NATURAL INTERACTION FRAMEWORK FOR PEDESTRIAN NAVIGATION SYSTEMS ON MOBILE DEVICES

by

Ceren KAYALAR

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
Spring 2006

NATURAL INTERACTION FRAMEWORK FOR PEDESTRIAN NAVIGATION SYSTEMS ON MOBILE DEVICES

APPROVED BY

Selim Saffet BALCISOY …….
(Dissertation Supervisor)

Erkay SAVAŞ …….

Hakan ERDOĞAN …….

Hüsnü YENİGÜN …….

Meriç ÖZCAN …….

(İbrahim TEKİN) …….

DATE OF APPROVAL: …….

NATURAL INTERACTION FRAMEWORK FOR PEDESTRIAN NAVIGATION SYSTEMS ON MOBILE DEVICES

Ceren KAYALAR

EECS, MS Thesis, 2006

Thesis Supervisor: Assistant Professor Selim Saffet BALCISOY

Keywords: context-awareness, location based services, mobile computing, posture recognition, interaction techniques, augmented reality, inertial sensors, GPS receiver

Abstract

Mobile Augmented Reality applications based on navigation frameworks try to promote interaction beyond the desktop by employing wearable sensors, which collect user's position, orientation or diverse types of activities. Most navigation frameworks track location and heading of the user in the global coordinate frame using Global Positioning System (GPS) data. On the other hand, in the wearable computing area researchers studied angular data of human body segments in the local coordinate frame using inertial orientation trackers.

We propose a combination of global and local coordinate frame approaches and provide a context-aware interaction framework for mobile devices by seamlessly changing Graphical User Interfaces (GUIs) for pedestrians wandering in urban environments. The system is designed and tested on a Personal Digital Assistant (PDA) based handheld prototype mounted with a GPS receiver and inertial orientation tracker. It introduces a method to estimate orientation of a mobile user's hand. The recognition algorithm is based on state transitions triggered by time-line analysis of pitch angle and angular velocity of the orientation tracker. The prototype system can differentiate between three postures successfully. We associated each posture with different contexts which are of interest for pedestrian navigation systems: investigation, navigation and idle.

Thus, we introduce the idea that once orientation trackers became part of mobile computers, they can be used to create natural interaction techniques with mobile computers.

The prototype is tested successfully in two urban environments:

- Sabanci University campus area,
- 9th International Istanbul Biennial venues in Beyoglu, Istanbul.

MOBİL BİLGİSAYAR SİSTEMLİ YAYA NAVİGASYONU UYGULAMALARI İÇİN DOĞAL ETKİLEŞİM ALTYAPISI OLUŞTURULMASI

Ceren KAYALAR

EECS, Yüksek Lisans Tezi, 2006

Tez Danışmanı: Yard. Doç. Dr. Selim Saffet BALCISOY

Anahtar Kelimeler: kontekst-algılama, konum bazlı hizmetler, mobil bilgi işleme, beden duruşları tanıma, etkileşim teknikleri, artırılmış gerçeklik, atalet algılayıcıları, GPS alıcısı

Özet

Navigasyon altyapısı bazlı mobil artırılmış gerçeklik uygulamaları insan-bilgisayar etkileşimini masaüstü metaforundan daha ileriye götürmeyi amaçlamaktadır. Bu amaçla, kullanıcının pozisyonu, oryantasyonu veya çeşitli hareketleriyle ilgili verileri toplayan giyilebilir algılayıcılar kullanılabilir. Navigasyon altyapılarının çoğu kullanıcının konumunu ve hareket yönünü küresel koordinat sisteminde Küresel Konumlandırma Sistemi (GPS) verisi kullanarak izler. Ayrıca, giyilebilir bilgisayar sistemleri araştırmacıları, lokal koordinat sisteminde atalet oryantasyon izleyicileri kullanarak insan vücudu bölümlerinin açısal verisi üzerinde çalışmışlardır.

Bu tezde, küresel ve lokal koordinat sistemi üzerindeki çalışma yöntemleri birleştirilmiştir. Şehir ortamında yaya gezen mobil bilgisayar sistemi kullanıcıları için konteksti algılayabilen bir etkileşim altyapısı, gerektiğinde kesintisiz değişen Grafik Kullanıcı Arayüzleri (GUIs) ile oluşturulmuştur. Önerilen sistem, GPS alıcısı ve atalet oryantasyon izleyicisi bağlanmış bir Kişisel Dijital Yardımcı (PDA) bazlı avuçiçi bilgisayar prototipi üzerinde dizayn ve test edilmiştir. Mobil kullanıcının elinin oryantasyonunu tahmin etmek için bir metod geliştirilmiştir. Oryantasyon izleyicisinin eğim açısı ve açısal hız değerleri zaman çizgi analizinden geçirilmiş ve bu sonuçlar ile tetiklenen durum geçişleri üzerine bir tanıma algoritması üretilmiştir. Prototip sistem üç beden duruşunu ayırt edebilmektedir. Her bir beden duruşu yaya navigasyon sistemleri ile alakalı kontekstlerle ilişkilendirilmiştir: inceleme, navigasyon, boş zaman.

Bu şekilde, oryantasyon izleyicilerinin mobil bilgisayarların bir parçası olması durumunda doğal etkileşim teknikleri oluşturmak için kullanılabilecekleri fikri önerilmektedir.

Oluşturulan prototip iki kentsel ortamda başarılı bir şekilde test edilmiştir:

- Sabancı Üniversitesi kampüsü,
- Beyoğlu'nda düzenlenmiş olan 9. Enternasyonal İstanbul Bienal mekanları ortamı.

Babaannemin anısına...

ACKNOWLEDGEMENTS

I want to thank my thesis supervisor Selim BALCISOY for his valuable support and academic guidance during my thesis work. Also, I would also like to thank my coworkers and friends at CGLab; Başak ALPER, Can ÖZMEN, Selçuk SÜMENGEN and Ekrem SERİN for their friendship and support during graduate school. Thanks to Emrah DENİZ from mechatronics laboratory for helping me to design the necessary RS232 connection of our hardware prototype.

My special thanks to my friends "Culnamos" back in Izmir; Gül BOZTOK, Serkan SAKARYA, Burçin BÜYÜKÇAKIROĞLU, Berhan SOYLU, for supporting me even kilometers away and being there on my trips back home.

I would like to thank people in Notestik for the entertaining summer vacations, especially Meriç and Deniz ÜNLÜ and their parents.

Thanks to my roommate in Sabanci University dormitory Burcu AKYOL and her family for their support and friendship.

I want to thank my friends and 'voluntary family' Şerife KAPUKAYA and Selim BUDAKOĞLU in Istanbul for believing in me, supporting me, and giving me a living place in their home.

Special thanks to my cousin 'Çekirge' Elvin ERKUT for loving me, cheering me up, humorous chats and supporting me for my lifetime. She was always there when I needed her. Also thanks to my aunt Muhteşem ERKUT for her support by long talks on the phone.

Finally I would like to express my thanks and deepest love to my mother Fatma Muhterem KAYALAR for being my teacher, her everyday support even kilometers far away and last minute support in Istanbul to finish up my thesis; and my father Prof. Dr. Arif Şengün KAYALAR for his love, confidence in my work, valuable academic thoughts, and supporting me for my lifetime. My special thanks to Izmir, my homeland, for growing me up, to my friends and my all family members who always supported me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST of ABBREVIATIONS

2D	Two Dimensional
3D	Three Dimensional
AR	Augmented Reality
COM	Communications Port
DGPS	Differential Global Positioning System
DOF	Degree of Freedom
GIS	Geographic Information System
GPS	Global Positioning System
GUI	Graphical User Interface
FOV	Field of View
HMD	Head Mounted Display
LBS	Location Based Services
MR	Mixed Reality
NMEA	National Marine Electronics Association
PDA	Personal Digital Assistant
RFCOMM	Radio Frequency Communication
SDK	Software Development Kit
SPP	Serial Port Profile
UI	User Interface
VR	Virtual Reality

Chapter 1 INTRODUCTION

1.1 Motivation

In recent years, mobile computing integrated into our daily lives rapidly. While interacting with the urban environment, we are using mobile devices, i.e. mobile phones and Personal Digital Assistants (PDAs), to send/receive text messages, play games, check e-mails, make phone calls, write documents, use location based or navigation services etc. To offer better services, the trends in mobile computing are leaning towards:

- minimizing user's effort to access information,
- being aware of environmental conditions and user's activities,
- offering natural interaction techniques and alternatives for desktop metaphor,
- preserving ubiquity without distracting user's perception in the environment,
- visualizing relevant information in 2D/3D graphics or text format on small screen effectively,
- offering efficient communication frameworks to other mobile computers, wearable or stationary trackers, and powerful desktop computing systems.

Inspired by these ideas, researchers began to use mobile devices in Augmented Reality (AR) applications as computing device, through their increasing computing power, 2D/3D rendering and storage capabilities. A mobile AR system covers the basic properties of a standard AR system [5]:

- Combines real and virtual,
- Interactive in real-time,

- Registered in 3D.

In addition, a mobile AR framework collaborates with context-aware computing by sensing environmental information that is relevant to the interaction between user and application, and ubiquitous computing by integrating into the real world seamlessly.

Most applications in this promising research field are prototyped as location based services, such as pedestrian navigation systems, car navigation systems, mobile tour-guide applications [10, 9, 11, 13, 33, 39]. Those services heavily rely on interaction techniques estimating precise position and orientation of the user in the global coordinate frame. However to create natural interaction, upper body postures need to be tracked and recognized.

Several research groups working in wearable computing area studied recognition of upper body postures and gestures using body-worn orientation trackers, tilt sensors and reported high success rates [43, 26, 3]. Combining such a study with location based services could offer natural interaction mechanisms for mobile AR frameworks.

1.2 Contribution

The proposed mobile AR system provides a context-aware interaction framework by seamlessly changing Graphical User Interfaces (GUIs) for pedestrians wandering in urban environments. As studied by researchers working on location based services, it is important to acquire a user's geographic position and where she is looking at. In addition to these context-attributes, we propose a natural interaction mechanism to the navigation system by inferring application dependent arm posture (Figure 1.1):

- **Idle state**, where a user is not looking to the screen, moving her arm and rendering is minimized,

- **Navigation state**, where a user's hand is approximately in a horizontal position, and point of interests are represented 3D in global coordinate system,

- **Investigation state**, where a user's hand is approximately in a vertical position, point of interests are placed according to the user's coordinate system and represented in 2D.

Figure 1.1 Three states of interaction

These services are implemented and tested on a PDA based handheld prototype mounted with a Global Positioning System (GPS) receiver and inertial orientation tracker.

This research introduces the idea that once orientation trackers became part of mobile computers, they can be used to create natural interaction techniques with mobile computers.

1.3 Thesis Structure

Chapter 2 addresses related work on context-aware computing, interaction paradigms, effective visualization on small screen and mobile AR systems hardware. Chapter 3 explains the proposed system as hardware, software components in detail. Chapter 4 introduces a case study on the prototype: a mobile context-aware pedestrian navigation application.

Chapter 5 discusses different interaction techniques for mobile users and emphasizes contribution of this work. Chapter 6 summarizes overall work and offers future research ideas.

Chapter 2 RELATED WORK & BACKGROUND

As the technology evolves rapidly; faster, smaller and multifunctional mobile computing devices integrate into our daily lives. Most common mobile devices, i.e. mobile phones and PDAs, are not only serving to make phone calls, send/receive text messages, check e-mails, write documents or watch video; beyond, they are capable of rendering complex 3D graphical environments and connecting with diverse positioning or orientation devices through faster communication interfaces. Naturally, researchers begin to exploit mobile computing devices as core device for Virtual Reality (VR) and Augmented Reality (AR) applications.

However, these applications should take advantage of the system's mobility by offering novel interaction techniques and user interfaces, which reduce the users' effort while providing relevant data. The decision about what information is relevant in which condition and how to react to it is given by the application designer, so the mobile AR application is informative about the real environment without distracting the user. Extracting such results requires observing the environmental conditions, collecting data, analyzing it and obtaining statistical results about possible user activities and context.

These techniques and design challenges are covered in detail in the following subsections.

2.1 Context – Aware Mobile AR Systems

The users of mobile systems are pedestrians or traveling in vehicle, hence interacting with the environment while using a mobile phone or PDA. Most of such systems are deaf and blind to anything occurring in the environment other than change in position. But mobile AR applications, which combine the real world with computer generated graphics, are

- increasing the richness of human-computer interaction,
- preventing perception distraction,

- offering more useful computational services than regular mobile applications by increasing the perceived information level,

- minimizing explicit interaction effort of users.

These applications take full advantage of context-awareness and provide humans the sense of being acquainted by the application interactively and intelligently. Some research systems are *ArcheoGuide*, real-time virtual reconstruction of a cultural heritage site's remains [39] (Figure 2.1); *Backseat Gaming*, a mobile AR game about finding virtual clues of a kidnapping case by interacting with the roadside objects [9]; and *MARS*, a mobile AR tour-guide system [18].

Figure 2.1 *ArcheoGuide*. The system setup (left), ruins of the temple (top right), augmented temple with rendered model on top of live video (bottom right) [39]

2.1.1 Definition of Context and Context – Aware Applications

Existing work in context-aware computing defines what constitutes a context-aware application and what context is. The term "context-aware" is first introduced by Schilit and Theimer [emphasized in 1], where context is defined as location, identities of nearby people and objects, and changes to those objects. Several definitions in

literature provided synonyms for context or depicted enumerations of examples using context, before [1] and [35] come up with generic and clear definitions as follows:

Abowd and Dey's definition: "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."

Salber's definition: "Context is any environmental information that is relevant to the interaction between the user and the application, and that can be sensed by the application."

Context information can be categorized by its *providers*, i.e. people, places, physical objects, computing objects and *attributes*, i.e. location, identity, activity and state [35]. For example, in a context-aware tour guide, the user carrying a mobile device explores an unknown site. When the user enters a specific area, the application displays relevant information. In such possible application, the context provider is place and context attribute is location.

A car navigation system showing the user's location on a map and point of interest information, recomputing driving directions when the user misses a turn and tagging the waypoint data to the nearest cinema is an example of a context-aware application, which emphasizes three classes of context usage:

- presenting information and services,
- automatically executing a service,
- attaching context information for later retrieval.

In such car navigation systems, the location information is provided by a GPS receiver. Application designer should handle the connection between the GPS receiver and the mobile device, parsing of GPS data and mapping the geographical coordinates to the screen coordinates. A software component that provides such services is defined as *context widget*. It consists of three component levels [35]:

- Generators, which interface directly with the sensors

- Interpreters, which abstract the data provided by sensors

- Widget controller, which coordinates the other components and is the entry point for applications to query the widget state (Figure 2.2).

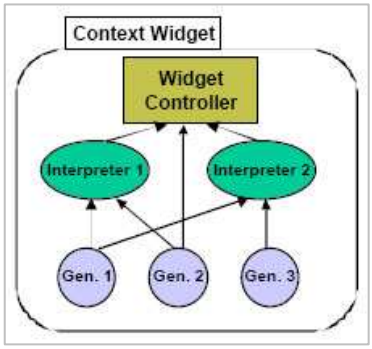


Figure 2.2 Anatomy of a context widget [35]

This abstraction provides benefits such as using a location widget in relevant applications, i.e. for a car navigation system it could be mobile tour-guide applications, and hiding the complexity of the actual sensors used from the application.

According to the definitions of context, context-awareness and context usage classes, two promising fields are collaborating with context-aware computing: *Mobile computing*, in which small and inexpensive computers and wireless networking allow users to roam the real world without being tethered to stationary machines, and *ubiquitous computing*, which aims to integrate computers in daily life in the sense of a more natural human-computer interaction. In mobile and ubiquitous computing applications, user's context is very dynamic. Under these circumstances, effective use of implicitly sensed context allows an application's behavior to be customized to the user's current situation, thus preserving ubiquity [1].

2.1.2 Location Based Services

For a mobile device to be context-aware, acquiring and processing location information is the obvious starting point, as enumerated in Salber's context attributes

definition (see section 2.1.1). The systems, which generate data according to the user's location thereby augment the real world with a layer of virtual information, are defined as Location Based Services (LBS). Considering a simple example, a LBS can help to show the closest cinema in a foreign city on a cell phone or PDA. In terms of augmented reality based LBS the user also participates in a scene as introduced in the prototype system by [13], which allows users gathering visual information about the real environment on the head-worn display and accessing detailed information through gaze-directed selection or interaction with handheld computer. This kind of illustration can be used for the creation of alternative learning environments, car and pedestrian navigation systems, and tour-guide applications.

Using 3D models and different interaction techniques in LBS applications provides a profound improvement in knowledge intermediation. But for some cases a moderate level detailed 2D map could be more informative than a 3D complex virtual environment. Also, a navigation mechanism should always be provided to view the environment from different directions.

LBS can be integrated both to outdoor and indoor applications, which use diverse techniques for location tracking, such as GPS, DGPS, RFID, IrDA, etc. (see section 2.4.3).

2.1.2.1 Mobile Tour-Guide Applications

A mobile tour guide typically runs on a handheld computer and allows a visitor to tour a particular indoor and outdoor space. As the visitor moves throughout the space, she is shown her current location on a map and is shown information about the interesting sites that she is near [35]. What really sets apart a mobile tour guide from more traditional solutions (such as paper guides and desktop guides) is location-awareness, which improves the usability of the guide.

Two crucial services that are usually provided by most mobile guides are navigation support and information delivery. Navigation support allows users to obtain directions to navigate in an environment and to locate themselves and points of interest in the surrounding area. Information delivery gives users the possibility to obtain information on the points of interest located in the visiting area [10].

To offer better services to users, existing work on mobile tour guides takes advantage of the contextual information, usually the position. *Cyberguide* [2] project defines several prototypes of a mobile context-aware tour guide, which are aware of the user's current location and as well as a history of past locations. The components of this project are referred as role playing as a cartographer, librarian, navigator and messenger. Cartographer (map component) is realized by a map of the physical environments that the tourist is visiting, librarian (information component) is realized as a structured repository of information relating to objects and people of interest in the physical world, navigator (positioning component) is realized by a positioning module that delivers accurate information on tourist location and orientation, and messenger (communications component) is realized as a set of wireless communication services. Prototypes are built on Apple MessagePad and pen based PC platforms acquiring position data from GPS (outdoor) and IR (indoor). Cyberguide was one of the first complete prototypes clarifying the thoughts on how context-aware computing provides value to the emerging technology promising to release the user from the desktop paradigm of interaction.

Another example, *LAMP3D* is a system for location-aware presentation of VRML (Virtual Reality Modeling Language) content on mobile devices (Figure 2.3). This system is used to provide tourists with a 3D visualization of the environment they are exploring, synchronized with the physical world through the use of GPS data; tourists can easily obtain information on the objects they see in the real world by directly selecting them in the VRML world (using a pointing device such as the PDA stylus or their fingers) [10].

Figure 2.3 LAMP3D field test showing the virtual counterpart of the real world scene [10]

2.1.3 Activity Recognition

As emphasized by Salber, another important context attribute is activity [35]. Beyond determining a person's current location, by recognizing what she/he is doing using diverse types of sensors or wearable computers, novel interaction mechanisms can be created.

For indoor systems, where GPS signals couldn't be received, different techniques (i.e. RFID, IrDA) are used for positioning purposes. Lee and Mase proposed an activity recognition approach to locate people in an office environment [26]. The hardware module consists of a Linux-based PDA, a digital compass module which is attached to the user's waist, and an accelerometer-gyroscope module which is carried in the user's trouser pocket (Figure 2.4). Without receiving positioning data from a transmitter, their location recognition system is collecting data from the sensors that the user wears. After executing some preprocessing tasks on PDA, predefined types of unit motions (sitting, standing, walking on level ground, going up a stairway and going down a stairway) are detected using a fuzzy logic reasoning method. When the system detects a walking behavior, a 3D displacement vector is updated using dead reckoning. Thus, this proposed system used activity recognition for positioning purposes and achieved a recognition ratio of 91.8%.

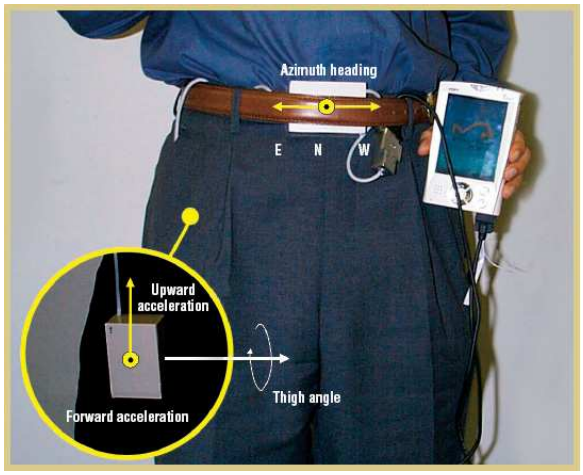
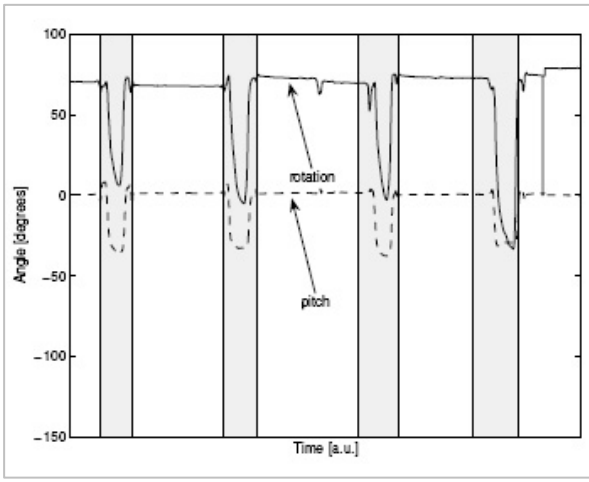


Figure 2.4 Wearable computing system proposed by [26]

Detection and recognition of upper body postures and gestures are also studied by several research groups to create natural human-computer interaction techniques. The proposed methods aim generically to aid daily life, reduce human effort to use computing systems and integrate computers into the environment seamlessly.

Amft et al. introduced a recognition system for detecting arm gestures related to human meal intake [3]. The idea of this project is based on dietary monitoring used by health professionals. They mounted two orientation sensors on the wrist and upper arm to detect gestures, i.e. moving the arm towards the mouth and back. First step of this system is segmenting the continuous sensor data into motion segments using the SWAB (sliding window and bottom-up) algorithm, which combines a bottom-up segmentation scheme with a sliding window algorithm, thus working on-line. A set of pre-defined eating and drinking gestures are then spotted using Hidden Markov Models (HMMs) from other usual movements, i.e. scratching head, touching skin, and turning pages of newspaper at a success rate of 94%. As a last step, the gestures are recognized and categorized using HMMs (Figure 2.5).



Category	Tools	Food type and gesture
Cutlery	fork, knife and plate	Lasagne; fork tap and loading, involved knife cutting activity
Spoon	spoon and bowl	Cereals with milk; spoon loading and manoeuvring
Hand	-	Chocolate bar; held in hand, moved to mouth and back
Drink	glass	Water; glass was moved to mouth and back on table

	Cutlery	Drink	Hand	Spoon
Relevant	31	31	31	31
Recognised	27	25	22	16
Insertions	9	4	12	10
Deletions	4	6	9	15
Accuracy	0.87	0.81	0.71	0.52

Figure 2.5 Sample sensor data and segmentation for "Drink" (left), gesture categories (top-right), and performance evaluation for spotting gestures (bottom-right) [3].

Recognizing arm postures is used to introduce a new technique for entering text into a mobile phone. Most common text entry technique is MultiTap, which forces ambiguity when a 26 letter Roman alphabet is mapped onto keys 2-9 on a standard 12-button mobile phone keypad. The novel technique, TiltText, uses the orientation of the tilt sensor mounted mobile phone to resolve this ambiguity, by tilting the phone in one of four directions to choose which character on a particular key to enter [43]. For example, pressing the 2 key and tilting to the left inputs character a, while tilting to the right inputs the character c (Figure 2.6).

There are three main values to determine the tilt value: key tilt, absolute tilt and relative tilt. With key tilt technique the amount of tilt is calculated as the difference in the value of tilt sensors at key down and key up, which requires the user to iterate three movements: push the button, tilt the phone, and release the button. The evaluations show that user performance is much slower than MultiTap. Absolute tilt technique compares the tilt sensor's value at any given time to a fixed absolute origin. Two movements are required to enter a key, tilt the phone and press the key. This approach is not ideal, since in practice users do not maintain a constant arm posture. Hence the fixed origin has to be reset every time the user's gross arm posture changes. Because of the need to return to the origin, the amount of movement is more than one may expect especially in the case of entering successive characters. Relative tilt approach calculates the tilt relative to a floating origin that is set when a tilt gesture begins. This approach

solves all the problems of the previous tilt determination methods, since all tilts are relative to the beginning of the gesture. They implemented relative tilt approach; however the tilt sensor allowed only 10 Hz sampling rate which caused repeated sampling of tilt angles. Mandatorily, they implemented absolute tilt approach by correcting it with a key functioning to fix the origin. Their results show that text entry speed including correction for errors using TiltText is 23% faster than MultiTap, despite a higher error rate for TiltText. They also reported that 20 to 50 Hz sampling rates are required for robust tilt implementations.

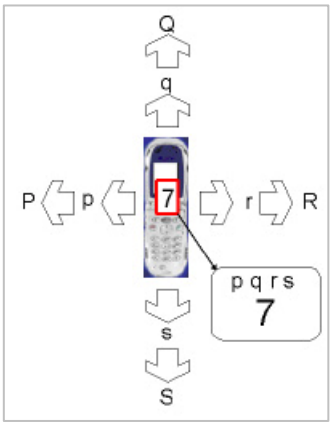


Figure 2.6 Text entry with TiltText. Uppercase letters are entered by tilting beyond a threshold [43].

2.2 Interaction Paradigms

Before the advent of wireless, mobile and handheld technologies, prevailing paradigm in interaction design was to develop applications for the desktop, where the user is interacting with keyboard, mouse and looking to a monitor. The term, which unifies concepts of GUIs representing the user's desk accessories and the whole desktop environment, is the *desktop metaphor*. Mostly such an interface is based on WIMP (windows, icons, mouse and pointers) using a regular monitor. However, recent trends in interaction paradigms try to promote beyond the desktop.

2.2.1 Ubiquitous (or Pervasive) computing

The interaction paradigm for ubiquitous computing is based on technology disappearing in the background, which means we would be no longer aware of the

computers in the environment while they are integrated seamlessly into the physical world, interacting with each other and extending human capabilities. Mark Weiser, the founder of ubiquitous computing, built a prototype system called “tabs, pads and boards” [42], which consist of hundreds of computers equivalent in size of post-it notes, sheets of paper and blackboards. These computing devices are to be used in office environments without noticing that they are computers and offering more functions than desktop metaphor.

Pervasive computing, “any time any where computing” is integrated with ubiquitous computing and the terms are used interchangeably. Some examples are cell phones, handheld devices, and prototypes such as, smart pans that beep when the food is cooked and intelligent fridges that signal the user when the stocks are low.

2.2.2 Wearable computing

Researchers try to embed technologies in everyday environment. Wearable computing focuses on systems which people can wear or mount on the clothes. Such a system allows the user interact with and take advantage of digital information while moving in the physical world.

A complete wearable framework, which consists of a backpack containing a laptop with a wireless interface, a positioning system, an orientation tracker, a see-through HMD and a camera (Figure 2.7) doesn’t introduce a natural interaction mechanism, because it prevents the user move freely. Rather than this heavy setup, a mixed reality platform consisting of a PDA with localization and orienting capabilities and a lightweight see-through HMD operates with same capabilities and allows a free movement, natural interaction to the user [31].

Figure 2.7 Lightweight MR platform [31] (left) and heavy backpacked AR system [18] (right).

Due to the small screen space of PDAs, screen based interaction mechanisms controlled with UI widgets are confusing and ineffective. Thus, small screen space forces the interaction mechanisms to be more natural. If the interaction is provided with widgets on the PDA screen, these widgets must be large enough to be distinguished from the content on display and to be practical for relevant interaction. This fact limits the displayed content size, and it is better to build natural interaction mechanisms and avoid virtual interface widgets.

Höllerer et al. introduced a gaze-directed selection mechanism for outdoor UI interaction in their mobile augmented reality system research. The display unit of this system is a see-through head-mounted display, which augments the real world with virtual labels and flags (Figure 2.7). Gaze-directed selection is accomplished by the user orienting her/his head so the desired object's projection is closer than any other to the center of the head-mounted display. If it remains the closest within that area for a half-second, the object's label is smoothly changing color over that interval to confirm selection [18].

Recognizing arm postures is used to introduce a new technique for entering text into a mobile phone. Orientation of the tilt sensor mounted mobile phone is used to resolve the ambiguity faced by standard text entry technique. Tilting the phone in one of four directions chooses which character on a particular key to enter (Figure 2.8) [43].

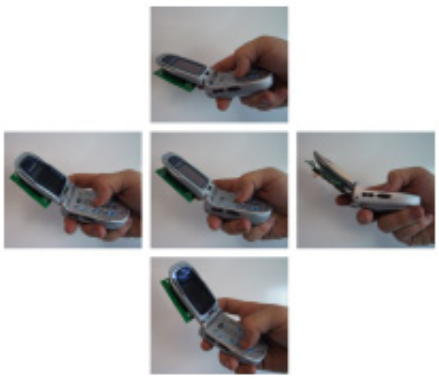


Figure 2.8 Novel interaction technique for entering text input into a mobile phone, TiltText [43]

2.3 Visualizing Data on Small Screen

Limitations of current mobile devices, such as the insufficient computational power and the limited space, make it difficult to obtain a smooth navigation of 3D representations, especially when complex environments (e.g. full cities) are taken into consideration.

The main technical challenge in cell-phone-based spatial-information system architecture is how to automatically make a compact, visually appealing map, even for small cell phone displays. The service also must compress the graphics data into a small area to enable transfer from server to client. Two elements are considered for data compression by Hitachi team [36], binary encoding and map summarization. These approaches drastically reduce the data volume that the server must supply to the mobile client, thus shortening data transfer time. Less than 1:5 compression rate is achieved and two cell-phone-based spatial-information services are implemented based on these techniques.

2.3.1 Interface Design for Mobile Computing

Most of the implemented location-aware mobile guides use 2D maps of the area where the user is located; pinpointing her position and usually providing visual information on the nearest points of interest and on the paths she has to follow to reach specific destinations. Maps are powerful tools for navigation because of the richness of information they can supply and the rate at which people can absorb this information.

Recently, some attempts have been made at exploring 3D graphics for mobile guides. Rakkolainen and Vainio have proposed a system that combines a 2D map of an area with a 3D representation of what users are currently seeing in the physical world, studying the effects of 3D graphics on navigation and way finding in an urban environment [33]. They found that 3D models help users to recognize landmarks (i.e., distinctive features of an environment, such as churches and squares, which can be used as reference points during navigation) and find routes in cities more easily than traditional 2D maps. Unfortunately, the prototype was implemented on a laptop computer, not on a PDA. 3D city models for route guidance have been tested also by [24] who obtained similar results but highlighted the need for detailed modeling of buildings and additional route information such as street names. Unfortunately, their prototype uses only predefined animations and sequences of pictures, not interactive 3D worlds. Moreover, both projects focused only on navigation support and no information delivery service about points of interest was provided.

Realistic visualization of large and complex 3D models, such as those used in mobile guides, is a very important task for other application areas as well: scientific simulation, training, CAD, and so on. However, mobile devices do not include the specialized hardware typical of high-quality graphics boards, and it is thus not always possible to obtain a good quality level for the visualization. A possible approach to this problem is to carry out rendering on a powerful remote server (or a cluster of workstations) connected through a wireless network and display the results on the mobile device as a video sequence [25]. This solution has two advantages: the data to be visualized is processed by specialized hardware, thus bypassing the problem of the low computational power of mobile devices, and the source data is not transmitted to the client device, thus allowing for data independence. On the other side, due to the limited bandwidth of current wireless networks, this remote computation solution needs complex algorithms for the preparation of the data to be transmitted.

Höllerer et al. described a mobile augmented reality system testbed which combines different UIs to allow indoor and outdoor users to access and manage information that is spatially registered with the real world [18]. The novelty of this system lies on hybrid UIs that combine different display technologies interacting real-time between outdoor and indoor systems. Outdoor users explore the real world with

different displays, a tablet PC and a see-through head-mounted display, which are used in conjunction. The handheld UI contains a 2D map and the HMD UI consists of screen-stabilized items, which are fixed to the display, i.e. menus, pointers, and world stabilized items, which are visually registered to spatial locations, i.e. flags. Thus, an object selected on the map is also selected on the HMD and the HMD menus are controlled from the handheld display (Figure 2.9). Indoor users interact with a desktop or projection-display UI, which allows them to provide guidance to outdoor users by sketching paths or pointing out objects of interest. Also, an immersive version of the indoor UI relying on the same display technique as the outdoor user is introduced.

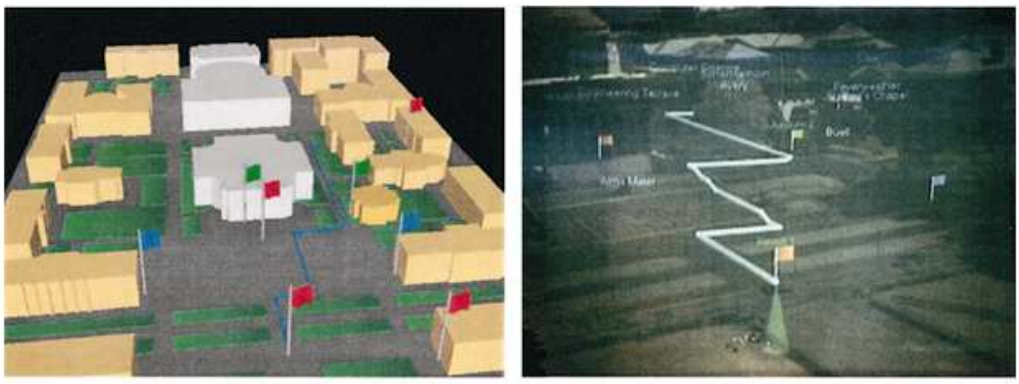


Figure 2.9 A path is created in desktop UI (left), same path on outdoor UI (right) [18]

In spite of increasing storage capacities, easier-to-read displays and faster communication interfaces, PDAs are limited by two constraining factors: the dimensions of displayed text and of the screens on which the text is displayed are unlikely to change very much. Considering these rigid factors, the design of user interfaces for PDAs must balance two opposing forces:

- the need to shrink the screen to a size that fits inside a very small box (physical limitation),

- the need to keep the screen sufficiently large to show enough information that the device is actually useful (functional limitation).

Kamba et al. focused on techniques for reducing the screen space that must be surrendered to widgets, thereby maximizing the available space for display of content

[22]. Their prototype is a text-based online newspaper, which supports limited interactivity, i.e. simple navigation between stories, searching for text strings, etc. They emphasized the use of semi-transparent widgets with a delayed response model, laid over text, so that both are present on the screen at the same time, but the text is able to fill nearly the entire screen. Usability studies of this research pointed out that after a learning period, the test subjects are able to select underlying screen objects, regardless of whether those objects were text or icons (Figure 2.10).

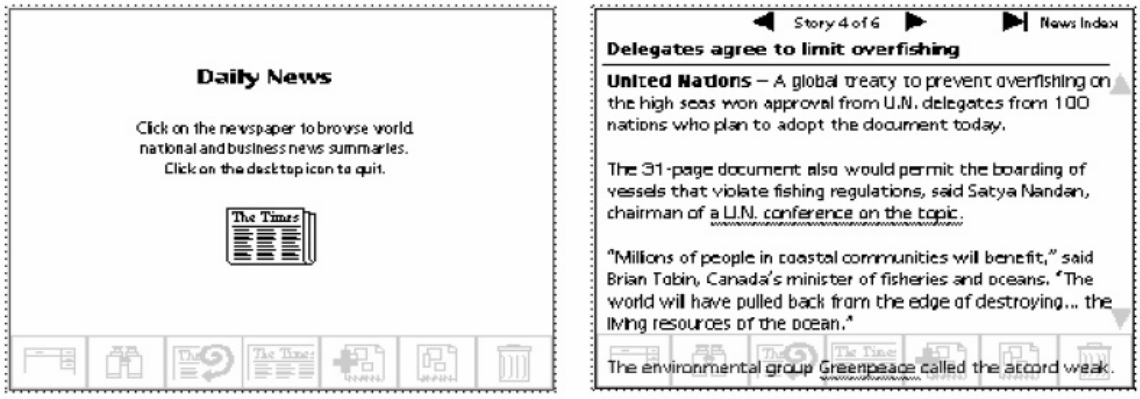


Figure 2.10 Typical screens in the prototype, with semi-transparent widgets (icons) [22].

2.3.2 View Management for AR Applications

Designing a 2D or 3D graphical user interface (UI) for augmented reality applications is a challenge in different manners;

- according to the changing viewing direction, the UI components must be relocated to maintain visibility,
- virtual objects can disappear in front of the real world scene because of lighting and rendering parameters,
- any annotations or virtual objects can overlap each other in a crowded scene.

Such problems have been discussed by researchers under the term, view management. Bell et al. defined view management for interactive 3D user interfaces as

of maintaining visual constraints on the projections of objects on the view plane, such as locating related objects near each other, or preventing objects from occluding each other [7]. Azuma and Furmanski handled this discussion from 2D point of view and according to their research; view management is about the spatial layout of 2D virtual annotations in the view plane of augmented and mixed reality applications [6].

Rose et al. used AR to annotate parts of an automobile engine in a mechanical repair scenario [34]. The visibility of the automobile engine's component annotation points is checked during rendering with a Z-buffer query. Hence visible ones are annotated. Also, they supplied a layout subsystem which arranges the annotations to provide the most readable display (Figure 2.11).

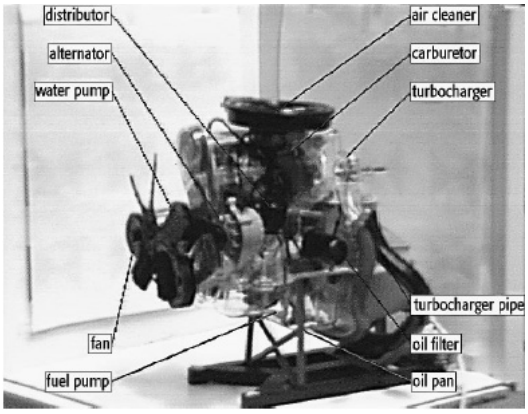


Figure 2.11 AR view of an engine with component annotations [34]

A view management component for 3D user interfaces in virtual and augmented reality applications is introduced by [7]. Each scene object is approximated by the upright rectangular extent of its projection on the view plane. Based on this approximation, they use rectangular space heuristics to determine any occlusions between objects, and binary space partitioning tree algorithm to obtain visible spaces in the 3D scene. Once the available scene fragments are identified, annotations are applied with appropriate font or image properties. According to their proposed algorithms, each object in the scene is constrained by some properties, which are visibility, position, size, transparency and priority. The system's temporal continuity is implemented and preserved on annotating a 3D city model based collaborative augmented reality environment using HMD display (Figure 2.12).

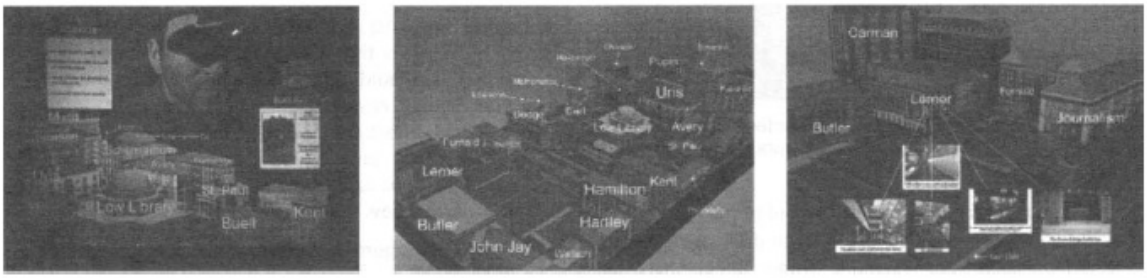


Figure 2.12 View management techniques applied to a 3D city model in collaborative AR environment [7]

Rather than dealing with general view management issues as introduced by [7], Azuma and Furmanski focused on evaluating the placement of 2D virtual labels that identify information about real counterparts (Figure 2.13) [6]. After evaluating three different real-time label placement algorithms (greedy depth first placement, discrete gradient descent and simulated annealing), they introduced a new cluster-based label placement method, which was originally designed for an air traffic control visualization application, and compared its results against the other methods. Their new method identified clusters, which are groups of labels, and then for each cluster, new sets of label positions are stochastically generated, where each set changed the position of every label simultaneously. The evaluation included both a statistical analysis and an empirical study guided by principles from human cognition. According to numerical analysis followed through, their new cluster-based method recorded the best average placement accuracy while requiring only relatively moderate computation time.

Figure 2.13 An audio mixer with knobs and dials are labeled in the video see-through system. Randomly chosen positions of labels (left), repositioned labels through cluster-based method (right) [6].

Other than the layout of annotations, text readability is affected from the interference of the background texture in the dynamically changing AR environment. Leykin and Tuceryan introduced a pattern recognition approach to automatically determine if a text placed on a particular background would be readable or not (Figure 2.14) [27]. They designed a total number of seven supervised classifiers that used both text and texture features, which are font size, font weight, intensity contrast and responses to a bank of Gabor filters convolved with the background texture, and utilized pattern recognition techniques. According to their research, the best classification results are given by support vector machine classifier with >87% classification accuracy. In fact, they didn't use color as a feature.

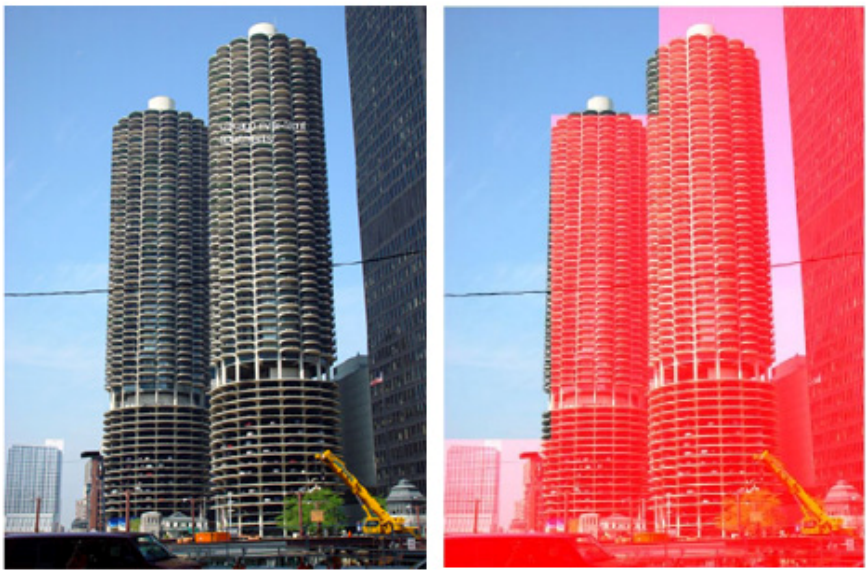


Figure 2.14 Text is placed randomly over the building, clearly unreadable (left). According to the results obtained from the classifier, red regions are marked as unreadable (right) [27].

2.4 Mobile AR Systems Hardware

2.4.1 Computing Devices

The core of a Mixed or Augmented Reality platform is a computing machine charged with the information retrieval and management, image synthesis, and acting as a crossroad between the different devices worn by the user (displays, trackers) and the environment (GPS, internet connections, etc). Since many years, the best candidate to

perform this task has been a laptop PC. In fact, since few years, notebooks come with an excellent computational power, onboard Wi-Fi and Bluetooth interfaces and a true embedded 3D graphics accelerator which dramatically increases the rendering capabilities of such machines. The choice of a laptop PC has been so far the best compromise.

Notebooks have weight and size restrictions imposed as mobility and wearability constraints:

- their hard-disks work badly when stressed by continued vibrations and shocks and may also be damaged,

- they become rapidly hot (especially when 3D acceleration is used) and their fans need openings to efficiently compensate the heating.

Peternier et al. tested the heating problem with a low-end assembled laptop (2Ghz Intel Pentium Centrino with an ATI Radeon Mobility 9700 graphic card) and a high-end model (IBM Thinkpad T43P with a FireGL 3200 3D accelerator) [31]. They concluded that after about 30 minutes of intensive 3D rendering, both computers showed temperature above the 40°C. This fact leads to some problem when the notebook is closed into a backpack or near the user's body. Finally, they are heavy and cumbersome, current generation average weight is above 2 kilos.

Nevertheless, some compact notebook model exists and features attracting specifications under reasonable encumbering constraints, but at the price of reduced 3D accelerated capabilities in regard to the full-sized laptop. For example, Dell's Latitude X1, weighting only 1.2 kilos and measuring 29x20x2.5 cm, or the tiny Oqo 01 are interesting machines which should be used at least as a maintenance update to typical backpack-based wearable frameworks. Despite of their reduced size and weight, they still suffer the same heating and fragility problems the full-sized notebooks showed. PDAs seem to solve some of those problems.

PDAs, originally conceived as evolution of handheld agendas, have greatly grown in versatility over the years and they are more and more getting similar to a compact

PC. In fact, the current generation features a powerful RISC microprocessor (today with a clock rate between 400 and 700MHz) with a Wi-Fi adapter, 640x480 VGA embedded display, audio card and a good set of expansion tools, like compact cameras, GPS and GSM adapters, micro hard-disks, USB/Bluetooth interfaces and external monitors. Mobile by definition, PDAs are an excellent compromise between the computational power offered by notebooks and a truly wearable and lightweight device: they are less sensible to shocks and do not need special cooling cares. Furthermore, they are cheaper than a laptop PC. Some of them are also equipped with a 2D and, recently, a 3D acceleration chip, like Dell's Axim x50v featuring PowerVR's MBX Lite graphics processor. Thanks to the heavy improvements on the software side, PDAs are today reasonably easy to program and use in new domains far from the simple schedule or address book they were originally conceived for.

Handheld gaming devices and mobile phones need also to be mentioned. Modern consoles like Sony's PlayStation Portable or Nintendo's GameBoy DualScreen offer great 3D rendering and computational power in compact sizes and weight. Unfortunately, these platforms are closed systems, entertainment-oriented and difficulty usable out of their original context. Moreover, the game industry does not seem to be interested in offering their hardware for Augmented or Mixed Reality applications. A few exceptions exist, like Sony's Eyetoy, or have existed, like Nintendo's Virtual Boy in 1995, but are far from being a worldwide topselling product or standard.

Mobile phone constructors are also implementing in their products PDA-like functionalities, on models like Motorola's A1000 or Sony Ericsson's P910i, as well as gaming features, like on Nokia's NGage. Their graphics and computational capabilities are also increasing thanks to a new generation of 3D acceleration chips conceived explicitly for mobile phones (like NVidia's GoForce and ATI's Imageon). Similar considerations about PDAs and handheld gaming devices are applicable to this category as well. It may also worth to cite some futuristic technologies foreseeing mobile phones with embedded see-through retinal displays. This approach would eventually eliminate the need of an external head-mounted display to superimpose synthesized images with reality, further improving the mobile aspect of a wearable Mixed Reality platform.

2.4.2 Display Devices

Both notebooks and PDAs do not own semitransparent displays to mix synthesized images with the real world scene. In order to achieve this effect, an external specific device needs to be connected. In a wearable Mixed Reality context, the ideal way to do that is using a see-through HMD or camera mounted PDA.

2.4.2.1 HMD

Two different categories of see-through head mounted displays exist: video-see-through and optical-see-through.

Video-see-through head-mounted displays have closed non transparent monitors and need an external camera to bring reality into the screens (Figure 2.15). To see the augmented virtual objects oriented to the real world while moving the HMD a tracking system is needed. In a video-see-through HMD this can be done with the built-in camera. Some commercial applications which use this kind of HMDs include visualizing 3D new designs, augmented video-see-through holograms, reconstruction of ancient ruins and monuments, design and visualization of architectural projects. Unfortunately, users may feel uncomfortable with those devices.

Optical-see-through head-mounted displays dispose of semi-transparent screens which allow the user to see the real world behind the lenses, like if she/he was wearing standard glasses. The user still perceives the reality directly through her/his own eyes. In optical-see-through systems an optional integrated camera or inertia sensors mounted on the user's head can be used for tracking purpose. This kind of HMDs are envisioned to be ideal for wearable computing, hence they are still prototyped. Diverse areas of mixed reality applications can take advantage of such displays, i.e. multimedia and telemedicine, training in medical, repair and maintenance of industrial equipment and machinery, etc.

Figure 2.15 Video-see-through HMD supporting 800x600 resolution and 40° FOV diagonally (left), monocular optical-see-through HMD prototype supporting 800x600 resolution and 32° FOV diagonally (right) [38]

The monocular displays which further improve user comfort and spatial perception by keeping one user's eye completely free, which are lighter and more wearable, particularly important when walking or running around. Trivisio is prototyping a monocular optical-see-through HMD especially usable with laptop (Figure 2.16) [38]. Other interesting models are Micro Optical's SV-6 PC Viewer [29] or Icuiti's M-920CF [19], explicitly conceived to work with a PDA. Unfortunately, these models do not belong to the optical see-through category. Peternier et al. built a wearable mixed reality framework using PDA as computing device and pointed out that Liteye's Liteye-500 is the best candidate for this setup, because it is lightweight, supports high resolution and is monocular optical-see-through supplied by a powered USB port [31]. Also Shimadzu's Data Glass 2/A is another candidate for this system (Figure 2.16).

Figure 2.16 Shimadzu's Data Glass 2/A (left) [37], Liteye's Liteye-500 (middle) [28], Icuiti's M-920CF with a PDA setup (right) [19].

2.4.2.2 PDA screen

PDA's screen itself can be used as a video-see-through display in mixed reality applications. An integrated camera or an external camera mounted on the handheld device with tolerable refresh rates (~10 Hz) can offer the magic lens effect, which means displaying the synthetic information through the PDA screen onto the real environment thereby acquiring additional data which cannot be perceived directly from the real world. Wagner and Schmalstieg introduced such a framework which supports optical tracking with an optional back-end server support (Figure 2.17) [41].

Figure 2.17 Handheld AR system tracking optical markers real-time [41].

2.4.3 Positioning Devices

2.4.3.1 GPS

The geo-localization standard, GPS is a network of 24 satellites that continuously transmit coded information, which makes it possible to precisely identify locations on earth by measuring distance from the satellites (Figure 2.18) [15]. GPS satellites, which are powered with solar energy, circle the earth twice a day in a very precise orbit. GPS receivers take this information and use triangulation to calculate the user's exact location.

Figure 2.18 GPS satellites in orbit and a GPS receiver capable of navigation [15].

A GPS receiver must be locked on to the signal of at least three satellites to calculate a 2D position (latitude and longitude) and track movement. With four or more satellites in view, the receiver can determine the user's 3D position (latitude, longitude and altitude). Once the user's position has been determined, the GPS unit can calculate other information, such as speed, bearing, track, trip distance, distance to destination, sunrise and sunset time and more.

GPS accuracy, which is typically about to 15 meters, largely depends on both the quality of the signals received and the receiver. The factors that affect GPS accuracy are ionosphere and troposphere delays, signal multipath, receiver clock errors, orbital errors, number of satellites visible, satellite geometry/shading, and intentional degradation of the satellite signal. This accuracy can be further improved by using a Differential GPS (DGPS). Differential GPS adds terrestrial signal emitters to use with signals sent from satellites. By knowing their spatial coordinates, the accuracy can raise up to 1-5 meters. Unfortunately, Differential GPS works only on DGPS covered area, while standard GPS works worldwide. Assisted GPS is a GPS technology often used by the mobile phone because it reduces the power requirements of the mobile phone and increases the accuracy of the location obtained. A server assisted GPS method, gpsOne, hands most of the heavy-duty calculations to the server over and the gpsOne equipped client mobile device supports only the measuring process [40]. This positioning service switches its method to triangulation, where the GPS satellite beams are impossible to reach. Such a correction mechanism is mostly necessary in *urban canyons*, which are

formed by buildings prevent GPS units from seeing enough satellites to get a “fix” position.

Due to the growing request, an impressive amount of GPS devices for notebooks and PDAs has been developed and is continuously updated with new products. Actually, it is straightforward to find a low-consumption, compact GPS unit weighting less than 100 grams. Although, GPS suffers one major drawback, it works only outdoor.

2.4.3.2 Wi-Fi

Alternative positioning techniques have also been created using non-localization specific devices, like GSM networks and Wi-Fi cards with access points. Howard et al. [16], and Cheng et al. [12] confirmed the feasibility of the Wi-Fi technique. Both technologies use the quality of signal received from different access points (APs) to estimate the position of devices within a known area. This approach works very well in a high-density Wi-Fi covered zone, either outdoor or indoor, but requires precise spatial coordinates of every available access point and some calibrations before being used.

The main advantage of using an already existing Wi-Fi infrastructure for this purpose is that specific GPS hardware is no longer required, reducing the weight and the price of a wearable localization-capable framework. Moreover, both technologies (GPS and Wi-Fi based positioning) can be used together to track the user when far from a Wi-Fied area or when inside of buildings, and thus out of the scope of GPS satellites.

Chapter 3 MOBILE AR SYSTEM

3.1 Hardware Components

The proposed mobile AR system provides a context-aware interaction framework for pedestrian, wandering in urban environments. As reported in relevant research based on navigation, it is important to acquire what the user's geographic position is and where she is looking at. In addition to these context-attributes, this research provides a natural interaction mechanism to the navigation system by inferring application dependent arm posture. These services are implemented and tested on a PDA based handheld prototype mounted with a GPS receiver and inertial orientation tracker (Figure 3.1). Functionalities and connections of hardware devices are presented in the following table:

Table 3.1 Hardware components of the prototype and their functionalities.

Hardware	**Type**	**Connection**	**Functionality**
Computing Device	PDA	- Offering Bluetooth service - Serial Port with a Sync port to RS-232 cable	-2D&3D rendering -Computations
Geo-localization Device	GPS Receiver	Via Bluetooth link to PDA	Latitude, longitude, heading, speed
Tracking Device	Inertial Orientation Tracker	Via Serial Port (RS-232) to PDA	-Roll, pitch, yaw -Angular Velocity -Heading (towards magnetic north)

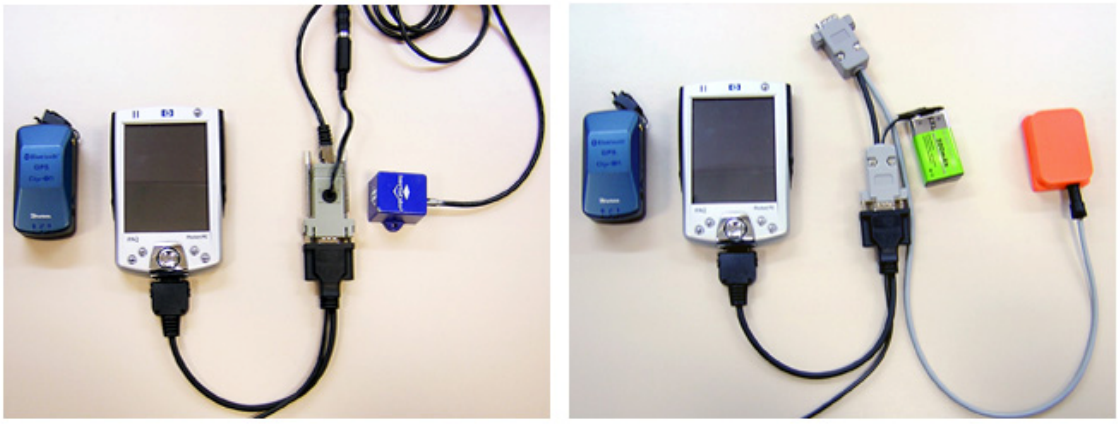


Figure 3.1 Hardware components of the prototype.

3.1.1 PDA

A basic PDA usually includes a clock, date book, address book, task list, memo pad and a simple calculator. One major advantage of using PDAs is their ability to synchronize data with desktop PCs and laptops. Pocket PC is a handheld computer and a specific type of PDA running Microsoft's Windows Mobile, Pocket PC edition operating system. Pocket PCs can be easily used with add-ons, like GPS receivers, cameras, RFID readers, and external keyboards.

We used a HP iPAQ Pocket PC h2200 series, with Intel XScale 400MHz processor, 64MB RAM, and 320x240 pixels 64K color TFT display (Figure 3.2). The version of its system is Pocket PC 2003. It has also an integrated Bluetooth chip and in this research GPS receiver connects with Pocket PC over Bluetooth. However, our Pocket PC doesn't support any serial or USB ports. Therefore an additional sync port to serial cable is needed to connect serial device hardware, i.e. inertial sensors. Pocket PC can be easily connected to a PC using a compatible USB cradle over its Sync port and synchronize data with ActiveSync, which is standard synchronization software for Pocket PCs. This capability eases the development process.

In the following sections, the terms PDA and Pocket PC are used interchangeably.

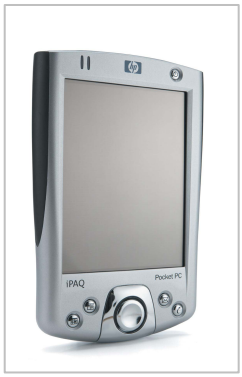


Figure 3.2 HP iPAQ h2200 series Pocket PC.

3.1.2 GPS Receiver

To acquire the exact geographic coordinates of the user, Fortuna's Clip-on Bluetooth GPS receiver is used (Figure 3.3). The Clip-on can work with any Bluetooth enabled device and this device's Bluetooth service must support SPP (Serial Port Profile), which is based on RFCOMM (Radio Frequency Communication) [14]. The Bluetooth protocol RFCOMM is a simple set of transport protocols, providing emulated RS-232 serial ports up to sixty simultaneous connections of a Bluetooth device at a time.

It is a lightweight device, only 95 g, measuring 74x41.5x30 mm, and can be easily carried by hanging with its neck cord. It has a removable Lithium-Polymer battery which works at 5V DC, 1200mAh. Positioning accuracy is about 10 meters; also in DGPS enabled area the accuracy increases up to 1-2 meters. Detailed technical specifications can be examined on Appendix A.

The Clip-on uses SiRF IIe/LP, which is a GPS microcontroller chip manufactured by the SiRF Corporation and responsible for interpreting signals from GPS satellites and deriving a position, based on the time it takes radio waves to travel from the satellites to the location of the chip. In addition, it has integrated SiRF Xtrac chip and supports a dual mode hardware switch between standard (ST) and XTrac (XT) chips, which is a patent pending technology. Standard mode is supported on all normal SiRF GPS Receivers using SiRF Star IIe/LP chipset and recommended to use under clear open sky conditions, where as XTrac is the firmware addition that brings in a better

stabilized and stronger signal in hard to reach places like when the user is under crowded environment, i.e. urban canyons.

Figure 3.3 Fortuna clip-on Bluetooth GPS receiver.

The GPS receiver provides data in NMEA 0183 (National Marine Electronics Association) format with 1 Hz update rate [30]. Data formatted according to this protocol consists of a sequence of ASCII characters, subdivided in different sentences identified by their first six characters: the '$' symbol; followed by the talker ID, which is always GP for GPS receivers; and the sentence ID. The identifier is followed by the sequence of actual data fields delimited by comma, and at the end an asterisk followed by a checksum value. The Clip-on provides GPS sentences beginning with following IDs as enumerated in its technical specifications: $GPGGA, $GPGSA, $GPGSV, $GPRMC, and $GPVTG. In addition, during GPS data collection and observation for this research, it is observed that the Clip-on provided also $GPMSS, $GPZDA beginning sentences (Table 3.2). Detailed sentence information having these IDs is given in Appendix B.

Table 3.2 GPS sentence ID's and provided information.

Sentence ID	**Description**
$GPGGA	GPS Fix Data
$GPGSA	GPS Dilution of Precision (DOP) and active satellites
$GPGSV	GPS Satellites in view
$GPRMC	Recommended minimum specific GPS/Transit data
$GPVTG	Track made good and ground speed
$GPMSS	Beacon Receiver Status
$GPZDA	UTC Date / Time and Local Time Zone Offset

In this research; latitude, longitude, heading (orientation angle from North), and speed information is needed and parsed from the relevant GPS sentences. The sentences beginning with $GPGGA provide latitude and longitude, while $GPVTG provides speed and heading. Sample sentences collected in Sabanci University campus area and their interpretations are given as follows:

Example 1:

$GPGGA,092858.779,4053.4723,N,02922.7091,E,2,06,1.0,187.3,M,39.7,M,1.4,0000*70
(Table 3.3)

Table 3.3 GPS sentence description of example 1.

Field	**Sample Data**	**Comments and Descriptions**
Sentence ID	$GPGGA	GPS Fix Data
UTC Time	092858.779	09:28:59 UTC
Latitude	4053.4723	40d 53.4723′ N
N/S Indicator	N	N = North / S = South
Longitude	02922.7091	29d 22.7091′ N
E/W Indicator	E	E = East / W = West
Fix Quality	2	0 = Invalid, 1 = GPS Fix, 2 = DGPS Fix, 3 = Valid PPS
Satellites Used	06	6 satellites in view
Horizontal Dilution of Precision (HDOP)	1.0	Relative accuracy of horizontal position
Altitude	187.3	187.3 meters above mean sea level
Altitude Unit	M	M = Meters
Geoid Separation	39.7	Geoid separation in meters according to WGS-84 ellipsoid
Geoid Separation Units	M	M = Meters
Time since last DGPS update	1.4	1.4 seconds
DGPS reference station ID	0000	Reference ID: 0000
Checksum	*70	Used by program to check for transmission errors
Terminator	CR/LF	

Example: $GPVTG,346.67,T,,M,2.47,N,4.6,K*63 (Table 3.4)

Table 3.4 GPS sentence description of example 2.

Field	Sample Data	Comments and Descriptions
Sentence ID	$GPVTG	Track made good and ground speed
True Course	346.67	346.67 degrees
Reference	T	T = True heading
Magnetic Course	Blank	
Reference	M	M = Magnetic heading
Speed	2.47	Horizontal speed
Units	N	N = Knots
Speed	4.6	Horizontal speed
Units	K	K = Kilometers per hour
Checksum	*63	Used by program to check for transmission errors
Terminator	CR/LF	

3.1.3 Inertial Orientation Trackers

In this research, an orientation tracker is employed to obtain,

- user's heading, and
- user's arm movement angles.

Although heading can be obtained from GPS receiver data, it isn't always reliable. When the user is moving, GPS receiver produces heading by calculating the angle between the receiver's displacement vector and north direction. Naturally, a GPS receiver provides erroneous heading data when the user stops. To maintain heading accuracy in any condition, the system must be supported with an orientation tracker, which provides accurate magnetic heading data. This requirement is also pointed out by previous research in outdoor navigation [10, 9].

The orientation tracker is also required to recognize arm posture and introduce a natural interaction mechanism. The proposed system is tested with two orientation trackers: Xsens's MTx and InterSense's InertiaCube2.

3.1.3.1 XSens's MTx

MTx is a miniature 3-DOF inertial measurement unit manufactured by XSens. It has integrated 3D magnetometers (3D compass), with an embedded processor capable of calculating roll, pitch and yaw drift-free in real time, as well as outputting calibrated 3D linear acceleration, rate of turn (gyroscope) and earth magnetic field data. Orientation of human body segments can be measured with this unit accurately. Also, its other sample fields of use are biomechanics, exercise and sports, VR, animation, and motion capture.

MTx's low power DSP (Digital Signal Processor) runs a proprietary sensor fusion algorithm developed by XSens [44]. The algorithm's functionality can be explained as follows: the measurement of gravity (accelerometers) and magnetic north (magnetometers) compensate for otherwise unlimited increasing drift from the integration of rate of turn (gyroscope) data (Figure 3.4). This type of drift compensation is often called attitude and heading referenced and such a system is often called an Attitude and Heading Reference System (AHRS).

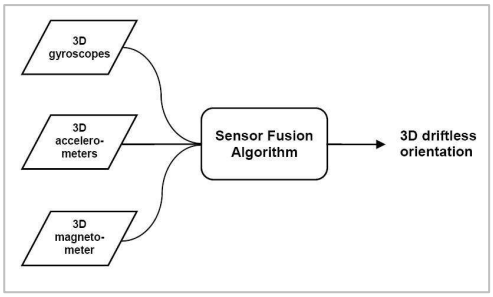


Figure 3.4 Functional structure of MTx

Data update rate of MTx is user settable, with a maximum of 120 Hz. The data provided by MTx is in two different formats: Calibrated data (acceleration, rate of turn, earth magnetic field) and Orientation data (roll, pitch, yaw).

The communication interfacing of MTx is provided either over USB port.

3.1.3.2 InertiaCube2

The InertiaCube2 is an inertial 3-DOF orientation tracking system, ideal for head tracking in simulation and training applications [21]. It obtains its motion sensing using a miniature solid-state inertial measurement unit, which senses simultaneously 9 physical properties: angular rate of rotation, linear accelerations and earth magnetic field along three perpendicular axes. The angular rates are integrated to obtain the orientation (yaw, pitch, and roll) of the sensor (Figure 3.5).

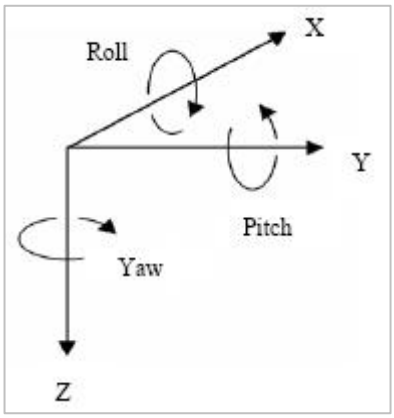


Figure 3.5 3-DOF Orientation: Rotation around Z-axis (Yaw), rotation around Y-axis (Pitch), and rotation around X-axis (Roll)

The geometry and composition of this inertial measurement unit is proprietary, but functionally it consists of gyroscopes, accelerometers and magnetometers for three axes (Figure 3.6). The basic computation of orientation from gyroscopic angular rates provides the very rapid dynamic response and high resolution of the system. The accelerometers and magnetometers are used to stabilize the orientation to the earth's gravitational and magnetic fields, thus eliminating the gradual but unbounded accumulation of gyroscopic drift errors.

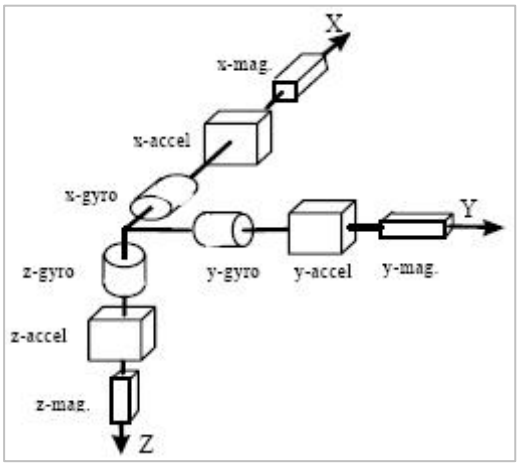


Figure 3.6 Functional structure of the inertial measurement unit

The InertiaCube2 plugs into RS-232 or USB port of a computer and it is compatible with Windows CE, 95, 98, 2000, NT and XP. Its sampling rate is 180Hz and SDK is supported for Win, Linux and Mac OS.

3.1.4 Connections and System Delay

In this work, we emphasized on how to connect the hardware components effectively while preserving mobility. The first step was to implement a navigation system working on PDA. For positioning purposes a GPS receiver is connected to PDA over Bluetooth. Because GPS receiver didn't perceive satellite beams indoors, GPS data is collected for possible navigation scenarios and these GPS logs are used during development phase. To implement arm posture recognition and receive correct heading while user is stopping, an orientation tracker is mounted to PDA. For orientation tracker connection, three different configurations are investigated:

1. Plugging MTx to USB port of a Laptop, and sending MTx data over Bluetooth to PDA (Figure 3.7),

2. Connecting MTx to PDA through a serial adapter cable (Figure 3.1 – right),

3. Connecting InertiaCube to PDA through a serial adapter cable (Figure 3.1 – left).

The first connection schema confounds the system's mobility constraint by adopting a Laptop connection in between. But MTx allowed only a USB interfacing, where PDA didn't offer any USB host port. The only connection idea was to mount MTx to a USB host offering heavily mobile Laptop, and run an application on Laptop to receive data from MTx and sending it over Bluetooth to PDA. Such a system isn't preserving mobility and an indirect Bluetooth connection between MTx to PDA is causing a system delay. Ideally, MTx should be connected to PDA directly.

Figure 3.7 Connection schema consisting of GPS receiver, PDA, Laptop, and MTx (left-to-right).

To connect MTx directly to PDA, an iPAQ serial adapter cable is purchased, which adds a 9-Pin DB9 male COM port to PDA's sync port for attaching any RS-232 serial device. In addition, serial interfacing of MTx is supplied by cutting off its data cable and solder with a 9-Pin DB9 female COM port. Power is supplied to this connection with a 9V battery. The operating supply voltage interval of MTx is 4.5V – 15V. Unfortunately, after running some tests with the whole prototype, MTx stopped working and didn't show any peak current at startup. This is due to a design shortcoming of MTx.

Intersense's InertiaCube2 package included a RS-232 interface for the device with 6V DC power adapter, so it is directly connected to PDA's serial adapter cable. After experiencing a power problem with MTx, we didn't risk InertiaCube2 for the sake of the development. We didn't plug any battery package to the system so it is only working indoor with previously collected GPS data.

The PDA used in this research is powerful enough, although not perfect for high-end visualization. But visualization quality is not the main constraint of the proposed system. As introduced by Lamberti et al., if a good quality level for visualization is needed, rendering can be carried out to a powerful server [25]. In their work, PDA connects to the server through wireless network and displays rendered models as a video sequence. Such client-server architecture suffers of limited bandwidth, networking delay, packet loss, and complex algorithms for the preparation of data to be transmitted. The proposed system in this thesis doesn't involve client-server architecture, thus avoiding networking problems. Necessary data is stored on PDA's main memory and calculations, rendering, and interfacing external devices are carried out on PDA.

3.2 Software Components

Software in the mobile and ubiquitous computing area is expected to be modular, simply modifiable to accommodate for new user needs, expectations, and a constantly changing environment.

To implement the navigation application a diverse set of APIs are integrated to the development environment:

- Microsoft eMbedded Visual C++ 4.0
- Microsoft eMbedded Visual C++ 4.0 Service Pack 4
- Pocket PC 2003 Software Development Kit (SDK)
- Vincent Mobile 3D Rendering Library (An OpenGL|ES implementation)
- UG Windowing Library
- GLUT|ES Windowing Library
- Virtual COM Port Library for Bluetooth Connection

- MTx SDK (A Pocket PC 2003 platform compatible version is developed over MTx's Windows PC platform SDK)

- InertiaCube2 SDK for Pocket PC 2003

To synchronize data with Pocket PC 2003 environment ActiveSync must be installed to the development PC. This is standard synchronization software used by the development environment, which is eMbedded Visual C++. In addition, eMbedded Visual C++ Service Pack 4 is installed to support services for Windows CE 5.0, 4.2, 4.1, and 4.0 based devices. The prototype system involves a Windows CE 4.2 based Pocket PC. Finally, Pocket PC 2003 SDK must be installed to create Pocket PC 2003 applications with eMbedded Visual C++.

All software components of the system are explained in the following subsections according to their functionalities (Figure 3.8).

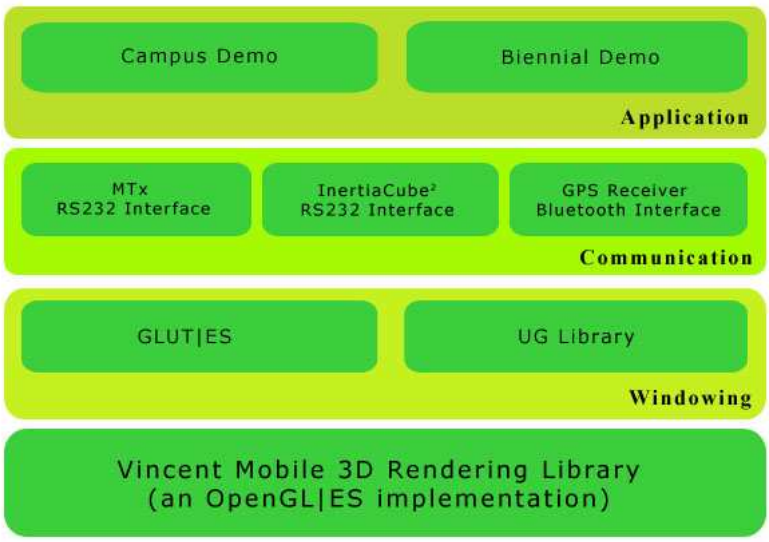


Figure 3.8 Software Layers

3.2.1 Rendering

Until 2003, since the first version of OpenGL|ES was released by Khronos group, gaming and graphics APIs for PDAs and cell phones were proprietary. OpenGL|ES is a

royalty-free, cross-platform API for full-function 2D and 3D graphics on embedded systems - including handheld devices, appliances and vehicles [23].

In this research, we used OpenGL|ES 1.1 (released in 2004), which is a subset of OpenGL 1.5 specification. Main modifications on OpenGL are delivered by Khronos Group to face the low-power and relatively limited computing capability of embedded devices: redundant APIs, functionality and data types are removed; expensive features (such as *texturing*) are limited; new versions of APIs are created to support smaller data types; GLU (OpenGL Utility Library) is discarded; new APIs supporting fixed point are implemented. The last limitation is especially important because many embedded devices don't include floating point support in hardware. All limitations of OpenGL|ES, its detailed specifications and comparison to OpenGL can be checked out on the web site of Khronos group.

For rendering purposes an OpenGL|ES implementation, Vincent Mobile 3D Rendering Library, is used. Libraries and include files extracted from this package must be copied to the Windows CE SDK directories on the development PC and the necessary dynamic link library must be copied to the Pocket PC. Installation details to setup the environment can be followed on the web [45].

3.2.2 Windowing

UG library, which comes standard in the Vincent library package, is used to simplify window creation. It provides GLUT (OpenGL Utility Toolkit) like functionalities: window creation, callbacks for keys, stylus and display methods, idle and reshape functions, etc. Although it is acceptable, GLUT|ES library, which is released by student researchers at University of Bordeaux, provides more features and it is more stable.

GLUT|ES is a GLUT implementation for WinCE and Win32 systems based on OpenGL|ES. Most of GLUT API v.3 functionalities are adopted in GLUT|ES such as window creation, callbacks, menus, idle routines and timers, support for stylus and keys, support for UNICODE strings with TrueType fonts, etc. GLUT|ES is compatible with Vincent library. GLUT|ES API 1.0 is used in this research for windowing purposes other than UG library.

3.2.3 Communication

Following communication modules are implemented in the prototype application:

- Bluetooth connection to GPS receiver,
- Serial port connection to MTx,
- Serial port connection to InertiaCube2 (Figure 3.8).

MTx and InertiaCube2 are never mounted to Pocket PC simultaneously, but one of them and GPS receiver communicate with the Pocket PC through their communication modules in the application framework at the same time. Though, there is an asynchronous communication mechanism built in the software to receive GPS data and orientation tracker data without blocking data flow each other.

Bluetooth Connection to GPS Receiver

Bluetooth communication is created using Microsoft Windows CE COM Port Emulator facility. The COM port emulator is the topmost layer of the Bluetooth Protocol Stack and provides access to RFCOMM based on a virtual COM port. Thus, the protocol stack enables devices to locate each other and establish a connection. To create the connection to the GPS receiver,

- One outgoing RFCOMM channel is opened,
- Outbound COM port of Pocket PC's Bluetooth framework (in this case it is COM8) is opened by using standard I/O function, *CreateFile*, to read data,
- Port settings, which are specified in Data Communication Block (DCB), are initialized with 38400 baud rate, one stop bit, no parity bits, 8 bits data length, and corresponding timeout values.

After opening Bluetooth connection, GPS data is read asynchronously until an instance of GPS connection is completed, which means in this work, one $GPGGA and

one $GPVTG sentences are read. The latitude, longitude, speed, and heading data are parsed from those sentences, and this navigation data is sent to the application layer of the software.

Serial Port Connection to MTx

MTx is mounted to the Pocket PC; some test data is collected and tested a few times successfully. Unfortunately it stopped working because of inconsistent voltages provided by different battery packages, despite the fact that the voltage range was always in the supply voltage interval as indicated in its technical specifications.

The SDK that XSens provided for MTx was only compatible with standard Windows PC platform. Therefore, some changes are made on this SDK to ensure that it works on Pocket PC platform. Most of the platform dependent problems are based on the character representation of two systems: Pocket PC platform uses UNICODE format, Windows PC platform uses ASCII format. All multi byte char (ASCII) based instructions are converted to wide char (UNICODE).

To create a connection with MTx,

- COM1 (sync port of Pocket PC) is opened at 115200 baud rate,
- Some configurations are done and output data format is set to Euler angles,
- Update rate is set to 120 Hz.

Afterwards, MTx data is read and roll, pitch, yaw angles and angular velocity of pitch angles are parsed. Yaw angle is heading to the magnetic north. Pitch angles and its angular velocity are the parameters for arm posture recognition system. Roll angles determine any possible tilt errors.

Serial Port Connection to InertiaCube2

Opening a connection and receiving data from InertiaCube2 is straightforward. COM1 (sync port of Pocket PC) is opened and roll, pitch, yaw angles are perceived at each step.

Even though Intersense provided a SDK of InertiaCube2 for Pocket PC 2003 platform, some character representation problems occurred and fixed as implemented for MTx's SDK.

To sum up, ideally, GPS receiver and an orientation tracker are connected to the Pocket PC and the system works outdoors. But necessary operating power is supplied to the orientation tracker only with a 6V DC adapter, thus limiting the working area as indoors. There is a functioning battery pack connection needed for a complete outdoor system.

Chapter 4 CASE STUDY: A MOBILE CONTEXT-AWARE PEDESTRIAN NAVIGATION APPLICATION

After connecting necessary hardware and building software components, the prototype system is studied with a context-aware pedestrian navigation application. Location, orientation, and activity of the user are the affecting context attributes. As a standard navigation system working outdoor, this application locates the user using GPS data and gives information about the point of interests around. Moreover, it offers different interfaces by changing them seamlessly according to the user's arm posture. This feature is achieved by mounting the orientation tracker to the Pocket PC's rear. Currently, power is supplied to the orientation tracker only with a power adapter. A battery pack connection must be built carefully. Therefore, the interface transition mechanism could only be tested indoor with previously collected GPS data.

The navigation application is tested for two urban environments and it is possible to add different environment data and run the navigation system without doing massive changes in code. This modularity increases the scalability of the prototype.

Test environments are:

- Sabanci University campus area (Figure 4.1)
- 9th International Istanbul Biennial venues in Beyoglu, Istanbul (Figure 4.2)

Figure 4.1 Three states and user interfaces for Sabanci University environment.

Figure 4.2 Environment selection interface and biennial demo.

4.1 Navigation in Urban Environment

During implementation of the navigation application, some design challenges are exceeded:

- Mapping buildings' and user's earth coordinates onto the local coordinate system of PDA screen,

- Relocating buildings according to user's heading,
- Representing point of interest buildings,
- Placing text labels of buildings while maintaining their visibility.

Buildings' earth coordinates (latitude and longitude values of their center) are collected from Google Earth and placed in a text file in the following format inspired by GPS data format of NMEA:

$PPCBA,MDBF,$PPCKO,40.89063,29.37923

Thus, $PPCBA indicates that building name comes afterwards, and $PPCKO is followed by latitude and longitude values of the building. This formatting allows adding more buildings and building properties to the scene by editing this text file.

3D Interface Mode

Buildings are placed onto a terrain texture or a plan of the environment in the 3D interface mode of the application. This texture file must be in targa format (.tga) having 8 bits, 24 bits, or 32 bits per pixel, and file size must be powers of 2, i.e. 256x256, 512x512 (these formats are used in this work), 256x128, 128x128, 64x64, etc. Minimum and maximum pixel values of building positions on the image file must be extracted to place the buildings correctly.

2D Interface Mode

In 2D interface mode, buildings are placed according to the user's position and heading. Because user's gaze direction (heading) is simulated using Pocket PC's heading. The angle between each building and user's gaze direction is calculated and after some distance calculations, building is placed onto its screen position at each frame (Figure 4.3).

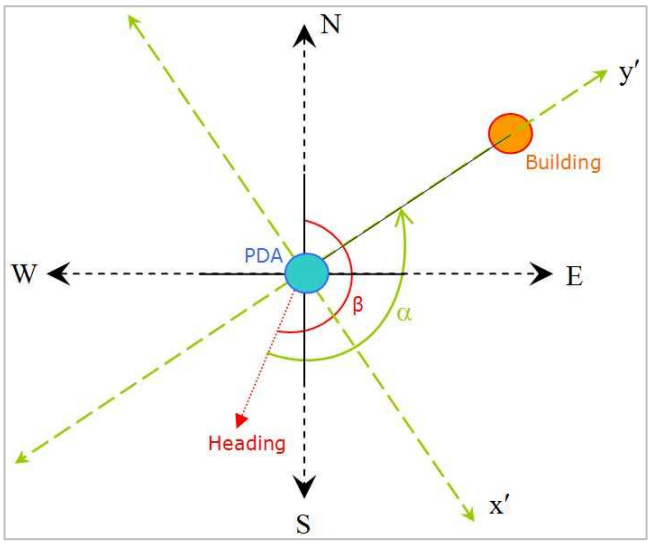


Figure 4.3 Direction calculation for buildings

Each building is represented with a rectangular box or a rectangle, which is scaled of a factor according to the distance between building and user. As the user approaches a building its scaling factor increases.

In addition, buildings are annotated with text labels, which are created with bitmapped fonts. The standard OpenGL|ES does not provide any vector font engine, thus there is a need to create and display characters efficiently. Bitmapped font is such a technique that all characters of a font are stored into a texture, where the requested character is created as a texture according to its offset in font texture and mapped and displayed onto a quad using blending options (Figure 4.4). Although GLUT|ES provides TrueType fonts, bitmapped font representation technique is more useful in this application where text labels are scaled and translated according to their position to the camera and to the buildings they annotate. The viewpoint camera, which is moved using the 4-way joystick of Pocket PC, is included in the 3D interface module to navigate in the environment.

Figure 4.4 Font characters

4.2 Posture Recognition

As presented in Chapter 2, all of the orientation tracker work is based on either to assist precise tracking and positioning of the user in space or gesture recognition using several sensors. In this work, the goal is to create a stable differentiation mechanism between several arm postures and map them to several application dependent contexts.

The developed recognition algorithm is based on state transitions triggered by time-line analysis of orientation and angular velocity of the sensor.

The angle between user's forearm and upper arm is obtained from the orientation sensor as pitch angle, α, and analyzed to recognize different postures. We have gathered sample data from mobile users with various walking speeds, while moving their hands between three postures:

- *vertical*, where pitch angle is around 0°,
- *horizontal*, where pitch angle is around 90°,
- *idle*, where the hand may move freely (Figure 4.5).

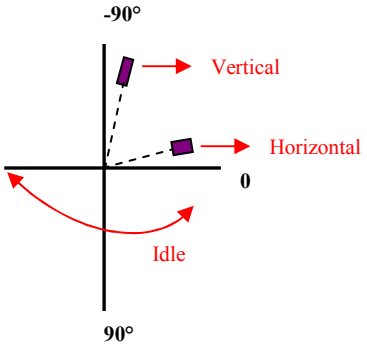


Figure 4.5 Drawing of target three hand postures from side view

Figure 4.6 shows pitch angle measurements of the user's arm movement in three different conditions: standing, walking, and running. Transitions between diverse arm postures can be inferred from the top left plot of Figure 4.6: For $0s \leq t < 5s$, the posture is on idle state. After this interval the user moves her hand up and stabilizes on horizontal posture until $t \approx 10s$. For $10s < t < 20s$, the user moves her hand down, stabilizes on idle state and moves her hand up. For $20s \leq t < 27s$, vertical posture is observed, and so on.

The measurements indicate that with the increase of velocity the noise on the measured signal increases significantly. The noise can be observed on the top right plot of Figure 4.6, where the transition from idle posture to horizontal posture is not clearly recognizable at $t \approx 40$. Our current algorithm performs acceptably with users walking with low speed but the accuracy decreases significantly with increased speed due to the high frequency noise introduced into data by walking and running motion.

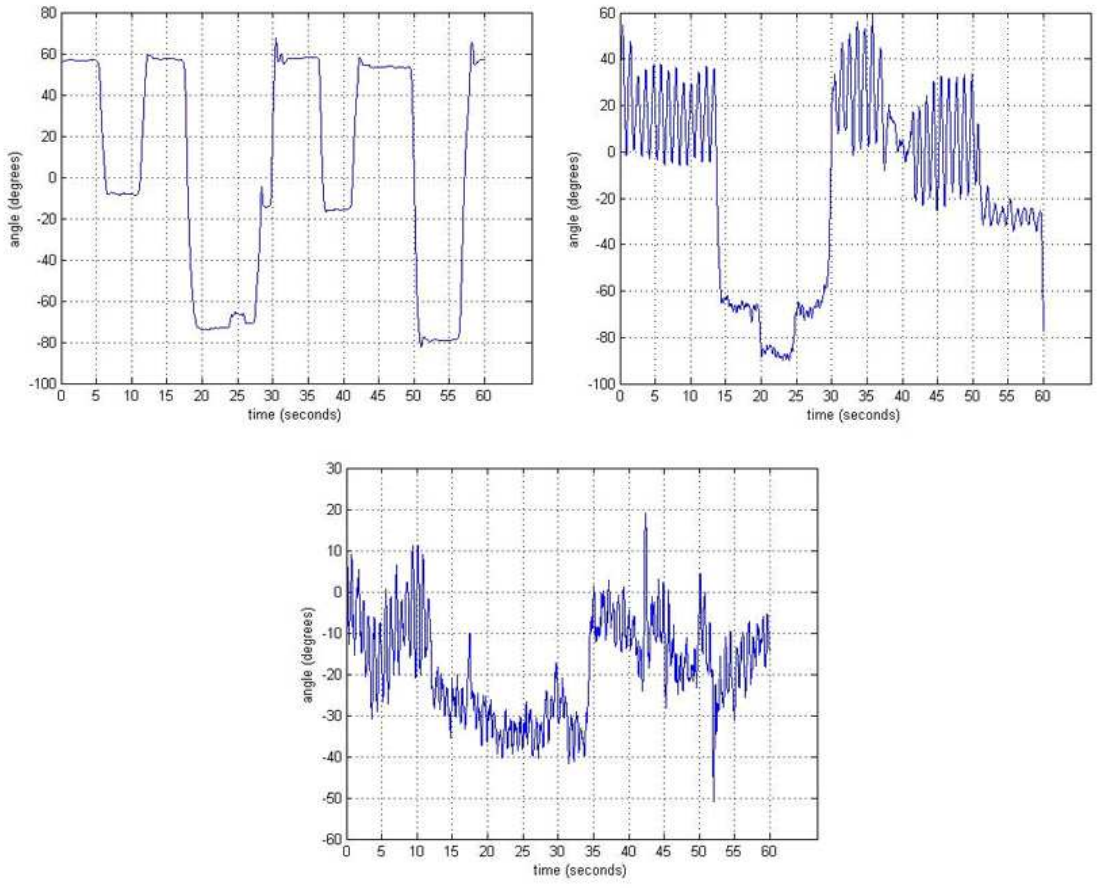


Figure 4.6 Pitch angle measurements while user is stationary (top left), walking (top right), and running (bottom). Data is collected with MTx.

We implemented a sliding window to detect changes of the hand on pitch angle, α. A window, which contains five angle values obtained in time interval [t-1, t-5], is created at each time step and upcoming angle is estimated by multiplying them with increasing weights.

$$0.1 * \alpha_i + 0.1 * \alpha_{i+1} + 0.1 * \alpha_{i+2} + 0.2 * \alpha_{i+3} + 0.5 * \alpha_{i+4} = \alpha_{estimated} \qquad (4.1)$$

The $\alpha_{estimated}$ angle is compared with the measured angle α_{i+5} to identify if the hand is moving up or down.

$$\alpha_{i+5} > \alpha_{estimated} \Rightarrow downside\ change \qquad (4.2)$$

$$\alpha_{i+5} < \alpha_{estimated} \Rightarrow upside\ change \qquad (4.3)$$

$$\alpha_{i+5} \cong \alpha_{estimated} \Rightarrow no\ change \begin{cases} \alpha_{i+5} \rightarrow -90^{\circ}, vertical \\ \alpha_{i+5} \rightarrow 0^{\circ}, horizontal \end{cases} \quad (4.4)$$

However using the pitch angle in one single direction is not sufficient enough to have robust posture recognition. We have also evaluated the case, where the user performs short tilts (rotations around the longitudinal axis) causing an inference on the state transition. For such cases, a filter is implemented on the system which increased the state transition accuracy.

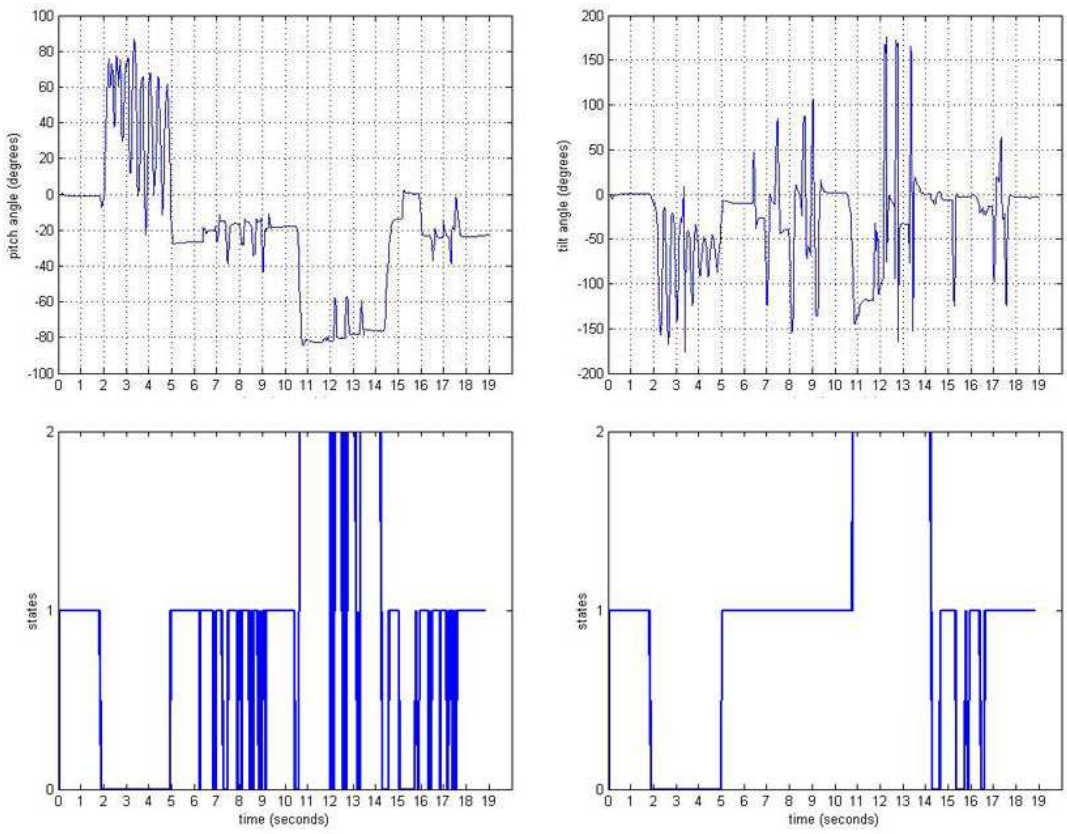


Figure 4.7 Sample pitch angle measurement (top left). Tilt measurement of the same posture (top right) causing erroneous estimation – 0: idle state, 1: navigation state, 2: investigation state (bottom left) and increased estimation accuracy with tilt filter (bottom, right). Data is collected with InertiaCube2.

Figure 4.7 shows plots of sample pitch and tilt angle measurements of the same motion and corresponding state estimations. For 7s< t <9s, tilt angle is increasing and decreasing instantly (top right plot, Figure 4.7) which affects the pitch angle. In spite of the fact that the user holds her hand stable around -20° during angular data

measurement, the top left plot of Figure 4.7 shows that the pitch angle is changing up to 20°. Same erroneous measurement can be observed for 11s < t < 15s. These unexpected changes cause inaccurate state estimation (bottom left plot, Figure 4.7). Therefore, estimation accuracy is increased (bottom right plot, Figure 4.7) by introducing the system with a tilt angle filter, which locks the state to the previous one if major changes occur on tilt angles.

The system becomes unstable and produces erroneous results when users perform other occasional movement patterns. Therefore we have introduced an additional data, angular velocity, to the recognition system. The change of angular velocity together with the angle allows us more stable recognition results.

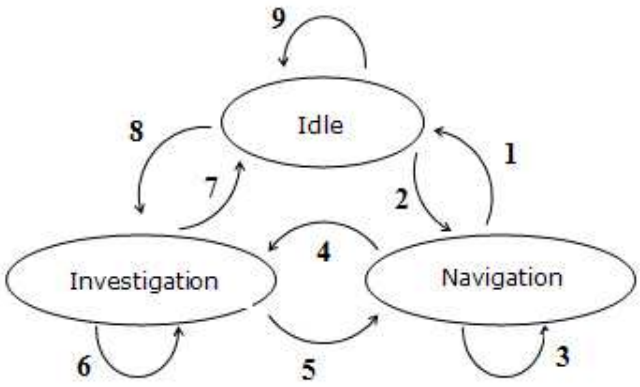


Figure 4.8 State transitions

Finally we developed a finite state machine to map all possible postures into one of the three states: investigation, navigation and idle (Figure 4.8). The investigation state is when a user holds a mobile terminal in vertical position to use it in an Augmented Reality context. In this condition the user needs to investigate point of interest buildings and receives environmental information according to her gaze direction in the local coordinate frame. The navigation state is when a user holds a mobile terminal in horizontal position to use it to render maps or GIS information. Thus, the user receives environmental information in the global coordinate frame (Figure 4.9 and Figure 4.10). There is a third idle state, where the user is not in either posture and moves her hand freely. In this state, rendering is minimized to allow power save property.

The conditions satisfying the state transitions in Figure 4.8 are defined in Table 4.1. In this algorithm, firstly, the estimated pitch angle value is compared with the angular value perceived from the orientation sensor at that time step. If they are approximately equal, user's arm posture is estimated to be stable and either in investigation or navigation state (3rd and 6th columns of Table 4.1). Other enumerated transitions include conditions which define possible changes between states, i.e. while arm posture is on idle state and the user moves her hand upwards, then it is possible to switch state to navigation or upside change on arm posture continues and state is switched to investigation. The state estimation algorithm is empowered by introducing angular velocity and tilt angle filter to the system.

Table 4.1 Description of State Transitions

1	2	3	4	5	6	7	8	9
$\alpha > 0°$ $\omega < -0.75$ $\Delta\alpha < 0$	$\alpha > 0°$ $\omega > 0.75$ $\Delta\alpha > 0$	$\alpha \approx 0°$ $\omega \approx 0$ $\Delta\alpha \approx 0$	$\alpha < 0°$ $\omega < -0.75$ $\Delta\alpha < 0$	$\alpha < 0°$ $\omega > 0.75$ $\Delta\alpha > 0$	$\alpha \approx 90°$ $\omega \approx 0$ $\Delta\alpha \approx 0$	$\alpha > 0°$ $\omega > 0.75$ $\Delta\alpha > 0$	$\alpha < 0°$ $\omega < -0.75$ $\Delta\alpha < 0$	$\alpha > 0°$ $\lvert\omega\rvert > 0.75$ $\Delta\alpha \neq 0$

$$\omega = angular\ velocity\ (rad/s)$$
$$\Delta\alpha = \alpha_{true} - \alpha_{estimated}$$

Unexpected arm movements of the user can affect the accuracy of the system. While the user holds the PDA in horizontal or vertical position (navigation or investigation state) and suddenly performs fast upward or downward movements with her hand, i.e. waving to somebody, the system is stabilized in the former state with a tolerably accuracy rate.

We performed a user study to examine the accuracy of our system. In this test, all possible state transitions emphasized in Figure 4.8 are performed. The overall accuracy rate is calculated as approximately %87. Performing sudden up-down movements in navigation state and investigation state produced some erroneous results.

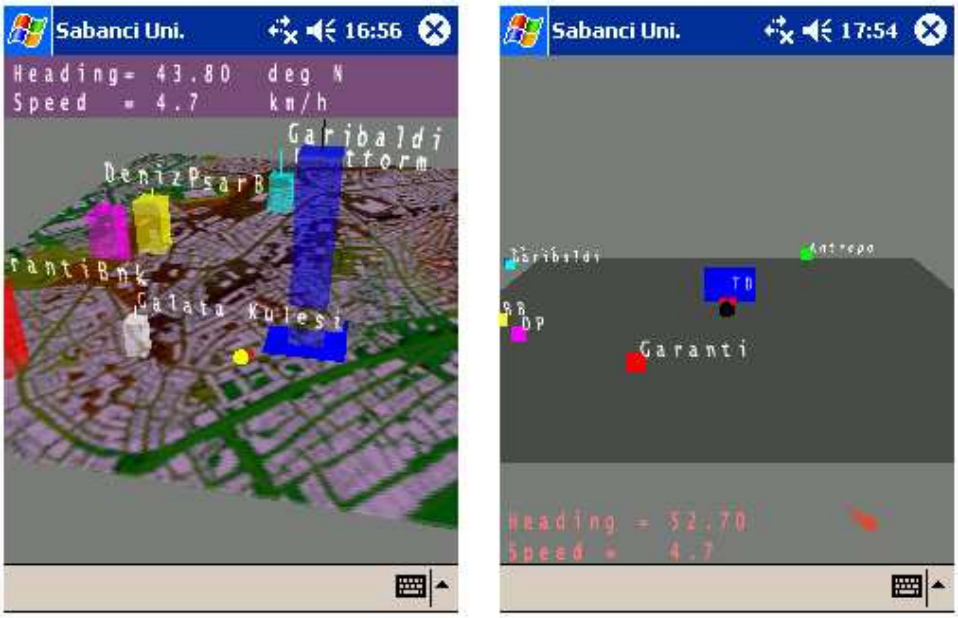


Figure 4.9 Screenshots of navigation and investigation states of Biennial environment. Yellow sphere represents the user on the left picture, black sphere represents the user on the right picture.

Figure 4.10 Screenshots of navigation and investigation states of Sabanci University campus environment..

Chapter 5 INTERACTION TECHNIQUES FOR MOBILE USERS

In case of a mobile context, users interact with the mobile device, access and modify data, and perceive environmental information continuously while the environment and user's state are dynamically changing. To minimize user's explicit interaction effort, a mobile application should be aware of the contextual attributes and offer natural interaction mechanisms to provide relevant data. Thus, offering alternatives to the most common interaction paradigm, desktop metaphor.

Mobile AR applications based on navigation frameworks try to promote interaction beyond the desktop by employing wearable sensors, which collect user's position, orientation or diverse types of activities. In this way, user is charged as her own interaction widget in the environment where the mobile framework preserves ubiquity.

5.1 Problem / Use Case

Several research groups studied interaction paradigms and offered new techniques for the mobile context. Most of them focused on developing systems which track location and heading of the user in the global coordinate frame and visualize relevant data on diverse types of UIs. Such systems offered navigation applications as case studies [10, 9, 18, 11]. On the other hand, some researchers in the wearable computing area studied angular data of human body segments in the local coordinate frame and offered interaction techniques without considering any global coordinate frame data [43, 8].

Interaction in Global Coordinate Frame

Mobile navigation systems offer information about point of interests in an unknown indoor or outdoor environment using different interaction techniques:

- User is located in the environment through the use of GPS data, and her heading is used to find which object of interest is in her gaze direction. 3D

representations of objects of interests in a city environment can be selected with a PDA stylus to perceive information about them [10].

- In a highly mobile situation, car navigation, different gaming scenarios are triggered according to the user's location, obtained from GPS receiver. Interaction with the surrounding road context is achieved using orientation tracker data [9].

- In a museum's exhibition room PDA acquires orientation information of the user and updates a panoramic view of the exhibition on the display, and user perceives detailed information about an exhibit by selecting it [11].

- A gaze-directed selection in outdoor environment is accomplished by the user orienting her head so the desired object's projection is closer than any other to the center of the head-mounted display. If it remains the closest within that area for a half-second, the object's label is smoothly changing color over that interval to confirm selection [18].

Interaction in Local Coordinate Frame

Wearable sensors allow recognizing user's activity and offering natural interaction mechanisms in local coordinate frame:

- Recognizing arm postures is used to introduce a new technique for entering text into a mobile phone. Orientation of the tilt sensor mounted mobile phone is used to resolve the ambiguity faced by standard text entry technique. Tilting the phone in one of four directions chooses which character on a particular key to enter [43].

- A set of interaction techniques, which support ad-hoc data retrieval, content manipulation and presentation while minimizing the need to look away from the projected image are developed using wrist-worn projector and computer equipped with position and orientation sensors [8].

5.2 Proposed Approach

We propose a combination of global and local coordinate frame approaches and provide a context-aware interaction framework for mobile devices by seamlessly changing Graphical User Interfaces (GUIs) for pedestrians wandering in urban environments.

As studied by researchers working on location based services, it is important to acquire a user's geographic position and where she is looking at. In addition to these context-attributes, we propose a natural interaction mechanism to the navigation system by inferring application dependent arm posture:

- **Idle state**, where a user is not looking to the screen, moving her arm and rendering is minimized,

- **Navigation state**, where a user's hand is approximately in a horizontal position, and point of interests are represented 3D in global coordinate system,

- **Investigation state**, where a user's hand is approximately in a vertical position, point of interests are placed according to the user's coordinate system and represented in 2D.

This research introduces the idea that once orientation trackers became part of mobile computers, they can be used to create natural interaction techniques with mobile computers.

Chapter 6 CONCLUSION & FUTURE WORK

Limitation

Currently, power is supplied to the Intersens' orientation tracker only with a power adapter. A battery pack connection must be built carefully. Therefore, the interface transition mechanism could only be tested indoor with previously collected GPS data. A complete system working outdoor is envisioned as future work.

As a former attempt, MTx was mounted to the Pocket PC; some test data is collected and tested a few times successfully. Unfortunately it stopped working because of inconsistent voltages provided by different battery packages, despite the fact that the voltage range was always in the supply voltage interval as indicated in its technical specifications.

Conclusions

We prototyped a pedestrian navigation system and have implemented a recognition algorithm with one orientation sensor attached to a PDA to distinguish between two different postures of the hand and an idle state. This data can be used to differentiate between three states to switch between different applications seamlessly. We tested our approach on two different orientation sensors.

In the global coordinate frame, we used GPS sensor data to locate the user, acquire her gaze direction, embed GIS data and provide information about point of interest buildings. In the local coordinate frame, we used orientation sensor data to allow the user interact with the mobile device while performing natural arm postures and perceive information on different user interfaces. By combining these interaction techniques of global and local coordinate frame, we provide a context-aware interaction framework for pedestrian navigation systems on mobile devices by seamlessly changing graphical user interfaces.

To sum up:

- Interaction mechanisms for pedestrian navigation systems can be improved by integrating angular data of human body segments, in addition to location and heading data.

- Once orientation trackers became part of mobile computers, they can be used to create natural interaction techniques with mobile computers.

- Pocket PC is the best candidate for an activity aware pedestrian navigation framework, considering its communication, rendering and computation capabilities.

- By implementing a sliding window mechanism, three arm posture states are distinguished successfully: idle, navigation, investigation.

- Distance information is given by scaling point of interests with to a factor according to the distance between a building and user, which eases to perceive information.

- Posture recognition was successful while user is stationary; error rate is few while walking.

- Mobile and ubiquitous system achieved using wearable sensors.

Future work

As a summary:

- Supply battery pack connection, test the prototype outdoors completely.

- Implementing indoor navigation system using Wi-Fi localization.

- Improving natural interaction: Tilt angle of orientation tracker navigates viewpoint camera, instead of joystick input.

- To achieve a precise state estimation while user is walking, another sensor can be included to the system, i.e. shoe-mounted inertial orientation tracker.

- Investigate environment with real-time camera input. Buildings can be annotated with virtual text labels, when arm posture is in investigation state and speed is zero.

- Improvement of visualization by using vector graphics, i.e. for terrain texture.

Bibliography

[1] Abowd, G. D., Dey, A. K., Brown, P.J., Davies, N., Smith, M., and Steggles, P. "Towards a Better Understanding of Context and Context-Awareness". In Proc. Handheld and Ubiquitous Computing: First International Symposium, HUC'99, Karlsruhe, Germany, September 1999.

[2] Abowd, G. D., Atkeson, C. G., Hong J., Long S., Kooper R., and Pinkerton M. "Cyberguide: A mobile context-aware tour guide". In ACM Wireless Networks, 1997, Vol. 3, No. 5 (Oct. 1997), pp. 421-433.

[3] Amft, O., Junker, H., and Troster, G. "Detection of eating and drinking arm gestures using inertial body-worn sensors". In Proc. 9th IEEE International Symposium of Wearable Computers, 2005. IEEE (2005) 160—163.

[4] Astle, D., and Durnil, D. "OpenGL ES Game Development". Thomson Course Technology, 2004.

[5] Azuma, R. "A survey of augmented reality". Presence: Teleoperations and Virtual Environments, 6(4):355-385, 1997.

[6] Azuma, R., and Furmanski, C. "Evaluating Label Placement for Augmented Reality View Management". In Proc. ISMAR 2003, Tokyo, Japan, IEEE (2003) 66—75.

[7] Bell, B., Feiner, S., and Höllerer, T. "View Management for Virtual and Augmented Reality". In Proc. UIST'01, Orlando, Florida, USA, ACM (2001) 101—110.

[8] Blask, G., Coriand, F., and Feiner, S. "Exploring Interaction with a Simulated Wrist-Worn Projection Display". In Proc. 9th IEEE International Symposium on Wearable Computers, ISWC 2005.

[9] Brunnberg, L. and Juhlin, O. "Motion and Spatiality in a Gaming Situation – Enhancing Mobile Computer Games with the Highway Experience". Proceedings of Interact 2003.

[10] Burigat S., and Chittaro L. "Location-aware visualization of VRML models in GPS-based mobile guides." 3D technologies for the World Wide Web: Proceedings of the tenth international conference on 3D Web technology, ACM Press, pp. 57–64, Bangor, United Kingdom, 2005.

[11] Chan, L.W., Hsu, Y.Y., Hung, Y.P., and Hsu, J.Y. "Orientation-Aware Handhelds for Panorama Based Museum Guiding System". In Proc. Smart Environments and Their Applications to Cultural Heritage, Ubicomp '05. September 11-14, 2005, Tokyo, Japan.

[12] Cheng, Y., Chawathe, Y., LaMarca, A., and Krumm, J. “Accuracy Characterization for Metropolitan-scale Wi-Fi Localization”. In Proc. 3rd International Conference on Mobile systems, Applications, and Services. Seattle, Washington. ACM (2005) 233—245.

[13] Feiner, S., MacIntyre, B., Höllerer, T., and Webster, A. “A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment”. In Proc. International Symposium on Wearable Computers, pp 74-81, 1997.

[14] Fortuna, http://www.fortuna.com.tw/clip.htm, Accessed July 2006.

[15] Garmin, http://www.garmin.com, Accessed July 2006.

[16] Howard, A., Siddiqi, S., and Sukhatme, G. S. “An Experimental study of Localization Using Wireless Ethernet”. The 4th International Conference on Field and Service Robotics, July 14—16, 2003.

[17] Höllerer, T. H., and Feiner, S. K. “Mobile Augmented Reality”. Telegeoinformatics: Location-Based Computing and Services, Chapter 6.

[18] Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., and Hallaway, D. “Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system”. Computers & Graphics 23, 779—785, 1999

[19] Icuiti, http://www.icuiti.com, Accessed July 2006.

[20] Rogers, Y., Sharp, H., and Preece, J. Interaction Design – Beyond Human-Computer Interaction.

[21] InterSense, InertiaCube2. http://www.isense.com/products/prec/ic2/index.htm, Accessed July 2006.

[22] Kamba, T., Elson, S. A., Harpold, T., Stamper, T., and Sukaviriya, P. “Using small screen space more efficiently.” Proc. CHI '96 (Vancouver BC, April 1996), ACM Press, 383-390.

[23] Khronos Group, OpenGL|ES API, http://www.khronos.org/opengles/1_X/, Accessed July 2006.

[24] Kulju, M. and Kaasinen, E. “Route Guidance Using a 3D City Model on a Mobile Device”. Workshop on Mobile Tourism Support, Mobile HCI Symposium, Pisa, Italy, 2002.

[25] Lamberti, F., Zunino, C., Sanna, A., Fiume, A., and Maniezzo, M. “An Accelerated Remote Graphics Architecture for PDAs”. In Proc. Web3D 2003, ACM Press, New York, 55—61.

[26] Lee, S., and Mase, K. “Activity and Location Recognition Using Wearable Sensors”. IEEE Pervasive 1 (2002) 24—32.

[27] Leykin, A., and Tuceryan, M. “Automatic Determination of Text Readability over Textured Backgrounds for Augmented Reality Systems”. In Proc. ISMAR 2004, Arlington, VA, USA, IEEE (2004) 224—230.

[28] Liteye, http://www.liteye.com, Accessed July 2006.

[29] Microoptical, http://www.microopticalcorp.com, Accessed July 2006.

[30] National Marine Electronics Association (NMEA), http://www.nmea.org, Accessed July 2006.

[31] Peternier, A., Vexo, F., and Thalmann, D. “Wearable Mixed Reality System in Less Than 1 Pound”. Euographics Symposium on Virtual Environments, 2006.

[32] Preece, J., Rogers, Y., and Sharp, H. “Interaction Design: Beyond Human-Computer Interaction”, John Wiley & Sons, 2002.

[33] Rakkolainen, T., and Vainio, T. “A 3D city info for mobile users”. Computers & Graphics, 25 (4), 619625, 2001.

[34] Rose, E., Breen, D., Ahlers K. H., Crampton, C., Tuceryan, M., Whitaker, R., and Greer, D. “Annotating Real-World Objects Using Augmented Reality”. Computer Graphics: Developments in Virtual Environments (Proceedings of CG International '95 Conference), (Leeds, UK), pp. 357-- 370, June 1995.

[35] Salber, D. “Context-awareness and multimodality”. Proc. First Workshop on Multimodal UI, 2000.

[36] Shimada, S., Tanizaki, M., and Maruyarna, K. “Ubiquitous spatial-information services using cell phones”. Micro IEEE, Volume 22, Issue 6, Nov.-Dec. 2002, Page(s):25 – 34.

[37] Shimadzu, http://www.shimadzu.com, Accessed July 2006.

[38] Trivisio, http://www.trivisio.com/products.html, Accessed July 2006.

[39] Vlahakis, V., Ioannidis, N., Karigiannis, J., Tsotros, M., Gounaris, M., Stricker, D., Gleue, T., Daehne, P., and Almeida, L.”Archeoguide: An Augmented Reality

Guide for Archaeological Sites". IEEE Computer Graphics and Applications, Vol. 22, no. 5, pp. 52-60, 2002.

[40] gpsOne by Qualcomm. http://www.cdmatech.com/products/gpsone.jsp, Accessed July 2006.

[41] Wagner, D., and Schmalstieg, D. "First Steps Towards Handheld Augmented Reality". In Proc. Seventh IEEE International Symposium on Wearable Computers, 2003. IEEE (2003) 127–135.

[42] Weiser, M. "The computer for the 21st century". Scientific American, pp. 94—104, September 1991.

[43] Wigdor, D., and Balakrishnan, R. "TiltText: Using Tilt for Text Input to Mobile Phones". In Proc. UIST 2003. ACM Press, 2003.

[44] XSens, MTx. http://www.xsens.com/download/MTx_leaflet.pdf, Accessed July 2006.

[45] ZeusCMD OpenGL|ES Tutorials, http://www.zeuscmd.com/tutorials/opengles/, Accessed July 2006.

Appendix A

Technical Specifications of Clip-On Bluetooth GPS Receiver (Fortuna)

Chipset:	SiRF Star IIe/LP and SiRF XTrac
Frequency:	L1, 1575.42 MHz
C/A:	1.023 MHz chi rate
Internal Antenna Type:	Built-in ceramic patch antenna
External Antenna Type:	MCX

Accuracy

Position:	10 meters, 2D RMS 7 meters 2D RMS, WAAS corrected 1-5 meters, DGPS corrected
Velocity:	0.1 meters/second
Time:	1 microsecond synchronised to GPS time

Datum	WGS-84

Acquisition Time (ST)

Reacquisition:	0.1 sec
Cold:	45 sec
Warm:	38 sec
Hot:	8 sec
Snap Start:	2 sec

Acquisition Time (XT)

Reacquisition:	100 msec
Continuous:	1 sec
Cold:	45 sec
Warm:	38 sec
Hot:	4 sec

Sensitivity XT Mode

Tracking:	16dB-Hz
Hot Start:	23dB-Hz
Warm Start:	28dB-Hz
Cold Start:	32dB-Hz

Dynamic Conditions

Altitude:	18,000 meters (60,000 feet) max
Velocity:	515 meters/second (1000 knots) max
Acceleration:	<4g

Main Interfaces

Connection:	Communication with host platform via Bluetooth Serial Port Profile
Protocol Messages:	NMEA 0183 output protocol
Baud Rate:	38400 bps

Data Bit:	8
Stop Bit:	1
Output Format:	GGA (1 sec), GSA (1 sec), GSV (5 sec), RMC (1 sec), VTG (1 sec)

Power

Operational Power:	3.3vDC
Input Power:	5vDC
Battery Source:	rechargeable and removable Lithium Polymer battery with 5vDC input charging circuit (1200 mA)
Operating Time:	8.5 hours continuous operation after full charge
Battery Charge Time:	3 hours approx
Battery Low:	approx 30 mins to stop operation

Environmental

Operating Temperature:	-20C to +60C
Humidity Range:	5% to 95% non condensing

Dimensions

Length:	74mm
Width:	41.5mm
Height:	30mm
Weight:	95g

Appendix B

Detailed Descriptions of GPS Sentences

$GPGGA

Global Positioning System Fix Data: Time, position and fix related data for a GPS receiver

eg. $GPGGA,170834,4124.8963,N,08151.6838,W,1,05,1.5,280.2,M,-34.0,M,,,*75

Name	Example Data	Description
Sentence Identifier	$GPGGA	Global Positioning System Fix Data
Time	170834	17:08:34 UTC
Latitude	4124.8963, N	41d 24.8963' N or 41d 24' 54" N
Longitude	08151.6838, W	81d 51.6838' W or 81d 51' 41" W
Fix Quality: - 0 = Invalid - 1 = GPS fix - 2 = DGPS fix	1	Data is from a GPS fix
Number of Satellites	05	5 Satellites are in view
Horizontal Dilution of Precision (HDOP)	1.5	Relative accuracy of horizontal position
Altitude	280.2, M	280.2 meters above mean sea level
Height of geoid above WGS84 ellipsoid	-34.0, M	-34.0 meters
Time since last DGPS update	blank	No last update
DGPS reference station id	blank	No station id
Checksum	*75	Used by program to check for transmission errors

$GPGSV

GPS Satellites in view.

eg. $GPGSV,3,1,11,03,03,111,00,04,15,270,00,06,01,010,00,13,06,292,00*74
$GPGSV,3,2,11,14,25,170,00,16,57,208,39,18,67,296,40,19,40,246,00*74
$GPGSV,3,3,11,22,42,067,42,24,14,311,43,27,05,244,00,,,,*4D
$GPGSV,1,1,13,02,02,213,,03,-3,000,,11,00,121,,14,13,172,05*62

1 = Total number of messages of this type in this cycle
2 = Message number
3 = Total number of SVs in view
4 = SV PRN number

5 = Elevation in degrees, 90 maximum
6 = Azimuth, degrees from true north, 000 to 359
7 = SNR, 00-99 dB (null when not tracking)
8-11 = Information about second SV, same as field 4-7
12-15= Information about third SV, same as field 4-7
16-19= Information about fourth SV, same as field 4-7

$GPRMC

Recommended minimum specific GPS/Transit data

eg. $GPRMC,225446,A,4916.45,N,12311.12,W,000.5,054.7,191194,020.3,E*68

225446	Time of fix 22:54:46 UTC
A	Navigation receiver warning A = Valid position, V = Warning
4916.45,N	Latitude 49 deg. 16.45 min. North
12311.12,W	Longitude 123 deg. 11.12 min. West
000.5	Speed over ground, Knots
054.7	Course Made Good, degrees true
191194	UTC Date of fix, 19 November 1994
020.3,E	Magnetic variation, 20.3 deg. East
*68	mandatory checksum

(for NMEA 0183 version 3.00 active the Mode indicator field is added)

1 = UTC time of fix
2 = Data status (A=Valid position, V=navigation receiver warning)
3 = Latitude of fix
4 = N or S of longitude
5 = Longitude of fix
6 = E or W of longitude
7 = Speed over ground in knots
8 = Track made good in degrees True
9 = UTC date of fix
10 = Magnetic variation degrees (Easterly var. subtracts from true course)
11 = E or W of magnetic variation
12 = Mode indicator, (A=Autonomous, D=Differential, E=Estimated, N=Data not valid)
13 = Checksum

$GPVTG

Track Made Good and Ground Speed.

eg1. $GPVTG,360.0,T,348.7,M,000.0,N,000.0,K*43
eg2. $GPVTG,054.7,T,034.4,M,005.5,N,010.2,K*41

054.7,T True course made good over ground, degrees
034.4,M Magnetic course made good over ground, degrees
005.5,N Ground speed, N=Knots
010.2,K Ground speed, K=Kilometers per hour

eg3. for NMEA 0183 version 3.00 active the Mode indicator field is added at the end

$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K,A*53
A Mode indicator (A=Autonomous, D=Differential, E=Estimated, N=Data not valid)

$GPMSS

Beacon Receiver Status

eg1: $GPMSS,55,27,318.0,100,*66

55 signal strength in dB
27 signal to noise ratio in dB
318.0 Beacon Frequency in KHz
100 Beacon bitrate in bps
*66 checksum

eg2: $GPMSS,0.0,0.0,0.0,25,2*6D

Field	Example	Comments
Sentence ID	$GPMSS	
Signal strength	0.0	Signal strength (dB 1uV)
SNR	0.0	Signal to noise ratio (dB)
Frequency	0.0	Beacon frequency (kHz)
Data rate	25	Beacon data rate (BPS)
Unknown field	2	Unknown field sent by GPS receiver used for test
Checksum	*6D	

$GPZDA

UTC Date / Time and Local Time Zone Offset

eg. $GPZDA,024611.08,25,03,2002,00,00*6A

Field	Example	Comments
Sentence ID	$GPZDA	

UTC Time	024611.08	UTC time
UTC Day	25	UTC day (01 to 31)
UTC Month	03	UTC month (01 to 12)
UTC Year	2002	UTC year (4 digit format)
Local zone hours	00	Offset to local time zone in hours (+/- 00 to +/- 59)
Local zone minutes	00	Offset to local time zone in minutes (00 to 59)
Checksum	*6A	

Appendix C

InertiaCube2 Technical Specifications

Degrees of Freedom	3 (Yaw, Pitch and Roll)
Angular Range	Full 360° - All Axes
Maximum Angular Rate*	1200° per second
Minimum Angular Rate*	0° per second
Accuracy*	1° RMS at 25°C
Angular Resolution*	0.01° RMS
Serial Interface Update Rate	180 Hz
Minimum Latency	2 ms for RS-232 (PC host OS dependent)
Prediction	up to 50 milliseconds
Serial Rate	115.2 kbaud
Interface	RS-232 Serial (shown above)
Size	28.89 mm x 24.38 mm x 33.91 mm
Weight	25.00 grams
Cable Length	4.572 m. - Max. 22.86 m
Power	6 VDC, 100 milliamps via AC adapter
Operating Temperature Range	0° to 50° C
O/S Compatibility	.dll for Windows 98/2k/NT/XP/CE .so for Linux and SGI IRIX libisense.dylib for Mac OS X
Software Support	SDK with full InterSense API Windows Control & Connectivity Software Ethernet w/ Windows Control Software

*Measurements with perceptual enhancement algorithm turned off (= 0)

Optional USB Adapter Specifications

InterSense USB Update Rate	180 Hz (Windows 98/2000/XP) 180 Hz (Macintosh OS X)
USB Interface Minimum Latency	2 ms for USB direct (PC host OS dependent)
Power Source	Direct from Host USB Port
USB Adapter Size	60 mm x 35 mm x 20 mm
Cable Length	3 meters