level' ', then he will get all the related URLs of those states on his PDA. In this way, the hierarchy of websigns is controlled by access range, level, and category automatically. First, PDA sends a request combining position (latitude, longitude), level, categories, to proxy, then a message containing area, level, and interested categories, is sent to the server, and finally, the server will only get related websigns in the corresponding area. As a result, we achieve scalability.

#### 4.2 WebSignSys Proxy Design

In WebSignSys, the proxy acts as a bridge between the client and the server. After receiving a client PDA's request, the proxy will send a bigger area to the WebSignSys server, and get more websigns rather than just the ones wanted by the client PDA, which provides privacy and efficiency by pre-caching.

In WebSignSys, the proxy takes charge of the following functionalities: searching activated websigns for clients in the cache; detecting new WebSignSys servers; monitoring the change of websigns in WebSignSys servers; deleting obsolete websigns from the cache, and so on. Each of the tasks can be implemented in some module. The architecture for the proxy is illustrated in Fig.4.

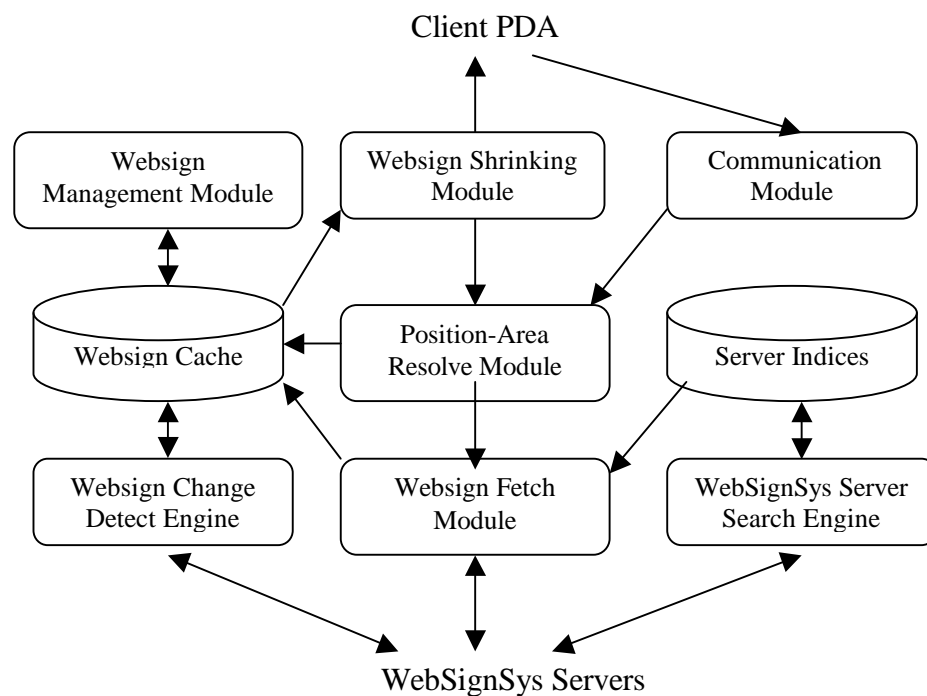


Fig. 4 WebSignSys proxy architecture

Websign searching in the cache is done by the Websign Shrinking Module. In the shrinking algorithm, latitudes and longitudes are sorted first, making it easy to ' 'shrink' ' the big set of websigns to the small set of websigns activated by the client PDA. This is one of the reasons why we use square-shape access range. On the other hand, squares or rectangles may be more flexible than circles, because some websigns may have bigger access ranges in one direction than others.

The Position-Area Resolve Module maps a position to an area, which provides privacy and caching as we mentioned before. As to other modules related to websign management

and WebSignSys server management, their functionalities are obvious. Websign Change Detect Engine is responsible for detecting the change of websigns in WebSignSys servers and Websign Management Module is responsible for inserting new websigns or deleting the obsolete ones, while WebSignSys Server Search Engine takes charge of what new WebSignSys server comes up, and what has changed for the servers in the server indices.

### 4.3 WebSignSys Client Design

The architecture of WebSignSys client is illustrated in Fig. 5. A client PDA sends a request containing position tuples to its proxy and then gets a set of activated websigns back. The Anchoring Module filters the websigns in the direction pointed at by the device and forwards them to the User Interface. Before the request is sent to the proxy, the local cache will be checked first. If the client's position is not changed significantly, no further request is sent out and the websigns in the local cache will be fetched for the anchoring algorithm.

The Positioning Accuracy Management Module deals with the inaccuracy of positioning technique used. Neither outdoor nor indoor positioning techniques shave 100 percent accuracy. In fact, it is almost impossible to get an accurate position because of the limitations of the measuring instruments. The inaccurate position may have heavy influence on the anchoring algorithm. Here we give an example. Suppose you are using GPS which shows your current position  $P$ , but due to the accuracy of GPS, your real position is  $P'$ .  $P'$  may be 10 meters away from  $P$ . If the websign you want to get is in 10 meters range, without accuracy control, the URL you see on the PDA may be totally different from what you need. Thus, fault tolerance mechanisms must be designed.

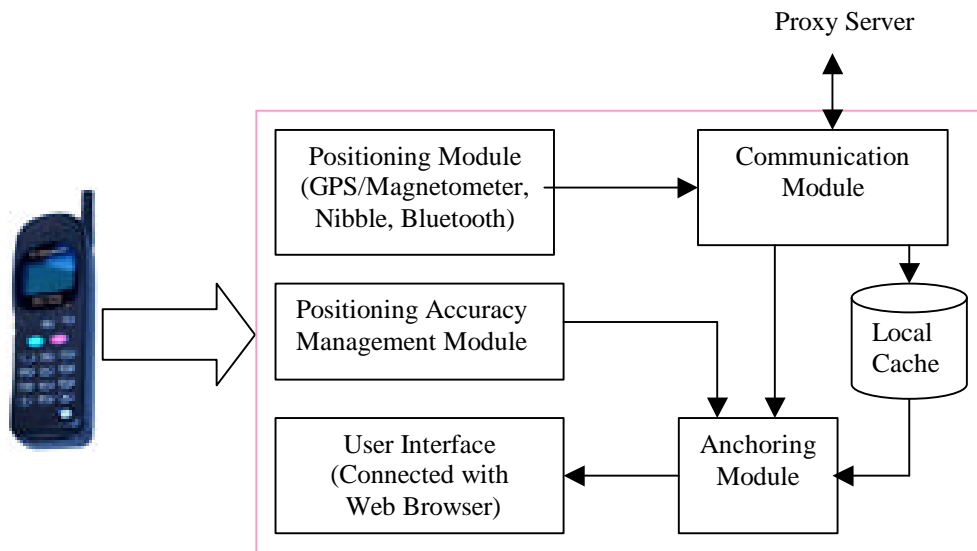


Fig. 5 Architecture of WebSignSys client

## 5. Applications and Related Work

In recent years, building location-aware systems to provide users with augmented reality becomes a very active research field. There are many such systems or prototype systems developed ([1], [2], [4], [5], [8], [9], [10], etc). In all these systems, some kind of

' 'bridge' ' is needed to connect physical objects with the users. This ' 'bridge' ' can be physical or virtual. According to the feature of the ' 'bridge' ', we classify the location-aware information systems into two categories: Physical Beacon-based and Virtual Beacon-based. In Physical Beacon-based systems, small devices or tags are attached to physical objects. The devices can use radio waves or infrared wireless links to communicate with the users. Or the users can detect the tags on the physical objects and get the related information. In both cases, physical objects (wireless devices or special tags) play the role in the communication. Augment-able Reality [8] uses special tags; Cyberguide [4] and Active Badge [10] employ wireless devices, while CoolTown [1] utilizes both of them. Though this type of system has its own application, it has much limitation in access range and also it causes deployment scalability and flexibility issues for many applications. As to Virtual Beacon-based systems, WebSignSys, SpaceTag [9], and Touring Machine [2] fall in this category. None of them use physical signals from the physical objects or detect tags attached to the physical objects. And all of them utilize the location attributes of physical objects, and retrieve the related information by computing the accessible resources (users' position and direction achieved from GPS and Compass, for example). However, WebSignSys differs from Space Tag and Touring Machine.

WebSignSys and Touring Machine are all web-based in general. But they are designed for different types of applications. Touring Machine is more localized web-based system. The HTTP server in Handheld PC is initialized by the tour application running on the backpack PC, while tour data on the backpack PC includes all the useful information for the tour place. During the navigation, the tour application will continuously receive input from the GPS position tracker, the orientation tracker, and also user input (such as menu selection). Based on this input and the database of information about the tour place, it will generate the graphics that are overlaid on the real world by the head-worn display. Compared with WebSignSys, Touring Machine is more localized and more customized. Users need to customize a new tour application for a new place to navigate. On the other hand, WebSignSys uses the concept of access range and also employs proxy, which make it much easier to deploy. WebSignSys service can be available to every cellular-phone or PDA user.

Though both WebSignSys and SpaceTag adopts access range concept, the essential difference is that SpaceTag is a broadcasting system, which is similar to the information distribution services from paging service or cellar-phone companies. But WebSignSys is web-based infrastructure. SpaceTag is trying to tell the user ' 'what is available around' ', while WebSignSys puts efforts in ' 'what I am pointing to' ', though it can also do the first job by giving omni-direction.

## **6. Conclusions**

In this report, we have described a design of WebSignSys, which bridges the physical world and the virtual world (World Wide World) and provides a simple form of augmented reality. It is scalable, flexible, efficient, and easy to deploy. Further more, most of the technologies used in the designed WebSignSys are mature enough. Our recent work is to implement this design and provide a prototype system in order to give a better performance evaluation.

## **Acknowledgement**

This work is based on the class project of CS244A (supervised by Prof. Wesley Chu).

## References

1. CoolTown: <http://cooltown.hpl.hp.com>
2. S. Feiner, B. MacIntyre, T. Hiller, and T. Webster, A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. In *Proc. of ISWC '97 (First Int. Symp. on Wearable Computers)*, Cambridge, MA, October 13-14, 1997. Also as: *Personal Technologies*, 1(4), pp. 208-217, 1997
3. T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, J. Morris, J. Schettino, and B. Serra, People, Places and Things: Web Presence for the Real World, *Tech report HPL-2000-16*, Hewlett-Packard Laboratories, 2000
4. Sue Long, Dietmar Aust, Gregory D. Abowd and Chris Atkeson, Cyberguide: Prototyping Context-Aware Mobile Applications, Short paper in *Proc. of 1996 Conference on Human Factors in Computing Systems (CHI '96)*, Vancouver, CA, April 13-18, 1996
5. J. Loomis, R. Gollidge, and R. Klatzky, Personal Guidance System for the Visually Impaired Using GPS, GIS, and VR technologies. In *Proc. of Conf. On Virtual Reality System and Persons with Disabilities*, Millbrae, CA, June 17-18, 1993
6. Nibble: <http://mmsl.cs.ucla.edu/~castrop/nibble.html>
7. Salil Pradhan, Cyril Brignone, Jun-Hong Cui, Alan McReynolds, Websigns: Hyperlinking from Physical Locations to the Web, and Mark Smith, *IEEE Computer special issue on Location-aware Computing*, 34(8), pp. 42-48, August, 2001
8. Jun Rekimoto, Yuji Ayatsuka and Kazuteru Hayashi, Augment-able Reality: Situated Communication through Physical and Digital Spaces, In *Proc. of ISWC '98 (Second Int. Symp. on Wearable Computers)*, 1998
9. Hiroyuki Tarumi, Ken Morishita, Megumi Nakao, Yahiko Kambayashi, Space Tag: An Overlaid Virtual System and its Applications, In *Proc. of International Conference on Multimedia Computing and Systems (ICMCS '99)*, Vol.1, pp. 207-212, 1999
10. R. Want, and A. Hopper, Active Badges and Personal Interactive Computing Objects. *IEEE Transactions on Information Systems*, 38(1), pp. 10-20, 1992