Human Intelligible Positioning

by

Vishwanath Venugopalan

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2004

Author .....
Department of Electrical Engineering and Computer Science
January 16, 2004

Certified by .....
Robert C. Miller
Assistant Professor
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Human Intelligible Positioning

by

Vishwanath Venugopalan

Submitted to the Department of Electrical Engineering and Computer Science
on January 16, 2004, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science and Engineering

Abstract

We use street addresses to refer to locations in a city. Street addresses are easy to remember and communicate because they follow a symbolic addressing scheme, containing human intelligible symbols. However, street addresses can often be ambiguous or confusing and don't provide complete coverage of outdoor spaces. Latitude and longitude coordinates, a metric addressing scheme, are unambiguous and accommodate locations that may not have street addresses. However, latitude and longitude coordinates are unusable on a daily basis because they must be specified to many digits to be useful at human-level scales. This thesis describes the design and implementation of a new hybrid addressing scheme, Human Intelligible Positioning (HIP), which uses a metric addressing scheme as its substrate. Addresses in this metric addressing scheme are mapped to two-dimensional offsets within named coordinate systems. HIP addresses combine the easy memorability and communicability of street addresses with the precision and universal outdoor coverage of latitude and longitude coordinates.

Thesis Supervisor: Robert C. Miller
Title: Assistant Professor

Acknowledgments

I would like to dedicate this thesis to my parents and sister, whose blessings and constant support enabled me to study computer science at MIT. Without their emphasis on the value of education and the importance of hard work, this thesis would not have been possible.

I thank my thesis supervisor, Prof. Rob Miller, for his technical élan, energy, support and feedback. This thesis is greatly inspired by his vision for pervasive location modeling and intelligent user interfaces. I also thank the faculty members in the EECS and Brain and Cognitive Science departments, who laid the bedrock for this thesis by imparting to me a distinctive mix of knowledge about computers and human minds.

Finally, special thanks go to my colleagues in Prof. Miller's research group, Matt Notowidigdo, Michael Bolin and Alisa Marshall and the students in *6.893: User Interface Design and Implementation*, whose insights helped improve the usability of HIP. I thank all the marvelous individuals who participated in the user studies for this thesis. I thank Bryt Bradley and Adel Hanna of the Computer Graphics Group for their kindness and tremendous willingness to help. Heartfelt thanks to Karolína, whose love and support made several long nights worthwhile. Thanks also to my friends on Fourth East for their support and for always being there when I needed them.

Contents

Chapter 1

Introduction

In recent years, there has been a significant growth in technologies for the domain of ubiquitous computing. Ubiquitous computing envisions a world with cheap hardware and network resources, where computing extends beyond its traditional stronghold of desktop computing into a diverse assortment of "smart devices". The prevalence of devices such as personal digital assistants (PDAs) and cellular phones, coupled with recent advances in mobile and wireless technology research stands witness to the relevance of ubiquitous computing for the future. When computing shifts to mobile devices that are untethered from static desktops, the physical location of these devices assumes an important role. This thesis presents Human Intelligible Positioning (HIP), a technique for location modeling that combines the positive attributes of existing location models to extend their utility and reach.

1.1 Motivation

This section mentions good and bad properties of existing address schemes and clarifies how HIP addresses can be used.

1.1.1 Existing addressing schemes

Locations in a city may be located using two prevalent wide-area addressing schemes: street addresses and latitude and lontigude addresses.

Street addresses

Street addresses are commonly used to refer to locations in a city. Street addresses are one example of a symbolic addressing scheme. Street addresses are easy to memorize and communicate to other people. However, they have two disadvantages:

- Local idiosyncracies and ambiguities. Street addresses formed through a process of localized and idiosyncratic evolution. For this reason, one can get confused by frequent street names such as "Main Street". Other artifacts of this idiosyncratic evolution are observed when one street turns into another street with no indication, or is near another street of the same name in a densely populated region. Furthermore, as is the case in the Boston area, streets may not be well-labelled.

- Lack of wide coverage. Street addresses are only available for those positions that are serviced by streets. It is not possible to refer to a bench in a park or a particular door of a building using a street address.

Latitude and longitude addresses

Latitude and longitude addresses are one example of a metric addressing scheme. Latitude and longitude addresses refer to positions relative to a two dimensional imaginary reference grid superimposed on the earth's surface. Latitude and longitude addresses are completely unambiguous and can be resolved to a unique position. Furthermore, they also have complete coverage outdoors, in principle. In other words, even those positions that do not have street addresses can have latitude and longitude addresses. However, latitude and longitude addresses cannot replace street addresses because they are quite unusable at street scales. The difference in latitude and longitude coordinates between two ends of a house or a parking lot is at the order of

10^{-5} degrees. Using latitude and longitude addresses for positioning and navigation at typical human scales would require them to be specified to a total of 14-16 digits. Furthermore, the digits in a latitude or longitude address are similar, which increases the likelihood of transcription and transposition errors when communicating such an address. Thus, latitude and longitude coordinates are hard to remember and communicate because of their length and self-similarity.

1.1.2 HIP solution domain

Clearly, street addresses and latitude and longitude addresses are complementary in their strengths and shortcomings. HIP aims to improve the usability of latitude and longitude coordinates by superimposing a system of human-intelligible names on them. In doing so, it brings symbolic and metric location models together in a hybrid model. A HIP address contains a *coordinate system* and an *offset*. Coordinate systems have a fixed origin in latitude and longitude coordinates and extend over a rectangular region. They have human-intelligible names and may refer to landmarks, regions or other artifacts salient in a human context. HIP addresses may be visualized as having numerical horizontal and vertical offsets within these coordinate systems. In this way, HIP addresses can harness the precision of GPS in naming positions, but still be identifiable by human-intelligible names.

HIP addresses are intended to be used for everyday location and navigation. They could be used as postal addresses, or by people who wish to meet others at precisely specified locations. HIP addresses should be sufficiently usable and memorable that users feel comfortable exchanging them during telephone conversations and social encounters. Because HIP addresses frequently refer to human-intelligible landmarks, they could be used to give directions to lesser known positions near a well-known landmark. HIP addresses could add a whole new dimension to sports such as Geocaching[GRO] and scavenger hunts. They may also aid search and rescue missions by providing convenient and memorable points of geographical reference.

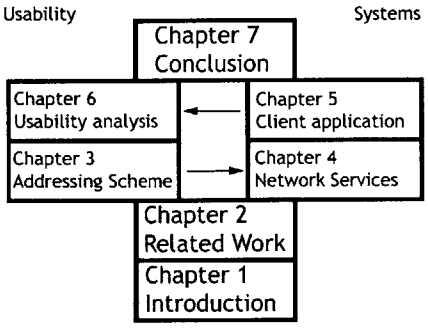


Figure 1-1: This thesis is organized along two tracks: a usability track and a systems track.

1.2 Thesis contributions

This thesis makes the following contributions:

- It defines Human Intelligible Positioning (HIP), a pervasive, wide-area, hybrid location model, which is influenced both by street addresses and latitude and longitude addressing schemes. It also considers factors essential to a usable addressing scheme in general and compares HIP with existing addressing schemes.
- It describes the system architecture and implementation of HIP's geographical information system (GIS) back-end and a client application aimed at a mobile device.
- It describes a method for testing the usability of HIP and analyzes results from a controlled user study that quantitatively and qualitatively compare HIP with existing addressing schemes.

1.3 Thesis Overview

Figure 1.3 shows a visual overview of this thesis. Work on the HIP addressing scheme has proceeded on two separate, but related tracks: a systems track and a usability track.

Chapter 2 provides some background and describes research related to HIP. Chapter 3, which belongs in the usability track, describes the design and essential properties

of the HIP addressing scheme. Chapters 4 and 5 belong in the systems track. Chapter 4 describes the network services that constitute the back-end of the HIP addressing scheme. Chapter 5 provides details about the HIP client application, its user interface and its evolution via an iterative design process. The thesis returns to the usability track with Chapter 6, which analyzes the results of a user study that compares HIP to existing addressing schemes. Finally, Chapter 7 concludes the thesis and offers suggestions for future work on HIP.

Chapter 2

Background and Related Work

This chapter provides some background pertinent to HIP and the rest of this thesis. It also recognizes research efforts similar to HIP and contrasts their design goals with those of HIP.

2.1 Terminology

The addressing schemes discussed in this thesis—street addresses and latitude and longitude pairs—may be classified abstractly as symbolic and metric addressing schemes[LEO98].

A **symbolic** addressing scheme uses symbols with linguistic meaning in its addresses. Symbolic addresses are frequently expressed hierarchically. Street addresses such as *77 Massachusetts Avenue, Cambridge, MA* contain a hierarchy of human intelligible place names and thus follow a symbolic addressing scheme.

A **metric** addressing scheme uses numeric coordinates to quantitatively express the distance of a point from a fixed origin. Latitude and longitude addresses are a metric addressing scheme. Latitudes are north-south distances from the earth's equator and longitudes are east-west distances from the Prime Meridian, a longitude line that passes through Greenwich in the United Kingdom. Latitude and longitude addresses such as *(42.359697N 71.056713W)* express these distances in degrees.

Terms such as position, location and address may have many different senses in common parlance. This thesis assigns these words well-defined meanings and uses

them in specific contexts.

The term **position** refers to a context-free, low-level property directly measurable using a global, absolute addressing scheme. Latitude and longitude coordinates will always be called positions.

The **location** of an object is an expression of its position in higher-level, non-absolute addressing schemes, such as street addresses and HIP. Locations usually need some context to be interpreted appropriately. HIP addresses need a coordinate system, whereas street addresses need a city and state or ZIP code.

An **address** is merely a name–the literal textual representation of a location. *77 Massachusetts Avenue, Cambridge, MA 02139* and *Boston/Boston Common .35 .84L* are addresses.

In the sense that the above terms are used in this thesis, addresses must conform syntactically to an addressing scheme, but need not correspond to an actual position. The string "25 Diagon Alley, Azkaban, VT 90210" apparently represents a valid street address, although the position it refers to probably does not exist. In contrast, locations and positions must correspond to places that exist in the real world. This distinction becomes relevant when we describe the ways in which HIP addresses are used later in this thesis.

Navigation is the process of finding directions from one position to another.

2.2 Related Work

This section examines technologies that form the bedrock of HIP and other efforts in location modeling which have had influences on HIP.

2.2.1 GPS & Cricket

Because HIP needs an absolute metric coordinate system to be its substrate, we will examine two absolute, metric location support systems on which HIP could be based.

The Global Positioning System (GPS) is a location support system developed by the United States military that is used to determine latitude and longitude positions[HOW].

It consists of a network of twenty-four satellites, each orbiting the earth twice a day. Their configuration ensures that every point on the surface of the earth has a line of sight to at least four GPS satellites. A GPS receiver maintains an *almanac* with the approximate trajectories of GPS satellites so that it can contact them to ascertain its own position. A GPS receiver estimates the straight-line distance to three or more GPS satellites by using a radio signal. These straight lines can be said to form the diameters of three imaginary spheres. The position of the GPS receiver can be computed from the intersection of the earth's surface with these three imaginary spheres. This process of determining one's position from three or more sources is called *triangulation.*

While the idea behind GPS location support is straightforward, its implementation runs into several practical difficulties. Radio signals from GPS satellites may get reflected from rivers or shiny skyscrapers and cause errors in a GPS receiver's distance computations. GPS service works reliably only outdoors, where reasonable line of sight to GPS satellites is available. However, in "urban canyons" such as the dense network of skyscrapers on Manhattan Island, the lack of a wide view of the sky may hinder positioning via GPS. Even when reasonable line of sight is available, GPS satellites cover the earth's surface with only a finite resolution. All latitude and longitude positions obtained using GPS have an associated uncertainty.

Cricket[PRI00] is an indoor location support system developed under Project Oxygen at MIT's Computer Science and Artificial Intelligence Laboratory. It can be thought of as an analogue of GPS for indoor use. Low-cost Cricket transmitters are distributed throughout a building. These transmitters emit radio as well as ultrasound signals simultaneously. A Cricket receiver listens for these signals and computes its distance to three or more receivers using the time lag between the electromagnetic and ultrasound signals from a receiver. This technique used by Cricket receivers is similar to how one can estimate how far away a thunderstorm is by measuring the time lag between a flash of lightning and its corresponding clap of thunder. Once the distance to three or more Cricket receivers has been calculated, a computation similar to that performed by a GPS receiver is used to determine one's indoor position.

The HIP implementation described in this thesis uses latitude and longitude coordinates obtained from GPS as a substrate for defining coordinate systems. In principle, HIP may also be based on Cricket, to provide indoor location support.

2.2.2 Japanese addresses

In Japan, street addresses, as found in the United States, are generally not used[PAN]. A typical Japanese address resembles "1-22-14 Jinan, Shibuya-Ku, Tokyo". Tokyo and other Japanese cities are divided into large sections called wards. Wards are subdivided into districts. Shibuya-ku in the above address is a ward, while Jinan is a district. The first number in the above address names a Chome, a subsection of a district. The second number names a subdivision of a Chome, usually the size of a city block. The last number is the building number within the block. The Japanese system of addressing can be viewed as successively zeroing in on a position.

The ward, districts and Chome of Japanese cities are analogous to hierarchical HIP coordinate systems. However, once an Japanese address has been localized to a block, the right building cannot always be located reliably, because building numbers are usually assigned in the order that the buildings were built. HIP addresses too enable a user to zero in on a position. However, in contrast with Japanese addresses, HIP addresses have two-dimensional Cartesian offsets that can be generalized across all coordinate systems.

2.2.3 Wide area addressing schemes

Wide area addressing schemes such as GEOREF and MGRS[DAN95] provide location and navigation services over larger regions.

The World Geographic Reference System (GEOREF), based on latitude and longitude, is used for airline navigation. The world is divided into twelve latitude zones and twenty four longitude zones, thus forming a grid of 15 degree quadrangular regions, each named by a pair of letters. Each of these 15 degree squares are divided into unit degree squares, also named by letters. Complete GEOREF addresses are

expressed as minute offsets relative to the latitude and longitude of a one degree square. An example of a basic GEOREF address is *FJHA1516*. GEOREF addresses reduce the self-similarity of latitude and longitude coordinates by using letters to refer to coordinate systems. However, because GEOREF is not intended to be used for addressing at human scales, letters are not used at scales smaller than one degree squares. Using GEOREF for everyday addressing would require specifying offsets to many significant digits, thereby rendering it less usable.

The Military Grid Reference System (MGRS) also creates rectangular regions identified by a number and a zone character, e.g. 14R. Additional characters identify 100-km-by-100-km squares within these regions. Finally, an even number of digits specify north and east offsets within these squares, to various precisions. One example of a MGRS address is *14RPU2116149894*. As many as 10 digits are needed to specify an offset to a precision of 1 meter. This specification also renders MGRS unviable for use in everyday addressing.

HIP also seeks to create coordinate systems like the two wide area addressing schemes described above. HIP coordinate systems will be named with familiar, human-intelligible names rather than names conforming to a global reference grid.

2.2.4 Intentional Naming System

The Intentional Naming System (INS)[ADJ99] was also developed under the auspices of Project Oxygen at MIT's Computer Science and Artificial Intelligence Laboratory. INS names use an expressive language of *attribute-value pairs* to identify services and resources in the immediate vicinity in a context-aware manner. For example, all public cameras in the Oval Office of the White House can be referenced with the INS name

```
[city = washington [building=whitehouse
                    [wing=west
                    [room = oval-office]]]]
[service = camera]
```

INS exists to enable scenarios such as a user requesting a document to be printed on "the nearest printer" without necessarily knowing the printer's name. An INS deployment consists of a number of name resolvers connected with a minimum spanning tree topology. These name resolvers propagate a name resolution request and responses and maintain records of known names in the system. Name resolvers are dynamically updated and destroyed in an INS deployment for load-balancing.

HIP is intended to provide context-awareness in name resolution like INS. HIP may also use name resolution techniques similar to INS, when resolving a HIP address across a wide area covered by numerous coordinate system resolvers.

2.2.5 Aura Location Identifier

The Aura Location Identifier(ALI)[JIA02] forms was developed under Project Aura at Carnegie Mellon University. ALI is a hybrid location model combining hierarchical, symbolic addresses and absolute, metric coordinates. ALI can be used to locate resources and services in the immediate vicinity. An ALI like *ali://cmu/wean-hall/floor3/3100-corridor#(10,10,0)* would be used to refer to the position of a resource in the corridor of Room 3100 on the third floor of Wean Hall at CMU. ALI provides the infrastructure for Project Aura's space service, which will enable queries such as locating printers within a certain distance of an ALI.

HIP and ALI share a high-level goal of modeling spaces using hybrid symbolic and metric identifiers. The ALI location model targets services and resources indoors. In contrast, HIP devises a hybrid location model suitable for wide-area, outdoors use.

2.3 Chapter Summary

This chapter defined some terms that will be used in the rest of this thesis and examined some research related to HIP. HIP's relatively simple central ideas bear influences from a number of research projects in location modeling. HIP builds upon previous research in location modeling to build a hybrid location model that covers a wide area, is context-aware and is easy to use.

Chapter 3

Addressing scheme design

This chapter describes the conceptual model behind the HIP addressing scheme. HIP provides a human-centric, context-aware, hybrid location model, which exhibits properties of both symbolic and metric addressing schemes. HIP's coordinate systems are based on latitude and longitude addresses. Figure 3 shows the position of Building 7, the main entrance to MIT, as a HIP address.

3.1 HIP addressing scheme

One example of a HIP address is

`US, MA, Cambridge, MIT .42 .35K`

A HIP address is composed of three parts:

- A coordinate system
- A two-dimensional offset and
- A check code

The constituent parts of a HIP address are explained below, with reference to the above address.

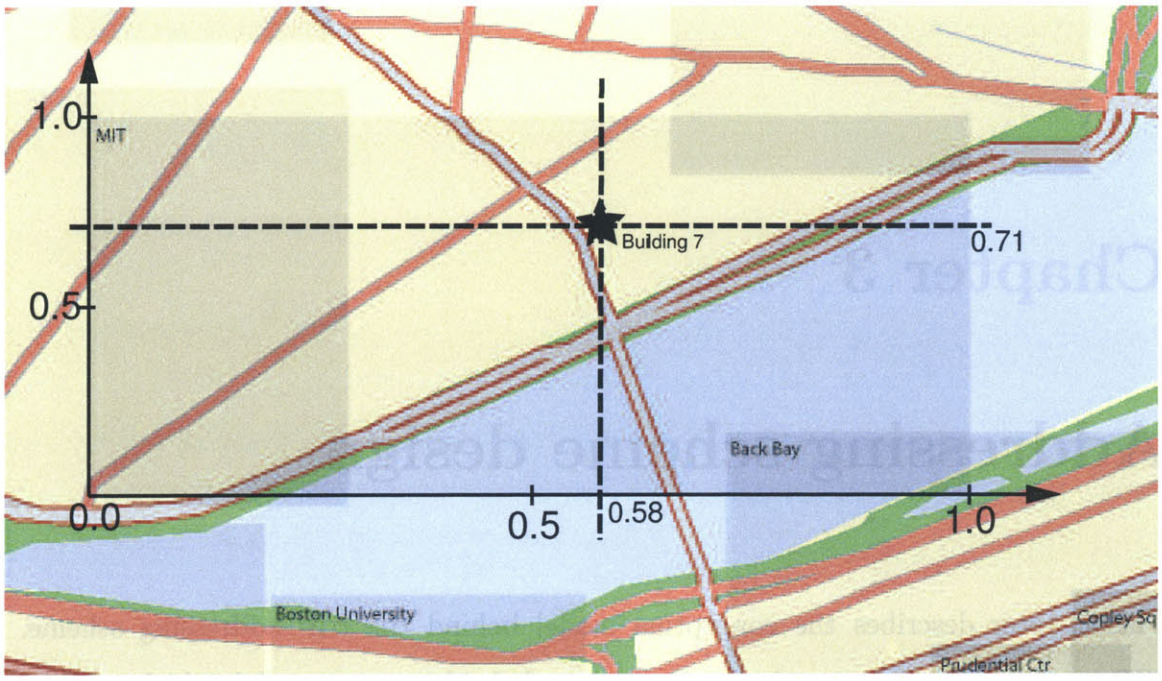


Figure 3-1: *MIT .58 .71*: the position of Building 7 at MIT expressed as a HIP address. Note the rectangular MIT coordinate system and the other coordinate systems adjoining it. The horizontal and vertical offsets of the HIP address are expressed as fractions of the length and width of the MIT coordinate system respectively.

3.1.1 Coordinate system

The coordinate system appears first in a HIP address. The coordinate system in the HIP address above is *US, MA, Cambridge, MIT*. A coordinate system is a place name with respect to which a HIP address is constructed.

A coordinate system is represented as a contiguous rectangular region(Figure 3). The rectangular region has an origin, a fixed latitude and longitude position, and width and height attributes that define its size in degrees of arc. It is possible for coordinate systems to overlap. For this reason, a single latitude and longitude position might be described by multiple HIP addresses, each expressed with respect to a different coordinate system.

The coordinate system of a HIP address has a human-intelligible name. A coordinate system may be named for any place, including towns, counties, states, and even landmarks. A coordinate system may contain other coordinate systems within

its domain. Coordinate system names represent the influence of symbolic addressing schemes on the hybrid location model that is HIP. Coordinate system names are intended to be created and named in a decentralized manner, as and when needed. Collections of coordinate systems could be created for the following purposes:

- The businesses at Harvard Square in Cambridge, MA could choose to create smaller coordinate systems around their own alcoves of Harvard Square and be incorporated into the surrounding Harvard Square coordinate system.
- The United States Postal Service could create coordinate systems named by ZIP codes, for locating delivery addresses physically removed from streets or rural routes.
- A weather researcher studying tornado patterns in sparsely inhabited areas of Oklahoma could create custom coordinate systems around regions of interest to identify the positions associated with collected data easily.

Each of these coordinate system hierarchies has a root and resembles a tree. It must be noted that a forest of multiple independent coordinate system hierarchies may cover a geographical region.

A coordinate system name exhibits internal structure. A separator appears between two components of a coordinate system name. Any character that is not a letter, number or period can be used as a coordinate system name separator. The canonical form of a coordinate system name shows components arranged hierarchically, in decreasing order of size. A coordinate system name is considered to be in its canonical form if all coordinate systems that contain it in the present hierarchy are included in the name. The coordinate system name in the example HIP address above happens to be in its canonical form.

3.1.2 Offset

The offset portion of a HIP address locates a position with respect to the origin of a HIP coordinate system. A HIP offset is two dimensional, and adheres to standard

Cartesian convention, with the first coordinate along the horizontal axis and the second along the vertical.

The two coordinates of an offset typically and preferably lie between 0 and 1. However, when a position is not enclosed by any coordinate system, it may make sense to refer to it using the origin of a coordinate system that does not enclose it. When referring to a position that lies outside the rectangular region of a coordinate system, the coordinates in the offset may exceed 1 or even be negative. The North Pole is an extreme case in point. Any rectangle enclosing the North Pole would be highly distorted due to the curvature of the earth at such a high latitude, making coordinate references within it difficult. The North Pole should typically be referenced from a coordinate system that lies on a flatter part of the earth. When even one coordinate of an offset is negative, both coordinates of the offset should be qualified explicitly with + or - signs, e.g. *US, MA, Cambridge -23.0 +47.5.*

We have previously stated that a HIP address corresponds to a latitude and longitude position. Because positioning technologies such as GPS have a finite resolution, every position obtained with the assistance of these technologies has an associated uncertainty. A latitude and longitude position obtained from GPS corresponds to a tiny *region*, rather than the mathematically accurate, zero-area definition of a *point.*

It is important to think carefully about the relationship between the size of a coordinate system, the accuracy of positions that are derived from HIP addresses within it and the offset of a HIP address. Let us suppose that a coordinate system A is 0.001 degrees wide. The position denoted by the HIP address A .5 .5 is five tenths along the width dimension. The order of magnitude of the step size, one-tenth, tells us that this position refers to a region precise to $0.001 \times 0.1 = 10^{-4}$ degrees. If the HIP address were expressed as A .50 .50, the latitude and longitude of the position derived from it would be the same as earlier. However, because the HIP address measures fifty hundredths along the width dimension, this position would refer to a region precise to $.001 \times .01 = 10^{-5}$ degrees. It is therefore important to realize that HIP offsets may define a position more or less tightly, depending on the number of significant figures in their literal representation.

3.1.3 Check code

The check code portion of a HIP address is optional and closely tied to the geographic position to which a HIP address corresponds. The check code portion of a HIP address has two purposes:

- Error detection and correction. If there is a discrepancy between the check code from a user-supplied HIP address, and that of the position to which a HIP address resolves, a typographical error in the input can be indicated. Its intended purpose is similar to the Luhn check digit algorithm [WIK03] used to test the surface integrity of credit card numbers.

- Address disambiguation. If a user does not use the canonical name of a coordinate system, a HIP address may correspond to multiple positions. When a check code is provided, the position corresponding to a HIP address can be picked out correctly.

It is important to devise a check code calculation algorithm that is transparent to HIP users. Ideally, check codes would vary predictably across different regions of the earth. Then, a user could estimate the position of an ambiguous HIP address by using the check code. If the algorithm used to calculate a check code is unclear to users, it merely adds to the cognitive load experienced when remembering and communicating a HIP address. See Section 3.3.2 for a discussion of the design considerations behind check codes.

3.2 Essential properties of HIP

The properties of HIP that are central to its definition are human intelligibility, context awareness and ease of use.

3.2.1 Human intelligibility

Human intelligibility is essential to HIP addresses. While there are no rigid restrictions on how a coordinate system can be named, their names are typically landmarks,

neighborhoods in a city, cities, states or countries.

Naming coordinate systems in a human-intelligible manner makes HIP addresses easy to localize. Let us consider positions along long streets passing through a region, such as Massachusetts Avenue through Boston and Cambridge or Broadway through Manhattan Island. An address such as *1374 Massachusetts Avenue, Cambridge, MA* would mean little to someone unless they were somewhat familiar with the blocks of street numbers on Massachusetts Avenue. However, expressing the same position as *Cambridge, Harvard Square .69 .43* immediately localizes it to the Harvard Square area. Similarly, the roughly equivalent addresses of *1585 Broadway, New York, NY* and *New York, Times Square .34 .75* each convey a different sense of the position they represent. The HIP usability study (see Section 6.3.3) showed that subjects usually located HIP addresses faster on a map. HIP addresses might help localize an address better because coordinate systems cover regions at granularities coarser than streets, while retaining human intelligible names.

3.2.2 Context awareness

When we communicate our present location to someone else, we unconsciously use the context we share with the other person. For example, someone located at *258 Dartmouth Street, Boston, MA* would describe their location as "258 Dartmouth Street" to someone else in Boston, because they share a common local context of Boston, MA. The same person would say "258 Dartmouth Street, Boston, MA" to someone who is in Chicago, IL, who does not share the local context of Boston, MA.

HIP coordinate systems contain other coordinate systems. This relationship of containment is intended to be context-aware. When someone located in the *US, NY, New York* coordinate system uses a HIP address such as *Times Square .65 .76*, the system must interpret it within the surrounding context of New York.

Context-awareness also includes knowledge of the how the HIP address is being used. For instance, when one walks, the coordinate system of their current HIP address would not usually change very frequently. In contrast, when one is traveling in a car at a considerably higher speed, they cross coordinate system boundaries quickly.

The process of generating HIP addresses should have some built in hysteresis, so that coordinate systems don't change too fast. For example, if one is driving west from Boston, HIP addresses should be given in terms of town coordinate systems, rather than in terms of neighborhood coordinate systems.

3.2.3 Ease of use

HIP addresses should be easy to remember and communicate to other people. Although GPS receivers are becoming cheap and commonplace, the long and entirely numeric latitude and longitude addresses generated by them are suitable mostly for receiver-to-receiver communication. Any navigation using GPS receivers is thus exclusively device-mediated. On the other hand, HIP addresses have a symbolic component, which enables human beings to write them down when they hear them from other human beings. HIP addresses could also be printed on business communication, such as letterheads and business cards.

It is worthwhile considering how HIP can complement or supplement existing uses of other addressing schemes. HIP locates positions with respect to human-intelligible landmarks, but may not be ideal for communicating how to navigate from one address to another. On the other hand, street addresses may not locate a position very well, but outline a procedure to reach one street address from another. For example, it may not be clear how to navigate from *New York, Empire State Building .40 .98* to *New York, Guggenheim Museum .82 .54* unless one has already internalized a good mental map of New York. On the other hand, even a newcomer to New York can follow directions such as "Heading northwards on Fifth Avenue, take the third right onto 38th street...". HIP addresses and street addresses complement each other in their suitability for location and navigation.

The above observation has interesting implications for the types of people who might use it. HIP addresses would probably be favored by local residents of a town, who maintain a rich, high-level context about its neighborhoods and can relate to positions being expressed in terms of familiar landmarks. On the other hand, newcomers to a place might initially prefer street addresses and procedural directions

which they can follow until they build a their own substantial context around that place. More interestingly, studies about gender differences in spatial ability suggest that when asked for directions, men tend to give abstract procedural directions such as "Take the next right and then the second left", whereas women tend to give directions based on landmarks[PRE, LAW02]. Such research may have influences on HIP's ease of use for different population groups.

3.3 HIP design process

The design of the HIP addressing scheme went through several changes since its inception. This section contrasts a few of the earlier designs with the present design to elucidate some decisions that led to the present design.

3.3.1 Coordinate system evolution

HIP coordinate systems initially had dimensions in units of length (m, km) rather than units of arc. This choice worked well for coordinate systems at the scale of landmarks, neighborhoods or even towns, because the negligible curvature of the earth at such scales enabled flat, Euclidean coordinate systems to approximate physical spaces well. However, for coordinate systems at the scales of entire states or countries, the curvature of the earth would introduce deviations from the coordinate system's Euclidean approximation, when using units of length.

The motivation behind using units of length in the first place was so that people could approximately place positions in a coordinate system by using their intuitive notion of the coordinate system's scale. For example, let us assume that SoHo, a trendy New York neighborhood, can be enclosed by a square of side 0.5 mile. If a position was a quarter mile north of SoHo's southern boundary and an eighth of a mile east of its western boundary, its approximate HIP address would be *New York, SoHo .50 .25*. Using units of length in this way would also make it easy to estimate the distance between two HIP addresses mentally, something which is not always possible with street addresses. Changing a coordinate system's dimensions to units

of arc enables similar judgements ("halfway east and a quarter of the way north"). The difference, however, is that units of arc are compatible with the curvature of the earth at a greater range of scales.

In the earliest stages of designing the addressing scheme, it was tempting to leave the shape of a coordinate system and the interpretation of its HIP offset largely unspecified. Some geographical regions might be approximated better using a polar coordinate system or an arbitrary polygon. In fact, a good case could be made from a systems point of view to leave HIP offset interpretation entirely up to the owner of a coordinate system, much like the mapping of URLs to the file system is left to the Web servers. The resolution query could then be propagated through a peer-to-peer network of coordinate system resolver nodes and responses could be collected from the entire network. This apparent end-to-end nature of a HIP address held significant technical appeal. However, allowing coordinate systems to assume arbitary shapes and offset interpretation schemes destroys the transparency of a HIP address to its users. Tasks important to the usability of an addressing scheme, such as direction estimation and linear interpolation, would cease to be uniformly applicable. Therefore, it was decided that HIP coordinate systems would not have arbitrary shapes.

HIP coordinate systems were then designed to be square. When all sides of the coordinate system had the same dimensions, there would be no distortion introduced by the aspect ratio of the coordinate system. In other words, a small increase in a coordinate would measure the same distance in both the horizontal and vertical directions. However, when actual coordinate systems were drawn on a map, it became clear that forcing coordinate systems to be square made them cover regions which couldn't reasonably be associated with them. As a case in point, the MIT campus, which is approximately a mile in the east-west direction but only a quarter of a mile in the north-south direction, would be covered with a square coordinate system encompassing one mile in width and height. This coordinate system included large portions of Boston's Back Bay across the Charles River, which could not reasonably be said to lie on the MIT campus!

It became clear that artificially constraining coordinate systems to be square was a bad idea. A rectangular coordinate system would provide a tighter fit to an approximately rectangular region, thus assuring that HIP addresses with reference to that coordinate system could reasonably be localized to that region. Rectangular coordinate systems were a generalization of square coordinate systems. It was also clear that every HIP address from a square coordinate system could be mapped into one from a rectangular coordinate system using translation and scaling, both linear transforms.

Extending the idea of using linear transforms, it was also proposed that rectangular HIP coordinate systems be rotated some number of degrees from the horizontal to align with the underlying geographical region better and cover it even more tightly. Rectangular coordinate systems rotated with respect to the horizontal would work well for regions such as the MIT campus, which, in addition to being vaguely rectangular, are not oriented roughly east-west either. Rotation, being a linear transform, would also establish a one-to-one mapping between an axis-aligned HIP address and rotated HIP addresses. However, choosing to rotate coordinate systems arbitrarily eliminates an intuitive calculation of heading from how one's HIP address is changing. For example, considering an arbitrarily rotated coordinate system, an increase in the Y coordinate of one's HIP address does not necessarily mean that one is heading north. Calculating the heading between two HIP addresses also becomes considerably more difficult, without prior knowledge of how the two coordinate systems are oriented. Although rotation would mean even tighter coverage of a region, it would mean that orientation of a HIP addresses would become opaque to the user and be completely device-mediated. This loss of intuition in heading was borne out by results of the user study, where subjects weakly preferred horizontal, axis-aligned coordinate systems, over arbitrarily rotated ones (see Section 6.3.5). As a result, rotation of coordinate systems with respect to the horizontal was not pursued.

3.3.2 Check code evolution

The check code portion of a HIP address is intended to serve two goals: error correction and address disambiguation. Because address disambiguation is a goal of the check code, it should be tied to the position to which the HIP address would resolve. In other words, only those HIP addresses that resolved to valid positions, i.e. those with valid semantic content, could have check codes. The check code should have little to do with the literal written form of the HIP address, because the HIP address entered by a user may be subject to contextual modification (see Section 3.2.2) before being resolved into a position. The ideal check code would have the following properties:

- Correlation with location. A check code would be tied to the geographical position of a HIP address. For example, check codes for positions in the earth's western hemisphere might always come from the first half of the alphabet.

- Ease of use and communication. A check code should be easy to communicate to other people. Letters such as I or O, which may be mistaken for numbers, should not be used. They should also lend themselves to easy auditory memory. For example, check codes may always be a vowel followed by a consonant or a consonant followed by a vowel, in order to be a pronounceable syllable.

- Sensitivity. The check code for a HIP address must not be too sensitive to perturbation. For example, check codes for two otherwise identical HIP addresses differing only in the number of significant figures in their offsets should not be markedly different.

- Ease of internationalization. Check codes should have support for being drawn from several international alphabets.

The check code, whose initial design stipulated a single alphabet, was augmented with a length attribute, to support multi-character check codes. Moreover, check codes were specified to be drawn from an arbitrary range of Unicode characters, to

support easy internationalization. A check code algorithm possessing the properties of sensitivity and correlation with location in their most ideal form as described above would require a sophisticated universal hash function that cast HIP addresses into different bins at an appropriate level of granularity. Designing a geographically sensitive hash function with the above properties remains as future work. The HIP implementation described in later chapters of this thesis employs only a basic hash function[BLO01] to generate the check code for a HIP address.

3.4 Chapter Summary

In this chapter, we have seen the design of the HIP addressing scheme and its essential characteristics. HIP is a human-intelligible, context-aware, hybrid location model, whose emphasis is on ease of use. HIP addresses consist of a coordinate system name, a two-dimensional offset and a check code. Coordinate systems are named with human-intelligible place names to enable easy localization of a HIP address. HIP offsets are required to be two dimensional and Cartesian to ensure uniformity of interpretation across coordinate systems. Check codes are intended to be transparent to users. The evolution of the HIP addressing scheme, also included in this chapter, elucidates the intentions behind the present design of HIP and explains which other options were considered. HIP was subject to a few iterations and design changes before reasonably accomplishing the above goals, which helped clarify my own thinking about the motivations behind HIP.

Chapter 4

Network Services

This chapter describes the architecture and implementation of the HIP addressing scheme. HIP is powered by a set of database-backed Web services that generate HIP addresses from latitude and longitude positions and resolve HIP addresses into latitude and longitude positions.

4.1 Design Considerations

This section describes the choices considered when implementing the HIP addressing scheme. The HIP back-end is built to allow platform-neutral access from a wide variety of devices.

4.1.1 Implementation goals and non-goals

At the core of HIP addressing scheme are two capabilities: *address generation* and *address resolution.* Address generation produces HIP addresses from latitude and longitude positions. Address resolution converts HIP addresses into latitude and longitude positions. The results of address generation and resolution can be used in a variety of ways because several of today's device-assisted mapping, navigation and other location-based applications (e.g. Mapquest, Mappoint) are backed by databases of latitude and longitude positions. The core definition of the HIP addressing scheme

does not include applications such as mapping and navigation, which can be attained by composing HIP capabilities with existing third-party solutions. Thus, HIP address generation and resolution are necessary and sufficient to implement the HIP addressing scheme.

4.1.2 Platform considerations

The Java programming language was chosen for implementation because it is object-oriented and platform-independent. The PostgreSQL relational database management system was chosen because it is a high-performance, ACID-compliant[GRE99], open source database, which supports user-defined data types[PGRES]. Contributions from the PostgreSQL free software community include PostGIS[PGIS], a library of spatial data types for PostgreSQL. PostGIS defines a geometric object model including points, lines, polygons and collections thereof. It also implements the Open GIS Consortium's OpenGIS Simple Features for SQL specification[RYD99], geared towards database-backed geographic information systems (GIS) applications.

4.1.3 Network interface

HIP network services are exposed over the network using the Simple Object Access Protocol (SOAP), a World Wide Web Consortium (W3C) standard[MIT03] to enable remote method invocation. SOAP enables a client to invoke methods on a server using structured XML messages over HTTP. SOAP was chosen over distributed computing architectures such as Java RMI and CORBA because it is based on HTTP, which is not bound to a particular software platform. SOAP client implementations exist even for mobile devices, towards whom the HIP addressing scheme is aimed (see chapter 5). From the plethora of available SOAP implementations, the Apache Axis SOAP Library was chosen because it supported Java and integrated well with open source Web products such as the Apache Web server and the Tomcat engine.

4.2 Abstract data types

The HIP implementation contains five core abstract data types:

- `GPSAddress`. A `GPSAddress` encapsulates latitude, longitude and the associated uncertainty fields. The latitude, longitude and uncertainty are expressed as integers representing ten-thousandths (10^{-5}) of a degree.

- `CoordinateSystem`. A `CoordinateSystem` has a `GPSAddress` object as its origin field. This `GPSAddress` object is defined to have an uncertainty of zero. A `CoordinateSystem` object also has a name field, which contains its canonical name. It has a number of other fields to store information about check codes: check code length, beginning and end characters of the Unicode range from which the check code is drawn and an integer seed, which is used to actually calculate the check code.

- `HIPName`. A `HIPName` represents the coordinate system part of a HIP address. It is a sequence of human intelligible name components separated with a separator character. A HIPName provides a means to enumerate its components.

- `HIPOffset`. A HIPOffset represents the offset portion of a HIP address. A `HIPOffset` encapsulates horizontal and vertical offsets of a position within a HIP coordinate system.

- `HIPAddress`. A `HIPAddress` encapsulates a `HIPName` and a `HIPOffset`. A subtype of `HIPAddress`, `CheckedHIPAddress`, encapsulates a check code in addition to the fields of `HIPAddress`.

The relationships between the abstract data types above are captured in Figure 4.2, a problem object model for HIP.

4.3 Data Model

The data model for HIP address generation and resolution contains one table. HIP coordinate systems are stored as PostGIS polygons. The table definition appears

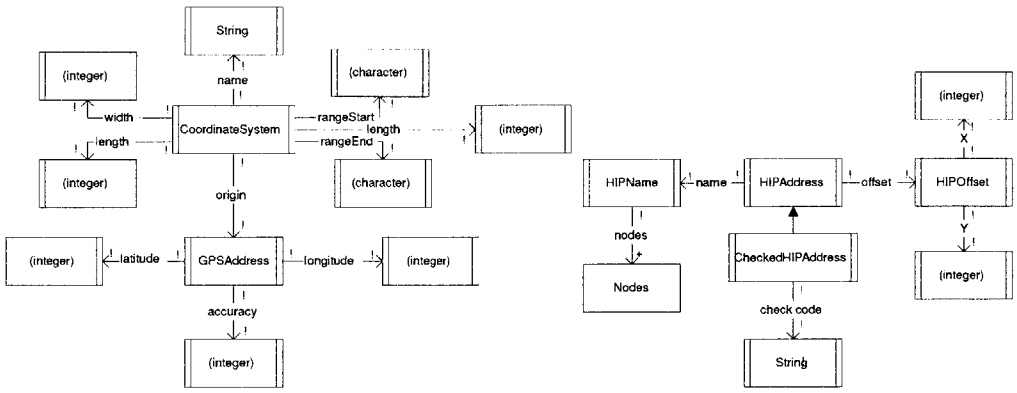


Figure 4-1: This is a problem object model of the abstract data types of HIP.

below:

```
CREATE TABLE coord_systems (
    -- coordinate system related columns
    polygon_id          serial       NOT NULL
    name                varchar(255),
    coord_system        geometry,
    parent_polygon_id   integer,
    -- check code related columns
    check_start         varchar(5),
    check_end           varchar(5),
    length              integer      NOT NULL,
    seed                integer,
    -- table constraints
    PRIMARY KEY(polygon_id),
    CHECK ((geometrytype(coord_system) = 'POLYGON'::text)),
    CHECK ((srid(coord_system) = 4269))
);
```

The table includes columns for basic attributes of a coordinate system, such as its name, its geometric definition and its parent coordinate system. All coordinate system names are stored in the database in their canonical form, e.g. "US/MA/Cambridge".

All coordinate systems are rectangles (4-sided polygons), with their width and height implicit in their geometric definition.

The geometric definition of a coordinate system has a Well Known Textual (WKT) representation, adhering to OpenGIS specifications. For example, the polygon object representing Boston can be expressed as the literal string

> POLYGON((-71.191198 42.208839,-71.191198 42.417755,-70.982281 42.417755,-70.982281 42.208839,-71.191198 42.208839))

The WKT representation for a two dimensional polygon is a list of its vertices that starts and ends with the same vertex. The coordinates of every vertex are separated with a space. OpenGIS specifications require all geometric objects to have a *spatial reference system*, which is a reference grid against which to interpret coordinate values. All polygons in the `coord_systems` table are assigned the spatial reference system identifier (SRID) 4269. This identifier corresponds to the World Geodetic System 1984[STA], an imaginary reference ellipsoid closely approximating the shape of the earth, upon which latitudes and longitudes are defined.

The `coord_systems` table also contains attributes important for check code generation. The `check_start` and `check_end` columns mark the range of characters from which check codes can be drawn. Provided that the PostgreSQL database is initialized with the appropriate character encoding, check codes can be drawn from non-Latin Unicode alphabets as well. The `length` column contains the length of the check code. Finally, the `seed` column contains an integer that will result in a check code drawn from the range specified above. This check code seed is calculated when inserting a coordinate system into the database.

4.4 Address generation

Address generation generates a HIP address from a latitude and longitude position. Because of contingencies in coordinate system layout, multiple HIP addresses may correspond to a latitude and longitude position. In this situation, the HIP client

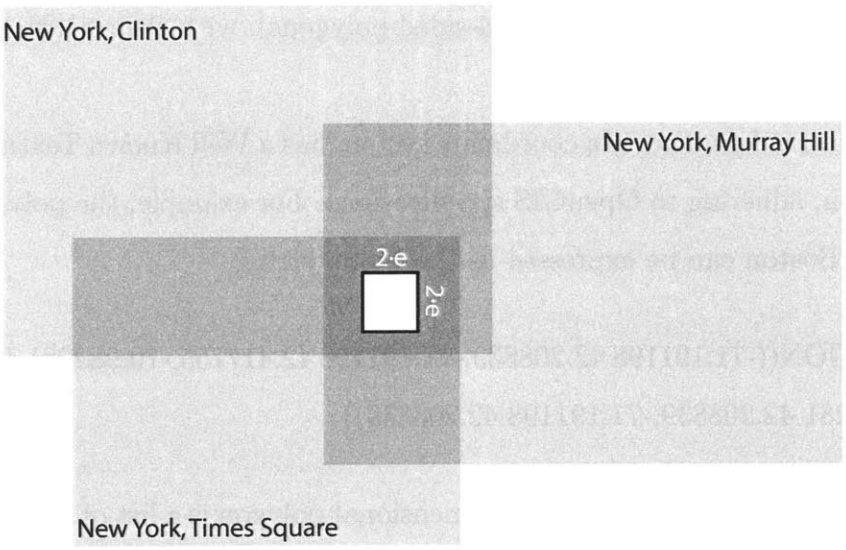


Figure 4-2: The `enclosing` method of the HIPAddressGenerator service returns all the coordinate systems that intersect with or enclose a square of side $2 \cdot e$, where e is the accuracy of the latitude/longitude parameter.

application is responsible for choosing an appropriate coordinate system and plotting a position within it. Most of the computation in generating a HIP address is thus delegated to the client application (see Section 5.2).

The network interface for HIP address generation consists only of the `enclosing` method, which returns a set of all the coordinate systems enclosing a position (Figure 4-2).

The following SQL query template, with placeholders for latitude and longitude values, returns the set of coordinate systems enclosing a position:

```
SELECT
 name, AsText(coord_system),
 seed, length, check_start, check_end,
 Distance(coord_system, GeometryFromText('POINT(<long> <lat>)', 4269)) AS distanc
FROM
 coord_systems
WHERE
 coord_system && GeometryFromText('BOX3D(<long> <lat>, <long> <lat>)', 4269)
```

```
ORDER BY distance
```

Note the use of the && relational spatial operator, which returns *true* if the bounding box of the left operand contains the bounding box of the right operand. The && operator applied between a POINT and a BOX3D returns *true* even if the POINT lies on the BOX3D. The above query template is valid for a position with no uncertainty.

The region denoted by a latitude and longitude position P_0 with uncertainty ϵ_0 may reasonably be visualized as a circle C_0 of radius ϵ_0 centered at P_0. The SQL query for returning the coordinate systems enclosing a position P with uncertainty ϵ appears below. The set of coordinate systems returned encloses a square S of side $2 \cdot \epsilon$ with centroid at P (Figure 4-3). The SQL query uses a square as an approximation to a circle because PostGIS does not have support for circles.

SELECT
name, AsText(coord_system),
seed, length, check_start, check_end,
Distance(coord_system, GeometryFromText('POINT(< *long* > < *lat* >)', 4269)) AS distance
FROM
coord_systems
WHERE
coord_system && GeometryFromText('BOX3D(< *long* > $-\epsilon$ < *lat* > $-\epsilon$,
< *long* > $+\epsilon$ < *lat* > $+\epsilon$)', 4269)
AND Distance(coord_system, GeometryFromText('POINT(< *long* > < *lat* >)', 4269)) <= ϵ
ORDER BY distance

The circle C_0 is the veridical representation of a position with an associated uncertainty. The square S, used in the query above, is its approximation. There could have been two ways to construct S as an approximation to C_0 (Figure 4-3):

- C_0 is the circumcircle of S. S would *undershoot* the veridical area by $\frac{\pi-2}{\pi}$, approximately 36%.

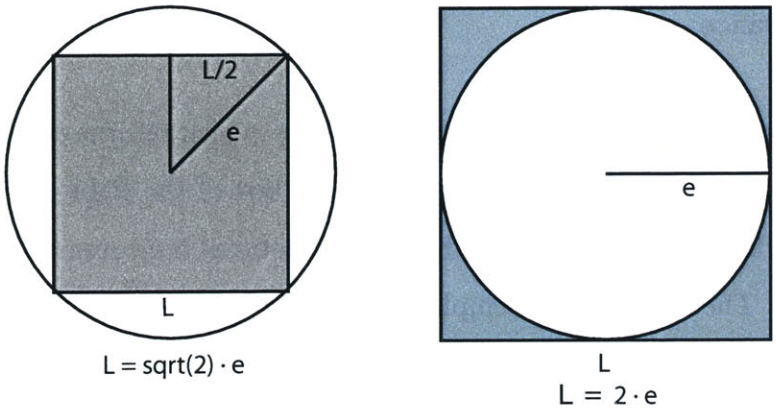


Figure 4-3: When C_0 circumscribes S, S undershoots the veridical area by 36%. When C_0 is inscribed in S, S overshoots the veridical area by 27%.

- C_0 is incircle of S. S would *overshoot* the veridical area with an error of $\frac{4-\pi}{4}$, approximately 27%.

The second option was chosen because the approximation is an overshoot. Returning more enclosing coordinate systems can only provide more information about the area surrounding a position. If the first option were chosen, then the approximation would exclude a large portion of the veridical area to be considered when looking for enclosing coordinate systems. The second option is also a better approximation because it has lesser relative error.

For example, if we instantiate the above query template with the position (42.369676N, 71.034267W), we get a result set of $\{US/MA/Medford, US/MA/Boston, US/MA/Everett, US/MA/Cambridge, US/MA/Somerville\}$.

Equipped with the knowledge of what coordinate systems surround the client, the actual HIP address is generated on the client automatically, or based on preferences set by the user (see Section 5.2). The process of HIP address generation is shown in Figure 4-4.

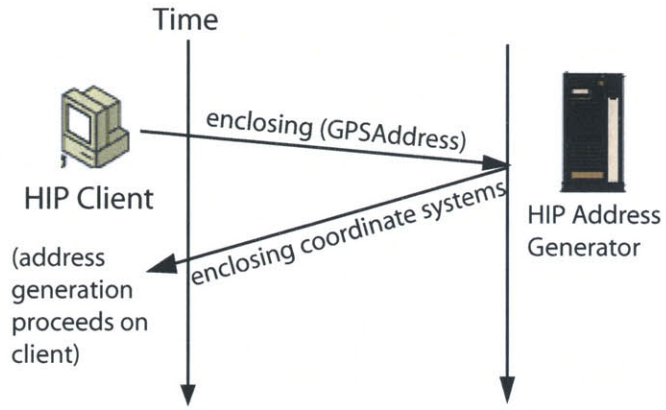


Figure 4-4: The HIP client application calls the `enclosing` method of the HIPAddressGenerator Web service to get the coordinate systems surrounding the phone's current position. After that, the HIP client generates its HIP address using local computation.

4.5 Address resolution

HIP address resolution converts a HIP address into a latitude and longitude position. The mapping between the space of HIP addresses to the space of latitude and longitude positions is many-to-one. HIP address resolution is an easier problem to solve because every HIP address corresponds to exactly one latitude and longitude position.

The network interface for HIP address resolution consists only of the `resolve` method. However, HIP address resolution is executed in three steps:*HIP address parsing*, *coordinate system resolution* and *address computation*. The first step is carried out by a suite of parsers on the server side. The second step is implemented in Java as the `CoordinateSystemResolver` interface and the third as the `AddressResolver` interface.

For the remainder of this section, let us consider how two HIP addresses, *US/MA/Boston/Boston Common .38 .64* and *US/MA/Boston/Central Square .38 .64* are resolved into latitude and longitude positions. The first HIP address lies within an actual coordinate system in Boston, MA. The second HIP address does not exist, because Boston does not have a locality called Central Square.

4.5.1 HIP address parsing

There is an easily extensible suite of HIP address parsers on the server side, which parses `Strings` into HIP addresses. All these parsers implement the `HIPAddressParser` interface. As a result, the suite of parsers can be extended easily. The suite of parsers internally uses the Java reflection API to instantiate parsers and execute them in sequence against the input. The suite stops as soon as one parser successfully constructs a HIP address from the input. Section 5.2 contains an explanation for why HIP address parsing was implemented on the server side rather than on the client, in apparent opposition to HIP's stated goal of context awareness.

4.5.2 Coordinate system resolution

A `CoordinateSystemResolver` initially tries to do a straightforward lookup on the coordinate system portion of a HIP address. If the lookup succeeds and results in a unique coordinate system, address resolution can continue to the address computation step. The coordinate system of the first HIP address listed above will be successfully resolved in this step. The second HIP address will not be resolved in this step.

If the straightforward lookup fails, the `CoordinateSystemResolver` enters a *best-effort stage*, where it looks for all coordinate systems that could plausibly match the given name. For the second HIP address listed above, it tries to resolve the following coordinate system names in the order listed below:

- US/MA/*/Central Square
- US/*/Central Square
- */Central Square

The asterisks are wildcards that match one or more characters. At each step above, it adds any results to a working set. There are three possible outcomes for the working set after the best-effort stage:

- The working set is empty. The `CoordinateSystemResolver` notifies the caller that the given coordinate system is unresolved.

- The working set contains one element. The coordinate system resolution stage will complete successfully and address resolution can continue.

- The working set contains more than one element. The `CoordinateSystemResolver` notifies the caller that the given coordinate system is ambiguous and needs further clarification.

At the end of the best-effort stage for the second HIP address listed above, the working set would contain the coordinate systems US/MA/Cambridge/Central Square and any other coordinate systems whose canonical names ended in Central Square.

If address resolution cannot proceed to the address computation step for any reason, the caller is notified. Any recovery is carried out by the client until a unique coordinate system corresponding to the given name is found.

4.5.3 Address computation

Once a unique coordinate system for a given HIP address is identified, an `AddressResolver` object multiplies the horizontal and vertical HIP offsets by the width and height of the coordinate system, respectively to get horizontal and vertical increments. It adds these increments to the origin associated with the coordinate system, resulting in a latitude and longitude position. The entire process of resolving a HIP address is shown in Figure 4-5.

Let us suppose that the coordinate system resolution step returns a coordinate system at (42.000000N, 71.000000W) with a height and width of 0.001 and 0.005 degrees respectively. Calculating increments for the first HIP address listed above, we have $\Delta X : .38 \times .001 = .00038$ and $\Delta Y : .46 \times .005 = .0023$. Therefore, the HIP address listed above corresponds to the position (42.00038N, 71.0023W).

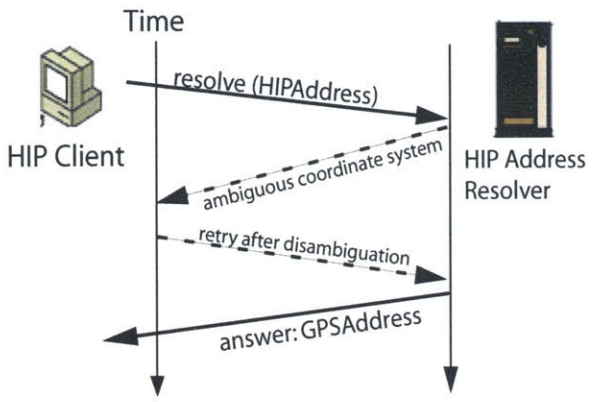


Figure 4-5: The client calls the `resolve` method of the HIPAddressResolver service with a HIP address. The service may ask for disambiguation of the coordinate system portion of the HIP address. When all ambiguities have been resolved, the service returns a latitude/longitude address (a GPSAddress) as the result of the resolution.

4.6 Chapter summary

The HIP network services are designed to provide a simple, minimalistic and well-defined interface to HIP clients. They are client-agnostic and intended to separate mechanism and policy. Most choices about generating and resolving HIP addresses are pushed out to the client application. If the client application runs on a computationally impoverished device, then providing an acceptable HIP user experience can be challenging. The next chapter will discuss the design, architecture and evolution of a HIP client application.

Chapter 5

Client Application

This chapter describes the design and evolution of the HIP client application and user interface.

5.1 Design Considerations

The previous chapters in this thesis have described the design and implementation of the HIP addressing scheme. By and large, the discussion so far has centered around the conceptual model behind HIP. High-level capabilities of the HIP addressing scheme, such as address generation and address resolution have also been discussed. The HIP client application brings the HIP addressing scheme to the user, along with some practicable applications of these high-level capabilities.

5.1.1 Usage scenarios

Users of a HIP client should be able to accomplish tasks that may be accomplished by using other addressing schemes. These tasks are:

- Location. A user must be able to see the HIP address of their current position.
- Mapping. Given a HIP address, a user must be able to receive a map of the position corresponding to it.

- Navigation. Given an origin HIP address and a destination HIP address, a user must be able to receive directions from one location to another.

- Recentering. A user must be able to receive a HIP address for their position relative to any of the coordinate systems that surround them. This task is a key manifestation of HIP's context awareness.

We will now expand the abstract descriptions of tasks above into some concrete usage scenarios:

Background

Alyssa P. Hacker is going to meet Ben Bitdiddle for dinner at Greenhouse Cafe at Harvard Square in Cambridge, MA. Ben lives in Newton, MA and has been to downtown Boston a few times. However, he is unfamiliar with interior Cambridge. Alyssa and Ben were supposed to meet at 5:00 pm, but Alyssa was not sure where she was going to be at the time. She and Ben agreed that Alyssa would call him about where they would meet.

Scenario 1–Location

Alyssa starts up the HIP client on her cellular phone. She ascertains her current HIP address. She calls Ben and asks him to meet her at *Cambridge, Harvard Square .65 .54B.*

Scenario 2–Map

It is 5:45 pm and Ben has not shown up. Alyssa is slightly annoyed and concerned that Ben has got lost. Ben calls her and says he is currently at *Boston, Prudential .32 .46L.* Alyssa now knows that he is in the vicinity of the Prudential shopping mall in downtown Boston. She wants to see exactly which intersection Ben is at, so that she can estimate how long it will take for him to get to Harvard Square. She chooses the mapping option on the HIP client and receives a map of Ben's current location.

Scenario 3–Directions

It's 6:00 pm. Ben calls Alyssa saying he is walking around in downtown Boston trying to find the way to Harvard Square. Alyssa is quite annoyed now and asks Ben to go back to the HIP location he had given her earlier. She says she'll just meet him there herself. After she hangs up, she enters her current HIP address as her origin and Ben's HIP address as her desination into the HIP client. She receives directions from her location to Ben's location.

Scenario 4–Recentering

Alyssa is on her way to meet Ben near the Prudential Center. She has just crossed Harvard Bridge and is now in Boston. She was in Cambridge all this time, and was getting HIP addresses with respect to the Cambridge coordinate system. Now that she is in Boston, getting HIP addresses with respect to Boston would make it easier for her to get to Ben's current position. In her HIP client, she finds that her current position can be expressed in terms of both the Boston and Cambridge coordinate systems. She chooses the Boston coordinate system and starts receiving HIP addresses with respect to Boston.

Eventually, she meets Ben near the Prudential Center and they eat at Legal Seafoods in the Prudential Center.

5.1.2 Choice of platform

HIP is a technique for pervasive wide-area location modeling. It became clear that the full range of applications this location based service can offer would be realized better on a mobile computing device with a changing location, rather than on a desktop with a fixed location. We considered two mobile device platforms for the HIP client application: personal digital assistants (PDAs) and cellular phones. We decided to choose cellular phones because built-in GPS support in PDAs was limited. In comparison to PDAs, cellular phones generally have less screen space, processor power and storage. Developing a user interface for the more computationally impoverished

cellular phone platform presented a challenging and interesting research problem.

After surveying the large number of GPS-enabled cellular phones in the market, we chose to develop with the Motorola iDEN platform[IDEN] on the Motorola i88s phone. Almost all the other GPS-enabled cellular phones made GPS data available only for emergency purposes, in compliance with Federal Communications Commission directives[FCC03]. On the other hand, Motorola's iDEN platform provided developers with Assisted GPS (AGPS), an API to access the GPS position of the phone. The iDEN platform's compliance with the Java 2 Micro Edition (J2ME) specification[J2ME] was fortunate because all modules of HIP could now be written in the same programming language, Java.

5.2 HIP client architecture

This section describes the architecture of the HIP client and details how it uses the GPS capabilities of the Motorola i88s phone to act as a front-end to the HIP network services.

Assisted GPS

The Motorola i88s cellular phone, in contrast to other GPS-enabled cellular phones, offers assisted GPS (AGPS), which makes a user's current position available for use in location-based services. The AGPS feature of the i88s is delivered through a GPS chipset. When a user or an application requests the phone's current position, it obtains a *GPS fix*, which is an attempt to triangulate its latitude and longitude position by contacting at least three GPS satellites in its line of sight. When obtaining a fix for the first time at a location, it could take the phone up to two minutes to locate satellites and compute its position. After the first fix has been obtained, subsequent fixes take under 10 seconds.

HIP address generation

As we have seen in Section 3.1.1, a latitude and longitude position may have multiple corresponding HIP addresses. The best HIP address is then the one that is most context-aware for the present situation. The HIP client application may have limited access to the items in the user's current context, as well as their relative salience. For instance, if a user were driving quickly through town after town in Massachusetts, they might want HIP addresses with respect to the larger Massachusetts coordinate system, rather than with respect to the individual towns. Similarly, a user new to a city might prefer HIP addresses with respect to the city as a whole, rather than individual neighborhoods within it. For these reasons, the HIP addressing scheme does not impose a unified policy for generating HIP addresses and lets the user decide the preferred coordinate system with respect to which HIP addresses will be generated.

The HIP client calls the `HIPAddressGenerator` Web service (Section 4.4) and obtains a list of coordinate systems surrounding the user (Figure 4-2). Because the list of coordinate systems is returned in increasing order of distance from the user's current position, the HIP client makes a locality assumption and automatically selects the first element of the list to bootstrap the preferred coordinate system for displaying HIP addresses. Note that in the absence of any other information, this choice would still lead to reasonable application behavior if the HIP client were executed in a geographical region for the first time. If a HIP user from Boston started the HIP client soon after landing in Los Angeles for the first time, they would probably receive HIP addresses with respect to Los Angeles airport's coordinate system. If they then wish to change their preferred coordinate system, they can recenter their HIP address by setting preferences in the user interface (Section 5.3.4).

The HIP client locally computes the offset of the current position within the currently preferred coordinate system. Another approach to generating a HIP address would be to contact a Web service with the phone's current position, thus offloading the computation to a more powerful resource. There are two advantages of computing HIP offsets on the cellular phone rather than using a remote Web service:

- The cellular phone does not transmit its exact latitude and longitude contiuously over a possibly unsecured network. It is possible to surmise the approximate position of the phone by observing the set of enclosing coordinate systems that it requests. However, this situation is far more desirable than the phone continuously transmitting its exact position over the network. Therefore, concerns about security and privacy of one's position are allayed to some extent.

- The cellular phone does not have to maintain an open HTTP connection to the Web service at all times to obtain its current HIP address. Opening an HTTP connection only when necessary reduces server load and network latencies. Not waiting for generated HIP addresses to arrive from the server also means a nimble HIP address display that is likely to be more reflective of the phone's current position.

- Reducing network communication may also lead to significant power savings from reduced transmission.

When either coordinate in the offset continues to rise or fall beyond certain upper or lower thresholds near the edges of a coordinate system, it means that the user is about to leave the present coordinate system. The HIP client then queries the `HIPAddressGenerator` service again to retrieve new coordinate systems that the user might enter.

Check code calculation is done using an algorithm similar to the hash algorithm in [BLO01]. Every coordinate system contains a seed integer that is useful in check code computation. A prime number is the starting point of the check code calculation. The coordinate system seed and the horizontal and vertical components of the HIP offset are multiplied in turn by a prime number and added to the initial prime number. The result is divided by the length of the permissible check code range for that coordinate system. The remainder of this division is then indexed into the range of permissible check code characters to derive the check code. A short Java language code snippet that calculates the check code appears on the next page.

```
public String check(HIPAddress h) {
```

```
  // ... retrieve cs, the coordinate system of h

  // start with a prime number and accumulate
  // the seed and coordinates of the HIP address
  // multiplied by the same prime
  int result = 41;
  result = 23 * result + cs.getSeed();
  result = 23 * result + (int) h.getOffset().getX();
  result = 23 * result + (int) h.getOffset().getY();

  // divide the result by the length of the permissible
  // check code range and use that as an index into the
  // check code range
  char offset = (char) (cs.rangeStart + (result % (cs.rangeEnd - cs.rangeStart));
  return new String(new char[] {offset});
}
```

Because both coordinates of the offset of the HIP address are accumulated into the result, the check code is sensitive to a user's movement within a coordinate system. It must be noted that the coordinate system of the address is independent of the literal written form of the address. For example, both *US/MA/Cambridge* and *Cambridge* may eventually resolve to the same coordinate system *US/MA/Cambridge*. Because the check code calculation proceeds from the coordinate system corresponding to the HIP address, the resulting check code is independent of how the HIP address is written. The above code calculates only a single letter check code.

HIP address resolution

HIP address resolution is used in the mapping and navigation modules of the HIP client application. The cellular phone resolves a HIP address into a latitude-longitude coordinate pair by calling the `HIPAddressResolver` Web service (Section 4.5, Figure 4-5) with a HIP address. For simplicity, the HIP client application passes user input

to the service directly as a `String`. In keeping with the goals of context-awareness in HIP, HIP address parsing would best be placed on the client side rather than the server side, because the HIP client application is likely to have better access to various items in the user's context. Unfortunately, MIDP 1.0 does not support Java reflection, a crucial component to creating pluggable parsers and an extensible parser suite.

If the coordinate system of the HIP address is unresolved, the user is informed to that effect and returned to the main menu. If there are multiple matches to the coordinate system of the HIP address, then the user must intervene to choose between them. This disambiguation interface is very similar to the recentering user interface (see Section 5.3.4). Once a unique coordinate system has been chosen, the service returns a latitude and longitude coordinate pair. This coordinate pair can then be fed to third-party Web services that produce maps and directions from coordinate pairs.

5.3 User Interface Design and Implementation

An iterative design process similar to the risk-driven spiral model described by Boehm[BOE88] was used to construct the HIP client application. The design of the HIP user interface went through three stages: paper prototyping, computer prototyping and a final implementation.

Each stage had three phases: a design phase, an implementation phase and an evaluation phase. In the design phase, the goals for that stage were articulated and any feedback from previous stages was incorporated. Once the design was implemented in the implementation phase, it was evaluated using techniques such as heuristic evaluation and user testing in the evaluation phase. The results from the evaluation phase were then fed back into the design for the next stage. This section will walk the reader through the final user interface of the client application (Figure 5-1) and contrast it with previous prototypes, when appropriate, to highlight its evolution.

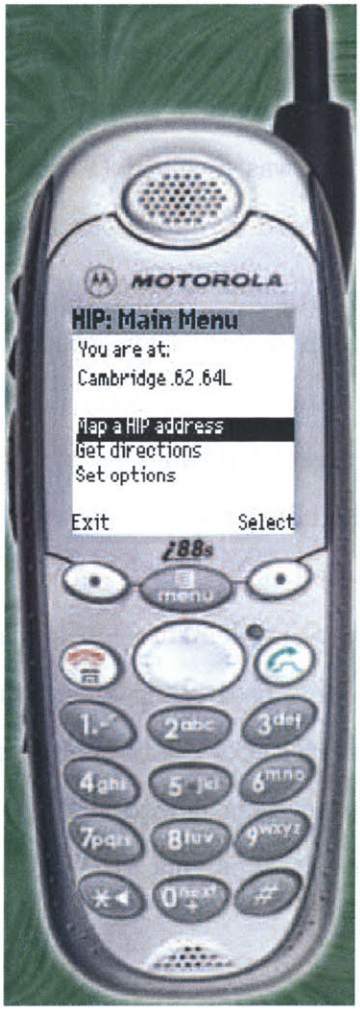


Figure 5-1: The main menu and opening screen for the final implementation of the client application.

5.3.1 Location

The location scenario is realized in the client application in a straightforward manner. The HIP address corresponding to a user's current position is always shown at the top of the screen (Figures 5-1, 5-3, 5-4). This scenario uses HIP's high-level capability of address generation.

Previous prototypes of the user interface featured a "Where am I?" option in the main menu, which was designed to contact the `HIPAddressGenerator` service once and display the HIP address that was received (Figure 5-2). This approach proved to be problematic because it imposed a rigid separation between different scenarios. For example, a user might be interested in receiving a map or directions from their present location. With the earlier design, they would have to remember their HIP address from the "Where am I?" module and enter it into the mapping or navigation modules. Showing the user's current HIP address at all times enables them use it as they please in all screens of the client application and reduces the need to recall it from other parts of the application.

The revised design of the final implementation is also more amenable to a truly mobile user. If a user were walking or driving when using the the client application for navigation, the HIP address of a user could change between accessing it through the "Where am I?" option and entering it into the navigation module (Figure 5-4). A live, updating display of the user's current HIP address would probably result in more useful directions.

5.3.2 Mapping and navigation

Mapping and navigation use HIP's high-level capability of address resolution. The latitude and longitude coordinate pair resulting from address resolution can be fed into third-party Web services that produce maps and directions from coordinate pairs. As a proof of concept, this HIP client application shows a map of the position, obtained from the U.S. Census. The directions module obtains walking directions for short distances from a service provided by the Massachusetts Bay Transportation

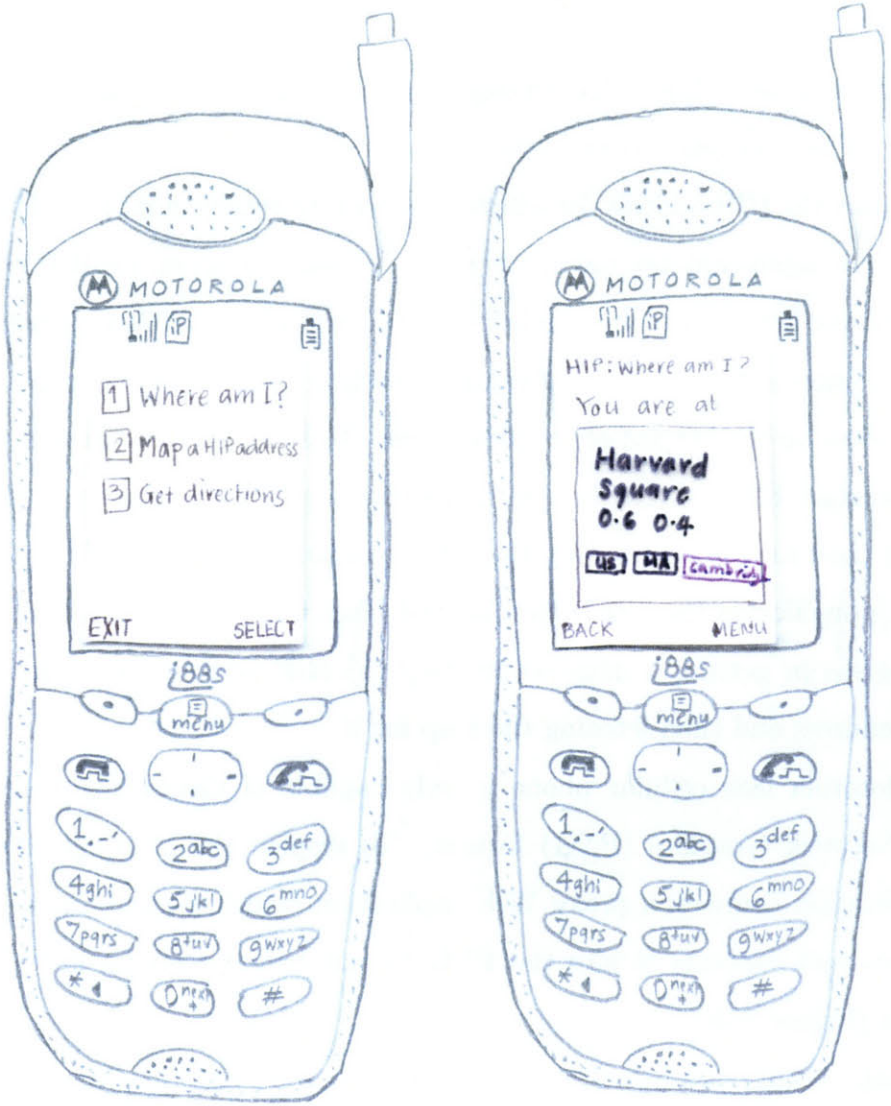


Figure 5-2: Recentering (right) was implemented through the "Where am I?" option (left) in the paper prototype.

Authority. Both these services are offered free of charge for general use. Other commercial services, such as MapQuest and MapPoint, may offer mapping and navigation solutions that are better suited to mobile devices.

Mapping

The user chooses the "Map a HIP address" option from the main menu of the client application. The resulting screen contains a form with a textbox (Figure 5-3), where the user enters the HIP address for which they wish to receive a map. HIP addresses tend to have a significant textual component. Typing out an entire HIP address on a cellular phone keypad can be cumbersome because upto four key presses may be necessary to enter just one letter. In contrast, each numeric character requires only a single key press on the keypad of the i88s phone. To assist the user in text entry, the client application fills the text box with the user's current coordinate system. The user would then have to enter only the offset portion of the HIP address, which is numeric. Upon clicking the soft button labeled Map, the display keeps them updated of the progress in getting a map—converting the HIP address into a latitude and longitude address and then fetching the map for it.

The Motorola i88s cellular phone is only capable of displaying images in the Portable Network Graphics (PNG) format. To display the map successfully, the client application contacts a proxy Web application, which fetches a map from the U.S. Census and converts it into the PNG format before relaying it to the client application (Figure 5-3).

Two task exceptions may arise during the address resolution prior to displaying the map:

- Unresolved addresses. If the HIP address that the user entered could not be resolved into a latitude and longitude address because its coordinate system could not be found, then the user is alerted to this event and returned to the main menu.

- Ambiguous addresses. If the coordinate system of the HIP address that the

Figure 5-3: When the user chooses "Map a HIP address" from the main menu, they are prompted to input the HIP address for which they want a map (left). The user's current coordinate system is filled into the textbox to reduce keypad entry of alphabets. If the HIP address is ambiguous, the user is asked to disambiguate it (center). Once an unambiguous HIP address is available, its corresponding coordinate pair is obtained from the `HIPAddressResolver` Web service and used as input to a U.S. Census mapping service, which results in a map (right).

user entered might match multiple coordinate systems, the user is shown a list of all these matches and asked to clarify which one they meant to use (Figure 5-3).

The map displayed by the client application is labeled with the HIP address. Note that the the user's current location also appears near the top of the screen. Once a user has viewed the map, they can return to the main menu by pressing the soft button labeled Menu.

The user interface workflow for mapping a HIP address is very simple. As a result, the mapping module of the client application remained unchanged from previous prototypes.

Navigation

The user chooses the "Get directions" option from the main menu of the client application. The resulting screen contains a form with a textbox (Figure 5-4), where the user enters the HIP address of their origin. Upon pressing the soft button labeled Next, the user is shown an almost identical form, where they enter the HIP address of their destination. The client application makes a reasonable assumption that the user would most likely seek directions from near their current position. To assist the user in text entry, their present coordinate system is already filled into the textbox where they enter their origin HIP address. After entering both HIP addresses, the client application resolves them into latitude and longitude coordinate pairs. Any task exceptions that occur during address resolution are dealt with in turn after each of the two addresses, in the same way as the mapping module above.

To display directions, the client application contacts a proxy Web application, which fetches walking directions from the Massachusetts Bay Transportation Authority's website and strips away extraneous HTML markup before relaying it to the client application. The walking directions appear in a full-screen, scrollable text display on the cellular phone (Figure 5-4). The user can then return to the main menu by pressing the soft button labeled Menu.

Figure 5-4: When the user chooses "Get directions" from the main menu, they are prompted to input the origin and destination HIP addresses in a form (left). The coordinate pairs corresponding to these are obtained from the `HIPAddressResolver` Web service and used as input to a walking directions service offered by the MBTA, resulting in walking directions (right).

Figure 5-5: If a user wishes to recenter their HIP address away from what is currently shown, they can choose the "Set Options" option from the main menu and choose from a list of the coordinate systems that enclose them.

5.3.3 Recentering

Recentering uses the `HIPAddressGenerator` Web service to obtain a list of coordinate systems surrounding the cellphone's current position. The user can choose the "Set options" option from the main menu of the client application to receive HIP addresses with respect to a different coordinate system surrounding them than the current one (Figure 5-5).

Recentering is an important part of the HIP addressing scheme because it directly manifests an important property of HIP, context awareness. Implementing an effec-

tive user interface for recentering was almost as tricky as it was important. The paper prototype of the client application integrated recentering into the "Where am I?" option. A display of the coordinate systems that contained a position would appear below a user's current HIP address (Figure 5-2). The address shown to the user in the top half of the screen could be interpreted sensibly with respect to the coordinate system that was highlighted in the bottom half of the screen. From the example we see in the figure, *Harvard Square 0.6 0.4* can be interpreted sensibly with respect to *US/MA/Cambridge*, which is highlighted. Operating systems show analogous behavior when naming hierarchical file names at a command prompt. The highlighted coordinate system at the bottom is analogous to the user's working directory. The HIP address in the top half of the screen is analogous to a filesystem path relative to the current working directory. The user could move highlight left or right to change the coordinate system with respect to which their current HIP address would be expressed, and the HIP address displayed above it would be updated accordingly. For example, if the user moved the highlight to the left by one position, the displayed HIP address would change to *Cambridge, Harvard Square 0.6 0.4.*

The recentering user interface had not been thought through carefully at the paper prototype stage. The hierarchical display at the bottom of the screen represents just one path from the current position to the root of the present coordinate system hierarchy. It is not clear how to display multiple coordinate systems at the same level (e.g. towns) that contain a position. Furthermore, the users that tested this early recentering user interface said that they could adjust the context in a HIP address by reading the appropriate parts of the coordinate system display themselves, rather than prepending text to the HIP address by moving the highlight left or right. Because the test users of the paper prototype failed to see why recentering was important, it was decided to integrate it into the client application in a later prototype, after being thought through more carefully.

5.3.4 Implementation issues

The HIP client was implemented in Java on the Motorola i88s cellular phone, based on the Java 2 Micro Edition platform specification. The Motorola i88s complies with the Mobile Information Device Profile (MIDP) 1.0[MIDP] specification. MIDP 1.0 also has a set of lowest common denominator user interface (LCDUI) widgets, whose organization is similar to the original Java AWT class hierarchy. Cellular phone applications that conform to the MIDP specification are called MIDlets.

Every MIDlet can be viewed as a storyboard of classes that extend the `Displayable` class. The MIDlet developer implements logic to switch `Displayables` in and out of the current display in response to key and command events generated by the phone. `Displayables` in a MIDlet are created by extending one of two subclasses of `Displayable`:

- Inheritance from `Screen`, a translated input event processor. The LCDUI toolkit contains four `Screen` subclasses–`Form`, `Alert`, `TextBox` and `List`–which process translated input events, such as key presses and soft key commands. A `Displayable` that inherits from one of the above objects automatically encapsulates a Model (such as that in the Model-View-Controller paradigm) and makes key properties, such as the selected index of a list, or the text within a text box available programmatically. Child components within these `Displayables` inherit from the class `Item`, which forms a hierarchy unrelated to `Screen`.

- Inheritance from `Canvas`, a raw input event processor. Subclasses of `Canvas` handle low-level input events such as key presses and releases and provide custom painting methods on the current graphics context. `Displayables` inheriting from `Canvas` typically have to maintain their own Model and handle input-output interactions at a low level.

From Figure 5.3.4, the observant reader will note that `Form` and `List` are peers in the LCDUI hierarchy, vis-à-vis `List` being an `Item` that can be added to a `Form`. MIDlet menus typically subclass `List`. Therefore, if a form-like component wishes

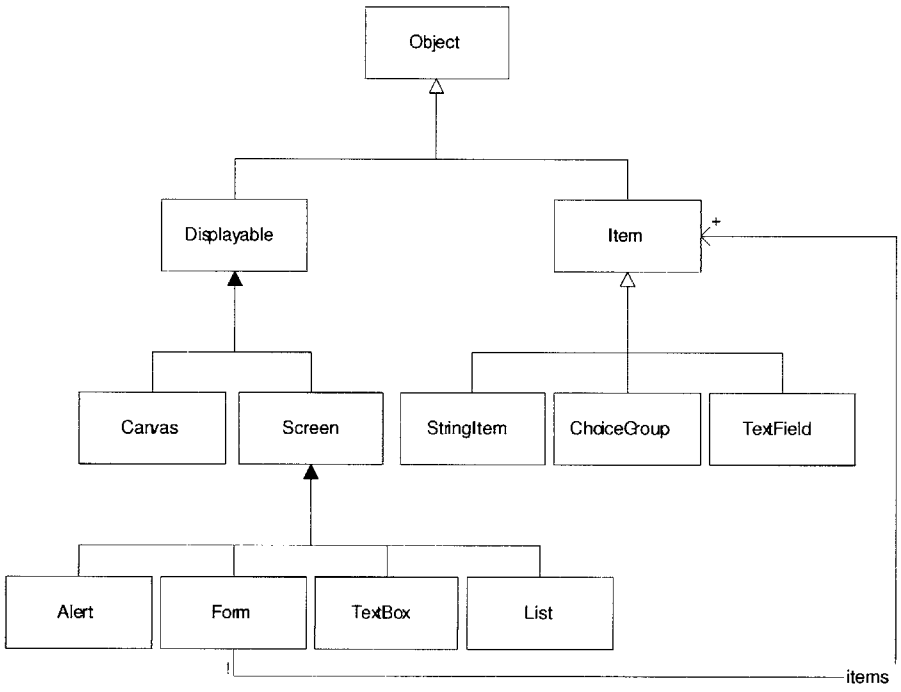


Figure 5-6: This is a simplified model of the inheritance hierarchy found in the package `javax.microedition.lcdui`. The designers of the LCDUI toolkit have chosen a flatter, higher-performance object hierarchy over a more traditionally object-oriented view hierarchy, such as Java AWT or Swing.

to incorporate a menu, it must inherit from `Canvas` rather than `Form` and perform low-level paint actions to display a menu. There also seem to be other instances, where the LCDUI object hierarchy seems to ignore rudimentary object-oriented design principles. For example, `TextBox` and `TextField` both encapsulate the display of an editable string of characters. The former occupies the entire screen and the latter is added to a `Form`. Despite their similar behavior and programmatic interface, they do not extend a common base class. The computational resource constraints of MIDP 1.0 compelled the designers of the LCDUI toolkit to forsake elegant object oriented design for an efficient and practical class hierarchy[BOO]. A relatively flat class hierarchy reduces the cost of runtime method dispatch. While it is possible to simulate a `TextBox` using a `Form` with a single `TextField`, this simulation would have a larger memory footprint. In summary, the final implementation of the HIP client used a toolkit that occasionally exposed a counter-intuitive API.

5.4 Chapter Summary

This chapter described the design and evolution of the HIP client application. The HIP client is a Java application written for GPS-enabled mobile devices such as the Motorola iDEN i88s cellular phone. The user interface for this phone was developed over the course of three prototypes, each going through design, implementation and evaluation phases. Because not everything that can go wrong with the design of a user interface can be foreseen, a risk-driven spiral model was especially well suited to the development of the HIP client. Even before the final implementation began, several improvements were identified and incorporated into the user interface in return for relatively small investments of time and effort. Finally, it forced me to think about problems from the user's perspective, an issue that is in danger of being ignored if only one person is working on a system.

An important part of implementing the HIP system was demarcating the split between client and server side functionality. From a design point of view, it made sense to design a smart client that would do most of the computation, using HIP network

services as sparingly as possible. However, this vision was limited by the relatively young MIDP/J2ME platform. Some crucial features that would have made the final implementation easier are support for floating point operations, a user interface toolkit rooted more in traditional object-oriented design principles and support for Java language features such as reflection. Features such as advanced HIP address parsing had to be moved to the server despite being designed for the client. Other than these considerations, writing an application for such a resource-constrained platform was technically challenging and stimulating. It is most likely possible to implement richer HIP client applications as location-based services become more pervasive and mobile devices as well as their development platforms mature.

Chapter 6

Usability Analysis

We have considered the design and implementation of HIP at considerable length so far in this thesis. It would be educative now to compare HIP with two other addressing schemes, street addresses and latitude and longitude, on a few key usability-related variables. The design of any system that claims to be usable would be incomplete without an analysis of results from a controlled user test. This chapter describes a controlled experiment used to validate the usability goals of HIP and analyzes the results obtained from it.

6.1 Usability indicators

Based on considerations stated earlier in this thesis, two variables are key to understanding HIP's success with respect to usability: understanding of the HIP conceptual model and how users might use HIP addresses in a real-world situation.

6.1.1 Conceptual understanding

A controlled usability test for HIP should test a user's understanding of the location model behind HIP. Specifically, it should test a user's ability to locate HIP addresses on a map. Measuring user competence in a map location task for HIP addresses and other addresses would enable us to make judgements about how effectively HIP can

complement or supplement existing addressing schemes.

6.1.2 Field use

A controlled usability test for HIP should also test how effectively HIP addresses can be used in a field situation. Specifically, it should test how well a users remember HIP addresses when they are subject to cognitive load from an unrelated activity. Measuring user competence in remembering HIP addresses under cognitive load would be a good subsitute to extensive field testing of HIP addresses.

6.2 Experiment design

The usability test had a within-subjects, multiple factorial design.

6.2.1 Hypotheses

The usability test set out to validate the following hypotheses:

- The accuracy with which users locate addresses on a map will vary with the type of address presented to them. Map location accuracy will be adversely affected by interpolation errors. Street addresses will be located with the best accuracy because street addresses always lie on street, thus reducing a degree of freedom.

- The time that users take to locate an address on a map will vary with the type of address presented to them. HIP addresses will take the least time on the location task because they are human-centric at an appropriate level of granularity. Note that street addresses are also human-centric, but usually at a finer level of granularity than HIP addresses.

- Users will experience different cognitive loads when trying to remember different types of addresses.

6.2.2 Experimental procedure

After giving their informed consent for participation, subjects provided some basic background information about themselves in a pre-test questionnaire. Questions probed their level of familiarity with location and navigation technologies such as GPS devices and online cartographic services such as Mapquest and MapPoint. Subjects were also asked to indicate their level of familiarity with the greater Boston area and the Pittsburgh area. Subjects were then given a brief, self-paced, written tutorial of street addresses, latitude and longitude coordinates and HIP addresses, the three addressing schemes that would be used in the study.

Then, subjects performed the following two tasks:

- Map location task. Ten positions each were chosen in the Boston and Pittsburgh area. Subjects were presented with a computer map interface and addresses corresponding to these ten positions from four different addressing schemes (Figure 6-1). Subjects would locate each address on the map by clicking on it. Subjects would process all addressing schemes for one city before moving on to the other city. Within each addressing scheme, the order in which addresses were presented was randomized. The order of addressing schemes as well as the order of cities was counterbalanced across the subject pool. Subjects were not told that the addresses in all addressing schemes referred to the same set of positions.

- Cognitive load task. Ten positions each were chosen in the Boston and Pittsburgh area and their locations were determined in three addressing schemes. Each address was verbally presented to subjects and then subjects were asked to repeat it over a cellular phone as they walked around the perimeter of a wide open indoor space. For example, the address *749 Massachusetts Avenue, Cambridge* was read aloud as "Seven hundred forty nine Massachusetts Avenue, Cambridge". The latitude/longitude coordinate pair *40 26.830N 79 56.808W* was read aloud as "Forty twenty-six point eight three zero north [short pause] seventy-nine fifty-six point eight zero eight west". The HIP address *Pittsburgh,*

Bellefield .13 .45L was read aloud as "Pittsburgh [short pause] Bellefield point one three point four five el".

Subjects changed the direction (clockwise or counterclockwise) of their walk after completing every round trip. Walking around an unfamiliar space and changing direction periodically was intended to increase the background cognitive load on users as they heard various addresses. Within each addressing scheme, the order in which addresses were presented was randomized. The order of addressing schemes and that of cities was counterbalanced across the subject pool.

After the two tasks, users were presented with a post-test questionnaire, where they gave their subjective opinions on HIP and how it compared with the other addressing schemes they saw.

Note on map preparation

Maps were obtained from Delorme Street Atlas, a desktop mapping and navigation application. Maps obtained from the application are heavily biased towards location and navigation using street addresses. To counteract this inordinate advantage conferred upon street addresses and to equalize the "visual noise" across addressing schemes, the maps were prepared deliberately. In particular, there were approximately equal numbers (approximately 20) of street, latitude and longitude and HIP coordinate system labels in each map.

There were structural differences between the maps of Boston and Pittsburgh, introduced by the lay of the land and the sizes of different neighborhoods. When the maps were prepared, the average lengths of the features relevant to each addressing scheme (street segments, latitude and longitude quadrants and HIP coordinate systems) were measured (see Table 6.1), so that they could be used as normalizing factors in the analysis of errors from the map location task.

Figure 6-1: Subjects were asked to locate addresses for the same position, expressed in four different addressing schemes. The figure shows a position on the eastern edge of the MIT campus, along with interpolation features for each addressing scheme. This position was expressed as *290 Main Street, Cambridge* (street, top left), *(42 21.744N; 71 5.115W)* (latitude/longitude, top right), *Cambridge, Kendall Square .47 .57F* (axis-aligned HIP, bottom left) and *Cambridge, Kendall Square .47 .43M* (rotated HIP, bottom right).

Addressing scheme	Feature	Boston		Pittsburgh	
		Horizontal	Vertical	Horizontal	Vertical
Rotated HIP	Coordinate system	251	147	294	271
Aligned HIP	Coordinate system	269	158	299	270
Latitude/longitude	Line	190	255	195	257
Street	Street segment	156		226	

Table 6.1: This table shows the screen dimensions in pixels of the features used to locate an address in the map location task. These dimensions are within reasonable ranges of one another and did not have a significant effect on the time to locate an address on the map.

6.2.3 Profile of subjects

The usability test was run on 11 users (N = 11, 7 men, 4 women). Subject ages ranged from 20 to 47, with an average of 27.9 years. Familiarity with the greater Boston metropolitan area and unfamiliarity with the Pittsburgh area were the criteria for subject selection. All subjects indicated in a pre-test questionnaire that they were comfortable getting around in the Boston/Cambridge area and that they had never been to the Pittsburgh area.

6.2.4 Independent variables

The independent variables manipulated were:

- Addressing scheme. This independent variable had four levels in the map location task and three in the cognitive load task. In the map location task, four types of addresses were presented: street addresses, latitude and longitude coordinates, HIP addresses with coordinate systems oriented rectilinearly along north-south lines and HIP addresses with coordinate systems that were rotated to arbitrary angles with respect to the horizontal. In the cognitive load task, three types of addresses were presented: street addresses, latitude and longitude coordinates and HIP addresses. There was one less condition than in the map location task because the spoken form of a HIP address is independent of how its coordinate system is oriented.

- City. Addresses were drawn from different metropolitan areas. This independent variable had two conditions: the Boston/Cambridge area and the Pittsburgh area. The purpose of varying this independent variable was to counterbalance the fact that all subjects were drawn from the Boston/Cambridge area. Varying this variable would also prove to be a useful test of how HIP would be used in unfamiliar cities.

6.2.5 Dependent variables

For the map location task, the dependent variables measured were:

- Location estimation error. The X and Y coordinates of each click were recorded and compared to those of the actual location of the address on the map. Euclidean (straight-line) errors were then calculated.

- Time for location. The time between two successive location clicks was recorded. Locating an address on a map involves two subtasks: locating the feature (street, latitude/longitude line or HIP coordinate system) in which the address is located and then interpolating that address within the feature between the given boundary parameters for that feature.

For the cognitive load task, the number of errors made by the users when repeating an address back to the experimenter was recorded.

6.2.6 Subjective opinion variables

In the post-test questionnaire, subjects gave their subjective opinions on the ease of using HIP relative to other addressing schemes. Subjects responded to the following statements:

1. The HIP addressing scheme makes sense to me.

2. HIP addresses were easier to locate on a map than street addresses.

3. HIP addresses were easier to locate on a map than latitude and longitude addresses.

4. HIP addresses were easier to remember than street addresses.

5. HIP addresses were easier to remember and repeat than latitude and longitude addresses.

6. I prefer horizontal HIP rectangles to slanted HIP rectangles,

7. I would be happy to use HIP as a daily addressing scheme.

Responses were collected on a discrete graduated scale with five levels: Strongly Agree, Agree, Neutral/No opinion, Disagree and Strongly Disagree. Subjects also gave the three addressing schemes they used in the study an overall rating for location and navigation tasks. This rating was on a scale of 1 (unsatisfactory) to 7 (very satisfactory)

6.2.7 Limitations

Asking users to locate HIP addresses on a map directly measures their understanding of the HIP conceptual model. A HIP address uses its coordinate system portion to give a user a rough idea of its position. However, precisely locating HIP addresses on a map is asking users to solve a harder problem than they would typically encounter when using HIP. HIP is expected to be used mostly in an device-mediated manner.

Not knowing how exactly HIP would be used would also influence the subjective opinions of subjects towards HIP. From the tasks they were asked to perform, users could think those would be the primary way of using HIP addresses, leading to a slight bias against HIP in some of the subjective opinion items above.

6.3 Results

Data collected from the two tasks were analyzed using the ANOVA test for main effects of and interactions between the addressing scheme and the city of the address.

Addressing scheme	Boston		Pittsburgh	
	Mean	Standard deviation	Mean	Standard deviation
Street	0.22	0.07	0.16	0.05
Latitude/longitude	0.15	0.04	0.32	0.44
Aligned HIP	0.11	0.06	0.18	0.16
Rotated HIP	0.15	0.05	0.26	0.12

Table 6.2: The average Euclidean errors made in each addressing scheme in both cities are shown. The errors above are ratios, because they have been normalized by the average feature sizes in table 6.1.

6.3.1 Location estimation error

The raw Euclidean errors in terms of pixels were normalized by the average feature lengths reported in the note on map preparation above. The addressing schemes in increasing order of mean Euclidean error in Boston were axis-aligned HIP addresses, rotated HIP addresses, latitude/longitude and street addresses. (see Table 6.2).

The addressing schemes in increasing order of normalized mean Euclidean error in Pittsburgh were street addresses, axis-aligned HIP addresses, rotated HIP addresses and latitude/longitude addresses. There was no significant effect of type of address ($F=0.98$, $p < 0.41$) on the location estimation error. There was no significant main effect of city ($F=3.69$, $p < 0.06$) on the location error. There was no significant interaction between the two factors ($F=1.74$, $p < 0.19$).

6.3.2 Classification of errors

The raw user clicks collected from the map location task were plotted on their respective map images, along with the actual locations of those positions. These plots (see Figure 6-2 for an example) were then examined to enable a qualitative classification of the types of errors that subjects committed. In total, there were 440 user clicks to locate aligned and rotated HIP addresses on the map. There were three main types of errors committed by users:

- Interpolation errors. The user clicks from the map location task were found to

deviate from the veridical positions on the map mostly due to imprecise interpolations within the appropriate feature for each addressing scheme. For latitude and longitude and both types of HIP addresses, where subjects performed two-dimensional interpolation, this was because there was no scale displayed on the latitude/longitude quadrants or coordinate systems. There were 395 interpolation errors (**89.7%** of all the errors).

- Coordinate transposition errors. Some users occasionally switched the order of the coordinates in the HIP offset. They mistakenly interpreted the first coordinate of the offset as the vertical displacement in a coordinate system and the second coordinate as the horizontal. The source of this error could be because in latitude and longitude addresses they might have seen earlier, latitudes (coordinates specifying north-south displacement) were specified first. There were 34 coordinate transposition errors (**7.7%** of all the errors).

- Miscellaneous errors. Some users clicked the mouse in the wrong feature on the map because of inadvertent double-clicking or problems with mouse responsiveness. There were 11 miscellaneous errors (**2.5%** of all the errors).

6.3.3 Location time

The addressing schemes in increasing order of mean time to locate addresses in Boston and Pittsburgh were axis-aligned HIP address, rotated HIP addresses, street addresses and latitude/longitude addresses(Table 6.3).

There was a highly significant main effect of addressing scheme ($F=5.03$, $p < 0.003$). There was no significant main effect of city. There were also no significant interactions between the two factors.

6.3.4 Errors under cognitive load

The addressing schemes in increasing order of number of repetition errors in both Boston and Cambridge were street addresses, HIP addresses and latitude and longi-

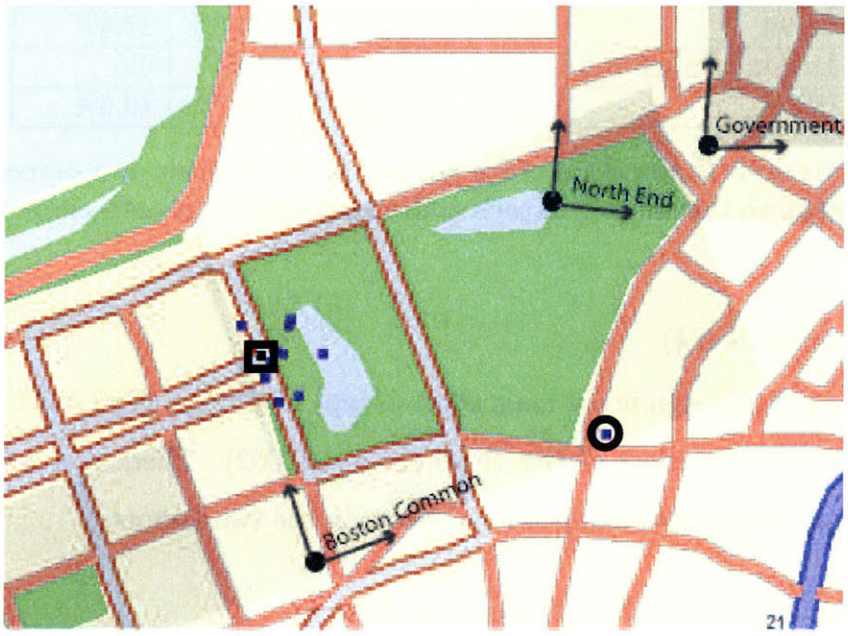


Figure 6-2: These are the clicks users made in response to the HIP address *Boston, Boston Common .05 .68Q*. The true position for this address is highlighted with a heavy square. A coordinate transposition error is marked with a heavy circle.

Addressing scheme	Boston		Pittsburgh	
	Mean	Standard deviation	Mean	Standard deviation
Street	13.82	6.44	18.87	10.91
Latitude/longitude	18.36	7.03	20.56	10.44
Aligned HIP	10.04	3.27	11.42	3.41
Rotated HIP	10.51	3.10	14.81	6.14

Table 6.3: The average times, in seconds, to locate addresses in each addressing scheme in both cities are shown above.

Addressing scheme	Boston		Pittsburgh	
	Mean	Standard deviation	Mean	Standard deviation
Street	12.7%	11.0%	34.5%	17.5%
Latitude/longitude	100%	0%	100%	0%
HIP	52.7%	19.0%	62.7%	18.5%

Table 6.4: This table shows the average number of errors made as a percentage of all addresses spoken to the subject in each addressing scheme in both cities.

tude addresses(Table 6.4).

There was a highly significant main effect of type of address ($F=170.54$, $p < 10^{-25}$) and a significant main effect of city ($F=9.83$, $p < 0.002$). There was a statistically significant interaction ($F=3.48$, $p < 0.04$) between the two factors.

6.3.5 Subjective opinion variables

The five gradations of agreement were coded as +2, +1, 0, -1 and -2, with the positive integers indicating agreement. Using these numeric values, responses to each question were averaged across all users (Tables 6.5, 6.6). Subjects also gave an overall rating to each addressing scheme.

Users generally had a strong consensus ($\mu = +1.18$) that the HIP addressing scheme made sense to them. There was weak agreement ($\mu = +0.55$) that HIP addresses were easier to locate on a map than street addresses. There was stronger agreement ($\mu = +1$) that HIP addresses were easier to locate than latitude and longitude addresses. Users generally disagreed ($\mu = -1.09$) that HIP addresses were easier to remember than street addresses. However, there was strong agreement ($\mu = +1.64$) that they were easier to remember than latitude and longitude addresses. There was a weak preference ($\mu = +0.36$) for axis-aligned coordinate systems over arbitrarily rotated coordinate systems. There was a mild disagreement ($\mu = -0.27$) with the suitability of HIP as an addressing scheme for everyday use. For overall use in location and navigation on a scale of 1 to 7, HIP fared better ($\mu = 4.55$) than latitude and longitude addresses ($\mu = 2.73$), but not as well as street addresses ($\mu = 5.64$).

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree	Average
HIP makes sense	1	-	-	5	5	+1.18
HIP easier to find than street	1	1	2	5	2	+0.55
HIP easier to find than lat/long	1	1	-	4	5	+1.0
HIP easier to remember than street	3	6	2	-	-	-1.09
HIP easier to remember than lat/long	-	-	-	4	7	+1.64
Aligned HIP better than rotated HIP	-	2	5	2	2	+0.36
I'd use HIP	2	3	3	2	1	-0.27

Table 6.5: For each question in section 6.2.6, the number of responses for each level of agreement are shown

Addressing scheme	Worst	2	3	4	5	6	Best	Average
Street addresses	-	-	1	2	1	3	4	5.64
Latitude/longitude addresses	3	3	1	2	2	-	-	2.73
HIP addresses	-	1	2	2	2	4	-	4.55

Table 6.6: Subjects rated each addressing scheme they used on a subjective scale of 1 (unsatisfied) to 7 (very satisfied). The number of users indicating each score is shown in this table.

6.4 Discussion

The results show that neither the addressing scheme nor city had significant main effects on the accuracy of subjects in the map location task, when normalized by average feature lengths on the various maps. We can confidently accept the null hypothesis and conclude that HIP is not significantly different from existing addressing schemes so far as magnitude of errors is concerned. To some extent, this bolsters arguments for using HIP as a common addressing scheme, because HIP may be deemed roughly equivalent to existing addressing schemes in terms of how subjects understood and used it.

The ranking of addressing schemes in both cities in ascending order of normalized Euclidean error is inconclusive with respect to their relative merits. Street address errors fared markedly differently in the two cities, worst in Boston and best in Pittsburgh, a fact that can only be attributed to structural differences between the maps. It is however clear that axis-aligned HIP coordinate systems fared better than rotated coordinate systems in both Boston and Pittsburgh. It is plausible that subjects mentally rotated the rotated coordinate systems back to the horizontal before interpolating HIP addresses within them. In contrast, axis-aligned addresses could be interpolated directly on the map. The mental rotation and subsequent interpolation using a mental image probably led to the rotated HIP addresses having higher mean error than the aligned HIP addresses.

The city had a significant main effect on the time required to locate a HIP address, with Boston performing better than Pittsburgh. This phenomenon probably occurred because all the subjects were more familiar with Boston than with Pittsburgh.

Furthermore, the addressing scheme had a significant main effect on the time to locate an address on the map. HIP addresses, both aligned and rotated, were located faster than the those from other addressing schemes. The time to locate an address can be distributed between two subtasks: locating the right feature (street, latitude/longitude quadrant or coordinate system) for interpolation and then actually performing the interpolation. The first subtask essentially requires subjects to

perform a linear search through the map for each feature. Because the number of labels that subjects had to read on each map was roughly equalized for all the addressing schemes, the actual interpolation subtask was probably the source of most of the variance in the time for locating an address. HIP captures neighborhoods in a town at a coarser level of granularity than street addresses. This feature led to faster performance in the first subtask above. HIP addresses were probably slower to interpolate than street addresses because interpolation was along two dimensions. It may be supposed that the longer the dimension of a feature (street segment length, distance between latitude and longitude lines or widths and heights of coordinate systems) within which interpolation occurs, the longer is the time for interpolation. However, sufficient diligence put into map preparation ensured that no unfair advantage was conferred on any particular addressing scheme due to differences in feature lengths.

Therefore, we can conclude that feature lengths were not a major factor in the time for interpolation once the right feature had been identified. HIP addresses confer a significant advantage over street addresses and latitude/longitude addresses in the time needed to locate an address. HIP addresses cover a region with human-intelligible symbols, but at a coarser level of granularity than street addresses. For people familiar with a town, this means that they can localize HIP addresses more easily to a small region of the map before they interpolate a HIP address. In other words, HIP addresses serve to give a quick intuitive idea of their position, relative to other addressing schemes. Despite HIP's apparent time disadvantage in being interpolated along two dimensions, it ends up being faster to locate than street addresses.

From the results for the cognitive load task, we see significant main effects of both city and addressing scheme. Boston addresses generally witnessed fewer errors than Pittsburgh addresses because subjects were more familiar with Boston street and neighborhood names than Pittsburgh. Looking at the addressing scheme dimension, we find that HIP addresses did better than latitude and longitude addresses, but not as well as street addresses. The greater rate of errors on HIP addresses can be attributed mostly to the number of offset digits that subjects needed to remember.

Street addresses also had a slight advantage in that subjects remember them far more frequently than HIP addresses in their everyday lives. Subjects had probably developed strategies for remembering and communicating street addresses effectively, whereas they were exposed to HIP addresses for the first time in this study. One of the aims of the cognitive load task was to find ways to redesign the HIP check code scheme. However, it was found that less than one error in five was due to the check code. Most of the errors subjects made were in remembering the digits of the HIP offset. For this reason, no useful data regarding redesign of the check code was obtained.

This slight "cultural" advantage of street addresses over the other addressing schemes is also seen in the results for subjective opinion variables described above.

6.5 Chapter Summary

This chapter described how HIP can be compared to existing address schemes in terms of a few usability indicators. These usability indicators were measured indirectly through a multi-factor, within-subjects usability study. The results of the study were quite encouraging overall. They showed that users' understanding of the conceptual model behind HIP was comparable with existing addressing schemes. HIP was not expected to perform better than street addresses overall because street addresses have been in use for years and have cultural and learnability advantages already. However, it was a pleasant surprise to see that HIP addresses came close to street addresses on some measured variables and even surpassed them in the time required to look up an address on a map. The results of the usability studies can be reasonably taken to mean that HIP is a tenable addressing scheme for common use.

Chapter 7

Future Work and Conclusion

While this thesis is important in establishing the tenability of HIP, it remains a fledgling addressing scheme with several possibilities for extension and improvement. This chapter describes ways in which HIP may be extended, improved or used for delivering location-based services. It also serves as the concluding chapter of this thesis.

7.1 Future Work

The implementation of HIP presented in this thesis is a good proof of concept for a hybrid, wide-area location model. However, not all its aspects could be implemented to a high degree of polish due to time constraints. This section describes a few ways in which HIP can be enhanced for every day navigation.

7.1.1 Other substrates

The implementation of HIP presented in this thesis is suitable for use with GPS technology, which is available only outdoors. There is nothing inherent in the design of HIP that ties it to GPS. HIP could in principle be implemented with indoor absolute coordinate systems such as Cricket acting as a substrate.

7.1.2 Check code algorithm

The current implementation of HIP check codes is tied, perhaps too closely, to the internal state of a position object. An ideal check code algorithm would be tied to the position underlying a HIP address with an appropriate sensitivity and in a manner transparent to users. It would also draw from alphabets or alphabet clusters that are easy to say and remember.

7.1.3 Usability improvements

Entering a complete HIP address, especially those with long coordinate system names, into a cellular phone can be time-consuming and inconvenient. Currently, HIP address parsers construct a HIP address only from the input given to them. HIP address parsers could be designed to take more cues from the surrounding context, such as how fast a HIP address is changing, or referring to a set of personalized HIP address aliases, to construct machine-intelligible HIP addresses requiring even less user input. New ways of representing a HIP address could be devised similar to how letters are used to represent phone numbers, e.g. 1-800-FLOWERS. These new ways of condensing HIP address components into more memorable chunks might make HIP addresses easier to remember, possibly at the cost of obscuring their internal structure.

As can be reasonably expected, almost all the context from which a HIP client application draws is located on the client side. The HIP client application developed in this thesis was itself based on the fledgling MIDP 1.0 specification. There was a tension between putting a lot of functionality on a computationally primitive MIDP 1.0 device and making the HIP client device do most of the computation. A newer and more advanced version, MIDP 2.0, was released in November 2002 and the first phone supporting it was released in November 2003. MIDP 2.0 provides remarkable improvements in the user interface toolkit, as well as the ability to open secure network connections. The changes in MIDP 2.0 imply direct improvements for HIP in user interface design as well as in security and privacy.

The usability of HIP can be improved by adding HIP address *hysteresis*, i.e. a mechanism that prevents one's coordinate system from changing too often. The coordinate system of a HIP address may be associated with significant context in a user's mind. If a user is passing through several neighborhoods of a town at a good speed, the neighborhood-scale coordinate systems of their HIP addresses would change frequently, possibly disorienting them. One way to remedy this situation is to generate addresses with respect to the larger coordinate system of the surrounding town, so that the coordinate system of a HIP address does not change too often.

7.1.4 Distributed computing considerations

The research presented in this thesis has two related but distinct aspects: a usability aspect and a systems aspect. This thesis has emphasized the usability aspect to establish that a hybrid addressing model such as HIP is feasible. However, there are quite a few issues to be thought through on the systems side of things. For instance, a well-defined protocol needs to be designed for a HIP client application to operate when multiple coordinate system resolvers are available. MIDP 2.0 supports "J2ME push", a technique for cellular phone service providers to deliver content proactively to mobile devices. The HIP client application could notify a service tower when it starts up and have information about coordinate system resolvers pushed to it.

When multiple coordinate system resolvers service an area, multicasting HIP resolution requests to all of them would not scale well with large numbers of HIP-enabled devices. One way to solve this problem would be for a HIP client to download coordinate system information from a resolver and associate an expiry date with it, not unlike a DHCP lease. Then HIP address resolution could also be moved to the client side. Clever caching techniques would have to be employed to store only the most relevant coordinate systems in the client-side data store at a given time.

7.2 Conclusion

Over the course of this thesis, we have seen the design, implementation and evaluation of HIP, a new addressing scheme that seeks to combine the positive attributes of street addresses and latitude and longitude addresses, while reducing their undesirable attributes. While the conceptual idea behind HIP is simple, both the addressing scheme as well as the software system went through a few major design changes through their evolution. These changes, sometimes unforeseen, were driven by input from users and from fast and cheap prototyping. Ultimately, these changes were borne out by positive results in user tests.

The HIP location model was sensible to understand and is a tenable addressing scheme in principle. A number of indicators point to the growing importance of location-based services. With some additional effort, a polished HIP client application could introduce this addressing schemes to the millions of location-aware cellular phones out there today.

Appendix A

User study materials

This appendix contains the following materials used in the usability study:

- The pre-test questionnaire
- The self-paced, written tutorial about the different addressing schemes in the study
- The maps from the map location task
- The post-test questionnaire

Please circle one: Male Female

Please tell us your age: _____

1. How would you describe your level of experience with the following technologies?

Technology	**None** (never used it, or don't know)	**Little** (used once or twice)	**Some** (used enough to be comfortable)	**Lots** (use regularly)
Global Positioning System (GPS)				
Computer-based map programs, such as MapQuest, Yahoo Maps, Delorme Street Atlas, or Microsoft MapPoint				

2. How would you describe your level of experience with the Boston/Cambridge area?

None (never been, don't know)	**Little** (been once or twice)	**Some** (been enough to be comfortable getting around)	**Lots** (know an extensive area well)

3. How would you describe your level of experience with the Pittsburgh area?

None (never been, don't know)	**Little** (been once or twice)	**Some** (been enough to be comfortable getting around)	**Lots** (know an extensive area well)

4. Do you own a cellular phone? Yes No

Welcome

We are evaluating new computer interfaces for location and navigation around a city. Your participation in this user study will help us better understand existing schemes for location and navigation and evaluate a new scheme we are developing.

You will be working with three location and navigation schemes during this user study:

- Street addresses
- Latitude and longitude coordinates and
- Human Intelligible Positioning (HIP) addresses.

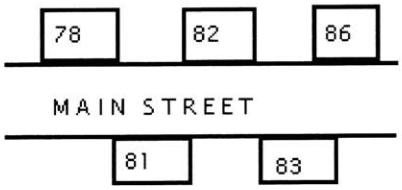
Please take a few minutes to familiarize yourself with how each of these three schemes work.

Street addresses

Street addresses are commonly used to mark locations in a city. One example of a street address is

77 Massachusetts Avenue, Cambridge.

House number | Street name | City



A typical street map

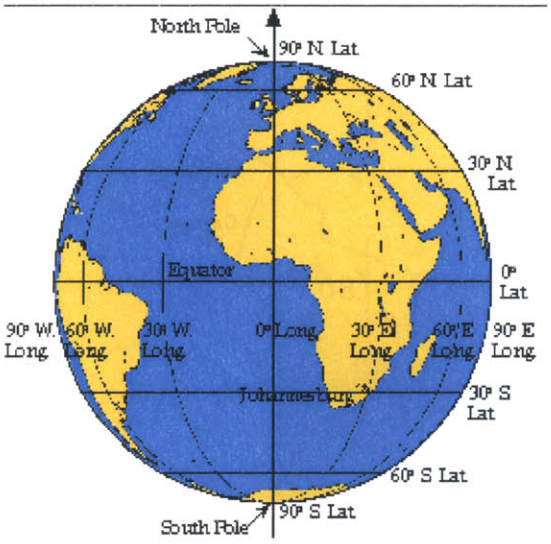
- Street addresses consist of a house number and a street name, usually followed by the city.
- Street numbers are usually whole numbers like 77, but numbers like 93A and 255½ are also found.
- Street numbers usually increase in one direction on a street and decrease in the other (see picture).
- Odd street numbers are usually found on one side of the street, while even numbers face them across the street.

Latitude and longitude coordinates

Latitude and longitude coordinates consist of a pair of decimal numbers. One example of set of these coordinates is

(42.358891N, 71.094115W)

Latitude Longitude



Latitude and longitude lines

- The Earth's surface has several imaginary lines drawn on it to aid location and navigation. The east-west lines are called latitude lines, while the north-south lines are called longitude lines.
- Every point on the Earth's surface can be represented by a pair of numbers, which denotes its location on the grid created by these lines.
- The latitude of a location may be anywhere between 0 and 90 degrees north or south of the equator. The longitude of a location may be anywhere between 0 and 180 degrees east or west of the Prime Meridian, a reference meridian of longitude that passes through Greenwich in the United Kingdom (see picture).
- In an address like (*42.358891N, 71.094115W*), the first number is the latitude and the second number is the longitude.
- One degree of latitude or longitude has 60 minutes. One minute means roughly a mile on the ground.

HIP addresses

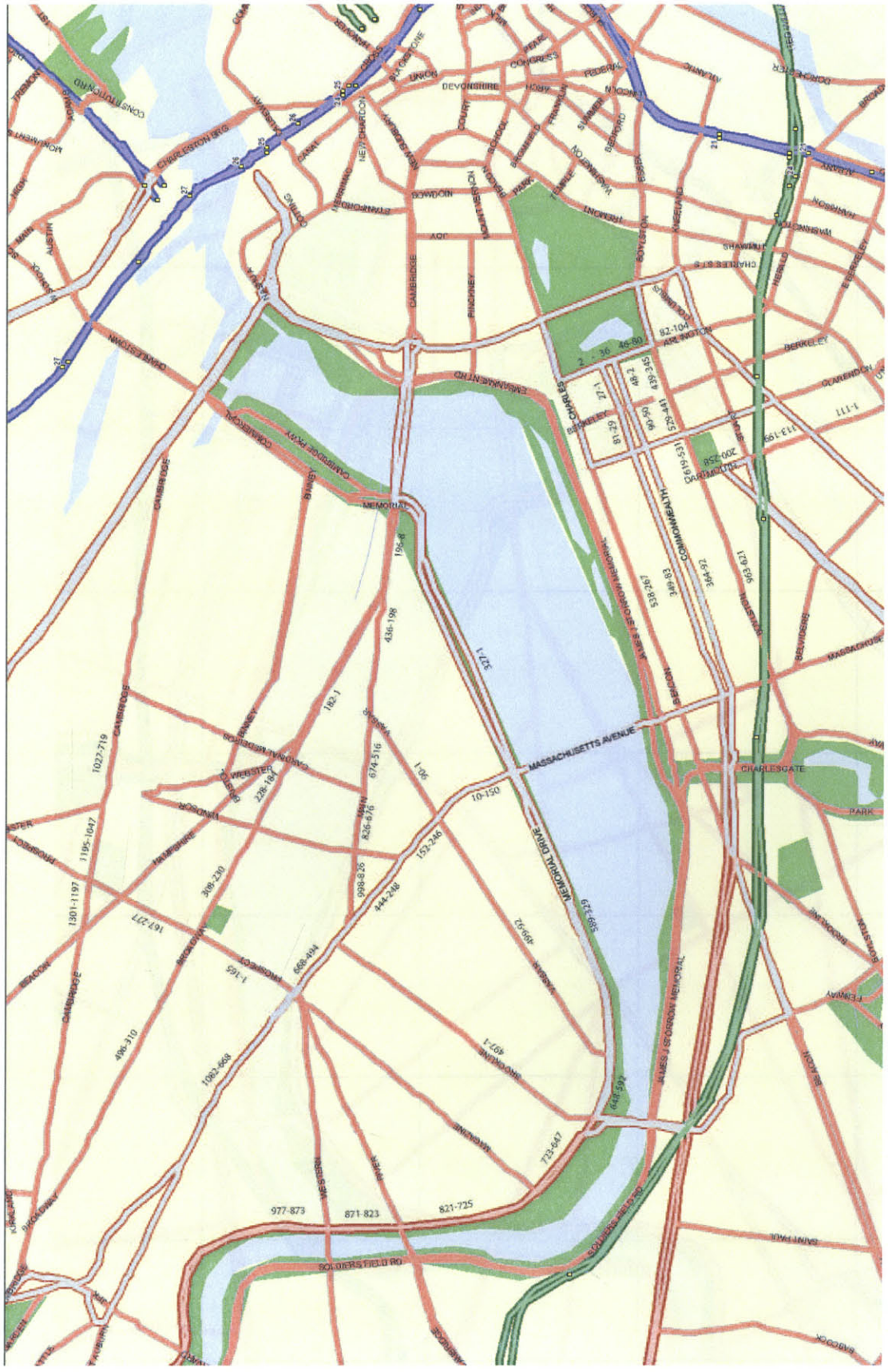
HIP addresses consist of a place name, a pair of numbers and optionally, a check code. One example of a HIP address is

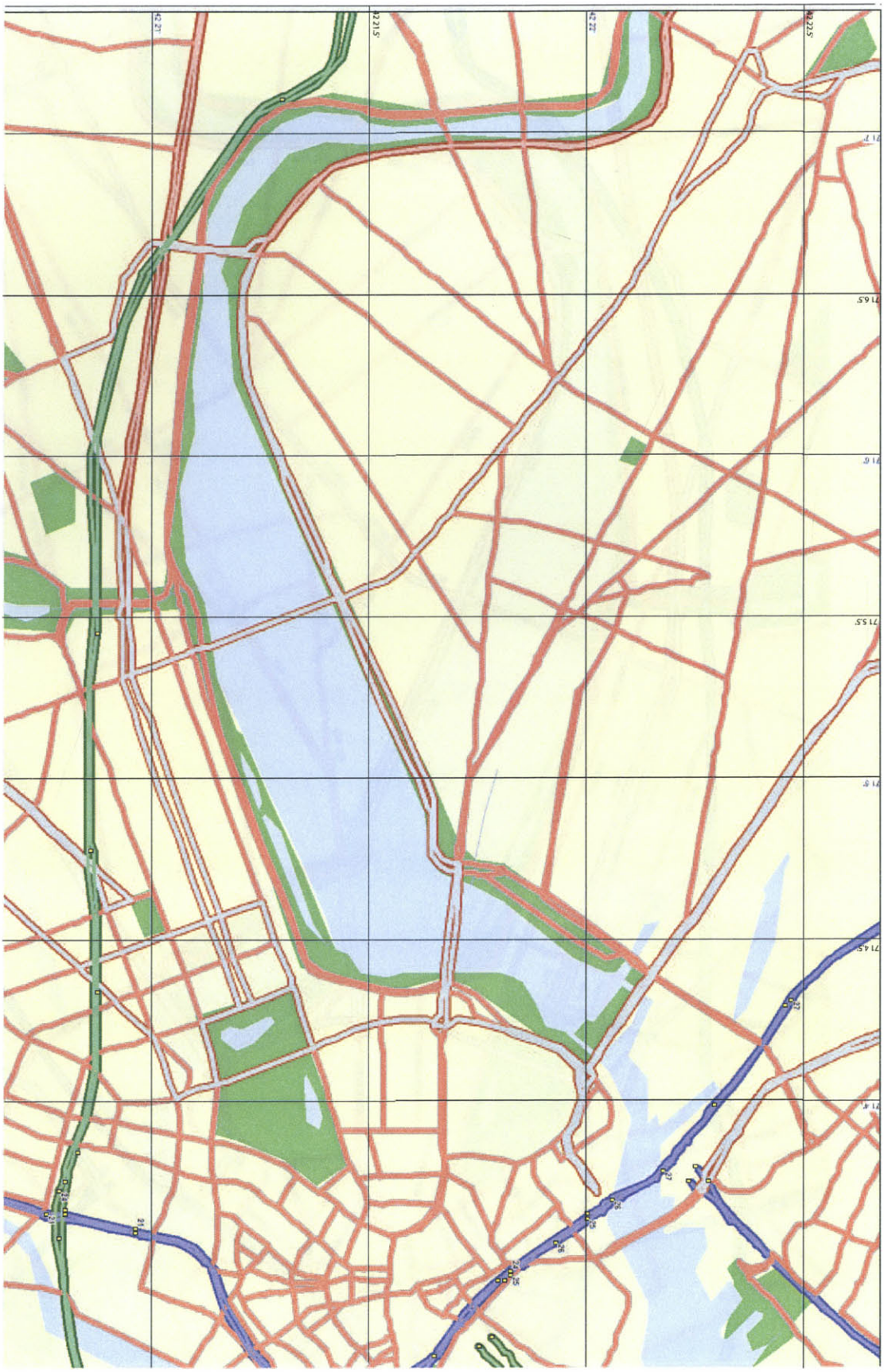
Canada, Ontario, Lake Ontario .80 .75V

Place name | Numbers | Check code

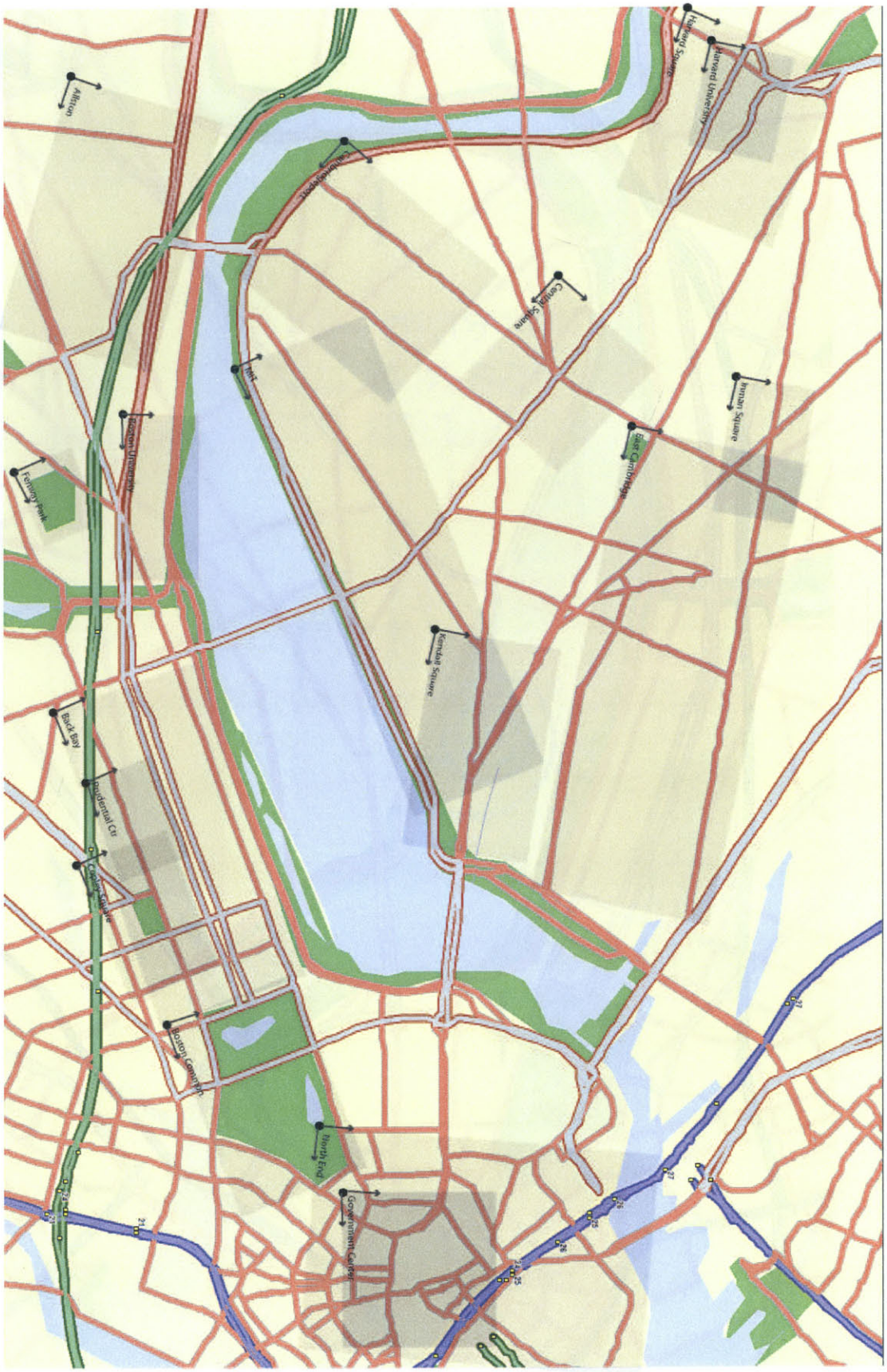
Location of Canada, Ontario, Lake Ontario .80 .75V

- Place names may refer to cities or well-known landmarks or localities in them. *New York, Times Square* and *Washington, White House* are examples of place names.
- The pair of numbers in a HIP address refers to its position inside an imaginary rectangle drawn around the landmark or locality (see Figure 3). The fractions indicate how far along the edge of the surrounding rectangle a point is located. The picture above shows the location of *Canada, Ontario, Lake Ontario .80 .75V*. The first number shows how far to go horizontally and the second shows how far to go vertically.
- The check code, a letter at the end of a HIP address, is an optional part of a HIP address. It is used to ensure that a HIP address is free of errors from user input.

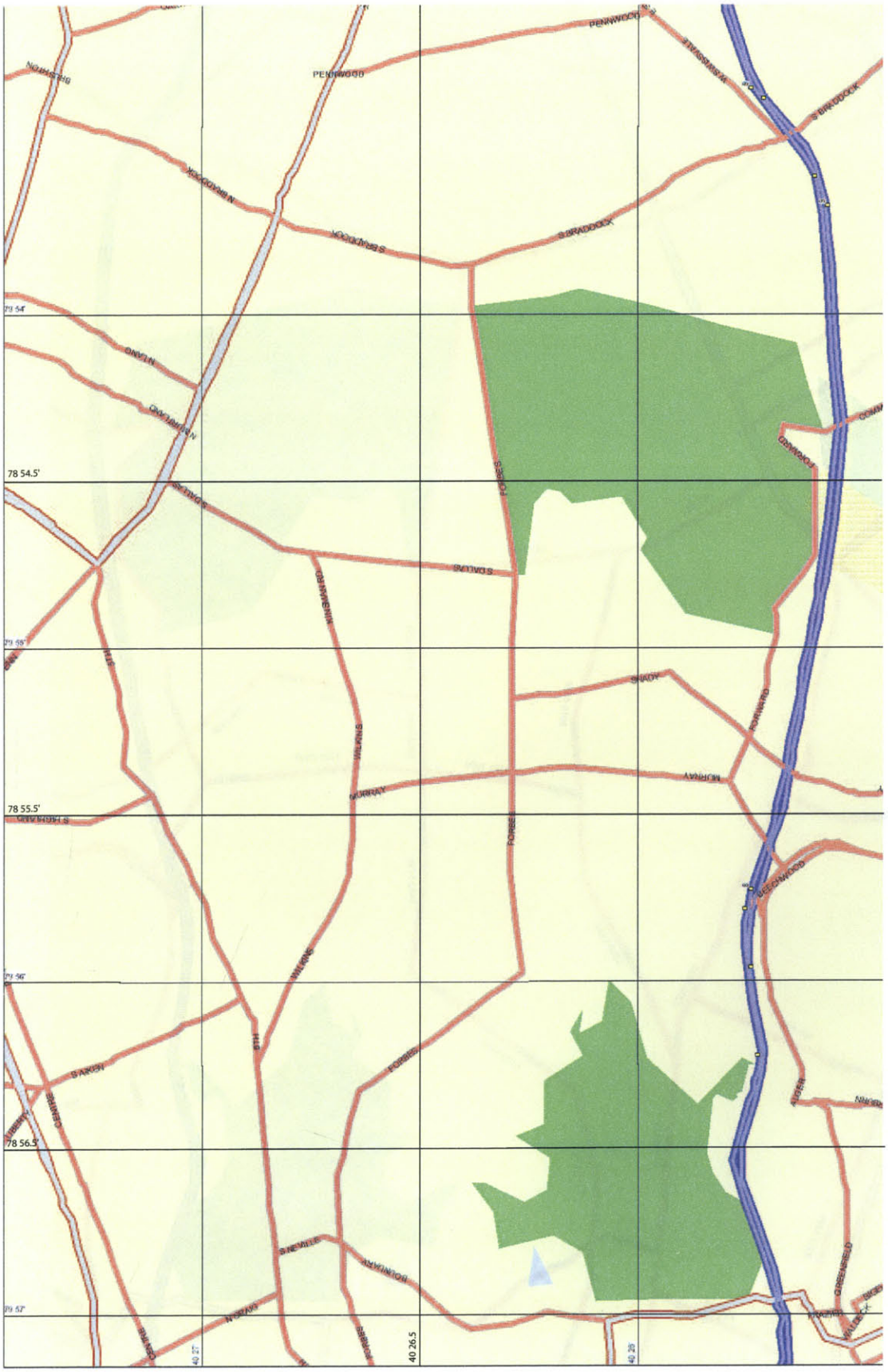

MASSACHUSETTS AVENUE
MEMORIAL DRIVE
SOLDIERS FIELD RD
CHARLESGATE
CAMBRIDGE
COMMONWEALTH
BERKELEY
CLARENDON
DARTMOUTH
PARK
977-873
871-823
821-725


42 21'
42 21.5'
42 22'
42 22.5'
71 7'
71 6.5'
71 6'
71 5.5'
71 5'
71 4.5'
71 4'


Harvard Square
Harvard University
Inman Square
East Cambridge
Central Square
Kendall Square
Cambridgeport
MIT
North End
Government Center
Boston Common
Back Bay
Allston
Boston University
Copley Square
Prudential Ctr
Fenway Park


Allston
Harvard Square
Harvard University
Charlestown
Central Square
Inman Square
East Cambridge
Boston University
Fenway Park
Kendall Square
Back Bay
Prudential Ctr
Copley Square
Boston Common
North End
Government Center


5442-5798
FORBES
5100-5440
4721-5099
4776-5198
5200-5994
5265-5920
501-999
4555-5399
298-100
1301-1699
1701-2399
5800-5898
5900-6598
6600-7598
833-1659
101-831
601-1599
101-599
5944-6734
6528-9109
1700-2248
2250-2598


PENNWOOD
S BRADDOCK
78 54.5'
78 55.5'
78 56.5'
40 26.5


Shadyside
Point Breeze
Wilkinsburg
Mellon Park
Westinghouse Park
Chatham College
Bellefield
Squirrel Hill
Frick Park
frick art museum
Carnegie Mellon
Panther Hollow
Schenley Park
Edgewood
Swissvale
Greenfield


Wilkinsburg
Edgewood
Swissvale
Westinghouse Park
Frick Park
Frick Art Museum
Mellon Park
Point Breeze
Squirrel Hill
Chatham College
Greenfield
Schenley Park
Panther Hollow
Bellefield
Oakland

For each sentence below, please write down how much you agree or disagree with it:

	Strongly agree	**Agree**	**Neutral or no opinion**	**Disagree**	**Strongly disagree**
The HIP addressing scheme makes sense to me.					
HIP addresses were easier to locate on a map than street addresses.					
HIP addresses were easier to locate on a map than latitude and longitude addresses.					
HIP addresses were easier to remember than street addresses.					
HIP addresses were easier to remember and repeat than latitude and longitude addresses.					
I prefer horizontal HIP rectangles to slanted HIP rectangles,					
I would be happy to use HIP as a daily addressing scheme.					

For location and navigation overall, rate the following for ease of use and how much you are satisfied with them (1=not satisfied; 4=neutral/no opinion; 7=very satisfied)

Street addresses	1	2	3	4	5	6	7
Latitude and longitude addresses	1	2	3	4	5	6	7
HIP addresses	1	2	3	4	5	6	7

Appendix B

Addresses in User Study

The following addresses were used in the user study.

B.1 Map Location Task

B.1.1 Street addresses

Boston

- 256 Broadway, Cambridge
- 796 Massachusetts Ave, Cambridge
- 200 Brookline St, Cambridge
- 290 Main St, Cambridge
- 527 Memorial Dr, Cambridge
- 367 Beacon St, Boston
- 282 Commonwealth Ave, Boston
- 780 Boylston St, Boston
- 34 Arlington St, Boston

- 188 Vassar St, Cambridge

Pittsburgh

- 5053 Centre Ave, Pittsburgh
- 699 S. Aiken Ave, Pittsburgh
- 5600 Forbes Ave, Pittsburgh
- 5762 Wilkins Ave, Pittsburgh
- 1999 Shady Ave, Pittsburgh
- 6086 5th Ave, Pittsburgh
- 1195 S Dallas Ave, Pittsburgh
- 1200 S Braddock Ave, Pittsburgh
- 1900 Murray Ave, Pittsburgh
- 198 N. Craig Ave, Pittsburgh

B.1.2 Latitude/longitude addresses

Boston

- (42 22.068N, 71 5.832W)
- (42 22.005N, 71 6.371W)
- (42 21.539N, 71 6.376W)
- (42 21.744N, 71 5.115W)
- (42 21.240N, 71 6.175W)
- (42 21.156N, 71 4.962W)
- (42 21.003N, 71 5.031W)

- (42 20.951N, 71 4.830W)
- (42 21.234N, 71 4.309W)
- (42 21.484N, 71 5.956W)

Pittsburgh

- (40 27.267N, 79 56.618W)
- (40 27.121N, 79 56.179W)
- (40 26.295N, 79 55.655W)
- (40 26.660N, 79 55.666W)
- (40 26.127N, 79 55.129W)
- (40 27.170N, 79 55.363W)
- (40 26.547N, 79 54.744W)
- (40 25.912N, 79 53.584W)
- (40 26.166N, 79 55.366W)
- (40 26.978N, 79 57.032W)

B.1.3 Aligned HIP addresses

Boston

- Cambridge, East Cambridge .18 .03L
- Cambridge, Central Square .33 .78G
- Cambridge, Cambridgeport .55 .65N
- Cambridge, Kendall Square .47 .57F
- Cambridge, MIT .17 .02H

- Boston, Back Bay .42 .57F
- Boston, Back Bay .53 .70D
- Boston, Prudential .68 .35N
- Boston, Boston Common .05 .40V
- Cambridge, MIT .43 .67H

Pittsburgh

- Pittsburgh, Shadyside .28 .63X
- Pittsburgh, Shadyside .55 .65B
- Pittsburgh, Squirrel Hill .23 .46K
- Pittsburgh, Bellefield .07 .92Z
- Pittsburgh, Squirrel Hill .77 .38F
- Pittsburgh, Point Breeze .03 .56V
- Pittsburgh, Frick Park .22 .68E
- Pittsburgh, Edgewood .14 .63D
- Pittsburgh, Squirrel Hill .52 .38A
- Pittsburgh, Shadyside .07 .29L

B.1.4 Rotated HIP addresses

Boston

- Cambridge, East Cambridge .13 .05N
- Cambridge, Central Square .22 .56Y
- Cambridge, Cambridgeport .63 .82H

- Cambridge, Kendall Square .47 .43M
- Cambridge, MIT .10 .03U
- Boston, Back Bay .52 .88J
- Boston, Back Bay .72 .67R
- Boston, Prudential .54 .35P
- Boston, Boston Common .05 .68Q
- Cambridge, MIT .17 .73W

Pittsburgh

- Pittsburgh, Shadyside .25 .95J
- Pittsburgh, Shadyside .55 .65V
- Pittsburgh, Squirrel Hill .25 .47B
- Pittsburgh, Bellefield .95 .90L
- Pittsburgh, Squirrel Hill .75 .37T
- Pittsburgh, Point Breeze .05 .58K
- Pittsburgh, Frick Park .93 .82G
- Pittsburgh, Edgewood .12 .68N
- Pittsburgh, Squirrel Hill .57 .47R
- Pittsburgh, Shadyside .04 .62W

B.2 Cognitive load task

B.2.1 Street addresses

Boston

- 674 Main Street, Cambridge
- 494 Massachusetts Avenue, Cambridge
- 497 Brookline Street, Cambridge
- 1195 Cambridge Street, Cambridge
- 182 Broadway, Cambridge
- 2 Mt. Auburn Street, Cambridge
- 2 Arlington Street, Boston
- 810 Boylston Street, Boston
- 258 Dartmouth Street, Boston
- 468 Beacon Street, Boston

Pittsburgh

- 201 S Neville St, Pittsburgh
- 5090 S Forbes Ave, Pittsburgh
- 5601 5th Ave, Pittsburgh
- 5672 Wilkins Ave, Pittsburgh
- 398 S Highland Ave, Pittsburgh
- 849 S Aiken Ave, Pittsburgh
- 236 Boundary St, Pittsburgh

- 131 Halket St, Pittsburgh
- 6698 Kinsman Rd, Pittsburgh
- 2399 Murray Ave, Pittsburgh

B.2.2 Latitude/longitude addresses

Boston

- 42 21.776N 71 5.619W
- 42 21.810N 71 6.029W
- 42 21.326N 71 4.339W
- 42 21.247N 71 6.625W
- 42 22.404N 71 5.889W
- 42 20.720N 71 5.296W
- 42 21.759N 71 5.395W
- 42 22.204N 71 6.780W
- 42 20.997N 71 4.664W
- 42 21.146N 71 5.452W

Pittsburgh

- 40 26.830N 79 56.808W
- 40 26.646N 79 56.375W
- 40 26.980N 79 55.853W
- 40 26.685N 79 55.761W
- 40 27.258N 79 55.517W

- 40 27.017N 79 56.113W
- 40 26.080N 79 57.013W
- 40 26.287N 79 57.632W
- 40 26.744N 79 54.727W
- 40 25.804N 79 55.397W

B.2.3 HIP addresses

Boston

- Cambridge, East Cambridge .13 .05N
- Cambridge, Central Square .22 .56Y
- Cambridge, Cambridgeport .63 .82H
- Cambridge, Kendall Square .47 .43M
- Cambridge, MIT .10 .03U
- Boston, Back Bay 0.52 0.88J
- Boston, Back Bay 0.72 0.67R
- Boston, Prudential 0.54 0.35P
- Boston, Boston Common .05 .68Q
- Boston, Copley Square .30 .05G

Cambridge

- Pittsburgh, Bellefield .13 .45L
- Pittsburgh, Carnegie Mellon .78 .47B
- Pittsburgh, Chatham College .24 .88Q

- Pittsburgh, Squirrel Hill .56 .73G
- Pittsburgh, Shadyside .82 .68T
- Pittsburgh, Shadyside .43 .98Y
- Pittsburgh, Panther Hollow .14 .87S
- Pittsburgh, Schenley Park .13 .38V
- Pittsburgh, Frick Park .36 .34D
- Pittsburgh, Squirrel Hill .42 .37U

Bibliography

[ADJ99] William Adjie-Winoto, Elliot Schwartz, Hari Balakrishnan, and Jeremy Lilley. "The Design and Implementation of an Intentional Naming System." In Proceedings of the ACM Symposium on Operating Systems Principles, pages 186- 201, Charleston, SC, 1999.

[BLO01] Joshua Bloch, "Effective Java programming language guide", Sun Microsystems, Inc., Mountain View, CA, 2001 pp. 38-41

[BOE88] B. Boehm, "A Spiral Model of Software Development and Enhancement," IEEE Computer, v21 n5, May 1988.

[BOO] K. Boone, "J2ME FAQ," [Online Document], (date unknown) [cited Dec 28, 2003], Available HTTP: http://www.kevinboone.com/j2me.html

[DAN95] P. H. Dana, "Coordinate Systems Overview, "[Online document], July 1995 (revised December 1999) [cited May 15 2003], Available HTTP: http://www.colorado.edu/geography/gcraft/notes/coordsys/coordsys_f.html

[FCC03] Federal Communications Commission, "FCC: Enhanced 911, " [Online document], Nov 2003 [cited Dec 28, 2003], Available HTTP: http://www.fcc.gov/911/enhanced/

[GRE99] Philip Greenspun, "Introduction to AOLServer, Part 2," [Online document], July 1999 [cited Dec 28, 2003], Available HTTP: http://philip.greenspun.com/wtr/aolserver/introduction-2.html

[GRO] Groundspeak, Inc., "GEOCACHING - Frequently Asked Questions," [Online Document], (date unknown) [cited May 15, 2003], Available HTTP: http://www.geocaching.com/faq.asp

[HOW] HowStuffWorks, Inc., "How GPS Receivers Work," [Online Document], (date unknown) [cited Dec 28, 2003], Available HTTP: http://electronics.howstuffworks.com/gps.htm

[IDEN] iDEN Developer Community, http://www.idendev.com

[J2ME] Java 2 Platform, Micro Edition (J2ME), http://java.sun.com/j2me/

[JIA02] C. Jiang & P. Steenkiste, "A Hybrid Location Model with Computable Location Identifier for Ubiquitous Computing," UbiComp '02, Goteborg, Sweden, 2002

[LAW02] C. A. Lawton & J. Kallai, "Gender differences in wayfinding strategies and anxiety about wayfinding: a cross-cultural comparison," Sex Roles: A Journal of Research, Nov 2002, Available HTTP: http://www.findarticles.com/cf_dls/m2294/2002_Nov/97728454/p1/article.jhtml

[LEO98] U. Leonhardt. "Supporting Location-Awareness in Open Distributed Systems," Ph.D. Thesis, Dept. of Computing, Imperial College London, May 1998.

[MIT03] Mitra, Nilo (ed), "SOAP Version 1.2 Part 0: Primer, " [Online Document] June 2003 [cited Dec 28, 2003], World Wide Web Consortium, Available HTTP: http://www.w3.org/TR/2003/REC-soap12-part0-20030624/

[MIDP] Mobile Information Device Profile (MIDP), http://developers.sun.com/techtopics/mobility/midp/index.html

[NIEL94] J. Nielsen, "Heuristics for User Interface Design," [Online Document] 1994 [cited Dec 28, 2003], Available HTTP: http://www.useit.com/papers/heuristic/heuristic_list.html

[PAN] Pandemic Media, "The Black Art of Finding a Japanese address," [Online Document] 1998 [cited Dec 28, 2003], Available HTTP: http://www.pandemic.com/tokyo/addressfinder.cfm

[PGIS] PostGIS, http://postgis.refractions.net

[PGRES] PostgreSQL documentation, "User-Defined Types," [Online Document], (date unknown) [cited Dec 28, 2003], Available HTTP: http://developer.postgresql.org/docs/postgres/xtypes.html

[PRE] PreventDisease.com, "Tell Men Directions, Give Women Landmarks," [Online Document], (date unknown) [cited Dec 28, 2003], Available HTTP: http://preventdisease.com/news/articles/tell_men_directions.shtml

[PRI00] Nissanka B. Priyantha, Anit Chakraborty, Hari Balakrishnan. *The Cricket Location-Support system*, Proc. 6th ACM MOBICOM, Boston, MA, August 2000.

[RYD99] Ryden, Keith (ed), "OpenGIS Simple Features Specification for SQL," [Online Document] May 1999 [cited Dec 28, 2003], Open GIS Consortium, Inc., Available HTTP: http://www.opengis.org/docs/99-049.pdf

[WIK03] WikiMedia Foundation Inc., "Wikipedia: Luhn Formula," [Online Document], September 2003 [cited Dec 28, 2003], Available HTTP: http://en2.wikipedia.org/wiki/Luhn_formula

[STA] STASYS Limited, "World Geodetic System 1984," [Online document], (date unknown) [cited Dec 28, 2003], Available HTTP: http://www.wgs84.com